



Das Geheimnis der Computer

Vorwort

Als ich in der elften Klasse war – das muss 1981 gewesen sein – hat meine Schule einen Computer angeschafft. Als Oberstufenschüler im Mathematik-Leistungskurs genoss ich das Privileg an diesem Gerät von der Größe einer Waschmaschine unterrichtet zu werden – von einem Mathematiklehrer, der sich jeweils am Abend zuvor mühsam die Grundlagen der Programmiersprache „Pascal“ selbst beigebracht hatte. In den Pausen wurde die Maschine später auch für die unteren Klassen freigegeben. Die machten uns Oberstufenschülern dann vor, wie man damit so selbstverständlich umgeht wie mit einem Telefon. Das war das erste Mal in meinem Leben – mit 16! – dass ich mir alt vorkam! Ich hatte einen Trend verpasst, der sich bis heute mit ungebremster Kraft fortgesetzt hat, und der immer noch weiter gehen wird. Inzwischen habe ich selbst Spaß an Computern und was man mit Ihnen alles machen kann. Ganz abgesehen davon, dass er ein selbstverständliches Werkzeug meiner täglichen Arbeit als Ingenieur ist.

Für meine Kinder wollte ich die erste Begegnung mit Computern nicht zu einem solchen Negativerlebnis werden lassen. Deshalb habe ich ihrer natürlichen Neugierde auch kaum Einhalt geboten, wenn es darum ging, sich mit Computern zu beschäftigen. Als ich allerdings im Jahre 2001 einen neuen Computer gekauft habe, und meine damals achtjährige Tochter Ansprüche auf den alten anmeldete: „Papa, dann kann ich ja deinen alten Computer haben!“, musste ich zumindest aufschiebend tätig werden: Ich habe sie auf ihren elften Geburtstag vertröstet, an dem sie dann einen geschenkt bekommen sollte.

Wie wir alle wissen, gehen erstens die Jahre furchtbar schnell ins Land und haben zweitens Kinder in solchen Dingen ein phänomenales Gedächtnis! Jedenfalls war es – nun schon wieder vor etwas längerer Zeit – so weit, dass Katharina zu ihrem elften Geburtstag einen Computer bekommen hat.

Wenn man sich ein Klavier kauft, ist man noch lange nicht musikalisch, und bis man es richtig spielen kann, sind eine Menge Lernen und Üben erforderlich! Komischerweise glauben sehr viele Leute, bei Computern sei das anders!

Gott sei Dank sind Kinder von Natur aus neugierig und wollen von sich aus alles wissen, so dass das „Lernen und Üben“ nicht so unangenehm ist – zumindest so lange man es nicht so nennt. Diesen Vorteil wollte ich für meine Tochter nutzen; da sie eine Leseratte ist, dachte ich mir, ein gutes Buch, das ihr die Zusammenhänge der Computertechnik erläutert, sei genau das Richtige.

Es sollte ein ansprechend aufgemachtes Buch sein, das zu lesen genauso viel Spaß macht wie eine nette Geschichte. Es sollten auch nicht einfach Schritt-für-Schritt-Anleitungen drin sein, sondern es sollte vermitteln, warum die Dinge so und nicht anders sind, denn nur dann wird aus Wissen *Verstehen*, und nur Verstehen ermöglicht es, Probleme zu lösen und kreativ zu arbeiten, und nur so kann man selbständig weitere Erfahrungen sammeln. Nicht zuletzt sollte das Buch auch sprachlich der jungen Leserin entgegenkommen, aber gleichzeitig alle Begriffe präzise erklären und benutzen.

Wer schon mal in einer Computerbuch-Abteilung gestöbert hat, weiß, wie viel Literatur es zu diesem Thema gibt; trotzdem habe ich kein Buch gefunden, das meinen Vorstellungen über das richtige Buch für meine Tochter entsprach!

So ist die Idee geboren, dieses Buch zu schreiben! Inzwischen habe ich es auch das eine oder andere Mal überarbeitet, denn nun wird auch mein Sohn bald elf Jahre alt und meldet natürlich die Gleichbehandlung mit seiner Schwester an. Er möchte auch einen eigenen Computer haben!

Michael Janßen

Eckernförde, Mai 2007

Inhaltsverzeichnis

1	Wie dieses Buch geschrieben ist	8
2	Was ist ein Computer?	9
3	Nullen und Einsen	12
3.1	Wir bauen einen einfachen Computer	13
3.2	Größere Zahlen	19
3.3	Bits und Bytes für Menschen – das 16er-System	23
3.4	Texte mit Nullen und Einsen	25
3.5	Bilder mit Nullen und Einsen	31
3.6	Ein Programm für unseren Computer – natürlich mit Nullen und Einsen	40
4	Halbleiter und richtige Computer	47
4.1	Der Transistor	47
4.2	Immer im Takt	48
4.3	Arbeitsspeicher	49
4.4	Die CPU	51
4.5	Befehle	53
5	Was man einstöpseln kann	56
5.1	Zentraleinheit	56
5.1.1	Netzteil	56
5.1.2	Mainboard	57
5.1.3	Speicher	59
5.2	Schnittstellen	73
5.2.1	Erweiterungskarten	74
5.2.2	USB	76
5.2.3	Plug & Play	77
5.2.4	COM:	77
5.2.5	LPT1:	78
5.2.6	PS/2	78
5.2.7	PCMCIA	78
5.2.8	Bildschirm	79
5.2.9	Netzwerk	80
5.2.10	Drahtlose Schnittstellen	80
5.2.11	Benutzerschnittstelle	81

5.3	Bildschirm	107
5.4	Beamer 108	
5.5	Drucker 108	
5.6	Verbindung mit anderen Computern	111
5.7	Die Augen des Computers	112
5.7.1	Scanner	112
5.7.2	Digitalkamera	113
6	Bedienung	115
6.1	Booten 115	
6.2	Betriebssystem	118
6.2.1	Windows Benutzerschnittstelle	121
6.2.2	Fenster	135
6.2.3	Zwischenablage	144
6.2.4	Arbeitsplatz	145
6.2.5	Explorer	146
6.2.6	Hilfe	156
7	Software	158
7.1	Installieren von Anwendungsprogrammen	158
7.2	Laden von Programmen und Daten	159
7.3	Anwendungsprogramme	160
7.3.1	Textverarbeitung	160
7.3.2	Tabellenkalkulation	181
7.3.3	Datenbank	190
7.3.4	Präsentation	194
8	Netzwerke	195
8.1	HTML 197	
8.2	Der Browser „Internet Explorer“	199
8.2.1	Adressen	200
8.2.2	Surfen	202
8.2.3	Suchmaschinen	202
8.3	E-Mail 202	
8.4	Gefahren des Internets	205
8.4.1	Viren	206
8.4.2	Trojaner	207

8.4.3	Dialer	207
8.4.4	Würmer	208
8.4.5	Datensammler	209
9	Programmieren	210
9.1	Unsere erste Programmieraufgabe	211
9.1.1	Testen des Algorithmus	213
9.1.2	Verbessern des Algorithmus	215
9.2	Programmelemente	218
9.3	Programmierumgebungen	222
9.4	Implementieren des Programms	224
10	Quiz	233
10.1	Fragen	233
10.2	Lösungen	247
11	Glossar	248
12	Index	255

1 Wie dieses Buch geschrieben ist

Die Entwicklung der Computer wurde wesentlich in Amerika vorangetrieben. (Obwohl die ersten grundlegenden Ideen dazu aus Deutschland stammten!) Aus diesem Grund, und weil moderne Sachen sich noch moderner anhören, wenn man ihnen englische Namen gibt, stammen viele Bezeichnungen der Computerwelt aus der englischen Sprache. Wann immer ein solcher Begriff auftaucht, werde ich eine Übersetzung oder zumindest Erklärung geben, was er bedeutet.

Überhaupt werden eine Menge neuer Worte und Begriffe auf dich einströmen. Viele erwähne ich nur, damit du sie einmal gehört – bzw. in diesem Fall gelesen – hast. Man lässt sich dann nicht mehr so schnell ins Boxhorn jagen, wenn jemand versucht, Eindruck zu schinden, indem er Wörter verwendet, die sein Gegenüber noch nie gehört hat.

Wichtige Begriffe und ihre Erklärungen, manchmal sind es auch so genannte Definitionen, also einfache Festlegungen, was ein Begriff zu bedeuten hat, sind in solchen Kästchen eingerahmt wie dieser Absatz hier.

Worte oder Wortteile, die ich besonders betonen möchte, sind *kursiv gedruckt*, so nennt man das, wenn die Buchstaben ein wenig *schräg* stehen.

Worte, zu denen du an anderer Stelle in diesem Buch etwas nachlesen kannst, haben einen → Pfeil vorangestellt. Du kannst sie im Stichwortverzeichnis (Index) ab Seite 255 nachschlagen und findest dort die Seitenzahl, wo es weitere Informationen darüber gibt. Eine Kurzerklärung findest du im Glossar ab Seite 248. Das soll eine Hilfe sein, wenn du eine Stelle mal nicht auf Anhieb verstehst, weil du das betreffende Wort noch nicht kennst oder seine Bedeutung wieder vergessen hast.

An einigen Stellen des Buches sind *Experimente* beschrieben. Dort kannst du das Gelernte an deinem eigenen Computer ausprobieren, damit du auch glaubst, dass das stimmt, was du gelesen hast. Diese Experimente sind als solche gekennzeichnet und beinhalten durchnummerierte Schritte, so dass du nicht viel falsch machen kannst.

Experimente sind in dieser Schrift geschrieben.

Wenn ich im Text bestimmte Tasten der Tastatur meine, sind diese in eckigen Klammern geschrieben: [X] bedeutet also die Taste für den Buchstaben X.

Na und den Rest wirst du ja sehen, deshalb jetzt genug der Vorrede!

2 Was ist ein Computer?

Es gibt einen berühmten, alten Film: „Die Feuerzangenbowle“. Darin fragt der Lehrer Hempel seine Schulklasse: „Wat is 'n Dampfmaschien?“ (Leider kann man die komische Aussprache beim Schreiben nicht so richtig nachmachen. Das soll heißen: „Was ist eine Dampfmaschine?“)

Er beantwortet die Frage daraufhin gleich selbst: „'n Dampfmaschien is ene jrooße schwarze Kiste, und da sin zwej Löcher drin! In dat ene Loch da jeht de Dampf rein, un dat annere Loch – dat krieje mer spääter!“

Zu der Zeit als dieser Film entstand, waren Dampfmaschinen weit verbreitet und in der Technik sehr wichtig. Bestimmt interessiert dich auch wie eine Dampfmaschine funktioniert! Sie wird dir aber kaum noch irgendwo im täglichen Leben begegnen. Computer hingegen sind heute sogar noch weiter verbreitet als früher die Dampfmaschinen und vielleicht hast du ja in deinem Zimmer sogar einen stehen?! (Vielleicht steht da ja auch eine Spielzeug-Dampfmaschine, was mich sehr neidisch machen würde!) Jedenfalls soll hier erst mal der Computer erklärt werden: Wofür man ihn braucht, wie er funktioniert und wie man ihn bedient. Und zwar genauso unkompliziert, wie der Lehrer Hempel in dem Film die Dampfmaschine erklärt:

Ein Computer ist eine kleine beige Kiste. Jedenfalls klein im Vergleich zu einer Dampfmaschine! Ungefähr so groß wie ein großes Weihnachtsgeschenk! Dazu gehört meist eine Art Fernseher.

Es gibt verschiedene Bauformen: Solche die man sich auf den Schreibtisch stellt, heißen Desktop-Computer (von englisch *desk* = Schreibtisch und *top* = oben drauf), solche die ein turmartiges Gehäuse haben, nennt man auch oft Tower (sprich: tauer, englisch für Turm); die verstaut man meist irgendwo unter dem Schreibtisch. Kleine schnuckelige tragbare, die man auf dem Schoß benutzen kann, heißen Laptop (sprich: läpptopp, englisch *lap* = Schoß) oder Notebook (sprich: noutbuck, englisch für Notizbuch) und es gibt sogar superkleine, die nur so groß sind wie eine Handfläche, die heißen deshalb Palmtop (englisch *palm* = Handfläche). Natürlich unterscheiden sich Computer auch in ihren Innereien und ihrer Leistungsfähigkeit, aber das war bei den verschiedenen Dampfmaschinen auch schon so und braucht uns erst mal nicht weiter zu interessieren.

Der Fernseher, der zu einem Computer gehört, heißt *Monitor* oder *Bildschirm*.

Löcher wie die Dampfmaschine haben die meisten Computer auch, sogar mehr als zwei, aber da geht kein Dampf rein sondern zum Beispiel Strom, Disketten, CDs, DVDs (was das genau ist und wofür man es braucht, kriegen wir später), und meistens gibt es auch ein Loch, in das Luft hineingeht; die braucht der Computer, damit er nicht zu heiß wird. Deshalb solltest du dieses Loch niemals

verschließen (z.B. durch Möbel, Stoff, Papier oder Ähnliches), sonst kann der Computer kaputt gehen.

Was macht nun ein Computer? Wörtlich übersetzt bedeutet das englische Wort *Computer* „Rechner“, aber wenn du schon mal etwas mit einem Computer gemacht hast, wird es wahrscheinlich nicht rechnen gewesen sein, sondern du hast vielleicht ein Spiel gespielt, einen Text geschrieben, ein Bild angesehen oder sogar gemalt. Rechnen ist für diese Sachen ein etwas komisches Wort, aber wir werden später noch sehen, dass all das für den Computer wirklich nichts anderes ist als Rechnen, nichts anderes als Zahlen addieren und hin und her schieben. Zu allem Überfluss rechnet der Computer mit *nur zwei verschiedenen* Zahlen: Null und Eins! Andere Zahlen kennt der Computer nicht! Ganz schön blöde was?

Ja, Computer sind dumm! – Aber dafür sind sie unheimlich schnell und deshalb sehen sie für uns so schlau aus! Ein Beispiel:

Wenn du 5×123 im Kopf rechnen willst, rechnest du normalerweise $5 \times 100 + 5 \times 20 + 5 \times 3$ sind $500 + 100 + 15$ macht am Ende 615.

So hast du eine relativ schwierige Aufgabe in kleinere, einfache Teilaufgaben zerlegt. Genau das macht ein Computer auch! Und die vielen kleinen Aufgaben kann der Computer dann unheimlich schnell rechnen.

Und woher weiß der Computer was er rechnen und wie er die Aufgaben zerlegen soll? Auch in dieser Hinsicht ist ein Computer sehr, sehr dumm: Man muss es ihm sagen! Alles!

Man sagt einem Computer was er tun soll in einem *Programm*. Dieses Wort hast du bestimmt schon einmal gehört! Im Fernsehen, gibt es das Fernseh*programm*, das uns sagt, auf welchem Kanal um welche Uhrzeit welche Sendung zu sehen ist. Auf dem Computer gibt es zum Beispiel ein Textverarbeitungs*programm*, mit dem man Texte schreiben und verändern kann, so wie ich das mit dem Text getan habe, den du gerade liest. Dieses Programm sagt dem Computer was er in welcher Reihenfolge tun soll.

Programme sind der Grund, warum es heute so viele Computer gibt: Wenn man eine Aufgabe zu erledigen hat, braucht man nur ein Programm zu schreiben, das dem Computer sagt, wie er die Aufgabe erledigen soll, und schwupps hat man keine Mühe mehr damit. Natürlich klappt das nicht bei allen Arten von Aufgaben – Zimmer aufräumen zum Beispiel können Computer nicht so gut! Aber es gibt eine unglaubliche Menge von Dingen, die Computer erledigen oder zumindest leichter machen können, und deshalb begegnet man ihnen fast überall – und viel öfter als Dampfmaschinen!

Natürlich macht es auch eine Menge Arbeit, ein Programm zu schreiben! Deshalb setzt man Computer und ihre Programme immer dann ein, wenn man eine Aufgabe so oft zu erledigen hat, dass es sich lohnt, *einmal* die Arbeit auf sich zu

nehmen, das Programm zu schreiben. Von da an erledigt der Computer die eigentliche Aufgabe.

Für die meisten Anwendungen, die man heute so für einen Computer hat (Texte schreiben, Bilder malen, Spiele spielen, usw.) haben schon andere Leute Programme geschrieben, die man kaufen kann. Man kann aber fast jeden Computer auch selbst programmieren! Und das macht sogar richtig Spaß! Allerdings muss man dafür auch einiges lernen, denn dein Computer versteht natürlich nicht, wenn du ihm sagst: „Rechne mir die Durchschnittsnote meines Zeugnisses aus!“ Ein wenig mehr darüber erfährst du im Verlauf dieses Buches und speziell im Kapitel 9.

Programme kann man auch ziemlich leicht verändern, zum Beispiel wenn man sie verbessern will, oder wenn sich die Aufgabe ein wenig geändert hat. Im Englischen sagt man deswegen zu Programmen auch *Software*, was so viel heißt wie „weiche Sachen“. Damit soll ausgedrückt werden, dass man Software leicht bearbeiten und verändern kann. Im Gegensatz dazu ist die *Hardware* (also die „harten Sachen“) alles was sich nicht so leicht verändern lässt, nämlich der Computer selbst und alle Geräte, die so dazugehören. Man kann auch sagen, dass man Hardware anfassen kann, und Software nicht.

Ein Computer braucht immer beides! Mit Programmen kann man nichts anfangen, wenn man keinen Computer hat, und jede Hardware ist nutzlos, wenn man keine Software dafür hat.

3 Nullen und Einsen

Im vorangegangenen Kapitel hast du gelesen, dass Computer nur mit zwei Zahlen arbeiten: Mit der Null und der Eins. Jetzt geht es darum, wieso das so ist und wie das geht.

Wenn man etwas Neues erfindet, muss man immer ganz einfach beginnen. Bei Computern war das auch nicht anders und in diesem Buch wollen wir es auch so halten! Stell dir also vor, du wolltest eine Rechenmaschine erfinden und müsstest ihr – damit sie rechnen kann – erst mal die Ziffern beibringen und ihr sagen was welche Ziffer bedeutet.

Ziffern nennt man die Zeichen, mit denen man Zahlen schreibt. Bei uns gibt es also die Ziffern 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 mit deren Hilfe man z.B. die Zahl Hundertvierundfünfzig so schreibt: 154

Um deiner Rechenmaschine die Ziffern beizubringen, müsstest du etwas erfinden, das die Ziffern darstellt. Dabei könntest du zum Beispiel auf die Idee kommen, Drähte in der Form der Ziffern 1, 2, 3, usw. zu biegen, und sie in den Computer einzubauen. Und dann? Es ist schwer vorzustellen, dass eine Maschine diese gebogenen Drähte entsprechend ihrer Bedeutung 1, 2, 3, usw. verarbeiten, also z.B. damit rechnen kann.

Also könntest du dir als nächstes überlegen, noch einfacher anzufangen: Was bedeutet eine Zahl überhaupt? Sie sagt dir, *wie viel*. Also zum Beispiel wie viele Bücher in deinem Regal stehen. Stell dir vor, dich fragt jemand: „Wie viele Bücher stehen in deinem Regal?“ Du würdest hingehen, sie zählen und zum Beispiel antworten: „Neun!“ Ein kleines Kind, das gerade erst sprechen kann, könnte diese Antwort aber nicht geben! Die Frage ist zu schwierig, weil es noch nicht zählen kann. Welche einfachere Frage zu dem Bücherregal, könntest du dir vorstellen, die auch ein ganz kleines Kind, beantworten kann?

Wie wäre es mit: „Stehen in dem Regal Bücher?“

Auf diese Frage gibt es nur die Antworten *ja* oder *nein*. Einfacher geht es nicht! Das ist die kleinste Menge an Information, die man sich überhaupt vorstellen kann: Die Unterscheidung zwischen zwei Möglichkeiten wie

- es ist etwas da oder es ist nichts da
- etwas ist wahr oder falsch
- etwas ist eingeschaltet oder ausgeschaltet
- ja oder nein

Diese kleinste mögliche Informationsmenge, die Unterscheidung zwischen zwei Zuständen, nennt man auch ein Bit.

Zurück zu unserem einfachsten Computer: Wie können wir nun diese zwei „Ziffern“ (etwas ist da oder eben nicht) in der Maschine darstellen? Hast du eine Idee?

Wir könnten zum Beispiel eine Lampe einbauen, die wir entweder ein- oder ausschalten können, oder ein Loch mit einem Deckel versehen, der entweder auf oder zu ist, oder einen Magneten, der seinen Nordpol oben oder unten hat, oder, oder, oder.

3.1 Wir bauen einen einfachen Computer

Bleiben wir der Einfachheit halber bei einer Lampe, die wir mit einem Schalter ein- und ausschalten können. Fertig ist unser Computer!

„Wie bitte? Was soll daran ein Computer sein?“ wirst du dich fragen.

Nun ja, er kann noch recht wenig, nämlich nur anzeigen, ob wir den Schalter betätigt haben oder nicht; genau genommen sagt er nur, ob wir den Schalter eine gerade oder ungerade Anzahl Male betätigt haben, bzw. ob Strom durch die Lampe fließt oder nicht. Aber immerhin: Wir können eine Zahl eingeben! Wenn wir den Schalter betätigen geben wir eine Eins ein, wenn wir ihn nicht betätigen, geben wir eine Null ein; so legen wir das hier einfach mal fest. Und der Computer teilt uns eine Zahl mit (man sagt auch er *gibt sie aus*): Wenn die Lampe brennt, gibt er eine Eins aus, wenn sie nicht leuchtet, gibt er eine Null aus – und wenn wir vergessen haben, den Strom anzuschließen, könnte das ausgegebene Ergebnis falsch sein!

Abbildung 1 und Abbildung 2 zeigen dir das ganze noch einmal als einfaches Bild.

Eine Lampe leuchtet immer dann, wenn der Strom durch ein Kabel bis zu ihr hin fließen kann. In Wirklichkeit muss er auch durch ein zweites Kabel wieder zurückfließen können, aber wir nehmen an, dass das immer der Fall ist und lassen dieses zweite Kabel in den Bildern einfach weg, damit es etwas übersichtlicher wird.

In den Bildern werden die Kabel als Striche gezeichnet. Immer wenn es einen durchgehenden Strich von links nach rechts gibt, leuchtet die Lampe und ist gelb ausgemalt. Ist der Strich irgendwo unterbrochen, bleibt die Lampe aus und in der Zeichnung weiß. Das durchgehende Kabel wird rot gezeichnet, damit du den Weg des Stroms sehen kannst. Kabel, durch die kein Strom fließen kann, bleiben schwarz.

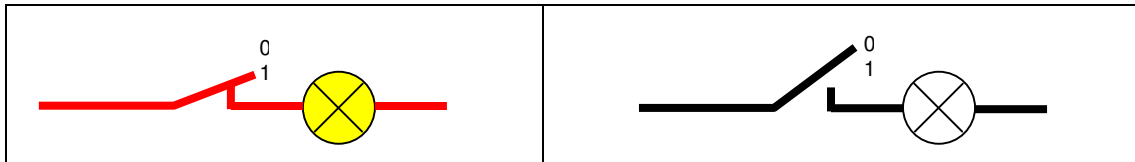


Abbildung 1: Schalter ein (er steht auf 1), die Lampe leuchtet *Abbildung 2: Schalter aus (er steht auf 0), die Lampe leuchtet nicht*

Fällt dir auf, dass wir in diesem einfachen Computer sogar zwei verschiedene Möglichkeiten verwendet haben, die Zahlen Null und Eins darzustellen? Der Schalter kann offen oder geschlossen sein – er stellt die Zahl dar, die wir eingeben – und die Lampe kann leuchten oder nicht – sie stellt die Zahl dar, die der Computer uns ausgibt!

Nun wirst du bestimmt schon ungeduldig, denn unser Computer soll ja rechnen!

Auch dabei wollen wir wieder ganz einfach anfangen: Das einfachste Rechnen ist das Addieren (zusammenzählen, „plus“ rechnen, „und“ rechnen) von zwei Zahlen. Oh! *Zwei* Zahlen! Bisher können wir nur eine Zahl eingeben. Wir müssen unseren Computer also erweitern! Die zweite Zahl wollen wir genauso eingeben wie die erste, nämlich mit einem Schalter. Das Ergebnis der Rechnung lassen wir wieder mit der Glühlampe anzeigen. Abbildung 3 zeigt dir einen solchen Computer:

So wie unser Rechner dort gezeichnet ist, haben wir ihm die Aufgabe gegeben, Null plus Null zu rechnen, denn der obere Schalter ist offen, die eine Zahl die wir eingeben ist also Null, und beim unteren Schalter ist es genauso.

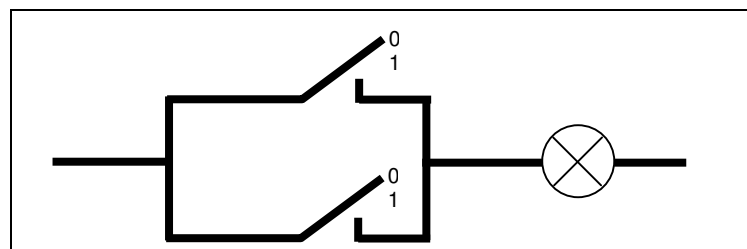


Abbildung 3: Computer, der zwei Zahlen addieren kann

Und was hat unser Computer ausgerechnet? Es gibt *kein* durchgehendes Kabel vom linken Ende der Zeichnung bis zur Lampe, sie leuchtet also *nicht*. Also ist das ausgegebene Ergebnis *Null*! Das stimmt! Hurra!

Aber halt! Was ist, wenn wir eine andere Aufgabe rechnen, zum Beispiel $0 + 1 = ?$

Übersetzt in die Sprache unseres Computers bedeutet diese Aufgabe: Ein Schalter ist offen (Null) und einer ist geschlossen (Eins), wie es in Abbildung 4 gezeigt ist:

Der Strom kann von links über den unteren Schalter bis zur Lampe fließen. Sie leuchtet! Der Computer sagt also „1“! Das Ergebnis ist wirklich Eins! Richtig!

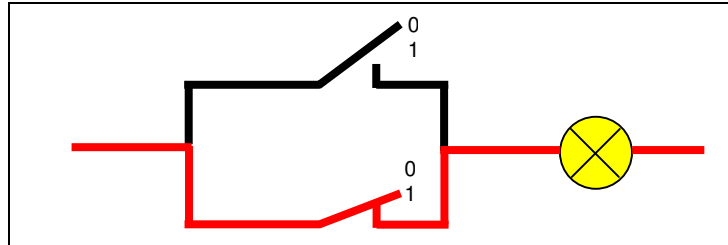


Abbildung 4: Der Computer rechnet $0 + 1$

Wie wäre es, wenn wir

$1 + 0$ rechnen wollten? Nun

das kannst du dir leicht überlegen: Wir würden den oberen Schalter schließen und der untere bliebe offen; der Strom könnte dann über den oberen Schalter bis zur Lampe fließen und sie würde auch leuchten! Auch dann wäre das Ergebnis richtig!

Und was ist, wenn wir $1 + 1$ rechnen wollen? Dann würden wir den oberen *und* den unteren Schalter schließen. Die Lampe würde leuchten, der Computer gibt also das Ergebnis „1“ aus! $1 + 1 = 1$, was ist denn das für ein Quatsch! Das richtige Ergebnis wäre doch *Zwei*!

Was ist denn jetzt falsch gelaufen? Warum gibt unser Computer nicht „2“ aus? – Ganz einfach: Er kann es nicht! Er kann es nicht, weil sein Ausgabegerät, sein „Bildschirm“, die Lampe nur die Zahlen Null und Eins anzeigen kann, indem sie aus oder an ist. Die Zahl Zwei haben wir unserem Computer noch nicht beigebracht!

Wie bringen wir unserem Computer das jetzt wieder bei? Das mit der Null und der Eins hat ja bisher ganz gut geklappt und es wäre doch schön, wenn unser Computer damit auskommen würde!?

Nun, wir Menschen benutzen zehn Ziffern: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Wir können aber doch trotzdem größere Zahlen darstellen als Neun. Also muss es doch auch möglich sein, mit zwei Ziffern größere Zahlen als Eins darzustellen! Wenn wir größere Zahlen als Neun darstellen wollen, also zum Beispiel „Zwölf“, dann helfen wir uns, indem wir eine weitere Ziffer an einer zweiten Stelle dazuschreiben: 12!

Jedes Mal wenn wir bei der ersten Stelle (der *rechten*) keine Ziffern mehr haben, um weiter zu zählen, dann erhöhen wir die zweite Stelle (die *linke*) um Eins. Also: Bis Neun haben wir genügend Ziffern, um mit einer Stelle auszukommen; wenn wir dann weiter zählen wollen, schreiben wir „10“. Wir erhöhen

also die zweite Stelle um eins – und fangen bei der ersten Stelle wieder mit Null an!

*Die Stellen einer Zahl werden in diesem Buch **von rechts nach links** gezählt, weil man die rechte Stelle beim Zählen als erstes braucht, und die anderen erst danach hinzukommen.*

Man kann das sehr schön am Kilometerzähler eines Autos sehen: Wenn das Auto fährt, verändert sich nur die rechte Stelle und zählt die Kilometer (bei manchen Autos werden ganz rechts die Hektometer (=100m) gezählt). Sobald an der rechten Stelle die 9 erscheint, wird beim nächsten Weiterzählen die zweite Stelle eins weiter gezählt und an der ersten Stelle erscheint wieder die Null.

Genauso könnte das doch auch unser Computer machen: Nehmen wir an, er würde zählen. Null bedeutet, die erste Lampe ist aus; Eins bedeutet, die erste Lampe ist an; jetzt kann er nicht mehr weiter und braucht eine zweite Stelle, also eine zweite Lampe. Zwei bedeutet also, die neue linke Lampe ist an – und die erste Lampe muss wieder bei Null anfangen, genau wie der Kilometerzähler; die erste Lampe muss also wieder ausgehen, wenn die zweite angeht.

Die folgende kleine Tabelle zeigt dir, wie unser Additionscomputer die Ergebnisse von 0 bis 2 mit zwei Lampen darstellen soll:







Schalter 1	Schalter 2	Additions ergebnis	Zweite Lampe	Erste Lampe
Null	Null	0		
Null	Eins	1		
Eins	Null			
Eins	Eins	2		

Tabelle 1: Mit zwei Lampen Ergebnisse von 0 bis 2 ausgeben

Wenn die Schalter so stehen wie in den ersten beiden Spalten gesagt, dann soll das Ergebnis der Addition (mittlere Spalte) mit den zwei Lampen so dargestellt werden, wie in den rechten beiden Spalten gezeigt.

Auf diese Tabelle werden wir später noch öfter zurück kommen, vielleicht solltest du dir ein Lesezeichen in diese Seite legen!

Wie bauen wir nun einen Computer, der genau das macht?

Fangen wir mit der ersten Lampe an. Die haben wir zwar schon in Abbildung 3 und Abbildung 4 fertig gehabt, aber wegen des falschen Ergebnisses bei unse-

rer 1 + 1 Aufgabe brauchen wir wohl eine andere Schaltung: Die Lampe soll aus bleiben, wenn beide Schalter gleich stehen (00 oder 11), und sie soll leuchten, wenn sie unterschiedlich stehen (01 oder 10). Das geht mit der Schaltung wie sie in Abbildung 5 gezeigt wird. Dabei bauen wir die Schalter etwas um: Sie schalten den Strom jetzt nicht ein oder aus, sondern sie schalten ihn von einem Kabel auf ein anderes *um*. Trotzdem haben sie nur zwei Schalterstellungen entsprechend Null und Eins: Hebel oben entspricht Null, Hebel unten entspricht Eins.

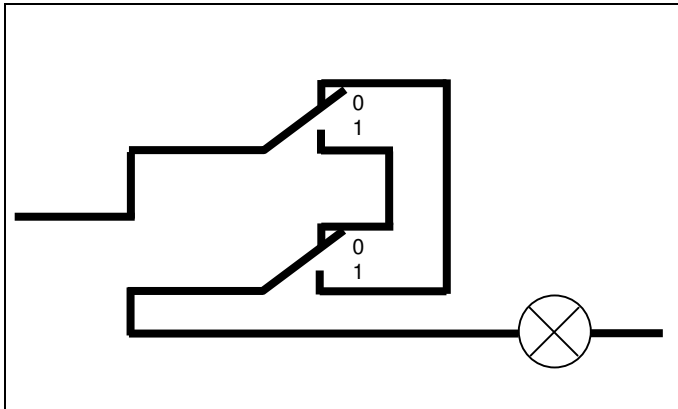


Abbildung 5: Kein Kabelweg geht bis zur Lampe durch. Sie leuchtet deshalb nicht.

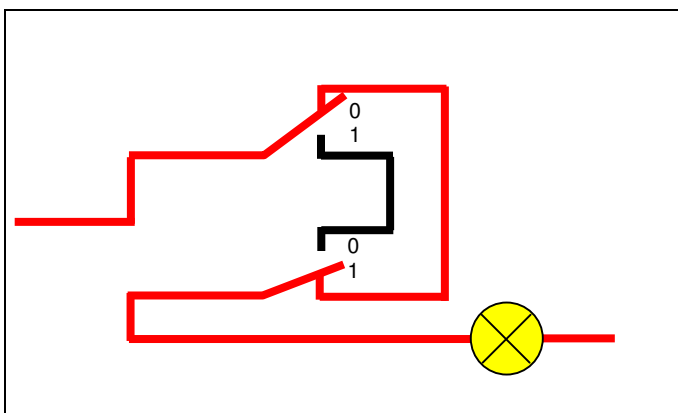


Abbildung 6: Die Lampe leuchtet, weil es einen durchgehenden Kabelweg gibt.

Abbildung 5 zeigt den Zustand, dass beide Schalter auf Null stehen. Du wirst kein Kabel finden, das bis zur Lampe durchgeht. Sie leuchtet also nicht und bedeutet damit „Null“. Vergewissern wir uns mit Tabelle 1: Wenn beide Schalter auf Null stehen, soll die rechte Lampe nicht leuchten, für diesen Fall ist unsere Schaltung also richtig!

Was ist nun, wenn ein Schalter auf Eins steht?

Dieser Zustand wird in Abbildung 6 gezeigt: Der untere Schalter steht auf „Eins“ und dadurch gibt es einen durchgehenden Kabelweg zur Lampe und sie leuchtet. Du kannst dir an Hand des Bildes schnell überlegen, dass die Lampe auch dann leuchtet, wenn der obere Schalter auf „Eins“

und der untere auf „Null“ steht. (Falls du Schwierigkeiten hast, male dir die Schaltung selbst noch einmal auf!)

Wenn beide Schalter auf „Eins“ stehen, gibt es wieder keinen durchgehenden Kabelweg, so dass die Lampe dann nicht leuchtet. Wenn wir noch mal in Tabelle 1 schauen und kontrollieren, stellen wir fest, dass unsere Lampe Nummer eins nun genau das tut, was sie soll: Sie leuchtet immer dann, wenn *ent-*

weder der eine *oder* der andere Schalter auf „Eins“ steht und Sie leuchtet nicht, wenn beide Schalter gleich stehen, also beide auf „Null“ oder beide auf „Eins“.

Eine solche Schaltung findest du bestimmt auch in eurer Wohnung! Meistens gibt es sie an Treppen oder in Räumen mit mehr als einer Tür (meist ein Flur). Man nennt sie auch „Wechselschaltung“.

Nun sind wir ein wenig vom Thema abgeschweift! Wir wollten ja einen Computer mit einer zweistelligen Anzeige bauen! Dafür fehlt uns noch die Schaltung der zweiten Lampe! Sehen wir mal in Tabelle 1 nach: Diese Lampe soll nur leuchten, wenn beide Schalter auf „Eins“ stehen. Das ist einfach:

Du kannst dir an Hand von Abbildung 7 leicht überlegen, dass es einen durchgehenden Kabelweg erst dann gibt, wenn *beide* Schalter geschlossen sind.

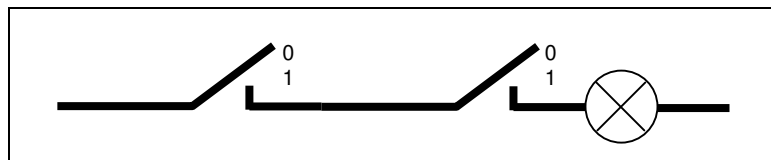


Abbildung 7: Die Lampe leuchtet nur dann, wenn beide Schalter auf „Eins“ stehen.

Die beiden Schalter in Abbildung 5 sind natürlich die selben wie in Abbildung 7, denn wir geben ja nur zwei Zahlen ein! Das bedeutet, dass die dargestellten Kabelwege gleichzeitig geschaltet werden. Solche Schalter gibt es auch, und in Abbildung 8 ist unser Computer noch einmal als Ganzes gezeichnet.

Die Schalter sind zur besseren Übersicht, was alles zu einem Schalter gehört, in dünnen Kästchen eingerahmt. In jedem Kästchen befinden sich zwei elektrische Schalter, die allerdings nur gleichzeitig betätigt werden können. Die hellblauen Bügel sollen das andeuten. Die Wechselschaltung für die erste Lampe (rechts) ist schwarz, die Verkabelung für die zweite Lampe (links) ist blau gezeichnet. In der abgebildeten Schalterstellung rechnet der Computer $0 + 0$ und das Ergebnis ist Null, also richtig.

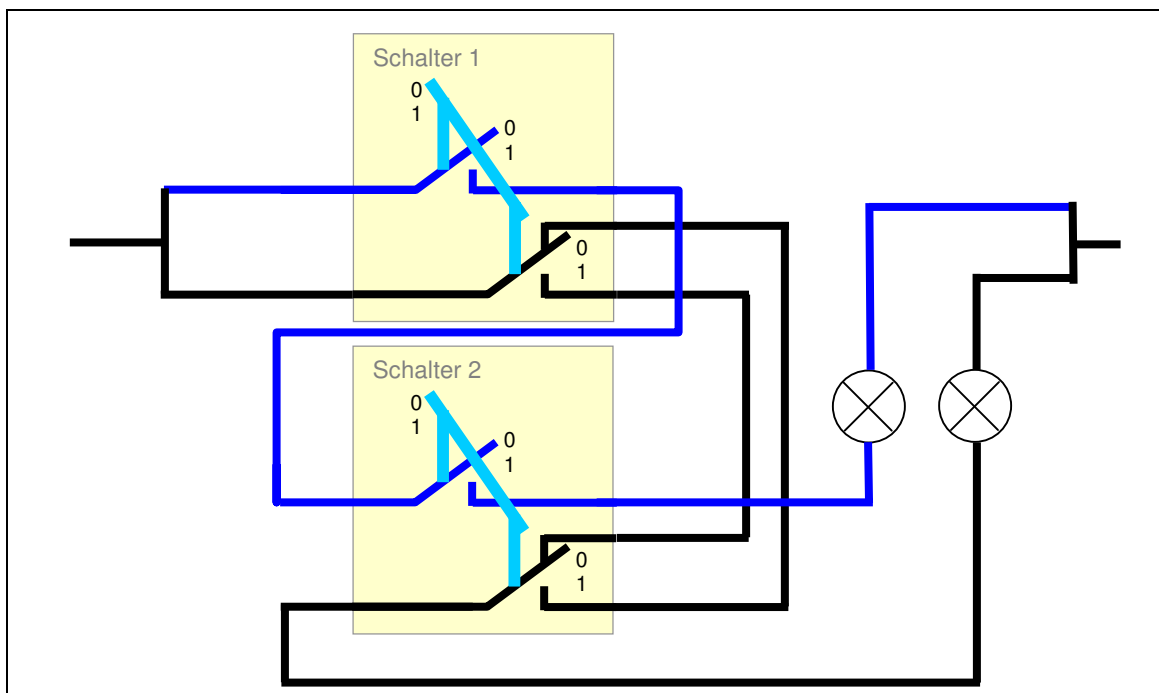


Abbildung 8: Wie die zwei Lampen mit den zwei Schaltern verbunden werden müssen, damit unser Computer richtig addiert

Dieser Computer kann nun wirklich die beiden Zahlen, die wir mit Hilfe der beiden Schalter darstellen, zusammenzählen! Je nachdem welche Zahlen wir an den beiden Schaltern einstellen, wird immer das richtige Ergebnis berechnet, wie wir es uns in Tabelle 1 überlegt hatten! Versuche an Hand von Abbildung 8 das zu überprüfen!

Bestimmt hast du immer noch das Gefühl, dass das noch kein richtiger Computer ist! Ganz Unrecht hast du damit natürlich nicht, denn ein Computer kann ja erstens mit viel mehr und viel größeren Zahlen rechnen und zweitens hast du ja im Kapitel 2 aufgepasst und gelernt, dass man einen Computer *programmieren* kann. Unser Computer rechnet aber immer nur zwei Zahlen zusammen und wir können ihm nicht sagen, dass er etwas anderes tun soll.

Jetzt bringen wir erst mal die Geschichte mit den größeren Zahlen zu Ende und später bauen wir dann auch noch einen Computer, der verschiedene Dinge tun kann!

3.2 Größere Zahlen

Denke noch einmal daran zurück, was wir gemacht haben, als wir die Zahl 2 mit unserer einen Lampe nicht darstellen konnten: Wir haben eine zweite Stelle eingeführt, die durch eine zweite Lampe dargestellt wurde. Können wir damit nur bis zwei zählen?

Wenn wir mit „richtigen“ Zahlen zählen, kommt nach der 9 die 10, das heißt die zweite Stelle wird eingeführt. Dann kommt die 11, dann die 12, 13, 14, 15, 16, 17, 18 und dann die 19. Wir erhöhen also wieder so lange unsere erste (rechte) Stelle, bis wir erneut keine Ziffern mehr haben, um weiter zu zählen. Was machen wir dann? Wir erhöhen unsere zweite Stelle und setzen die erste wieder auf Null und heraus kommt die 20.

Genau so können wir es mit unseren beiden Lampen auch machen! Wenn wir bis zwei gezählt haben, können wir immer noch die erste Lampe wieder einschalten, so dass dann beide brennen, und wir haben bis drei gezählt. Für unseren zuvor gebauten Computer brauchten wir das nur deshalb nicht, weil wir ihm keine Aufgabe mit dem Ergebnis „Drei“ stellen konnten.

Tabelle 2 zeigt dir die zwei Lampen wie sie bis 3 zählen. Die Spalte Bedeutung übersetzt dir die Spalte mit den Lampenbildern: Eine brennende Lampe bedeutet 1, und eine Lampe, die nicht brennt, bedeutet 0.

Können wir mit zwei Lampen noch weiter zählen? Nein! Es gibt keine An/Aus-Kombination mehr, die noch eine weitere Zahl darstellen könnte! Es gibt vier Kombinationen, weil wir zwei Lampen mit jeweils zwei Möglichkeiten haben.

Jeder dieser vier Kombinationen haben wir eine Zahl zugeordnet, nämlich die vier Zahlen von 0 bis 3. (Computerleute beginnen immer bei Null zu zählen!)









Zahl	Zweite Lampe	Erste Lampe	Bedeutung	
0			0	0
1			0	1
2			1	0
3			1	1

Tabelle 2: Zählen mit zwei Lampen

Zahl	Lampe3	Lampe2	Lampe1	Bedeutung		
0				0	0	0
1				0	0	1
2				0	1	0
3				0	1	1
4				1	0	0
5				1	0	1
6				1	1	0
7				1	1	1

Tabelle 3: Mit drei Lampen bis Sieben zählen

Du wirst dir schon denken, wie wir weiterzählen können: Wir führen eine dritte Stelle, oder – bei unserem Computer – eine dritte Lampe ein. Diese hat wieder zwei Möglichkeiten: Sie ist an oder aus. Bei jeder dieser beiden Möglichkeiten können wir die übrigen beiden Lampen in den vier Kombinationen aus Tabelle 2 leuchten lassen, so dass wir zwei mal vier gleich acht verschiedene Zahlen darstellen können. Wir können also mit drei Lampen von 0 bis 7 zählen! In Tabelle 3 steht, wie ein Computer mit drei Lampen zählen würde.

Nun kannst du dir vorstellen, dass wir zu den drei Lampen vorne auch noch Schalter dazubauen, mit deren Hilfe wir auch größere Zahlen *eingeben* können. Damit wir nicht wieder eine solche Überraschung erleben wie bei der Rechenaufgabe $1 + 1$, bei der das Ergebnis so groß war, dass unser Computer es nicht anzeigen konnte, könnten wir zwei Schalter für die erste einzugebende Zahl vorsehen und zwei Schalter für die zweite. Jedes Schalterpaar könnte dann Zahlen zwischen 0 und 3 darstellen (vgl. Tabelle 1), so dass das größtmögliche Ergebnis 6 wäre; das könnten wir mit unseren drei Lampen wie in Tabelle 3 gezeigt darstellen.

Wenn wir uns jetzt vorstellen, wie diese insgesamt vier Schalter und drei Lampen zu verschalten wären, sehen wir nur noch einen ziemlich Kabelsalat! Schon die Schaltung für zwei Lampen und zwei Schalter aus Abbildung 5 und Abbildung 7 war ja ziemlich unübersichtlich! Nun man kann das Ganze etwas geschickter lösen als wir es getan haben, und außerdem viel, viel kleiner; aber darauf komme ich später noch einmal zurück.

Für jetzt ist erst einmal klar, wie wir mit größeren Zahlen als Drei rechnen können. Auch Sieben ist noch keine so riesengroße Zahl, dass wir von unserem Computer beeindruckt sein könnten, aber wir haben das Prinzip verstanden: Wenn wir größere Zahlen darstellen wollen, nehmen wir einfach mehr Stellen! Nichts anderes machen wir mit unseren normalen Zahlen schließlich auch!

Mit jeder Stelle, die wir hinzufügen, verdoppeln wir die Anzahl der Zahlen, die wir darstellen können. Falls dir nicht klar ist warum, lies noch einmal auf Seite 21 nach und schaue dir Tabelle 3 noch einmal an.

Wenn du alles verstanden hast, kannst du sicherlich auch die Frage beantworten, wie viele verschiedene Zahlen man mit einer Hand darstellen kann!?

Fünf (1 bis 5)? Nein! Mehr!

Sechs (0 bis 5)? Schon besser! Aber es sind noch mehr! (Obwohl die meisten Menschen ihre Finger nur so benutzen.)

Denke daran, dass jeder deiner fünf Finger entweder ausgestreckt oder eingeklappt sein kann!

Man kann mit fünf Fingern 32 Zahlen darstellen! (In Worten: Zweiunddreißig, nämlich die Zahlen 0 bis 31)!

Okay, zugegeben: Es bedarf etwas Übung jeden einzelnen Finger unabhängig von den anderen zu bewegen, aber es geht!


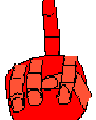



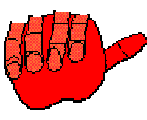


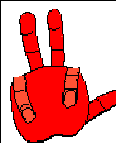


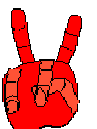





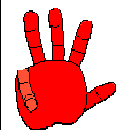

 0 00000	 4 00100	 8 01000	 12 01100	 16 10000
 1 00001	 5 00101	 9 01001	 13 01101	usw.
 2 00010	 6 00110	 10 01010	 14 01110	 30 11110
 3 00011	 7 00111	 11 01011	 15 01111	 31 11111

Tabelle 4: Mit den fünf Fingern einer Hand von 0 bis 31 zählen!

Die ersten Leute, die Computer gebaut haben, haben sich nicht mit drei Stellen („Bits“, siehe Seite 13) zufrieden gegeben und auch noch nicht mit fünf. Sie haben immer acht Bits verwendet.

Damit kann man $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 = 2^8 = 256$ Zahlen darstellen. (Genau so viele Seiten hat übrigens dieses Buch!) Das ist für heutige Verhältnisse auch noch nicht so furchtbar viel; heutige Rechner arbeiten mit 32 oder sogar 64 Bits gleichzeitig. Jedenfalls waren es zu Anfang acht Bits und deshalb hat man für solche Zahlen aus acht Bit ein eigenes Wort erfunden:

*Acht Bit heißen ein **Byte**. Das spricht man „Beit“ aus.*

Auch die frühen 8-Bit-Rechner konnten schon mit größeren Zahlen als 255 rechnen. Nur konnte eine solche Zahl dann eben nicht in einem Schritt verarbeitet werden, sondern sie musste in mehrere Teile zerlegt werden, mit denen nacheinander gerechnet wurde – wieder begegnet uns das Prinzip, schwierige Aufgaben in mehrere kleine zu zerlegen, um sie lösen zu können.

3.3 Bits und Bytes für Menschen – das 16er-System

Wir haben nun eine tolle Computer-Zahlensprache erfunden, oder? Wir haben einen Weg gefunden, unsere komplizierten Zahlen so einfach darzustellen, dass sogar eine dumme Maschine damit rechnen kann.

Leider können wir Menschen auch mit viel Übung nicht auf einen Blick erfassen, dass die Bitfolge 01100011 die Zahl 99 darstellen soll! Außerdem kommt man bei den vielen gleichen Ziffern, Nullen und Einsen, sehr schnell mit den Stellen durcheinander. Deshalb hat man eine Möglichkeit gesucht, acht Bit etwas übersichtlicher darzustellen.

Eine Möglichkeit ist natürlich, die Zahl zwischen Null und 255, die der Bitfolge entspricht, einfach als normale Zahl (man nennt sie auch *Dezimalzahl*) zu schreiben. Man hat aber etwas gesucht, mit dem man sehr schnell von der einen in die andere Darstellung wechseln kann. Wie immer wenn etwas schwierig ist, teilt man das Problem in kleinere Teile auf: Man teilt das Byte in zwei Hälften! Die vier Bits einer Hälfte stellt man mit einer einstelligen Zahl dar, so dass ein Byte als eine zweistellige Zahl dargestellt werden kann.

Ja, aber vier Bits können doch 16 verschiedene Zahlen darstellen!? Wie kann man das mit einer einstelligen Zahl darstellen? Nun man braucht nur 16 verschiedene Ziffern, also 16 verschiedene Zeichen, um eine Zahl zu schreiben (vgl. siehe Seite 12)!

Man hat sich darauf geeinigt, die zehn normalen Zahlen-Ziffern zu benutzen und zusätzlich die sechs ersten Buchstaben des Alphabets! Tabelle 5 zeigt die Bedeutung der Ziffern des 16er-Systems:

10er-System	16er-System	Nullen u. Einsen
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111

10er-System	16er-System	Nullen u. Einsen
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Tabelle 5: Ziffern des 16er-Systems

Wenn man also beispielsweise die Bitfolge 10101100 im 16er-System schreiben will, schreibt man für die vordere Hälfte dieses Bytes (1010) ein A und für die hintere Hälfte (1100) ein C, also „AC“.

Das Byte 11111111, das unserer normalen Zahl 255 entspricht, schreibt sich im 16er-System „FF“.

Ein kleines Problem ergibt sich, wenn wir zum Beispiel 10010111 im 16er-System schreiben wollen: Für die erste Hälfte 1001 schreiben wir 9, für die zweite Hälfte 0111 schreiben wir 7, zusammen also 97. Das sieht genauso aus wie unsere normale Zahl Siebenundneunzig, *bedeutet aber etwas völlig anderes, nämlich 151!*

Man kann deshalb durchaus darüber streiten, ob es so sinnvoll war, für das 16er-System die selben Ziffern zu verwenden wie für das Zehnersystem. Man behilft sich, indem man Zahlen im 16er-System besonders kennzeichnet, meist mit einem vorangestellten „0x“ oder einem angehängten „H“.

10010111 schreibt man also 0x97 oder 97H.

*Man nennt das Zahlensystem mit 16 Ziffern auch das **Hexadezimalsystem**.*

*Die Hälfte eines Bytes nennt man manchmal auch **Nibble**.*

*Unser normales Zahlensystem mit zehn Ziffern nennt man das **Dezimalsystem**.*

*Das Zweiersystem, mit dem wir unseren Computer gebaut haben, und das nur die zwei Ziffern Null und Eins kennt, nennt man auch **Binärsystem**. Manchmal liest man dafür auch den Begriff **Dualsystem**.*

*Alle diese Zahlensysteme sind sogenannte **Stellenwertsysteme**. Man nennt sie so, weil jede Stelle einen bestimmten Wert hat: Im Zehnersystem 1, 10, 100, usw., im Zweiersystem 1, 2, 4, 8, usw.*

Im Gegensatz dazu ist z.B. das System der römischen Zahlen kein Stellenwertsystem!

Nach diesem ziemlich abgehobenen Ausflug in die Welt der Zahlensysteme sollten wir uns aber wieder praktischeren Dingen zuwenden!

3.4 Texte mit Nullen und Einsen

In der Einführung stand so eine komische Andeutung, dass all die tollen Sachen, die du vielleicht schon mit deinem Computer gemacht hast, wie Bilder malen, Texte schreiben, usw. für den Computer auch nichts anderes ist als rechnen mit Nullen und Einsen.

In den vorangegangenen Kapiteln haben wir ja schon gesehen, wie der Computer Zahlen mit Nullen und Einsen darstellt (vgl. Tabelle 1 bis Tabelle 5). Einer bestimmten Bitfolge wird eine bestimmte Zahl zugeordnet – sinnvoller Weise natürlich so, dass es auch mathematisch Sinn macht, nämlich auf die selbe Weise, wie wir auch im Dezimalsystem zählen.

Es hindert uns doch niemand daran, einer Bitfolge statt einer Zahl einen Buchstaben zuzuordnen, oder? Warum soll 01000001 nicht „A“ bedeuten statt „65“?

Betrachten wir noch einmal die Addition (das Plus-Rechnen), die unser Super-Einfach-Computer im Abschnitt 3.1 durchgeführt hat: Das Ergebnis, also welche Lampe wann geleuchtet hat, hing doch nur davon ab, wie die Schalter gestellt waren, also welche Eingabe wir gemacht haben, und davon, wie unser Computer verdrahtet – man könnte auch sagen programmiert – war. Für die Schaltung war es völlig egal, welche Bedeutung wir den Schalterstellungen oder dem Leuchten der Lampe(n) zugeordnet haben; sie hat den Strom so geführt, wie eben die Verdrahtung war. Nur *wir* waren es, die die Verdrahtung *genau so gemacht* haben, dass die Bedeutung unseres Ergebnisses zur Bedeutung un-

serer Eingabe gepasst hat! Das hört sich jetzt an wie geschummelt, ist es aber nicht! Genau so arbeitet auch ein echter Computer!

Wir können uns doch jetzt eine einfache Aufgabe für die Bearbeitung von Texten einfallen lassen, zum Beispiel könnten wir aus einem großen Buchstaben einen kleinen Buchstaben machen wollen und umgekehrt. Damit unser Computer das kann, müssen wir nur festlegen, welches Bitmuster welchem Buchstaben entspricht, und müssen uns danach eine Verdrahtung überlegen, die bei jedem Großbuchstaben-Bitmuster an der Eingabeseite (Schalter) das zugehörige Bitmuster des Kleinbuchstabens an der Ausgabeseite (Lampen) erzeugt, bzw. umgekehrt.

Das ist einfacher als es sich anhört!

Fangen wir mit der Festlegung der Buchstaben an: Wenn wir bei den acht Bit bleiben, auf die sich die Erfinder des Computers irgendwann geeinigt haben, dann können wir 256 verschiedene Zeichen darstellen. Das Alphabet hat 26 Buchstaben; die gibt es als große und kleine Buchstaben, das macht zusammen schon mal 52 Stück. Dann kommen noch die Umlaute ä, ö und ü in groß und klein sowie das „ß“ hinzu. Vermutlich werden wir auch mal ganze Sätze mit unserem Computer schreiben wollen; dafür brauchen wir dann auch noch Satzzeichen wie Punkt, Komma, Ausrufungszeichen, usw. und bestimmt finden wir es irgendwann toll, auch so komische Zeichen wie %, \$, & benutzen zu können. Außerdem gibt es mit Sicherheit auch noch Computerfreunde in anderen Ländern, die noch weitere besondere Zeichen brauchen. Aber für den Anfang werden uns 256 Zeichen erst mal genügen!

Es ist sinnvoll, dass alle Computer die selbe Zuordnung von Buchstaben zu Bitfolgen verwenden, denn sonst hätte ein Text, den man auf dem einen Computer geschrieben hat, auf einem anderen Computer eine ganz andere Bedeutung und dieser Satz sähe vielleicht so aus:

Kl43jsdc l rlk 2ehr Es´fhg e4\$“ öoiyh rfgheISöfdh verjhgo i reobjGRnSh oer ergPner oer rönkFGbjs.renor ihg9h 25QQnön548p 5gi uARsfd

Damit das nicht passiert, hat man sich auf die gemeinsame Festlegung in Tabelle 6 geeinigt, die man ASCII genannt hat.

ASCII ist dabei eine Abkürzung und bedeutet „**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange“. Das ist Englisch und bedeutet auf Deutsch: Amerikanische Standard Code für den Informationsaustausch. Obwohl in dem Namen „Amerikanisch“ steht, benutzt man das auf der ganzen Welt.

Ein Code ist eine Zuordnung von Zeichen zu anderen Zeichen oder zu einer bestimmten Bedeutung.

00100000	32	01000001	65	A	01100001	97	a	10000101	133	...	10101011	171	«	11001101	205	í	11101111	239	ï
00100001	33	01000010	66	B	01100010	98	b	10000110	134	†	10101100	172	¬	11001110	206	î	11110000	240	ð
00100010	34	01000011	67	C	01100011	99	c	10000111	135	‡	10101101	173	‡	11001111	207	ï	11110001	241	ñ
00100011	35	01000100	68	D	01100100	100	d	10001000	136	^	10101110	174	®	11010000	208	Ð	11110010	242	ò
00100100	36	01000101	69	E	01100101	101	e	10001001	137	%	10101111	175	˘	11010001	209	Ñ	11110011	243	ó
00100101	37	01000110	70	F	01100110	102	f	10001010	138	Š	10110000	176	°	11010010	210	Ò	11110100	244	ô
00100110	38	01000111	71	G	01100111	103	g	10001011	139	<	10110001	177	±	11010011	211	Ó	11110101	245	õ
00100111	39	01001000	72	H	01101000	104	h	10001100	140	€	10110010	178	²	11010100	212	Ô	11110110	246	ö
00101000	40	01001001	73	I	01101001	105	i	10001101	141	□	10110011	179	³	11010101	213	Õ	11110111	247	÷
00101001	41	01001010	74	J	01101010	106	j	10001110	142	Ž	10110100	180	´	11010110	214	Ö	11111000	248	ø
00101010	42	01001011	75	K	01101011	107	k	10010001	145	‘	10110101	181	μ	11010111	215	×	11111001	249	ù
00101011	43	01001100	76	L	01101100	108	l	10010010	146	’	10110110	182	¶	11011000	216	Ø	11111010	250	ú
00101100	44	01001101	77	M	01101101	109	m	10010011	147	“	10110111	183	·	11011001	217	Ù	11111011	251	û
00101101	45	01001110	78	N	01101110	110	n	10010100	148	”	10111000	184	¸	11011010	218	Ú	11111100	252	ü
00101110	46	01001111	79	O	01101111	111	o	10010101	149	•	10111001	185	¹	11011011	219	Û	11111101	253	ý
00101111	47	01010000	80	P	01110000	112	p	10010110	150	–	10111010	186	º	11011100	220	Ü	11111110	254	þ
00110000	48	01010001	81	Q	01110001	113	q	10010111	151	—	10111011	187	»	11011101	221	Ý	11111111	255	ÿ
00110001	49	01010010	82	R	01110010	114	r	10011000	152	˘	10111100	188	¼	11011110	222	Þ			
00110010	50	01010011	83	S	01110011	115	s	10011001	153	™	10111101	189	½	11011111	223	ß			
00110011	51	01010100	84	T	01110100	116	t	10011010	154	š	10111110	190	¾	11100000	224	à			
00110100	52	01010101	85	U	01110101	117	u	10011011	155	›	10111111	191	¿	11100001	225	á			
00110101	53	01010110	86	V	01110110	118	v	10011100	156	œ	11000000	192	À	11100010	226	â			
00110110	54	01010111	87	W	01110111	119	w	10011101	158	ž	11000001	193	Á	11100011	227	ã			
00110111	55	01011000	88	X	01111000	120	x	10011111	159	ÿ	11000010	194	Â	11100100	228	ä			
00111000	56	01011001	89	Y	01111001	121	y	10100001	161	ı	11000011	195	Ã	11100101	229	å			
00111001	57	01011010	90	Z	01111010	122	z	10100010	162	ç	11000100	196	Ä	11100110	230	æ			
00111010	58	01011011	91	[01111011	123	{	10100011	163	£	11000101	197	Å	11100111	231	ç			
00111011	59	01011100	92	\	01111100	124		10100100	164	¤	11000110	198	Æ	11101000	232	è			
00111100	60	01011101	93]	01111101	125	}	10100101	165	¥	11000111	199	Ç	11101001	233	é			
00111101	61	01011110	94	^	01111110	126	~	10100110	166	ı	11001000	200	È	11101010	234	ê			
00111110	62	01011111	95	10000000	128	€	10100111	167	§	11001001	201	É	11101011	235	ë				
00111111	63	01100000	96	10000010	130	,	10101000	168	”	11001010	202	Ê	11101100	236	ì				
01000000	64	@		10000011	131	f	10101001	169	©	11001011	203	Ë	11101101	237	í				
				10000100	132	„	10101010	170	ª	11001100	204	Ì	11101110	238	î				

Tabelle 6: ASCII-Code

Lass dich von der Tabelle nicht verwirren! Jeweils drei Spalten, die durch einen dickeren Strich getrennt sind, gehören zusammen: Links steht das Bitmuster aus acht Bit, dann kommt die Dezimalzahl, der dieses Muster entspricht, und in der dritten Spalte steht der Buchstabe oder das Zeichen, das man dem Bitmuster zugeordnet hat.

Unsere Buchstaben A bis Z bzw. a bis z findest du in dem blau hinterlegten Bereich.

Wenn du aufmerksam hinsiehst, wirst du merken, dass gar nicht alle 256 möglichen Bitmuster in der Tabelle gezeigt sind (z.B. 0 bis 31 nicht). Das liegt daran, dass diese mit etwas belegt sind, das man gar nicht drucken kann, zum Beispiel Zeichen, die einem Drucker sagen, dass er ein neues Blatt nehmen oder eine neue Zeile beginnen soll. So etwas nennt man *Steuerzeichen*.

Noch etwas komisches fällt dir vielleicht an der Tabelle auf: Sie enthält auch Bitfolgen, denen die Ziffern 0 bis 9 zugeordnet sind. Sie stehen in dem mit gelb hinterlegten Bereich. Wofür brauchen wir das denn? Wir haben doch im Kapitel 3.2 schon Zahlen durch Nullen und Einsen dargestellt! Warum machen wir das hier noch einmal, und noch dazu mit einer anderen Zuordnung als wir sie vorher hatten? Die „9“ ist in der ASCII-Tabelle „00111001“ und nicht „00001001“, wie wir sie beim Rechnen geschrieben hätten! Nun die Antwort ist die: Auch beim Schreiben eines Textes wollen wir ja unsere Ziffern schreiben können; allein in diesem Absatz, den du gerade gelesen hast, kommen die Ziffern 0, 1, 2 und 9 vor, aber rechnen im eigentlichen Sinne wollen wir damit ja gar nicht. Immerhin waren die Leute, die sich die ASCII-Tabelle ausgedacht haben, so schlau, dass die rechte Hälfte (das rechte Nibble) der Bytes, mit denen Ziffern dargestellt werden, identisch ist mit der mathematischen Codierung der Zahlen: Die rechten vier Bits des ASCII-Codes für die 9 heißen 1001 und das entspricht genau dem Muster, das entsteht, wenn wir mit Nullen und Einsen bis Neun zählen. Hexadezimal geschrieben (vgl. Kapitel 3.3!) ist der ASCII-Code für die Ziffern 0 bis 9: 0x30, 0x31, 0x32, ..., 0x39; die rechte Stelle der Hexadezimalzahl entspricht also immer der dargestellten Ziffer.

Zurück zu unserer Aufgabe: Unser Computer soll das Bitmuster eines Großbuchstabens in das Bitmuster eines Kleinbuchstabens umrechnen. Auch hier lohnt es sich genauer hinzusehen: Groß- und Kleinbuchstaben sind in ihrem Bitmuster identisch, außer an der sechsten Stelle von rechts! Großbuchstaben haben dort eine Null, Kleinbuchstaben eine Eins stehen.

Der Computer, der unsere Aufgabe bewältigen kann, muss also nur das Bitmuster von unseren acht Eingabeschaltern auf die Ausgabelampen übertragen. Lampe Nummer Sechs muss dabei leuchten, wenn auf der Eingabeseite eine 0 steht und ausgeschaltet sein, wenn dort eine 1 steht.

Abbildung 9 zeigt einen Computer der das kann. Dargestellt ist die Umrechnung von „M“ in „m“. (Suche das Bitmuster für m und für M in Tabelle 6!)

Bit Nr. 8 ist ganz oben, Bit Nr. 1 ganz unten gezeichnet:

Beachte: Der Schalter für Bit Nr. 6 (Pfeil) steht in Stellung „Null“ also oben. Trotzdem leitet er den Strom. Würden wir von „m“ in „M“ umrechnen, müssten wir diesen Schalter nach unten in Stellung „Eins“ drücken; in dieser Stellung wäre das Kabel dann unterbrochen, so dass auf der Ausgabeseite eine Null erscheint. Man sagt das Bit wird *invertiert*, also umgedreht: Am Ausgang (Lampe) steht immer das Gegenteil vom Eingang (Schalter).

Eine zweite Besonderheit: Eigentlich bräuhete unser Computer für diese Aufgabe nur sechs Schalter und sechs Lampen, weil die Bits Nummer 7 und Nummer 8 sowohl bei Groß-, als auch bei Kleinbuchstaben immer auf Null bzw. Eins stehen (vgl. Tabelle 6). Wenn du unseren Computer wirklich benutzen und mit den Schaltern Buchstabencodes eingeben würdest, könntest du feststellen, dass du diese beiden Schalter nie verstellen müsstest.

Und in der Tat: Alte Fernschreiber, das sind Geräte die Texte über eine Telefonleitung übertragen haben, haben nur mit fünf Bit gearbeitet: Sie haben nicht zwischen Groß- und Kleinschreibung unterschieden und konnten mit fünf Bit 32 verschiedene Zeichen darstellen. Heute braucht man diese Geräte nicht mehr; das Telefax hat sie völlig verdrängt!

Unser Computer zur Umrechnung von Groß- auf Kleinschreibung und umgekehrt funktioniert jedenfalls!

Das war jetzt ziemlich viel Theorie und wahrscheinlich glaubst du noch gar nicht, dass so etwas in einem *echten* Computer auch so geht. Wenn du einen Computer hast – für dieses Beispiel brauchst du einen PC – kannst du es gleich ausprobieren! Falls dir die Unterscheidung zwischen PC und Computer nicht klar ist, ist das erst mal nicht schlimm, das kommt später. Probiere einfach folgendes aus; lass dir von jemandem helfen, wenn dir die Anleitung nicht klar ist,

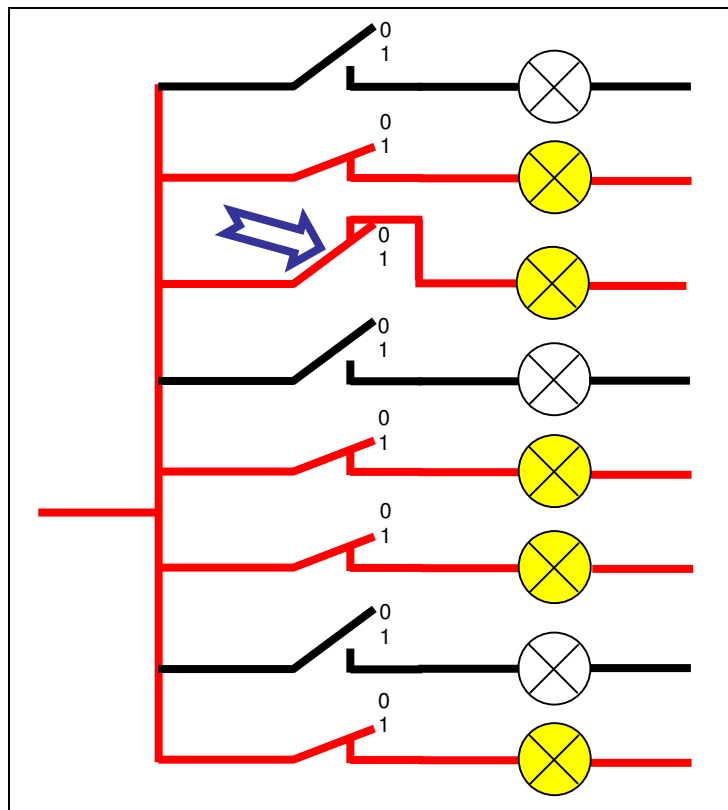


Abbildung 9: Computer, der von Groß- auf Kleinbuchstaben umrechnet und umgekehrt

oder lies erst weiter; alles was dir fremd vorkommt und du nicht verstehst, wird später noch erklärt!

Experiment 1: ASCII-Codes sind im PC versteckt!

Wo welche Taste sitzt, wird in Abbildung 10 gezeigt!

1. Starte deinen Computer.
2. Starte ein beliebiges Programm, bei dem man mit der Tastatur etwas eingeben kann, z.B. Word, Excel, Notepad, etc.
3. Das linke der drei Lämpchen über dem Ziffernblock muss leuchten; dann kannst du den Ziffernblock zum Zahlen tippen benutzen. Falls das noch nicht der Fall ist, betätige die NumLock-Taste!
4. Drücke die Alt-Taste und halte sie fest.
5. Tippe jetzt einen dreistelligen ASCII-Cade aus Tabelle 6 auf dem Ziffernblock. Für ein „M“ tippst du also beispielsweise „077“, für eine geschweifte Klammer „{“ tippst du „123“ oder für das at-Zeichen „@“ tippst du „064“.
6. Lasse die Alt-Taste wieder los! Das Zeichen erscheint!



Abbildung 10: PC-Tastatur

Wenn du bei diesem Versuch ASCII-Codes verwendest, die größer sind als 127, kann es vorkommen, dass du andere Zeichen angezeigt bekommst als in Tabelle 6 gezeigt sind. Das liegt daran, dass von dieser oberen Hälfte der Tabelle (Code Nummer 128 bis 255) verschiedene Versionen gibt, weil man

mit den 255 möglichen Zeichen doch nicht ausgekommen ist, und man diesen Teil der Zuordnungen wechseln kann.

Was zeigt uns dieser Versuch? Nun: Der Computer interpretiert deine Zahleneingabe entsprechend der in Tabelle 6 gezeigten Zuordnung. Man hat sie ihm also einprogrammiert. Früher hat man dieses Verfahren mit der Alt-Taste gebraucht, um dem Computer Zeichen einzugeben, die nicht auf der Tastatur stehen. Heute bietet fast jedes Programm eine bequemere Möglichkeit, solche Sonderzeichen einzugeben.

Aber immerhin bist du jetzt sicher, dass dein Computer unsere Zuordnung der Buchstaben und Zeichen zu den Nullen und Einsen richtig versteht!

Noch ein kleiner Hinweis: Obwohl du mit dem Ziffernblock die selben Zahlen eingibst wie mit den anderen Zahlentasten (vergleiche Abbildung 42 auf Seite 93) unterscheidet dein Computer diese Tasten! Das kannst du daran erkennen, dass das Experiment mit einer Tastatur ohne Ziffernblock (zum Beispiel auf einem Laptop) nicht funktioniert!

3.5 Bilder mit Nullen und Einsen

Jetzt können wir schon Zahlen und Texte mit Hilfe von Nullen und Einsen darstellen! Aber was machen wir mit Bildern? Du hast schon mal Bilder auf einem Computerbildschirm gesehen, also weißt du, dass es auch dafür eine Lösung gibt. Aber welche?

Können wir dem Computer wieder eine Zuordnung geben, also zum Beispiel „01101001“ entspricht einem Baum auf unserem Bild? Wenn du Computer wärst, würdest du sofort fragen:

„Eine Tanne oder eine Eiche? Im Sommer oder im Winter? An welcher Stelle des Bildes?“

Du merkst, so klappt das noch nicht ganz! Aber die Idee, einem Byte wieder eine Bedeutung zuzuordnen, ist schon mal nicht schlecht, denn sie hat ja bereits zweimal funktioniert. Nur die Bedeutung „Baum“ ist etwas ungeschickt gewählt! Wir müssen also wieder einfacher werden. Wir haben ja nur die 256 Möglichkeiten eines Bytes zur Verfügung, um unser Bild oder Teile davon zu beschreiben, und es gibt mit Sicherheit viel, viel mehr Möglichkeiten, einen Baum zu malen, ganz zu schweigen von Häusern, Autos, Menschen und Tieren. Der Teil des Bildes, den wir mit einem Byte beschreiben können, muss also viel kleiner sein als ein Baum. Und der Teil muss sich dazu eignen, aus mehre-

ren davon sowohl einen Baum, als auch ein Auto, als auch alles mögliche andere zu malen.

Kennst du den Maler *Vincent van Gogh*? Abbildung 11 zeigt dir ein Bild von ihm, auf dem er sich selbst gemalt hat.

Fällt dir daran etwas auf? Das Bild besteht aus ganz vielen kleinen Farbklecksen! Wenn du einen einzelnen Klecks davon siehst, kannst du nicht erkennen, ob er zu einem Baum, einem Menschen oder einem Auto gehört. Erst alle Kleckse zusammen ergeben das vollständige Bild. Das ist die Lösung unseres Problems! Der gute Vincent ist schon seit über hundert Jahren tot († 1890), und er hätte sich bestimmt nicht träumen lassen, dass es mal Maschinen geben würde, die Bilder nach dem selben Prinzip erzeugen wie er. In der Kunst nennt man diese Stilrichtung übrigens *Impressionismus*.

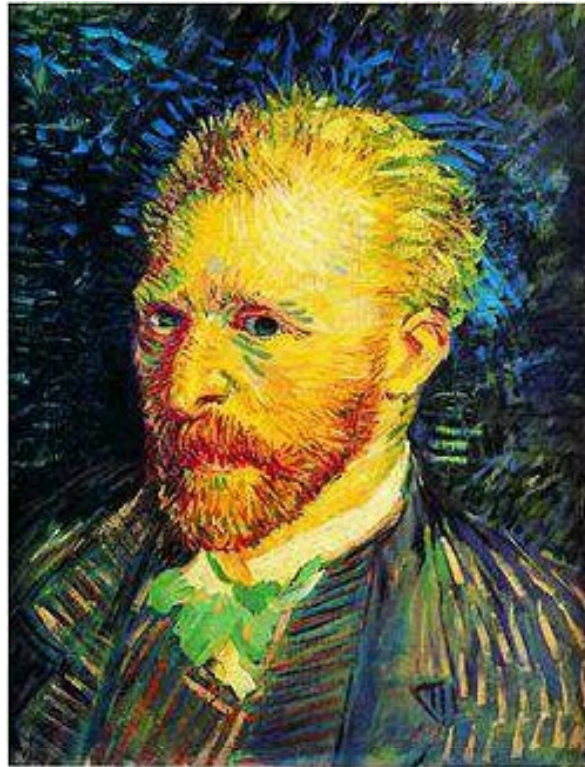


Abbildung 11: Selbstbildnis von Vincent van Gogh

Da unser Computer kein Künstler, sondern – wie wir schon mehrfach gesehen haben – ziemlich dumm ist, müssen wir das Ganze noch ein wenig weiter vereinfachen, indem wir die Kleckse zur Darstellung eines Bildes erstens alle gleich groß machen und ihnen zweitens allen die gleiche Form geben. Was unterscheidet die Kleckse dann noch voneinander? Ach ja richtig: Die Farbe! Mit Hilfe unserer 256 Möglichkeiten die Bits eines Bytes zu schreiben, können wir nun die Farbe codieren.

Natürlich unterscheiden sich die Kleckse auch durch die Stelle, an der sie im Bild sitzen. Müssen wir die auch codieren? Wir beschreiben ja auch nicht, wo das Byte 01000001 hingehört, das diesen Buchstaben **A** beschreibt. Das ergibt sich einfach aus der Reihenfolge der Bytes! Bei einem Bild ist es ganz genauso: Wir legen nur fest, wie viele Farbkleckse – vornehmer sagt man auch *Bildpunkte* oder *Pixel* – wir nebeneinander und wie viele untereinander brauchen, um ein Bild darzustellen, und wenn wir dann beispielsweise oben links in der Bildecke anfangen, und Zeile für Zeile jeden Farbkleck – pardon: Pixel – mit seiner Far-

be beschreiben, haben wir unser Bild komplett mit Nullen und Einsen nachgebildet.

Das folgende ganz einfache Bild in Abbildung 12 soll dir das Prinzip verdeutlichen:

Du erkennst einen Tannenbaum, eine Blume und die Sonne mit ein paar Sonnenstrahlen in diesem Bild. Es handelt sich um ein sehr grob gerastertes Bild: Es besteht nur aus $16 \times 16 = 256$ Bildpunkten, aber trotzdem kann man schon etwas erkennen!

Insgesamt habe ich fünf verschiedene Farben in dem Bild verwendet. Fünf? Nein sechs! Denn auch die Hintergrundfarbe weiß ist eine Farbe, die ich beschreiben muss! Ganz ähnlich wie wir in Tabelle 6 jedem Buchstaben eine Bitfolge zugeordnet haben, müssen wir es jetzt auch für die Farben tun!

Tabelle 7 enthält eine willkürliche Zuordnung, die ich mir dafür überlegt habe.

Eine solche Beschreibung aller verwendeten Farben nennt man auch eine *Palette*.

	00000000	000
	00000001	001
	00000010	010
	00000011	011
	00000100	100
	00000101	101

Tabelle 7: Farbcodierung

Da ich nur so wenig Farben verwendet habe, brauche ich nicht alle 8 Bit; die rechte Spalte zeigt daher nur die rechten drei Bits, auf die ich mich im Folgenden beschränken will, um mir Schreibarbeit zu sparen! Wenn wir jetzt das Bild mit Hilfe von Nullen und Einsen beschreiben wollen, tun wir das einfach, indem wir die Bitcodes für jede Farbe benutzen. Ich schreibe die Codes für jedes Pixel zeilenweise hin und beginne oben links. Kontrolliere es!

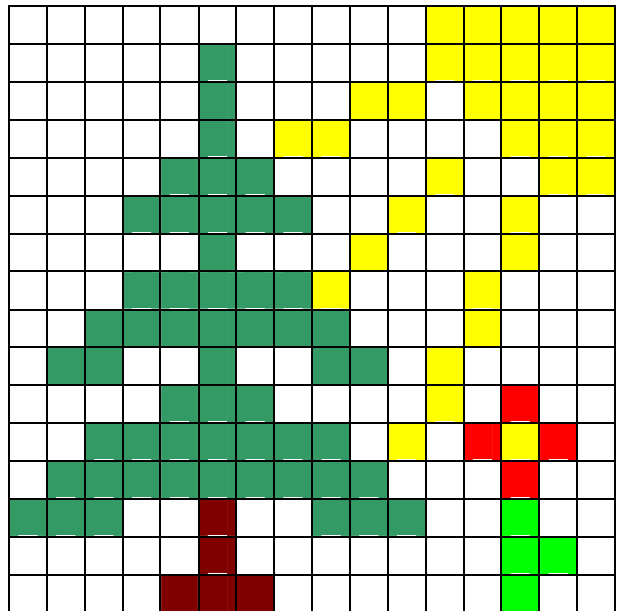


Abbildung 12: Bild mit 256 Pixeln

```
000 000 000 000 000 000 000 000 000 000 000 001 001 001 001 001
000 000 000 000 000 011 000 000 000 000 000 001 001 001 001 001
000 000 000 000 000 011 000 000 000 001 001 000 001 001 001 001
000 000 000 000 000 011 000 001 001 000 000 000 000 001 001 001
000 000 000 000 011 011 011 000 000 000 000 001 000 000 001 001
000 000 000 011 011 011 011 011 000 000 001 000 000 001 000 000
000 000 000 000 000 011 000 000 000 001 000 000 000 001 000 000
000 000 000 011 011 011 011 011 001 000 000 000 001 000 000 000
000 000 011 011 011 011 011 011 011 000 000 000 001 000 000 000
000 011 011 000 000 011 000 000 011 011 000 001 000 000 000 000
000 000 000 000 011 011 011 000 000 000 000 001 000 100 000 000
000 000 011 011 011 011 011 011 011 000 001 000 100 001 100 000
000 011 011 011 011 011 011 011 011 011 000 000 000 100 000 000
011 011 011 000 000 101 000 000 011 011 011 000 000 010 000 000
000 000 000 000 000 101 000 000 000 000 000 000 000 010 010 000
000 000 000 000 101 101 101 000 000 000 000 000 000 010 000 000
```

Ein Computer schreibt natürlich keine Zwischenräume und macht auch keine Zeilenumbrüche; die habe ich nur hinzugefügt, damit du die Darstellung leichter kontrollieren kannst! Die Breite und die Höhe des Bildes sowie die Farbcodierung muss man auch immer mit dazu schreiben oder dem Computer auf irgend eine Weise zugänglich machen, wenn man das Bild aus den vielen Nullen und Einsen richtig wieder herstellen will. Wie man dem Computer eine Farbe mitteilt, steht in Kapitel 5.3, in dem es um den Monitor geht.

Wir haben also jetzt eine Möglichkeit gefunden, auch Bilder nur mit Hilfe von Nullen und Einsen darzustellen.

*Die Darstellung eines Bildes auf die oben beschriebene Weise nennt man eine **Bitmap**. Das ist Englisch und bedeutet so viel wie eine Landkarte von Bits.*

Dir ist sicherlich aufgefallen, dass wir selbst für ein ganz einfaches Bild ziemlich viele Nullen und Einsen gebraucht haben!

Das Bild aus Abbildung 12 ist mit richtigen Pixeln gezeichnet so klein: 🌲
Daran erkennst du wie viele Bytes man für ein größeres Bild braucht!

Das ist ein grundsätzliches Problem bei Bildern, dass sie viel Platz für ihre Darstellung brauchen. Noch schlimmer ist es bei Filmen, bei denen man ca. 25 Bilder für eine Sekunde Film braucht, wenn er flüssig dargestellt werden soll! Man hat deshalb andere Verfahren entwickelt, wie man die Bilder (und im Übrigen auch alle anderen binären Daten, also alles was mit Nullen und Einsen darge-

stellt werden kann) Platz sparer darstellen kann – aber natürlich immer noch ausschließlich durch Nullen und Einsen!

Um dir zu zeigen, dass auch unser van-Gogh-Bild am Bildschirm aus Punkten zusammengesetzt wird, habe ich Vincents rechtes Auge einmal stark vergrößert. Abbildung 13 zeigt links das Bild, das du schon kennst, mit der Markierung der Stelle, die rechts vergrößert dargestellt wird.

Man kann gut erkennen, dass die Bildpunkte noch viel kleiner sind als die Farbkleckse, die van Gogh selbst verwendet hat. Die Größe eines einzelnen Pixels ist in der Vergrößerung rot markiert.

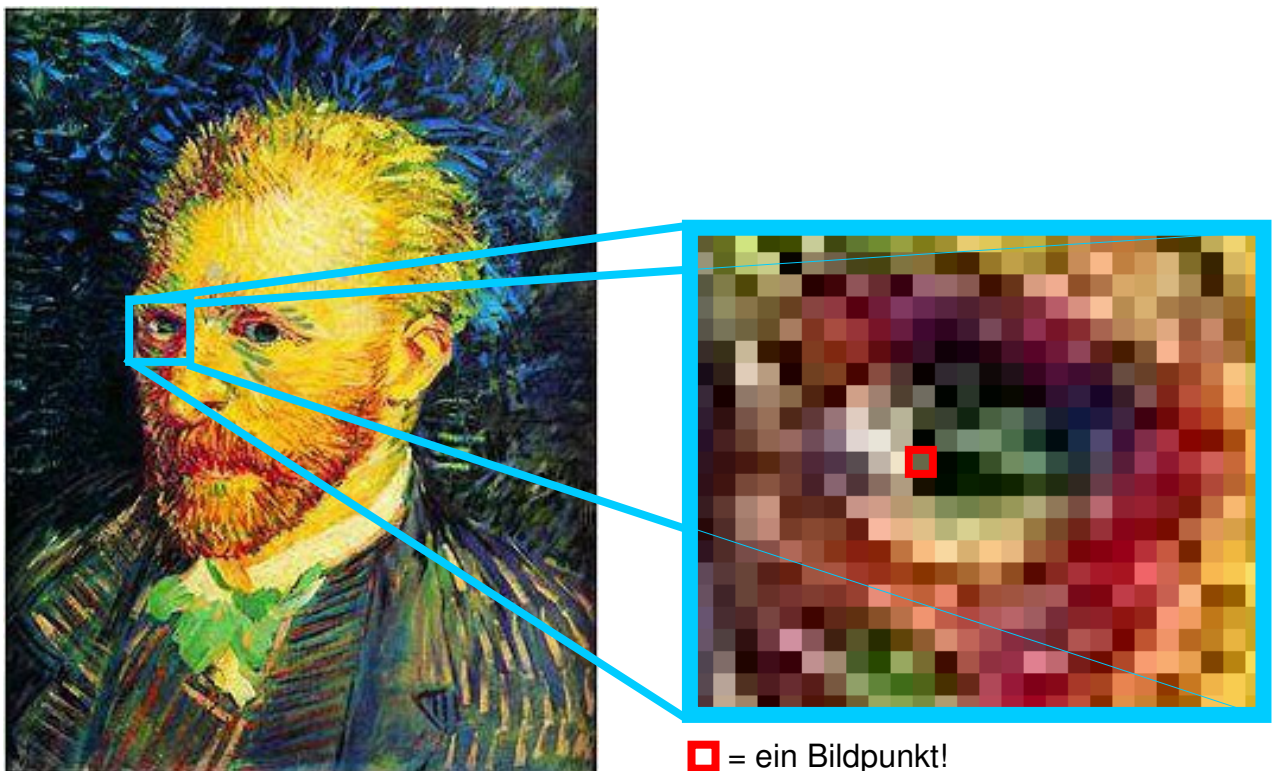


Abbildung 13: van Gogh stark vergrößert zur Verdeutlichung der Bildpunkte

Denken wir einmal darüber nach, was die Darstellung eines Bildes mit Hilfe so vieler kleiner Punkte und der den Farben zugeordneten Nullen und Einsen bedeutet:

Es ist doch zum Beispiel super einfach geworden, ein Bild verkleinert darzustellen! Wenn man zum Beispiel jeden zweiten Bildpunkt, bzw. jede zweite Spalte und jede zweite Zeile einfach nicht darstellt, ist das Bild nur noch ein Viertel so groß.

Zum Vergrößern des Bildes braucht man nur Zeilen und Spalten von Bildpunkten einzufügen, und die neuen Pixel in einer Farbe darzustellen, die so ähnlich ist wie die benachbarten Original-Pixel.

Wenn ich das Bild spiegeln will, brauche ich die Bildpunkte nur zeilenweise von rechts nach links zu lesen statt von links nach rechts, und wenn ich es auf den Kopf stellen will, brauche ich beim Lesen nur mit der letzten statt mit der ersten Zeile anfangen.

Besonders einfach ist das Ändern der Farben eines Bildes: Dafür muss man nicht einmal das Bild anfassen, sondern nur die Zuordnungstabelle, die festlegt, welches Bitmuster welcher Farbe entspricht (im Beispiel Tabelle 7).

Andererseits hat die Methode mit den Pixeln auch Nachteile: Wenn man ein Bild stark vergrößert, werden die Pixel sichtbar (wie in Abbildung 13) und das will man meistens nicht. Außerdem sind gerade Linien, die nicht genau senkrecht oder waagrecht durch das Bild laufen, keine Linien sondern Treppen, und das sieht man auch schon bei kleinen Vergrößerungen. Diese Linie ist ein Beispiel:



Ein weiterer Nachteil ist, dass man sehr schlecht bestimmte *Bildinhalte* gezielt verändern kann. Nehmen wir an, wir wollten Vincents Ohr größer machen! Wir können die Begrenzungslinie des Ohrs nicht als Ganzes ansprechen (also zum Beispiel darauf klicken und die ganze Linie ist markiert), sondern müssen alle Pixel, die dazu gehören, selbst auswählen.

Wegen dieser Nachteile hat man noch ein anderes Verfahren gefunden, um Bilder mit Nullen und Einsen zu beschreiben, die so genannte *Vektorgrafik*. Vektor kommt aus dem Lateinischen und bedeutet zu deutsch „Zeiger“.

Natürlich braucht man wie immer eine Zuordnungstabelle, in der die Bedeutung der Nullen und Einsen festgelegt wird. Nur werden jetzt nicht Farben zu Bildpunkten zugeordnet sondern die Nullen und Einsen stehen für bestimmte Formen: Die Bildbeschreibung ist so aufgebaut, dass Figuren beschrieben werden. Das könnte in unserer Sprache zum Beispiel so aussehen:

Kreis, Mittelpunkt 40 Bildpunkte nach rechts, 30 nach oben, Radius 20 Bildpunkte, Liniendicke 2 Bildpunkte, Liniensfarbe 0011, nicht ausgefüllt.

Gerade Linie, Beginn 20 Bildpunkte nach rechts, 50 Bildpunkte nach oben, Ende 100 Bildpunkte nach rechts, 10 Bildpunkte nach oben, Liniendicke 5 Bildpunkte, Liniensfarbe 0101

usw.

Das oben beschriebene Bild würde dann so aussehen:



Abbildung 14: Beispiel für eine Vektorgrafik

Dabei sind das grüne Raster im Hintergrund, die dünnen schwarzen Striche und die Zahlen nur eine Hilfe für dich, damit du die Bildpunkte leichter zählen kannst, um die Beschreibung nachvollziehen zu können! (Ein Kästchen entspricht einem Bildpunkt.)

Die Linienfarbe 0011 entspricht in diesem Beispiel blau, 0101 entspricht rot.

So etwas lässt sich dann wieder mit Hilfe von Zuordnungstabellen in Nullen und Einsen übersetzen: Für „Kreis“, „Gerade“ und alle anderen Formen, die man zeichnen können soll, gibt es ein Bitmuster. Das Programm, das die Bildbeschreibung später liest, muss wissen, wie viele Eigenschaften einer Figur jeweils genannt werden und wie sie dargestellt werden. Solche Festlegungen nennt man auch ein *Format*. Eine Eigenschaft der Figuren ist natürlich auch die Farbe, die wieder in einer eigenen Tabelle festgelegt und durchnummeriert werden.

Eine Vektorgrafik hat den Vorteil, dass man sie beliebig vergrößern kann, ohne dass die Bildpunkte größer werden. Man rechnet beim Vergrößern einfach alle Bildpunktangaben für die größere Darstellung aus. Der Kreis im obigen Beispiel hätte bei einer vierfachen Vergrößerung (Verdopplung von Länge und Breite des Bildes) seinen Mittelpunkt bei 140 Bildpunkten nach rechts und 100 Bildpunkten nach oben und einen Radius von 80 Bildpunkten. Erst wenn diese Werte ausgerechnet sind, zeichnet das Programm den Kreis. – Der wird dann natürlich trotzdem wieder mit den Bildpunkten des Bildschirms dargestellt.

Man kann bei einer solchen Darstellung natürlich sehr leicht die Eigenschaften des Kreises ändern und ihm z.B. eine andere Farbe geben, weil man sich nicht

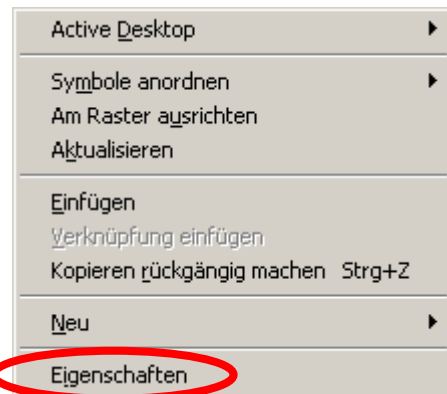
alle Pixel zusammensuchen muss, die dazu gehören, sondern einfach nur die Farbangabe ändern muss.

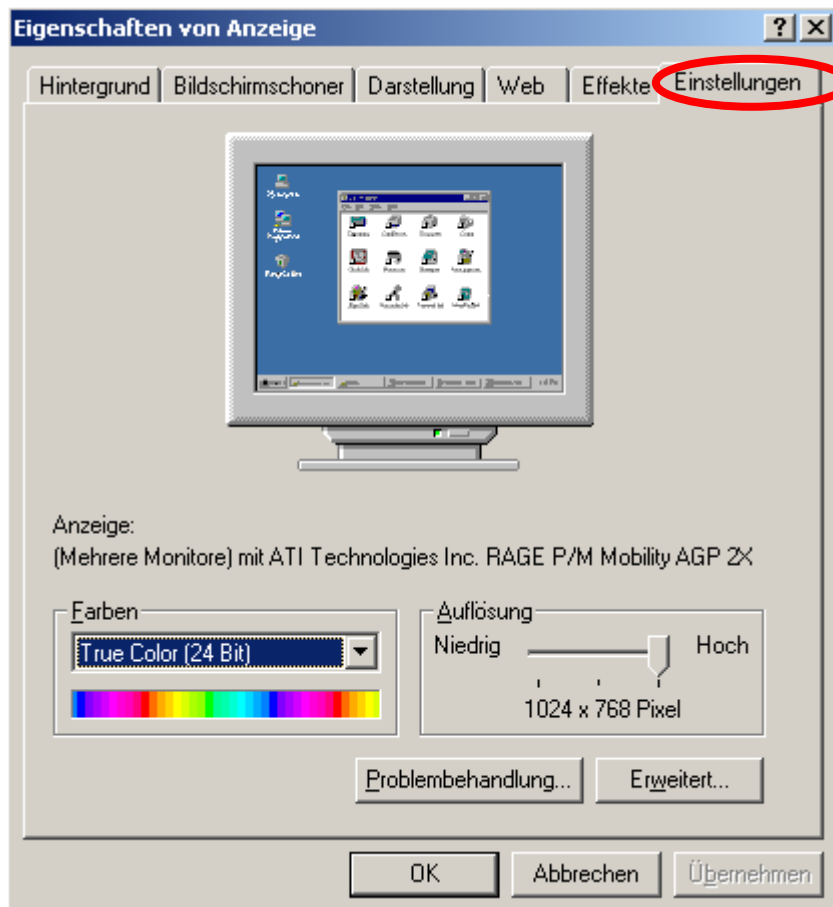
Die Darstellung eines Bildes auf dem Bildschirm oder dem Drucker passiert aber immer mit Hilfe von Bildpunkten. Und weil du so tapfer bis hier hin die graue Theorie gelesen hast, machen wir dazu wieder ein kleines Experiment mit deinem PC:

Experiment 2: Bildpunkte und Farben auf dem Monitor

Wenn die folgenden Erklärungen bei deinem Rechner überhaupt nicht stimmen sollten, frage bitte einen Erwachsenen, ob er dir die Bildeinstellungen zeigen kann. Kleinere Abweichungen in der Darstellung sind aber möglich und kein Grund zur Panik! Störe dich nicht an unbekanntem Begriffen in der Beschreibung; du wirst sie später noch lernen!

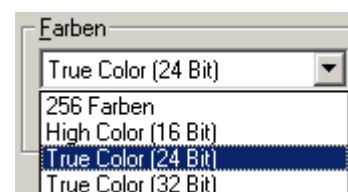
1. Starte deinen Rechner!
2. Klicke mit der **rechten** Maustaste auf eine freie Fläche auf dem Bildschirm; du erhältst ein so genanntes → *Kontextmenü*, das einige Dinge enthält, die du mit der angeklickten Sache - in dem Fall der Windowsoberfläche, dem *Desktop* - machen kannst.
3. Klicke auf „Eigenschaften“ (dabei ist es jetzt egal ob du mit der rechten oder linken Taste klickst).
4. Klicke auf die *Registerkarte* „Einstellungen“. Du solltest ein ähnliches Bild sehen wie dieses:





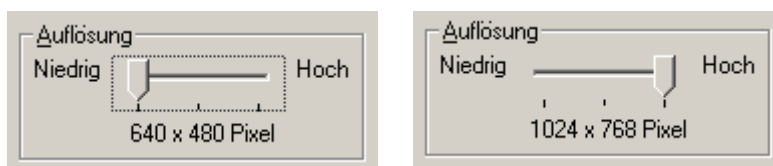
Hier findest du genau das, worüber wir die ganze Zeit geredet haben: Unter „Farben“ ist ein Auswahlfeld, das sich öffnet, wenn du auf den kleinen Pfeil klickst. (Man nennt das ein *Drop-Down-Feld*, weil es beim Öffnen quasi nach unten fällt.)

In diesem Feld kannst du auswählen, wie viele verschiedene Farben dein Bildschirm darstellen können soll. Bei der Einstellung „256 Farben“ wird genau wie oben beschrieben für jeden Bildpunkt ein Byte benutzt, um die Farbe des Pixels festzulegen. Bei den anderen Einstellungen steht es schon dahinter: 16 Bit, also zwei Byte pro Bildpunkt, ermöglichen die Darstellung von 65536 verschiedenen Farben, bei 24 Bit, also drei Byte sind es schon 16.777.216 verschiedene Farben und bei 32 Bit oder vier Byte 4.294.967.296, also über vier Milliarden verschiedene Farben! Bestimmt mehr als das menschliche Auge überhaupt unterscheiden kann!



Warum kann man das überhaupt einstellen, wie viele Byte man pro Bildpunkt nehmen möchte, um die Farbe zu beschreiben? Warum nimmt man nicht einfach die beste Darstellung, also die mit 32 Bit? Nun, das ist ganz einfach: Die Farbe, die jeder einzelne Bildpunkt haben soll, muss ja auch irgendwo ausgerechnet werden, und das ist umso schwieriger und dauert um so länger, je mehr Farben es gibt, und je mehr Bildpunkte man hat.

Mehr Bildpunkte? Wo stellt man das denn ein? Unter dem Wort „Auflösung“! Wie fein ein Bild für seine Darstellung durch Bildpunkte unterteilt wird, bezeichnet man als *Auflösung*. Je mehr Bildpunkte, desto besser die Auflösung.



Auf meinem Computer kann man Einstellungen zwischen 640x480 und 1024x768 Bildpunkten vornehmen. In der guten Auflösung sind das also 786.432 Pixel. Wenn ich für jedes Pixel vier Bytes für die Farbfestlegung benutze, sind das immerhin 3.145.728 Bytes, oder 25.165.824, also über 25 Millionen Bits (und jedes Bit ist in unserem Computermodell ein Schalter!), die ich nur dafür brauche, den Bildschirm darzustellen.

Nun hast du Gelegenheit gehabt, nachzusehen, dass die Geschichte mit dem Darstellen der Bilder im Computer wirklich so einfach funktioniert, wie wir das in den Beispielen zuvor schon gesehen haben.

Du solltest das unter Punkt 4 des Experiments gezeigte Fenster mit „Abbrechen“ schließen. Dann bleiben die Einstellungen deines Computers unverändert.

3.6 Ein Programm für unseren Computer – natürlich mit Nullen und Einsen

Nun ging es schon ziemlich lange darum, wie man mit Nullen und Einsen alles mögliche darstellen kann: Zahlen, Text, Bilder. Es fehlt uns nur noch eine aber dafür die alles entscheidende Sache, die auch mit Nullen und Einsen dargestellt werden muss: Das Programm! Darauf kommen wir in diesem Kapitel zu sprechen!

Was ist das überhaupt, ein Programm?

Du kennst vielleicht das Fernsehprogramm, das *Programm* für die Waschmaschine, das *Programm* einer Zirkusaufführung, auch deinen Stundenplan in der Schule könnte man ein *Programm* nennen.

Ein Programm ist ein im Voraus festgelegter Ablauf. Es legt fest, was wann gemacht wird.

Im Zirkus steht im Programm also zum Beispiel drin, dass erst der Direktor das Publikum begrüßt, dann kommt die Nummer mit den Hunden, dann die Hochseilartisten, dann der Clown, dann die Raubtiernummer usw.

Bei der Waschmaschine ist das Programm durch eine technische Vorrichtung festgelegt, meist ein Drehschalter mit einer Uhr, der dafür sorgt, dass zuerst Wasser in den Behälter mit dem Vorwaschmittel läuft, dass sich dann die Trommel abwechselnd dreimal linksrum und dreimal rechtsrum dreht, dass dann das Waschwasser abgepumpt, dann zwischengeschleudert, dann das Hauptwaschmittel eingespült, dann wieder abgepumpt, dann Weichspüler eingelassen, und am Ende geschleudert und die Tür freigegeben wird. Du siehst ein Programm kann ziemlich lang werden. Bei der Waschmaschine hat man sogar die Möglichkeit, zwischen verschiedenen Programmen zu wählen: Wenn man das Programm für die pflegeleichte Wäsche einstellt, dreht sich die Trommel vielleicht immer nur zweimal links- und rechtsrum, oder sie dreht sich langsamer oder was die Erbauer der Waschmaschine sich sonst ausgedacht haben.

Entscheidend bei diesem langen Absatz über die Waschmaschine ist nur, dass du verstehst, was ein Programm ist, und dass man zwischen mehreren Programmen wählen kann, damit unterschiedliche Dinge passieren.

Genau so ist das beim Computer nämlich auch: Wenn du das Textverarbeitungsprogramm wählst, macht der Computer etwas anderes als wenn du ein Grafikprogramm aufrufst. Das besonders tolle am Computer – und das geht bei der Waschmaschine nicht! – ist, dass man sogar ein Programm ausführen kann, mit dem man neue Programme erzeugen kann! Mehr dazu im Kapitel 9 über das Programmieren.

Unser Computer soll also jetzt verschiedene Dinge können und wir wollen auswählen können, welches er davon macht.

Fassen wir zunächst zusammen, was wir schon haben: In Kapitel 3.1 haben wir schon einen Computer gebaut, der zwei Zahlen addieren kann. Der Einfachheit halber, benutzen wir hier nur die Version aus Abbildung 5 mit nur einer Lampe als Ausgabe und gehen davon aus, dass die Aufgabe $1 + 1$, deren Ergebnis dieser Computer ja nicht darstellen konnte, nicht eingegeben wird, sondern nur $0 + 0$, $0 + 1$ oder $1 + 0$.

Wie wäre es, wenn wir als zweites Programm eine Multiplikationsaufgabe (Malnehmen) stellen? Natürlich wieder nur mit den ganz einfachen Zahlen Null und Eins! Unser Computer müsste die Aufgaben 0×0 , 0×1 , 1×0 und 1×1 rechnen können.





Schalter 1	Schalter 2	Ergebnis	Lampe
0	0	0	
0	1	0	
1	0	0	
1	1	1	

Tabelle 8: Ergebnistabelle für Multiplikation

Überlegen wir uns, wie wir die Schalter und Lampe verkabeln (programmieren!) müssen, damit bei der Multiplikation das richtige Ergebnis herauskommt:

Links stehende Tabelle 8 zeigt, wie die Lampe leuchten muss, je nachdem wie die Eingabeschalter stehen.

Ja aber

Das ist doch genau die gleiche Tabelle, wie wir sie schon für die zweite Lampe unseres Additionscomputers gemacht haben! Schau dir Tabelle 1 auf Seite 16 noch einmal genau an! Die Spalte „Zweite Lampe“ dort sagt genau das selbe wie Tabelle 8!

Also haben wir auch das Programm (die Verkabelung) für die Multiplikation schon fertig in der Schublade liegen! Abbildung 7 auf Seite 18 zeigt uns die Verschaltung die wir hier auch brauchen!

Um also jetzt einen Computer zu bauen, bei dem man sich aussuchen kann, ob man addieren oder multiplizieren will, brauchen wir doch nur die beiden Computer aus Abbildung 5 und Abbildung 7 zusammenschalten und einen dritten Schalter einbauen, mit dessen Hilfe man sich aussuchen kann, ob man addieren oder multiplizieren möchte!

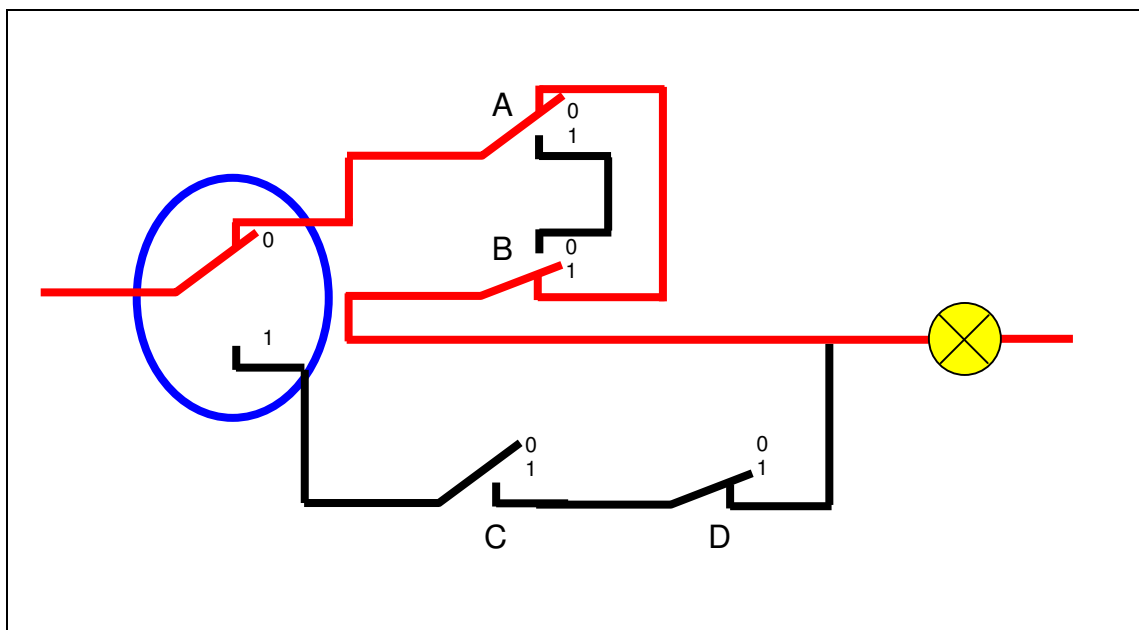


Abbildung 15: Computer, bei dem man wählen kann, ob man addieren oder multiplizieren will

Der blau eingekreiste Schalter ist neu hinzugekommen. Mit ihm können wir wählen, ob wir addieren oder multiplizieren wollen. Er ist unser Programmwahlschalter!

Abbildung 15 zeigt einen Computer, bei dem man wählen kann zwischen den Programmen „Addiere die Zahlen A und B“ oder „Multipliziere die Zahlen C und D“. (A, B, C, D sind die Bezeichnungen der Schalter.) Eigentlich wollten wir aber die Wahl haben zwischen „Addiere die Zahlen A und B“ und „Multipliziere die Zahlen A und B“, das heißt, wir müssen wieder dafür sorgen, dass die Schalter C und D die selben sind wie A und B. Das haben wir beim Computer in Abbildung 8 auf Seite 19 aber schon einmal gemacht. Du kannst unseren Computer bestimmt selbst so modifizieren, dass er unseren Wünschen entspricht! Er müsste dann aussehen wie in Abbildung 16:

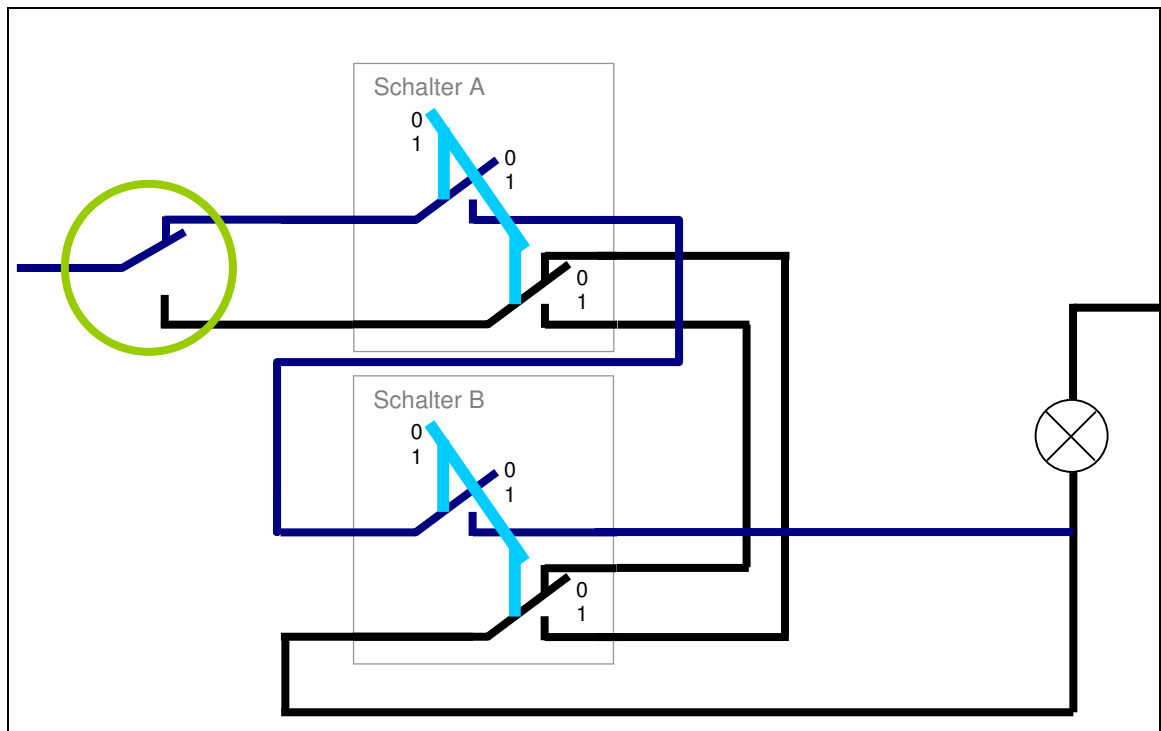


Abbildung 16: Computer, der die Schalter A und B entweder addieren oder multiplizieren kann, je nachdem wie der Programmwahlschalter (grüner Kreis) steht.

Bei den Computern in Abbildung 15 und Abbildung 16 können wir also zwischen den Programmen „Addieren“ und „Multiplizieren“ mit einem Schalter wählen. Ups! Ein Schalter! Unser Programm ist also schon → *binär* dargestellt! Der Schalter steht auf 0 oder 1, auf Addieren oder Multiplizieren!

Hätten wir mehrere Befehle als Addieren oder Multiplizieren zur Auswahl, würden wir mit mehreren Schaltern Kombinationen erzeugen, um den Befehl auszuwählen, den wir haben wollen!

Noch einmal zur Klarheit: *Wie* der Computer addiert oder multipliziert, hängt von der Verdrahtung der Ein-/Ausgabe-Schalter ab! Mit dem Programmwahlschalter wählen wir nur eine von mehreren möglichen Verdrahtungen, die sozusagen für den Befehl „addiere“ oder „multipliziere“ stehen, aus.

Wir sehen also: Es ist ganz einfach, auch Programme mit Nullen und Einsen darzustellen.

Überhaupt gibt es fast nichts, das sich nicht mit Nullen und Einsen darstellen ließe. Wenn man etwas, das eigentlich anders dargestellt wird, in Nullen und Einsen umwandelt, sagt man dazu auch *digitalisieren*:

Digital bedeutet, dass etwas in genau festgelegten Stufen dargestellt wird und nur bestimmte Werte annehmen kann.

Dein Schulzeugnis zum Beispiel ist eine digitale Darstellung deiner Leistungen: Dort steht entweder ein „gut“ oder ein „befriedigend“ (oder irgendeine andere Note) aber es steht dort nicht „irgendwo zwischen mittelprächtigt und echt klasse“.

Unsere Schalter oder unsere Lampe sind also auch digital: auf oder zu, bzw. an oder aus. Wir unterscheiden nicht, ob die Lampe hell oder nicht so hell leuchtet.

Das Gegenteil von digital heißt **analog**. Das bedeutet, dass alle Werte (meist zwischen zwei Grenzen) möglich sind. Ein Zeigerinstrument (auch eine Uhr mit Zeigern) ist zum Beispiel analog, weil der Zeiger jede beliebige Position auf der Skala einnehmen kann.

Wenn man eine analoge Größe digital darstellen möchte, dann unterteilt man die Skala einfach künstlich in Stufen. Diese Stufen nennt man **Quanten** (genau wie Käsefüße). Eine Waage zum Beispiel, die nur ganze Kilogramm in Zahlen anzeigt, ist digital. („Digit“ bedeutet im Englischen → „Ziffer“.) Wenn du dich damit wiegst, zeigt sie zum Beispiel 43kg an. Das bedeutet aber nicht, dass du genau 43kg wiegst, sondern dass du zwischen 42,5kg und 43,49999...kg wiegst. Der Fehler, der dabei zwangsläufig entsteht, heißt **Quantisierungsfehler**. Im Beispiel beträgt er $\pm 0,5\text{kg}$. Wenn man einen kleineren Quantisierungsfehler haben möchte, muss man einfach mehr Stufen für die Unterteilung nehmen.

Sind wir mit dem Thema „Programm“ schon fertig? Sieh dir noch einmal die Definition auf Seite 41 an und denke an die Waschmaschine! Dass wir zwischen verschiedenen Dingen wählen können, die die Maschine macht, war ja nur eine Seite der Medaille! Je nachdem welches Programm wir auswählen, soll die Maschine doch auch bestimmte Dinge in einer bestimmten *Reihenfolge* ausführen! Es würde uns ja nicht viel helfen, wenn die Waschmaschine erst schleudern und am Ende des Programms erst das Wasser einlassen würde, oder die Reihenfolge sich sogar von Waschgang zu Waschgang ändern würde!

Unser Computer hat aber nur, je nachdem welches Programm wir gewählt haben, entweder zusammengezählt oder malgenommen. Danach ist er wieder in einen Dornröschenschlaf gefallen und hat nichts mehr gemacht!

Wir müssen also noch einen Weg finden, wie wir dem Computer mehrere Befehle geben können, die er dann in einer bestimmten Reihenfolge abarbeitet.

Auch das ist erst mal ganz einfach: Wir brauchen doch aus dem blau eingekreisten Schalter in Abbildung 15 nur einen Drehschalter mit Uhr zu machen, wie er auch in der Waschmaschine sitzt! Schon wird unser Rechner nacheinander die Addition und die Multiplikation ausführen! Und wenn wir noch mehr Verschaltungen an weiteren Positionen des Drehschalters anschließen, können wir das Programm auch länger machen.

Schwieriger wird es dann, wenn Rechenergebnisse aus vorangegangenen Schritten in späteren Teilen des Programms verwendet werden sollen. Wenn also die Addition zum Beispiel das Ergebnis Eins hat (die Lampe leuchtet) und dieses Ergebnis die Eingabe für die Multiplikation werden soll. Dafür muss doch einer der Eingabe-Schalter gedrückt werden, wenn die Ausgabe-Lampe leuchtet! Schwierig!

Erinnern wir uns daran, wie wir ganz am Anfang überlegt haben, wie man die Zahlen 0 und 1 im Computer darstellen kann (s. Seite 14). Die Lampe war doch nur eine von mehreren Möglichkeiten; eine zweite Möglichkeit war der Schalter, den wir ebenfalls verwendet haben! Wie wäre es denn, wenn der Computer sein Ergebnis nicht mit Hilfe einer Lampe sondern mit einer Schalterstellung ausgeben würde? Dann könnten wir diese Schalterstellung sofort als Eingabe für unseren nächsten Programmschritt verwenden! Tolle Idee, was?!

Dafür bräuchten wir natürlich Schalter, die sich elektrisch betätigen lassen! Wenn unsere Lampe leuchten würde, weil ein Strom ankommt, muss unser Schalter auf Eins schalten. Solche Schalter gibt es; sie heißen *Relais* (sprich: relä). Das ist nichts anderes als ein Elektromagnet, der einen Schalter betätigt, sobald Strom durch ihn fließt.

4 Halbleiter und richtige Computer

Nun ist es an der Zeit, darüber nachzudenken, ob wir die Milliarden von Schaltern, die wir brauchen, wirklich mit herkömmlichen Schaltern, mit denen wir auch das Licht in unserer Wohnung einschalten, oder mit Relais', die auch nicht kleiner sind, realisieren wollen! Denke einmal an unser Experiment 2 zurück (Seite 39): Über vier Milliarden Schalter sind nötig, um einen Bildschirm so darzustellen, wie wir ihn kennen!

Wo kriegen wir also kleinere Schalter her?

4.1 Der Transistor

Was macht einen Schalter aus? Er soll umschalten können zwischen „Strom leiten“ und „keinen Strom durchlassen“. Metalle wie Eisen, Aluminium, Kupfer, Gold und Silber leiten den Strom. Einen Schalter bauen wir damit, indem wir eine Lücke im Stromweg entstehen lassen oder sie schließen.

Nun gibt es Materialien, die leiten den Strom nur unter bestimmten Umständen, nämlich dann, wenn man ihnen ein wenig Strom hinzugibt; solche Materialien heißen Halbleiter. Man kann daraus einen Schalter bauen, indem man in die Schalterlücke ein solches Material macht. Wenn man dann daran einen Strom anschließt, leitet der Schalter, wenn nicht dann nicht! Das ist ja toll! Wir können mit Halbleitern also nicht nur Schalter bauen, sondern das sind sogar Relais, also Schalter, die man elektrisch betätigen kann!

*Solche Schalter nennt man **Transistor**.*

Transistoren können noch viel mehr, nämlich zum Beispiel viel oder wenig Strom durchlassen, je nachdem wie groß der Steuerstrom ist, aber das ist für uns momentan nicht wichtig. Wir wollen ganz einfache Schalter.

Aber jetzt kommt das tollste: Transistoren kann man winzig klein bauen! So klein, dass diese vier Milliarden Schalter auf die Fläche eines Daumennagels passen – na ja es muss schon der Daumennagel eines Erwachsenen sein, aber das ist doch trotzdem eine tolle Leistung, oder? Ganz so einfach wie ich das hier schreibe ist es auch nicht, aber es gibt Leute, die das können und für uns tun, und das soll uns erst mal genügen.

*Einen solchen Daumennagel voller Transistoren nennt man auch einen **Integrierten Schaltkreis**, englisch abgekürzt **IC** für *Integrated Circuit*, oder wenn man sich nicht ganz so vornehm ausdrücken will, einen **Chip**.*

Fassen wir also noch einmal zusammen, was wir bei unseren einfachen Modell-Computern alles gebraucht haben, und das wir jetzt in einem richtigen Computer mit Transistoren bauen müssen:

4.2 Immer im Takt

Einer der Schalter in unserem Modellcomputer – der Programmwahlschalter – war ein Drehschalter mit Uhr! Wie kriegen wir das denn hin? Es gibt elektronische Bausteine, die in einem regelmäßigen *Takt* Strom abgeben. Die können wir benutzen, um unseren Computer die Programmschritte der Reihe nach ausführen zu lassen, indem wir diesen Takt als Steuerstrom für den Drehschalter-Transistor benutzen.

Die Geschwindigkeit dieses Taktes ist übrigens ein Maß dafür, wie schnell ein Computer arbeitet. Deshalb findet man sie in fast allen Werbeanzeigen für Computer. Dort steht zum Beispiel „3,4GHz Taktfrequenz“.

Hertz ist die Einheit in der man Häufigkeiten misst. Sie ist nach dem Physiker Heinrich Hertz benannt und wird „Hz“ abgekürzt. Das Fremdwort für Häufigkeit ist **Frequenz**. Eine Frequenz von 1Hz heißt „ein Mal pro Sekunde“, 100Hz demnach „hundert Mal pro Sekunde“.

Du weißt sicherlich, dass ein Kilometer (1km) das selbe bedeutet wie 1000m. **Kilo** bedeutet also „Tausend“. Dementsprechend ist ein Kilohertz (1kHz) das selbe wie „tausend Mal pro Sekunde“.

Mega bedeutet Million und **Giga** Milliarde.

1MHz heißt also 1.000.000 (eine Million) Mal pro Sekunde, 1GHz bedeutet 1.000.000.000 (eine Milliarde) Mal pro Sekunde und 3,4GHz ist ein Takt von 3.400.000.000 (drei Milliarden vierhundert Millionen) Mal pro Sekunde.

Der Bemerkung aus Kapitel 2, dass Computer zwar unheimlich dumm, dafür aber sagenhaft schnell sind, kannst du nun sicher zustimmen, oder?

Du hast gesehen, wie einfach die Aufgaben sein müssen, damit ein Computer sie rechnen, oder besser gesagt schalten kann. Aber wenn ein Computer – mit 32 Bit gleichzeitig! – mehr als drei Milliarden Mal in der Sekunde etwas tun kann, dann kann man schon ganz schön schwierige Aufgaben bewältigen; man muss sie eben nur in ganz viele kleine Aufgaben zerlegen.

Das ist übrigens auch eine der tollen Eigenschaften von Transistoren, dass man sie nämlich so schnell und so oft schalten kann. Mit mechanischen Schaltern ginge das nicht!

4.3 Arbeitsspeicher

Welche Schalter außer diesem nun sagenhaft schnellen Drehschalter brauchen wir noch?

- Schalter mit denen wir die Zahlen (oder besser „Daten“) eingeben, die für unsere Rechnung benötigt werden.
- Schalter mit denen der Computer das Ergebnis darstellt (das waren zuerst Lampen, wir haben aber gesehen, dass es günstiger ist, dass man dafür auch Schalter nimmt).
- Schalter mit denen der Befehl festgelegt wird, den der Computer ausführen soll (also zum Beispiel Addition oder Multiplikation). Jeder Befehl gehört später zu einem Programm.

Können das nicht alles die selben Schalter sein? Doch können Sie, mit ganz wenigen Ausnahmen!

Die Schalter in einem richtigen Computer haben auch einen anderen Namen, der dich zunächst überraschen wird. Sie heißen *Speicher*.

Die wenigen Ausnahmen heißen *Register*, aber dazu kommen wir später.

„Wieso sind Schalter das selbe wie Speicher?“ wirst du dich jetzt fragen. Nun, wenn du an unseren Modellcomputer zurückdenkst, dann hast du doch die Zahlen, die der Computer beispielsweise addieren sollte, an den Schaltern eingetippt, wie es ihrer binären Darstellung entspricht (vgl. Tabelle 3 auf Seite 21!). Die Schalterstellungen sind auch so geblieben, bis du sie geändert hast. Das bedeutet, der Computer hätte jederzeit diese Zahl für eine andere Rechnung verwenden können, vorausgesetzt, die Schalter der Eingabezahl wären mit den richtigen Programmschaltern verbunden worden.

Die Zahl, die du eingegeben hast, wurde also gespeichert. Der Computer hat sich die Zahl „gemerkt“.

In einem echten Computer ist also ein oder mehrere Speicherbausteine eingebaut. Der sieht zum Beispiel so aus wie in Abbildung 17:



Abbildung 17: Speicherbaustein

Ziemlich unscheinbar, nicht? Aber in einem dieser Bausteine ist Platz für zum Beispiel 16MB! (Speicher mit mehr oder weniger Platz sehen sehr ähnlich aus.) Meist werden acht Stück davon auf einer kleinen *Platine*, so heißt das kleine Plastikbrettchen, auf dem die Teile festgelötet und elektrisch miteinander verbunden werden, gemeinsam untergebracht, das macht schon 128MB!

MB steht für **Mega-Byte**. Du erinnerst dich: Mit einem Schalter können wir ein → Bit

an Information darstellen. Acht Bits haben wir zu einem → Byte zusammengefasst und → Mega bedeutet Million.

Weil Speicher Bits speichern und weil zehn Bits 1024 Kombinationsmöglichkeiten haben, etwas darzustellen, nimmt man es bei Speicher mit den „kilo“ und „Mega“ nicht so genau und meint mit kilo auch manchmal 1024 und mit Mega manchmal $1024 \times 1024 = 1.048.576$.

Wie auch immer das gemeint ist, in diesem Bauteil sind jedenfalls über eine Milliarde Transistoren untergebracht! In Wirklichkeit braucht man für eine Speicherzelle auch mehr als einen Transistor, aber das ist gar nicht so schlimm: Viel schlimmer ist, dass man diese vielen Speicherzellen ja auch anschließen muss! Wir wollen ja schließlich auch irgendwie „lesen“ können, was in dem Speicher drinsteht, also Strom anschließen können, um zu sehen wie die einzelnen Schalter gerade stehen. Und dabei wollen wir jedes Byte Speicher (also acht Schalter) direkt auswählen können, egal wo es in dem Baustein eingebaut ist.

Diese wichtige Eigenschaft, dass man wenn nicht auf jedes Bit so doch wenigstens auf jedes Byte einzeln zugreifen können möchte, hat dieser Art Speicher seinen Namen gegeben: Auf Englisch heißt er *Random Access Memory*, abgekürzt *RAM*, was so viel heißt wie „Speicher mit wahlfreiem Zugriff“. Auf Deutsch bezeichnet man diesen Speicher auch als *Arbeitsspeicher*, weil die Daten, mit denen der Computer arbeitet, in solchen Speichern stehen und auch die Programme, die der Computer abarbeitet – also alle Schalter auf der linken Seite unserer Modellcomputer-Bilder.

Warum ist dieser wahlfreie Zugriff beim Arbeitsspeicher so wichtig? Wir werden später noch sehen, dass wir nicht immer genau vorhersagen können, wo genau der nächste Befehl steht, den der Computer abarbeiten soll. Auch die nächste Zahl, mit der er rechnen soll, wissen wir nicht immer im voraus, weil sich beides oft erst aus dem Ablauf des Programms ergibt. Wir müssen also jederzeit frei

bestimmen können, welches Speicherbyte (man sagt auch „Speicherzelle“) als nächstes gelesen werden soll.

Daten sind jede Art von Information, also Zahlen, Texte, Bilder, Ton usw., und werden im Computer immer durch Nullen und Einsen dargestellt. Die Art und Weise, wie sie dargestellt werden, also quasi die Übersetzungsvorschrift, den \rightarrow Code nennt man **Datenformat**.

4.4 Die CPU

Wenn du dich richtig an unsere Modellcomputer erinnerst, hing es davon ab, wie die Eingabeschalter (die zum Beispiel die Zahlen dargestellt haben, die addiert werden sollten) miteinander verdrahtet waren, damit das richtige Ergebnis herauskommt. Wo steckt in einem echten Computer also nun diese Verdrahtung, die ja den eigentlichen Befehl – in diesem Beispiel „addiere Zahl 1 und Zahl 2“ – ausführt?

Der Programmwahlschalter hatte ja nur ausgewählt, dass wir addieren wollen. Dessen Schalterstellungen befinden sich im Arbeitsspeicher. Die Verdrahtung, also alle Befehle, die der Computer ausführen kann, sind in der sogenannten *Central Processing Unit*, oder kurz *CPU*, untergebracht. Das ist natürlich mal wieder Englisch und bedeutet soviel wie zentrale Recheneinheit. Man sagt auch einfach kurz *der Prozessor*. Da muss man sich nicht viel bei denken, irgend einen Namen muss das Ding ja haben!

Nichtsdestoweniger ist die CPU das Herzstück eines jeden Computers! Danach richtet sich, was der Computer kann! Neben der \rightarrow Taktfrequenz ist der Typ der CPU das zweite wichtige, das man auf jeder Werbeanzeige für einen Computer findet. Wie eine CPU aussieht, siehst du in Abbildung 18. Solltest du einmal Gelegenheit haben, in deinen eigenen Rechner hineinzuschauen, wirst du die CPU vielleicht zuerst gar nicht finden, denn manchmal hat sie Huckepack noch Kühlrippen und/oder einen Ventilator oben aufgeschnallt, damit sie bei der vielen schnellen Arbeit nicht zu heiß wird! Das sieht dann aus wie in Abbildung 19.



Abbildung 18: Central Processing Unit (CPU) eines Computers

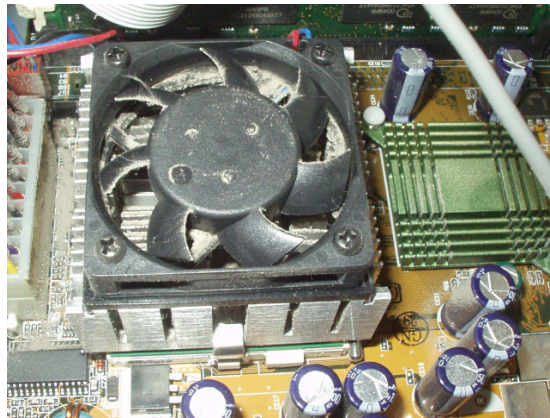


Abbildung 19: CPU eingebaut mit Kühlrippen und Lüfter

In einer CPU gibt es übrigens auch jede Menge Transistoren, denn natürlich ist nicht jeder Befehl einzeln verdrahtet, sondern die Verdrahtungen für die einzelnen Befehle werden umgeschaltet, und das macht man natürlich mit den bewährten Transistoren.

Jetzt haben wir also die Programmwahlschalter und die Schalter für die Eingabe und für die Ausgabe im Arbeitsspeicher untergebracht, und die Verkabelung in der CPU. Wie kriegen wir jetzt beides zusammen? Nehmen wir an, wir haben die CPU-Verdrahtung auf „addieren“ geschaltet; wir müssen ihr aber doch noch sagen, welche Zahlen, d.h. welche Schalter sie addieren soll!

Nun das geht auch wieder ganz einfach: In der CPU sind ein paar Byte Speicher enthalten; das sind die *Register*. Sie sind die Ausnahmen, die ich am Beginn von Kapitel 4.3 kurz erwähnt habe. Alle Befehle, die in der CPU verdrahtet sind, arbeiten mit diesen Registern. Zusätzlich spendieren wir noch Befehle, die in der Lage sind, ein Byte (oder mehrere) aus dem Arbeitsspeicher auszulesen und in diese Register zu kopieren.

Das Ergebnis eines Befehls schreibt die CPU ebenfalls in ein Register und deshalb brauchen wir natürlich Befehle, die aus den Registern in den Arbeitsspeicher zurück kopieren können. Du siehst, die Aufgaben werden noch ein Stück weiter untergliedert, und auf den ersten Blick umständlicher: Statt zu sagen „addiere Zahl 1 und Zahl 2!“ (indem wir einige Schalter betätigen), müssen wir sagen:

- „Kopiere das Byte Nr. 4711 des Arbeitsspeichers in das Register A!“
- “Kopiere das Byte Nr. 4712 des Arbeitsspeichers in das Register B!“
- “Addiere Register A und Register B und schreibe das Ergebnis in Register C!“
- “Kopiere Register C in Byte Nr. 4711 des Arbeitsspeichers!“

Jede Zeile dieser Befehlsfolge entspricht einer Schalterstellung unseres Programmwahlschalters! – Aber der schaltet ja so schnell, dass das gar nicht ins Gewicht fällt!

Was gewinnen wir durch dieses umständliche Vorgehen? Damit wird genau das realisiert, was ich vorhin so leichtfertig als selbstverständlich möglich hingestellt habe: Sowohl die Zahlen, die wir eingeben wollen, als auch die Ergebnisse, die der Computer jeweils ausrechnet, können in dem selben Speicher stehen. Ach ja, und das sollte ja deshalb so sein, damit wir Rechenergebnisse für weitere Berechnungen wiederverwenden können!

4.5 Befehle

Welche Befehle müssen in so einer CPU noch verdrahtet werden? Kopieren muss die CPU können, haben wir gesehen. Was noch?

Um einen leistungsfähigen Computer zu erhalten, ist es wichtig, den Befehlsatz, den der Prozessor bereitstellt, möglichst geschickt so anzulegen, dass die Befehle möglichst einfach verdrahtet werden können, dass sie möglichst schnell, das heißt in möglichst wenigen Takten abgearbeitet sind, und dass man möglichst gut jede denkbare komplexe Aufgabe in diese Befehle zerlegen kann.

Es gab mal eine Zeit, da hat man Prozessoren gebaut, die hatten einen besonders kleinen Befehlsvorrat! Dadurch konnte man den Computer so viel schneller machen, dass der zusätzliche Aufwand dadurch, dass man die komplexen Aufgaben in noch mehr kleine Aufgaben zerlegen musste, mehr als ausgeglichen wurde. Solche Prozessoren hießen *RISC*, was ein Akronym¹ ist und als englisches Wort „Risiko“ bedeutet; ausgeschrieben heißt es *Reduced Instruction Set Computer* und bedeutet „Computer mit reduziertem Befehlsvorrat“.

Manche Befehle, die man in einem Prozessor-Befehlssatz findet, wundern einen, und man fragt sich wofür man die wohl brauchen kann. Man muss sich aber eigentlich keine weiteren Gedanken darüber machen, denn wie du im Kapitel 9 noch sehen wirst, braucht man Programme gar nicht mit den komischen und umständlichen Prozessorbefehlen zu schreiben, sondern es gibt Computerprogramme, die Befehle, die für den Menschen einleuchtend sind, bereitstellen und in Befehle für einen Prozessor übersetzen. Solche Programme heißen *Compiler* (sprich: kompeiler).

Selbst wenn man mit Prozessorbefehlen programmieren will oder muss, tut man das nicht mit den eigentlichen Prozessorbefehlen – denn die bestehen ja

¹ Ein Akronym ist eine Abkürzung, die selbst wieder ein richtiges Wort ergibt.

nur aus Nullen und Einsen! – sondern mit einem speziellen Compiler, dem sogenannten *Assembler*.

Ein Beispiel:

mov al,61h

Das ist ein Assemblerbefehl und bedeutet „schreibe die → hexadezimale Zahl 61h in das Register *al*!“ *al* ist dabei nur ein Name für das Register. Es könnte genauso gut *Schuhkarton* heißen.

Als Prozessorbefehl sieht das ganze so aus: *10110000 01100001*

Das erste Byte ist der Befehl „schreibe in das Register *al*!“ Das zweite Byte ist die Zahl 61h: Dezimal geschrieben ist $61h = 6 \times 16 + 1 \times 1 = 97$. Die acht Stellen des Bytes als Zahl interpretiert bedeuten von rechts 1, 2, 4, 8, 16, 32, 64, 128. Das Byte 01100001 bedeutet also $0 \times 128 + 1 \times 64 + 1 \times 32 + 0 \times 16 + 0 \times 8 + 0 \times 4 + 0 \times 2 + 1 \times 1 = 97$

Hier noch einige Beispiele von Prozessorbefehlen, die einen vielleicht wundern:

Es gibt Befehle, mit deren Hilfe man das in einem Register stehende Bitmuster nach links oder rechts verschieben kann! So etwas lässt sich relativ leicht verdrahten, aber wofür soll das gut sein? Dazu muss man sich überlegen, was das Verschieben nach links bedeutet, wenn die Bits, die in dem Register stehen, gerade eine Zahl darstellen. Wenn du genau überlegst, fällt dir vielleicht auf, dass wir mit unseren Dezimalzahlen genau das selbe machen, wenn wir sie mit zehn multiplizieren: Wir schieben das Ziffernmuster nach links und hängen hinten eine Null dran. Im binären Zahlensystem entspricht der selbe Vorgang einer Multiplikation mit Zwei! Umgekehrt bedeutet das Verschieben nach rechts eine Division durch zwei – jedenfalls bei geraden Zahlen, denn deren erstes (rechtes) Bit ist immer Null! Bei ungeraden Zahlen muss man sich noch etwas einfällen lassen, was man mit der Eins im ersten Bit macht, die beim Verschieben quasi herunterfällt!

Ein weiterer wichtiger Befehl ist das Erhöhen oder Erniedrigen der Zahl in einem Register um Eins. Du wirst denken: „Um plus oder minus Eins zu rechnen, brauche ich doch keinen Computer!“ Aber trotzdem kommt dieser Befehl bei Computern so häufig vor, dass es sich lohnt ihn separat zu verdrahten, weil der Befehl dann viel schneller abgearbeitet werden kann, als wenn man eine normale Addition mit Eins rechnen würde (unter anderem spart man dadurch das Kopieren der Eins aus dem Arbeitsspeicher in ein Register).

*Man nennt das Erhöhen einer ganzen Zahl um Eins auch **inkrementieren**. Das bedeutet so viel wie eine kleinste Einheit (Inkrement) weiterzählen.*

*Das Erniedrigen um Eins nennt man **dekrementieren**.*

Eine Gelegenheit, bei der der Computer das Inkrementieren ständig braucht, ist das Abarbeiten eines Programms:

Wir haben ja gesagt, dass das Programm im Arbeitsspeicher steht. Genau wie die Ein- und Ausgabedaten irgendwie zur CPU gelangen müssen, muss der Prozessor auch wissen, welchen der vielen Befehle im Arbeitsspeicher er als nächstes abarbeiten soll, bzw. welche der vielen Nullen und Einsen im Arbeitsspeicher überhaupt Befehle *sind*, denn da steht ja noch alles mögliche andere 'rum!

Dazu hat jeder Prozessor ein extra Register, in dem ein *Befehlszähler* steht. In diesem → Register steht die Nummer (man sagt auch *Adresse*) der Speicherzelle im Arbeitsspeicher, in der der nächste auszuführende Befehl steht.

Sehr oft folgen viele nacheinander auszuführende Befehle direkt aufeinander. Das bedeutet, dass der Befehlszähler jedes Mal, wenn ein Befehl fertig ist, um Eins erhöht – inkrementiert – werden muss, damit der Computer seinen nächsten Befehl lesen kann.

5 Was man einstöpseln kann

Nun sind wir doch ziemlich einfach von unseren Modellcomputer-Bildchen zu richtigen Computern gelangt, oder? Dann wollen wir uns jetzt auch richtige Computer einmal näher ansehen!

In diesem Kapitel geht es um all das, was zu einem Computer dazugehört und irgendwo – sei es auch nur in der Steckdose – eingestöpselt wird. Demnach geht es *nicht* um Programme. In der Sprache der Computerfachleute würde man sagen, es geht jetzt um Hardware, nicht um Software.

Neben den Dingen, über die wir im Zusammenhang mit unseren Modellcomputern gesprochen haben – wie dem Prozessor zum Beispiel, kommen nämlich noch ein paar weitere Sachen dazu, denn bisher kann der Computer in unserer Vorstellung seine Rechenergebnisse nur in den Arbeitsspeicher schreiben. Dort können wir Menschen sie aber nicht lesen! Wir brauchen also irgend etwas, das uns die Ergebnisse in einer für Menschen verständlichen Form darstellt und vielleicht sogar dauerhaft auf ein Stück Papier zaubert.

5.1 Zentraleinheit

Zentraleinheit ist ein selten benutztes Wort! Meistens sagen wir dazu Computer oder Rechner. Das ist die Kiste, in der wirklich gerechnet wird, in der also die → CPU drin sitzt und wo die ganzen Kabel rein und raus gehen. Und warum machen wir um die CPU eine Kiste? Natürlich weil da noch mehr drin sitzt: Wir wissen doch schon aus Kapitel 4.3, dass der Prozessor den Arbeitsspeicher braucht, in dem seine Programme und Daten stehen. Der befindet sich auch in der Zentraleinheit; aber es kommt noch mehr:

5.1.1 Netzteil



Abbildung 20: Netzteil mit Schalter, Stecker für das Netzkabel und Lufteinlass

Die vielen Schalter im Arbeitsspeicher und im Prozessor, die wir Transistoren getauft haben, werden mit Strom betrieben. Das war ja gerade das Tolle daran, weil das so unheimlich schnell geht! Der Strom muss aber irgendwoher kommen! Aus der Steckdose, klar! Die Spannung an der Steckdose (230Volt) ist für so empfindliche Dinge wie einen Arbeitsspeicher oder Pro-

zessor aber viel zu stark! Sie brauchen zum Beispiel nur 5Volt. Deshalb sitzt in der Zentraleinheit ein sogenanntes *Netzteil*, ein Teil also, das unseren Computer mit dem Stromnetz verbindet, und das den Steckdosenstrom für unsere Zwecke umformt. Ein solcher Umformer heißt auch *Transformator*.

Abbildung 20 zeigt dir ein solches Netzteil. Du siehst auf dem Bild auch die Lüftungsschlitze, von denen ganz zu Anfang in Kapitel 2 schon die Rede war.

Wenn du später mal daran denkst, irgendwelche Erweiterungen in deinen Rechner einzubauen, musst du dich immer vergewissern, dass das Netzteil auch in der Lage ist, den für dieses zusätzliche Teil erforderlichen Strom zu liefern.

5.1.2 Mainboard

Alle wichtigen Komponenten der Zentraleinheit, vor allem also der Prozessor, sitzen auf einer großen Platine; weil Amerikaner eine bildliche Sprache lieben, heißt sie auch *Motherboard*, was so viel heißt wie Mutterplatine. Die Speicherbausteine aber auch Erweiterungen, zu denen ich weiter unten noch komme, sitzen wie Kinderplatinen in sogenannten *Steckplätzen* auf dieser Mutterplatine. Manche Menschen nennen das Ding auch *Mainboard*, also Hauptplatine.

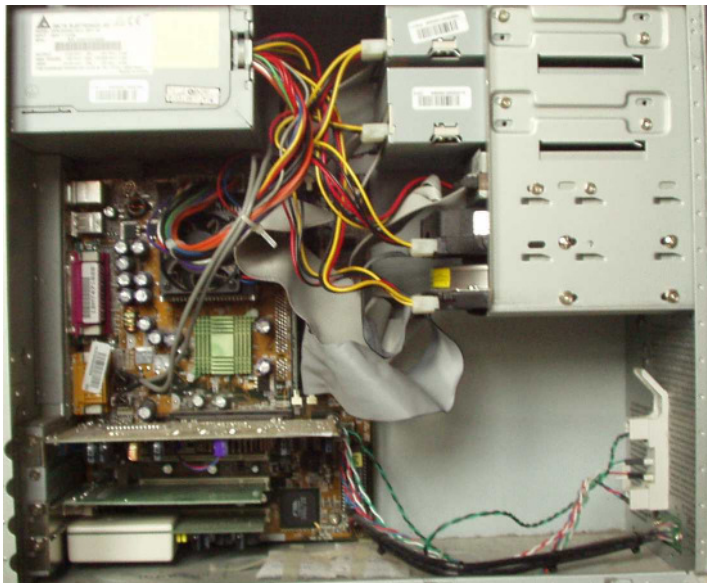


Abbildung 21: PC von innen

Abbildung 21 zeigt dir einen PC von innen: Links oben erkennst du das Netzteil wieder; da kommen die ganzen Kabel raus, die die anderen Teile mit Strom versorgen.

Rechts oben sieht man mehrere → Laufwerke (von oben nach unten: CD, DVD, 3,5“-Diskette, zwei Festplatten). Und das braune Teil links ist das Motherboard. Drei Kinderplatinen oder besser *Steckkarten* sitzen auch drauf (unten links).

Wenn man sich einen neuen Rechner kauft, kann es durchaus auch wichtig sein, sich zu überlegen, wie viele Erweiterungen im Laufe seiner Lebensdauer wohl nötig sein werden, und darauf zu achten, dass das Mainboard auch die entsprechenden Möglichkeiten bietet.

Das Mainboard beinhaltet Schaltkreise, die die Tastatur- und Mauseingaben verarbeiten. Es gibt einen Baustein der eine Uhr beinhaltet und deinem Rechner ermöglicht, die Zeit anzuzeigen oder in Programmen zu verwenden. Unter anderem wegen dieser Uhr, sitzt auch immer eine Batterie mit auf der Hauptplatine, denn die Uhr muss ja auch dann weiterlaufen, wenn der Rechner ausgeschaltet ist und das → Netzteil keinen Strom liefert!

Manchmal sind sogar schon die → Grafikkarte und/oder die → Soundkarte auf dem Mainboard enthalten. Die Sorgen dafür, dass die Nullen und Einsen im Arbeitsspeicher auch wirklich in ein Bild oder in Ton umgewandelt werden.

Eine letzte wichtige Komponente der Hauptplatine möchte ich noch erwähnen: Das *BIOS*.

BIOS *steht für Basic Input Output System und bedeutet „Grundlegendes Ein- und Ausgabe System“.*

Im Kapitel 6.1 wirst du lesen, was alles passiert, wenn der Computer eingeschaltet wird. Unter anderem muss irgendwo ein Programm stehen, das dem Computer sagt, wie groß sein Arbeitsspeicher ist (damit er nur Speicherzellen verwendet, die es auch wirklich gibt), welche Geräte angeschlossen sind und wo das → Betriebssystem steht, das die Befehle für den Betrieb des Computers enthält.

Das BIOS-Programm steht in einem kleinen Speicher, der seinen Inhalt auch dann behält, wenn er nicht mehr am Strom angeschlossen ist, auf den man aber nicht so ohne weiteres schreiben kann.

*Einen solchen Speicher, der nur gelesen werden kann, nennt man **ROM**. Das steht für **Read Only Memory** (engl. nur lese Speicher; sprich: ried ounli memmore).*

Später haben sich dann dazu Varianten entwickelt, weil man das ROM in bestimmten Fällen eben doch löschen und überschreiben wollte. Daraus entstand das *EPROM*, Erasable Programmable ROM, also lösch- und programmierbares ROM. Diese Bausteine konnte man mit UV-Licht (ultraviolettes Licht, von dem man im Sommer auch den Sonnenbrand bekommt, das hier aber mit speziellen Lampen erzeugt wurde) löschen. Das war ziemlich umständlich und deshalb gab es irgendwann das *EEPROM*, Electrically Erasable Programmable ROM, also das elektrisch löschrare, programmierbare ROM.

5.1.3 Speicher

Über den → Arbeitsspeicher haben wir schon gesprochen. Der sitzt auch in einem oder mehreren Steckplätzen auf der Hauptplatine. Jedes Programm, das der Computer ausführt (inklusive des Betriebssystems) und alle Daten (Zahlen, Texte, Bilder, ...), die er dazu braucht, müssen im Arbeitsspeicher stehen, denn nur von dort kann der Kopierbefehl des Prozessors sie abholen und in die → Register kopieren.

Der Arbeitsspeicher ist eine tolle Erfindung, wie wir oben schon gesehen haben: Wir bzw. der Computer kann an jeder beliebigen Stelle des Speichers lesen, was dort steht (und muss sich natürlich immer merken, wo was steht!). Das Lesen und Schreiben aus dem bzw. in den Speicher geht auch noch sagenhaft schnell!

Der Arbeitsspeicher hat eigentlich nur zwei „Schönheitsfehler“:

- 1.) Er ist ziemlich teuer!
- 2.) Er funktioniert nur, wenn er mit Strom versorgt wird! Ist der Strom einmal ausgefallen, hat der Arbeitsspeicher seinen gesamten Inhalt „vergessen“!

Das sind die Gründe, warum es auch noch andere Möglichkeiten geben muss, Daten und Programme zu speichern und über die wird in den nächsten Abschnitten berichtet.

5.1.3.1 Disketten und Festplatten

Im Kapitel 3.1 hatten wir uns Gedanken gemacht, wie man die Nullen und Einsen, von denen nun schon so oft die Rede war, im Computer darstellen kann und sind auf Lampen und Schalter gekommen. Die Lampe haben wir später wieder verworfen und sind schließlich bei den Millionen von Schaltern gelandet, aus denen der Prozessor und der Arbeitsspeicher nun bestehen. Wir suchen jetzt also eine weitere Möglichkeit, Nullen und Einsen darzustellen und zwar eine, die ohne Strom auskommt, oder die wenigstens ihre jeweilige Stellung (Null oder Eins) auch ohne Stromzufuhr behält. Natürlich wollen wir die Information in diesem Speicher auch wieder in elektrische Signale umwandeln können, damit wir unsere Schalter im Arbeitsspeicher wieder bewegen können!

Weißt du, wie elektrischer Strom erzeugt wird? Er kommt zum Beispiel aus Batterien, das sind chemische Prozesse; es gibt Solarzellen, die aus Licht Strom machen, okay, aber der weitaus größte Teil des Stromes, den wir jeden Tag benutzen, kommt doch aus der Steckdose! Die Steckdose hängt an einem Kabel, das durch die Wände deines Hauses läuft, und über einen Sicherungskasten, mehrere Umspannwerke und viele, viele Kilometer Hochspannungsleitung mit einem Kraftwerk verbunden ist. Dort wird der Strom erzeugt! Es gibt Kohle-, Gas-, Öl-, Atom- und Wasserkraftwerke. Und trotz der vielen unterschiedlichen Namen erzeugen sie den Strom letztlich auf die selbe Weise: Sie drehen einen

Generator. Ein Generator arbeitet genauso wie der Dynamo an deinem Fahrrad: Es werden Drahtspulen durch das Feld eines *Magneten* bewegt. Dadurch wird in den Spulen ein Strom erzeugt (man sagt auch *induziert*)! Damit das kontinuierlich, also fortlaufend, passiert, dreht man die Spulen (oder den Magneten).

Bei einem Elektromotor macht man übrigens genau das Umgekehrte: Man lässt Strom durch eine Drahtspule fließen, dadurch erzeugt diese ein Magnetfeld, das sich von einem anderen Magneten abstößt.

Also so viel zum Zusammenhang zwischen Strom und Magnetismus. Aber ein Magnet hat doch zwei unterschiedliche Seiten, zwei *Pole*, einen Nord- und einen Südpol sagt man, weil Magnete sich immer in Nord-Süd-Richtung ausrichten (wenn sie können), die eine – und zwar immer die selbe – Seite zeigt nach Norden, die andere nach Süden, weil auch die Erde ein Magnet ist, und sich unterschiedliche Pole von Magneten gegenseitig anziehen und gleiche Pole abstoßen.

Da haben wir doch etwas, womit wir unsere Nullen und Einsen darstellen können! Wir nehmen für jedes Bit einen kleinen Magneten. Wenn er Null anzeigen soll, ist der Nordpol oben, wenn er Eins anzeigen soll, ist der Südpol oben.

Wenn wir lesen wollen, was unsere Magneten darstellen, bewegen wir eine Spule über sie hinweg und wenn wir die Bits umschalten wollen, wenn wir also etwas in unseren Speicher hineinschreiben wollen, schicken wir einen Strom durch die Spule, der die Magnete umdreht! Dieser Vorgang entspricht dann dem Ein- oder Ausschalten eines Schalters in unseren Modellcomputern.

Abbildung 22 zeigt dir eine einfache Darstellung, wie wir mit Hilfe von Magneten unseren Speicher realisieren können:

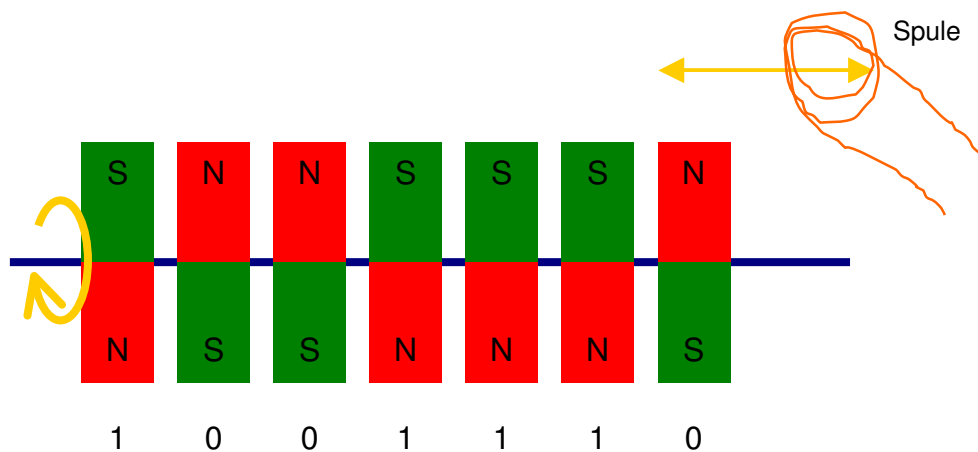


Abbildung 22: Speicherung der Zahl 142 mit Hilfe von Magneten

Du siehst acht Magnete, die einzeln drehbar (der gelbe Pfeil links soll das andeuten) auf einer Achse (der blau gemalten Stange) angeordnet sind. Dadurch kann entweder der Nordpol oder der Südpol nach oben zeigen. Rechts im Bild siehst du eine kleine Drahtspule, die wir entlang der Reihe der Magnete bewegen können. Wenn wir lesen wollen, wie die Magnete stehen, benutzen wir den Strom, der beim Vorbeifahren erzeugt wird, um unsere Schalter im Arbeitsspeicher entsprechend zu stellen. Wenn wir die Magnete einstellen wollen, schicken wir einen Strom entweder links herum oder rechts herum durch die Spule, jeweils wenn sie sich über dem Magneten für das betreffende Bit befindet und der Magnet wird sich dementsprechend auf der Achse umdrehen.

Wenn wir keinen Strom mehr haben, können wir den Speicher zwar nicht lesen (weil wir die Spule nicht bewegen können), aber zumindest bleiben die Magnete stehen wie sie sind, und sobald wieder Strom da ist, können wir den alten Inhalt wieder lesen.

Von diesem Modell zur Wirklichkeit braucht es noch folgende Schritte: Vor allem müssen die Magnete winzig klein sein, weil wir ja möglichst viel Information auf möglichst wenig Raum unterbringen wollen. Um das zu lösen nimmt man keine Magnete, die sich auf einer Achse drehen, sondern man beschichtet einen Träger – ein Band oder eine Scheibe – mit Eisenpulver, das zuerst nicht magnetisch ist, sondern erst durch den Strom der durch die Spule fließt zum Magneten wird. Wenn der Magnet sich dann „umdrehen“ soll, weil dieses Bit jetzt einen anderen Inhalt bekommen soll, wird das Eisen ummagnetisiert, das heißt es bewegt sich nichts, sondern man stellt quasi einen neuen, anders herum gepolten Magneten her.

Dann hat man sich noch überlegt, dass es ja viel einfacher ist, das Eisenpulver an der Spule vorbei zu bewegen als umgekehrt, denn an der Spule hängt ja ein Kabel!

Wenn du Musik- oder Hörspielkassetten hast, kennst du das ganze schon! Dort benutzt man das selbe Material, um die Töne zu speichern, allerdings nicht digital.

Früher hatte man wirklich Bänder aus ähnlichem Material wie Tonbänder, auf denen die Programme und Daten auf die beschriebene Weise gespeichert wurden! Fällt dir ein Unterschied zu unserem Arbeitsspeicher auf? Nimm an, das Band ist aufgewickelt und du willst das letzte darauf gespeicherte Bit lesen. Du musst erst das ganze Band abwickeln, bevor die richtige Stelle an deiner Spule vorbeikommt! Nichts mehr mit wahlfreiem Zugriff!

Deshalb ist man schon bald dazu übergegangen, Scheiben mit dem Eisenpulver zu beschichten. Diese Scheiben lässt man drehen und die Spule bewegt man in einer Richtung darüber, wie es Abbildung 23 zeigt:

Das braune sind mit Eisenpulver beschichtete Scheiben, die sich drehen. In der Vergrößerung sieht man die Spule, die mit Hilfe des Arms bewegt werden kann. Auf diese Weise kann jede Stelle der magnetisierbaren Fläche recht schnell erreicht werden. Allerdings können wir immer noch nicht unmittelbar ein bestimmtes Byte ansprechen; im schlimmsten Fall müssen wir eine komplette Bewegung der Spule plus eine Umdrehung der Scheibe warten, bis das Byte, das wir lesen wollen, vorbeikommt.

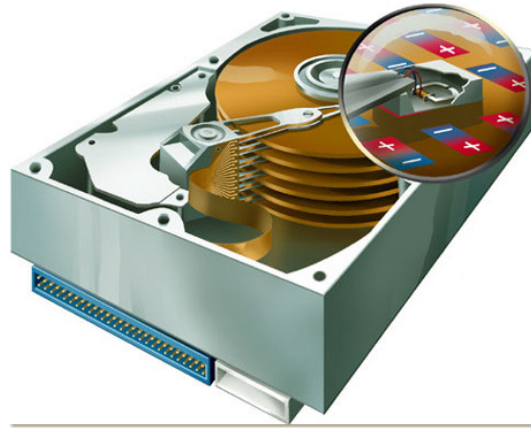


Abbildung 23: Magnetspeicher in Scheibenform

Das Bild zeigt gleich mehrere Scheiben übereinander, wie man es in *Festplatten*, die sich nicht herausnehmen lassen, macht, um den Speicherplatz zu vergrößern.

Scheiben zum Herausnehmen gibt es auch; die nennt man *Diskette*. Sie werden traditionell immer nach ihrem Durchmesser, gemessen in Zoll benannt. (Ein *Zoll*, geschrieben als 1" oder englisch *inch* entspricht 2,54cm.) Da die Scheiben aus einem flexiblen Material sind, werden sie auch manchmal als *Floppy-Disk* bezeichnet. Begonnen hat es mit 8,5"-Disketten, später waren 5,25"-Disketten gebräuchlich und heute noch übrig geblieben sind 3,5"-Disketten; sie können bis zu 1,44MB speichern (→ Megabyte). Sonderformen von Disketten, die sich wegen ihres Preises nicht durchgesetzt haben, kommen auf bis zu 100MB, wofür man aber auch spezielle Laufwerke braucht.

Laufwerk nennt man bei den herausnehmbaren Disketten, den Antrieb, den Lesekopf (die Spule) und all die anderen Dinge, die nicht mit herausgenommen werden. Solche Laufwerke sind in der Zentraleinheit eingebaut und haben einen Schlitz, in den man die Diskette hineinstecken kann, und einen Knopf, auf den man drücken kann, damit sie wieder herauskommt.

Festplatten sind auch in der Zentraleinheit eingebaut (s. Abbildung 21!); man sieht sie in der Regel von außen nicht, weil man ja auch nichts daran machen muss. Meist gibt es nur ein Lämpchen, das immer dann leuchtet, wenn die Festplatte benutzt wird. Häufig kann man die Festplatte hören!

Das Drehen der Platte hört man nicht immer, weil vielleicht der Lüfter des Netzteils zu laut ist, aber man kann fast immer ein leises Klickern wahrnehmen, wenn von der Festplatte gelesen oder auf sie geschrieben wird, also immer dann, wenn das Festplattenlämpchen leuchtet. Was man hört ist der Spulenkopf, der sich hin und her bewegt.

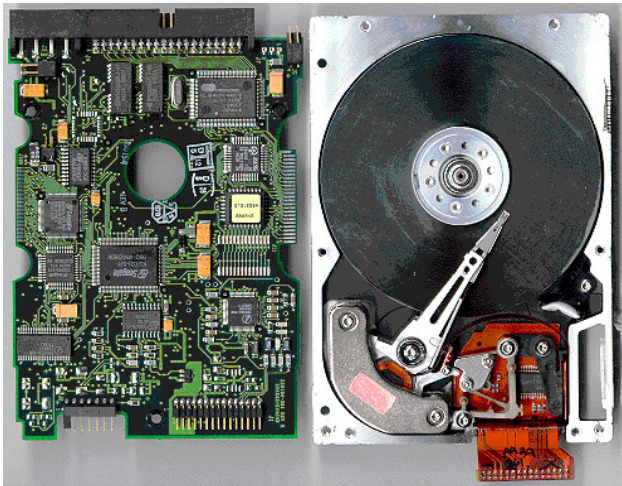


Abbildung 24: Geöffnete Festplatte mit zugehöriger Elektronik

Festplatten speichern in einer kleinen Kiste, halb so groß wie ein Taschenbuch zum Beispiel 120GB! (→ Gigabyte).

Abbildung 24 zeigt dir eine geöffnete Festplatte. Normalerweise befinden sie sich in einem luftdichten Gehäuse, weil sie wegen des kurzen Abstandes zwischen der Spule und der Magnetplatte sehr staubempfindlich sind.

5.1.3.2 Silberscheiben: CD und DVD

Viel vertrauter als diese magnetischen Scheiben wird dir eine andere Art von Speicher sein, die man heute auch meistens in der Zentraleinheit finden kann, die *CD*. Genaugenommen findet man natürlich erst mal nur das Laufwerk für die CD in der Zentraleinheit und die CD selbst muss man wie eine Diskette erst dort hinein tun. Das geschieht meistens nicht über einen Schlitz sondern über eine Schublade, die sich mit einem Knopf öffnen und schließen lässt. Das kennst du bestimmt schon von einem CD-Spieler für Musik-CD's.

CD heißt *Compact Disc* (englisch für Kompakte Scheibe), und damit soll ausgesagt werden, dass auf eine kompakte, kleine Scheibe ziemlich viele Daten passen, nämlich ca. 700MB.

Bei der CD werden keine Schalter oder Magnete verwendet, um Nullen und Einsen darzustellen, sondern Löcher! Ein Loch ist entweder auf oder zu; dementsprechend wird beim Lesen der CD ein ganz dünner Lichtstrahl entweder zurückgeworfen (man sagt auch *reflektiert*) oder nicht.

Der Scheibe ist es egal, ob die Löcher auf ihr nun Musik bedeuten oder irgend etwas anderes, und deshalb konnte man CDs problemlos auch für Computer gebrauchen, obwohl sie ursprünglich mal für die → digitale Speicherung von Musik entwickelt wurden.

Die Laufwerke konnten CDs ursprünglich nur lesen. Inzwischen kann fast jeder Computer auch CDs beschreiben. Diesen Vorgang nennt man *brennen*.

Man muss bei CDs verschiedene Typen unterscheiden: Der ursprüngliche Typ, den man als Anwender nur lesen kann, und der auch heute noch verwendet wird, wenn man zum Beispiel Software kauft. Die steht meistens auf einer sol-

chen *CD-ROM*. Die Abkürzung ROM bedeutet hier das selbe wie im Kasten auf Seite 58 erklärt.

Dann gibt es CDs, die man einmal beschreiben kann; die heißen *CD-R* für *CD-recordable*, was das englische Wort für „beschreibbar“ ist. Genau genommen kann man eine solche CD auch mehrmals beschreiben, nämlich in mehreren Abschnitten so lange bis sie voll ist. Man kann aber einmal aufgebrannte Daten nicht wieder löschen oder gar durch andere Daten ersetzen.

Das kann man bei den sogenannten *CD-RW*, RW steht für *rewritable*, das englische Wort für „wiederbeschreibbar“.

Auf CD-Brennern, also CD-Laufwerken, die auch CDs beschreiben können, stehen immer drei Zahlen wie zum Beispiel „48x, 32x, 16x“. Damit soll ausgesagt werden, wie schnell das Laufwerk ist, und die Zahlen beziehen sich auf die Geschwindigkeit der ersten Laufwerke; die konnten pro Sekunde 1.200kB (→ Kilobyte) lesen. 48x bedeutet also, dass das Laufwerk 48 mal schneller ist und demnach 57.600kB pro Sekunde lesen kann. Die zweite Zahl bezeichnet nach dem gleichen Schema, wie schnell das Laufwerk auf eine CD-R schreiben kann und die dritte, wie schnell es auf eine CD-RW schreiben kann. Auf CD-Rohlingen – so nennt man noch nicht beschriebene CDs – steht immer drauf, mit welcher Geschwindigkeit man sie maximal beschreiben kann. Das ganze ist eigentlich gar nicht so wichtig. Man sollte nur darauf achten, dass man nicht schneller zu schreiben versucht, als es der Rohling zulässt. Das kann man in dem Programm, das man zum Brennen von CDs benutzt (zum Beispiel Nero Burning ROM) einstellen.

CDs auf denen Musik drauf ist heißen Audio-CDs. Audio ist Latein (ausnahmsweise mal kein Englisch!) und bedeutet „ich höre“. Normalerweise kann man die an einem Computer auch hören, man braucht ja nur ein Programm, das aus den Nullen und Einsen auf der CD wieder die richtigen Töne macht – und natürlich Lautsprecher und eine → Soundkarte, um sie zu hören.

Dann gibt es noch eine Weiterentwicklung der CD, die sogenannte DVD. Das ist wieder eine englische Abkürzung und heißt ausgeschrieben „Digital Versatile Disk“, was so viel bedeutet wie digitale, vielseitige Scheibe.

DVDs arbeiten nach dem selben Prinzip wie CDs, nur dass sie viel mehr Daten speichern können. Ursprünglich wurden sie entwickelt, um Filme darauf zu speichern. Normalerweise können DVD-Laufwerke auch CDs lesen, aber nicht umgekehrt. Auch für DVD's gibt es seit einiger Zeit Brenner zu kaufen.

5.1.3.3 Elektronische Speicher

Inzwischen hat man es auch das ursprüngliche Problem unseres Arbeitsspeichers, dass er nämlich alles vergisst, wenn der Strom weg ist, gelöst! Es gibt

Speicherbausteine, deren eingebaute Schalter ihre Stellung behalten. Man findet sie in den verschiedensten Typen von Speicherkarten, wie sie meist in → Digitalkameras zum Einsatz kommen, oder in sogenannten USB-Sticks; das sind kleine Stöpsel, die man an die → USB-Schnittstelle (s. dazu Kapitel 5.2.2) stöpseln kann, und auf denen man dann etwas speichern kann wie auf einer Diskette oder Festplatte. Gängige Größen dafür sind heute 256MB (→ Megabyte) bis 1GB.

Warum hat man nun so viele unterschiedliche Typen von Speichermedien²? Zum einen hat das die Zeit so mit sich gebracht, weil immer neue Systeme entwickelt wurden, zum anderen haben die verschiedenen Typen unterschiedliche Vor- und Nachteile. Ein grundsätzlicher Unterschied ist immer, ob man das Medium, also das worauf die eigentlichen Informationen – die Nullen und Einsen – stehen, wechseln kann oder nicht. Falls ja, nennt man sie *Wechseldatenträger*. Ein zweites Kriterium ist immer die Geschwindigkeit, mit der gelesen und geschrieben werden kann und nicht zuletzt spielen auch die Kosten eine Rolle, und zwar die für das Laufwerk und die für die Medien.

5.1.3.4 Ordnung halten im Speicher

Dieses Kapitel passt eigentlich gar nicht unter die Hauptüberschrift, denn es geht nicht um Hardware sondern um etwas Organisatorisches, um *Ordnung*! Erwachsene predigen ihren Kindern meist, wie wichtig Ordnung sei. Darüber sind die Generationen seit Menschengedenken uneinig und ich möchte mich hier gar nicht weiter darüber auslassen! Im Zusammenhang mit Computern und seinen verschiedenen Speichern, von denen zuvor die Rede war, ist die Ordnung jedenfalls so wichtig, dass ich sogar meine Kapitelsystematik dafür durchbreche! Es ist mir sehr wichtig, dass du die Ordnung in den Speichern eines Computers bereits an dieser Stelle verstehst!

Du solltest dieses Kapitel unbedingt lesen!

In diesem Kapitel 5.1.3 hast du bisher von verschiedenen Speichern gehört, und alle sind in der Lage, große Mengen an Daten über lange Zeit (sie behalten

² Ein Medium nennt man den Träger oder Vermittler einer Information. Das kann eine Zeitung, das Fernsehen oder ein Stück Papier sein, aber eben auch eine CD, eine DVD oder ein Tonband.

Das Wort kommt aus dem Lateinischen und bedeutet „Mitte“. Das Medium steht ja in der Mitte zwischen der Informationsquelle und dir, dem Empfänger der Information.

die Daten nämlich auch, wenn sie nicht mit Strom versorgt werden) zu speichern.

Wenn du schon einmal etwas weggelegt hast, und es nach längerer Zeit wieder gebraucht hast, wirst du wissen, dass man es meist nur dann wiederfindet, wenn man die Sache nach einem halbwegs logischen Verfahren weggelegt hat. Wenn du ein Foto statt in ein Fotoalbum in die Gefriertruhe gelegt hast, wirst du es nach drei Jahren nicht mehr wiederfinden, wenn du es suchst! (Abgesehen davon, dass deine Mutter es sicher keine drei Jahre in der Gefriertruhe geduldet hätte!)

Eine weitere alltägliche Erfahrung, die du schon gemacht hast, ist die: Es ist umso schwieriger etwas zu finden, je mehr Orte es gibt, an denen man suchen kann, oder je mehr Teile man betrachten muss, um das Richtige zu finden: In einer LEGO[®]-Kiste findest du ein bestimmtes Teil schneller, wenn weniger Teile drin sind, und du findest es auch schneller, wenn du nur eine LEGO[®]-Kiste hast statt zehn.

Andererseits können sich die zehn LEGO[®]-Kisten dann bezahlt machen, wenn du die Steine darin einsortiert hast, und du deshalb genau weißt, in *welcher* Kiste du suchen musst.

Diese Beispiele sollen dir zeigen, dass man auf den verschiedenen Speichermedien Ordnung halten muss, denn es passt unheimlich viel rein und bleibt manchmal auch sehr lange drin!

Zum Glück wirst du beim Ordnung halten prima unterstützt, und es ist kaum schwieriger als das Einsortieren von LEGO[®]-Steinen in verschiedene LEGO-Kisten.

5.1.3.4.1 Namen geben

Neben dieser Notwendigkeit für Ordnung kommt noch ein ganz logischer Grund hinzu:

Bitte rufe dir noch einmal den Unterschied zwischen Arbeitsspeicher (dem RAM mit wahlfreiem Zugriff auf jede beliebige Speicherzelle) und den anderen Medien (Festplatte, Diskette usw.) in Erinnerung: Beim Arbeitsspeicher können wir jedes einzelne → Byte lesen, indem wir nur die → Adresse, das ist einfach die Nummer der Speicherzelle angeben. Wie aber sagen wir beispielsweise einer Festplatte, *welche* der vielen Nullen und Einsen, die auf ihr stehen, wir denn nun lesen wollen?

In Kapitel 3 haben wir gesehen, wie wir Zahlen, Texte, Bilder und Programme mit Nullen und Einsen darstellen können. Diese → Bits sollen natürlich auch auf der Festplatte stehen können. Irgendwie müssen wir doch bestimmen können, welche Bits zu einem bestimmten Bild und welche zu einem bestimmten Text oder Programm gehören. Wie geht das?

Ganz einfach: Wir geben diesen Dingen *Namen!* Den Rest erledigt der Computer für uns! Du könntest beispielsweise das Bild in Abbildung 11 auf Seite 32 „Selbstbildnis“ nennen und es unter diesem Namen auf der Festplatte speichern. Der Computer schreibt diesen Namen auf die Festplatte und dahinter schreibt er hin, wo auf der Festplatte die Bytes stehen, die zu diesem Bild gehören. Wenn du dann irgendwann Jahre später das Bild ansehen möchtest, kannst du deinem Grafikprogramm den Befehl geben, das Bild „Selbstbildnis“ von der Festplatte zu lesen und in den Arbeitsspeicher zu → laden. Der Computer sieht dann auf der Festplatte, an der Stelle wo die ganzen Namen stehen, nach, ob ein Bild mit diesem Namen gespeichert ist, und welche Bytes auf der Festplatte zu diesem Bild gehören. Dann kopiert er sie in den Arbeitsspeicher. Sobald das Bild dort steht, kann das Grafikprogramm damit arbeiten.

Wenn du nicht mehr weißt, wie das Bild heißt, das du sehen möchtest, kannst du dir auch alle Namen anzeigen lassen, die auf der Festplatte stehen. Dann kommt man meistens drauf, welches der richtige ist.

*Alle Bytes, die zu einem Bild, einem Text, einem Programm oder einer anderen Einheit gehören, fasst man unter einem Namen zusammen. Das was unter einem Namen zusammengefasst ist, nennt man eine **Datei**; deren Namen nennt man den **Dateinamen**. Dateinamen müssen eindeutig sein, das heißt sie dürfen nur einmal vorkommen!*

*Die Art und Weise, wie die Daten innerhalb der Datei angeordnet sind, nennt man das **Dateiformat**.*

*Das englische Wort für Datei ist **File** (sprich: feil).*

Der Bereich auf einer Festplatte oder Diskette, in dem die Dateinamen und ihre Speicherorte auf dem Speichermedium stehen, heißt *File Allocation Table*, abgekürzt FAT, was so viel heißt wie Datei-Zuordnungs-Tabelle.

Wenn du einmal Gelegenheit dazu hast, achte einmal darauf, dass das Speichern einer großen Datei auf einer Diskette oder Festplatte einige Sekunden dauern kann, das Löschen der selben Datei, aber sehr viel schneller geht. Hast du eine Erklärung dafür?

Wenn die Datei gespeichert wird, werden alle dazugehörigen Bytes auf die Platte geschrieben und dann der Name in die Zuordnungstabelle. Dafür müssen genau so viele kleine Magnete in die richtige Lage gebracht werden, wie die Datei Bits enthält.

Beim Löschen der Datei wird nur der Name aus der Zuordnungstabelle ge-

löscht, der Rest bleibt wie er ist! Das geht viel schneller! Da die Bereiche der Festplatte, wo die große Datei stand (bzw. nach dem Löschen zunächst auch immer noch steht), in der Zuordnungstabelle nicht mehr auftauchen, wird der Computer beim nächsten Speichern einer neuen Datei, diese dorthin schreiben.

5.1.3.4.2 Verschiedene Sorten

Während der Entwicklung der Computer hat es sich schnell herausgestellt, dass es ganz sinnvoll ist, einen Teil des Dateinamens dafür zu verwenden, die Art der Datei anzuzeigen, also ob es sich um einen Text, ein Bild, ein Programm oder etwas anderes handelt. Auf diese Weise hat man eine einfache und bis heute benutzte Möglichkeit geschaffen, auch dem Computer mitzuteilen, um welche Art Daten und welches Dateiformat es sich handelt, was meist auch dazu führt, dass der Computer weiß, was er mit diesen Daten machen soll.

Früher musste man mit Speicherplatz auch auf Festplatten geizen und so wurde damals festgelegt, dass ein Dateiname aus acht Zeichen bestehen darf, an den – getrennt durch einen Punkt – weitere drei Zeichen angehängt wurden, die den Dateityp anzeigten. Die Beschränkung auf acht Zeichen ist inzwischen entfallen bzw. auf 255 Zeichen erweitert worden, aber die drei Zeichen für den Dateityp gibt es immer noch, obwohl es auch dafür keine technische Beschränkung mehr gibt. Manche Dateityp-Endungen sind deshalb auch vierstellig. Tabelle 9 zeigt dir einige Beispiele:

Endung	Dateityp
.exe	ausführbare Programme (kommt von englisch <i>executable</i> = ausführbar)
.com	kleine Programme, die zum alten → Betriebssystem DOS gehören. com für englisch <i>command</i> = Befehl; z.B. edit.com
.tmp	temporäre Datei. Manche Programme (zum Beispiel Word) legen solche Dateien an, um irgendwelche Sachen zwischenspeichern. Normalerweise verschwinden diese Dateien wieder, wenn man das Programm beendet, aber wenn etwas schiefgegangen ist, bleiben sie manchmal auch übrig. Wenn kein Anwendungsprogramm gestartet ist, kann man sie löschen.
.sys	Systemdateien. Darin stehen Informationen, die das → Betriebssystem beim Starten braucht.
.bat	von englisch <i>batch</i> = Stapel. In solchen Textdateien werden Befehle an das → Betriebssystem gespeichert. Man kann diese Dateien dann ausführen wie ein Programm und der Computer verhält sich so, als würde man alle die enthaltenen Befehle der Reihe nach eintippen. Das ist ganz praktisch, wenn man bestimmte Aufgaben regelmäßig machen muss.

Endung	Dateityp
.lnk	von englisch <i>link</i> = Verbindung, Verknüpfung. Diese Dateisorte enthält nur einen Hinweis darauf, wo eine andere Datei steht. Sie ist sozusagen ein Wegweiser. Zum Beispiel hinter der Liste von Programmen im → Startmenü oder den Symbolen auf der Windowsoberfläche verbergen sich .lnk-Dateien.
.txt	Textdateien mit Text im → ASCII-Code
.doc	von englisch <i>document</i> = Dokument, Text aus dem Textverarbeitungsprogramm Microsoft Word
.rtf	steht für <i>Rich Text Format</i> und ist ein einfaches Textformat, bei dem auch Formatierungen (Fett- oder Kursivdruck und solche Dinge) mit ASCII-Zeichen gespeichert werden.
.htm .html	steht für Hypertext Markup Language. Auch das sind ASCII-Text-Dateien, die Befehle an das Darstellungsprogramm enthalten. In diesem Format werden Internetseiten geschrieben. (Siehe Kapitel 8.1)
.bmp	von englisch → <i>bitmap</i> , Bilddatei bei der das in Kapitel 3.5 beschriebene Verfahren zur Darstellung von Bildern benutzt wird.
.gif	von englisch <i>graphics interchange format</i> , anderes Bildformat
.jpg	noch ein Bildformat (manchmal auch .jpeg)
.zip	sogenanntes <i>Archiv</i> , das als eine einzelne Datei viele weitere Dateien beinhalten kann, das aber für die Speicherung weniger Bits benötigt. Man sagt auch die Daten werden komprimiert. Die Dateiendung kommt vom englischen Reißverschluss: Man macht sozusagen eine Dateitasche um die beinhalteten Dateien und schließt den Reißverschluss.

Tabelle 9: Beispiele für Dateitypen und ihre Dateinamensendungen

Es ist sehr hilfreich, wenn der Computer erkennen kann, um welche Art von Daten es sich in einer Datei handelt, denn dann muss der Benutzer nicht mehr jedes Mal sagen, was der Rechner mit der Datei tun soll. Eine Textdatei wirst du fast immer mit deinem Textverarbeitungsprogramm bearbeiten wollen und eine Bilddatei willst du dir ansehen oder mit deinem Grafikprogramm bearbeiten. Außerdem muss der Computer ja merken können, wenn eine Datei nicht irgendwelche Daten enthält, sondern ein Programm, das er ausführen soll (Endung .exe).

Aus diesem Grund gibt es im Betriebssystem eine Tabelle, in der alle Dateitypen aufgelistet sind, für die der Computer ein Programm hat, um sie zu bearbeiten. Du kannst dir diese Liste in folgendem Experiment einmal ansehen:

Experiment 3: Dateitypen

1. Starte deinen Rechner.
2. Klicke auf der Windows-Oberfläche auf „Arbeitsplatz“.

3. Wähle im Menü Extras „Ordneroptionen“.
4. Klicke auf die Registerkarte „Dateitypen“ und du erhältst folgendes Bild:

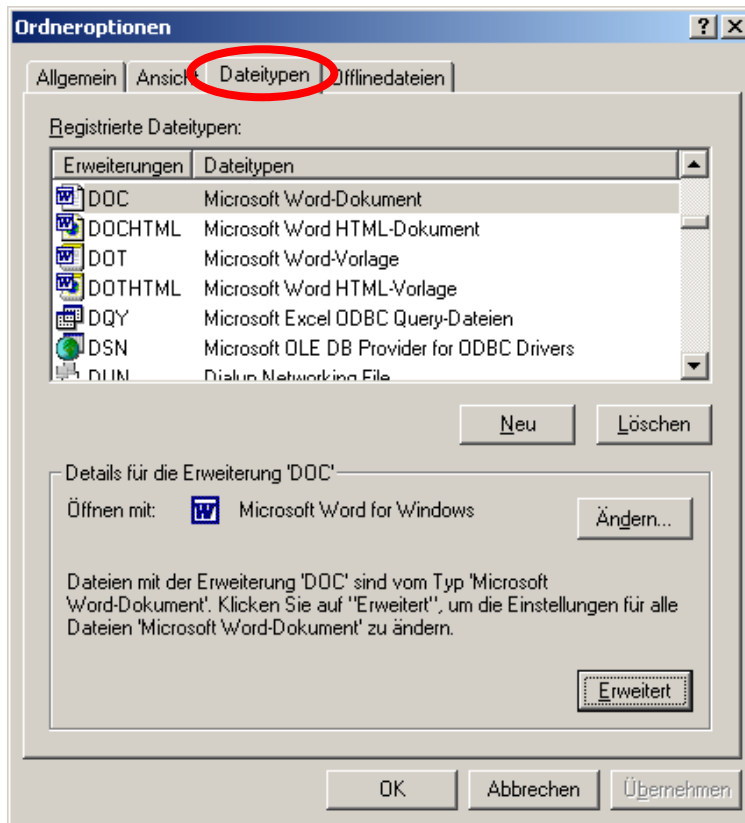


Abbildung 25: Ansicht der registrierten Dateitypen

In dieser Liste stehen alle Dateitypen, die dein Computer kennt. Sobald du auf eine Datei mit der entsprechenden Endung doppelklickst, wird der Computer das in dieser Liste genannte Programm und diese Datei in den Arbeitsspeicher laden, damit du die Datei bearbeiten kannst.

Du kannst die Einstellungen auch ändern. In der Regel ist das aber nicht notwendig.

5.1.3.4.3 Verschachtelte Kisten

Wir haben die vielen Bits eines Bildes oder Textes nun handhabbar gemacht, indem wir sie zu einer Datei zusammengefasst und dieser einen Namen gegeben haben.

Unsere Speicher sind aber so groß und wir brauchen so viele Dateien, dass auch das noch keine ausreichende Ordnung ermöglicht, mit der wir eine Datei, die wir brauchen, schnell finden können.

Darum machen wir mit den Dateien das selbe, was wir vorher mit den Nullen und Einsen gemacht haben: Wir fassen Sie zusammen und geben dieser Gruppe von Dateien wieder einen Namen.

Du kannst das wirklich mit einer Kiste LEGO®-Steine vergleichen: Die Dateien sind die LEGO®-Steine. Es gibt sie in verschiedenen Größen, Formen und Farben. Um dir das Wiederfinden zu erleichtern, könntest du sie zunächst nach Farbe sortieren. Du machst eine Kiste mit roten, eine mit gelben, eine mit blauen und eine mit schwarzen Steinen. In jeder dieser Kisten sortierst du dann nach Größe: Alle 1x1-Steine (die mit einem Noppen) in eine Kiste, alle 1x2-Steine in eine weitere usw. In jeder dieser Formen-Kisten kannst du dann noch zwei weitere Kisten haben, in denen die eine flache und die andere hohe Steine enthält. Nachfolgendes Bild soll dir das Beispiel veranschaulichen:

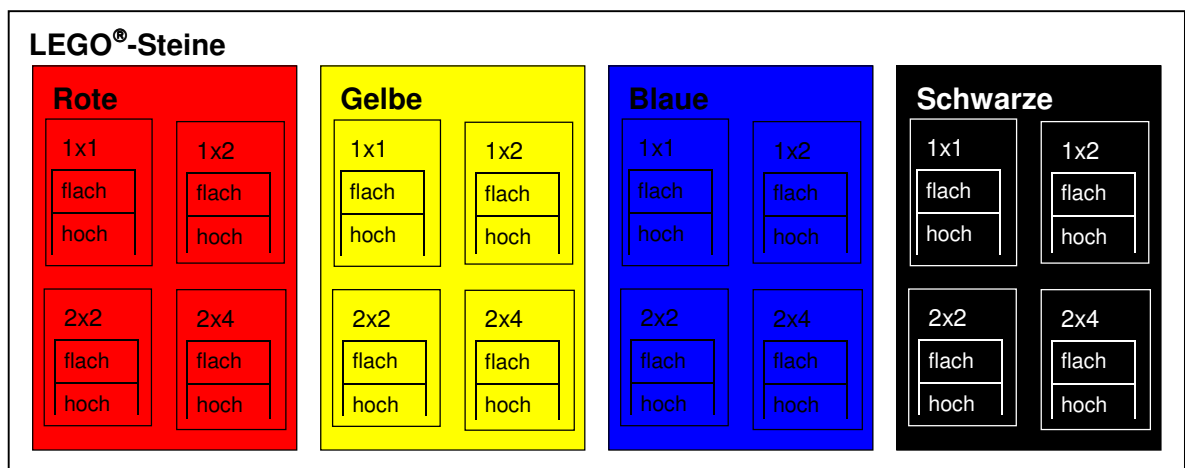


Abbildung 26: So könnte man LEGO®-Steine sortieren

Die Kisten beschriftest du natürlich alle fein säuberlich, was nichts anderes bedeutet, als ihnen Namen zu geben. Wenn du nun einen blauen, flachen 2x4-Stein suchst, weißt du sofort in welcher Kiste du ihn finden wirst! Natürlich könntest du dir auch eine beliebige andere Systematik ausdenken, nach der du die Steine sortierst! Du könntest zum Beispiel erst nach Größe und dann nach Farbe sortieren. Trotzdem würdest du jeden Stein immer sehr schnell finden können.

Bei den LEGO®-Steinen hat das Prinzip einen Haken: Woher bekommst du die ganzen Kisten? Und noch dazu welche, die alle ineinander passen und dann auch noch Platz für LEGO®-Steine haben?

Dieses Problem gibt es beim Computer nicht! Wenn du deinem Betriebssystem sagst, du möchtest eine weitere Kiste haben, um ein paar Dateien (oder auch weitere Kisten) hineinzutun, dann kriegst du die! Alles was der Rechner von dir noch verlangt, ist ein Name, den die Kiste tragen soll! Natürlich gibt es in Wirklichkeit doch eine Grenze für die Anzahl der Kisten, nämlich die Größe deiner

Festplatte, denn dein Computer muss sich den Namen der Kiste und welche Dateien darin enthalten sind natürlich irgendwo hinschreiben und wenn die Festplatte voll ist, kann er das nicht mehr! Diese Begrenzung hat aber keinerlei praktische Bedeutung!

*Beim Computer heißen die Kisten, in denen man Dateien zusammenfasst **Ordner oder Verzeichnis**. Im Englischen spricht man von einem **Directory** (sprich: deirektorie). Jedes Verzeichnis kann Dateien oder weitere Verzeichnisse enthalten. Verzeichnisse, die sich in einem Verzeichnis befinden, nennt man **Unterverzeichnisse**.*

*Die komplette Angabe von Laufwerk, Verzeichnis und allen Unterverzeichnissen nennt man **Pfad**. (In → Netzwerken kommt auch noch der Computernamenname dazu.)*

Wenn du dir in Abbildung 26 vorstellst, wir wollten bei den flachen 1x1-LEGO®-Steinen noch weitere Unterverzeichnisse einführen, zum Beispiel für runde und eckige Steine, dann wird diese Art der Darstellung ziemlich unübersichtlich. Da man bei Computern ja unheimlich viele Unterverzeichnisse anlegen kann, wählt man ein anderes Bild, nämlich das eines Baumes: Dateien sind Blätter und Verzeichnisse sind Äste bzw. Zweige. An jedem Ast können Blätter, aber auch weitere Äste sitzen. Das Laufwerk, also zum Beispiel die Festplatte, entspricht dann dem Stamm, und man nennt das auch das *Stammverzeichnis*. Auf Englisch nennt man es *Root* (sprich: ruut, = Wurzel).

Wenn man eine Verzeichnisstruktur als Text in einer Liste darstellen möchte, wird der Inhalt eines Verzeichnisses oft eingerückt, also zum Beispiel:

```
Stammverzeichnis (Laufwerk)
  Verzeichnis 1 im Stammverzeichnis
    Unterverzeichnis 1 in Verzeichnis 1
      Datei 1 in Unterverzeichnis 1
      Datei 2 in Unterverzeichnis 1
    Unterverzeichnis 2 in Verzeichnis 1
      Datei 1 in Verzeichnis 1
      Datei 2 in Verzeichnis 1
  Verzeichnis 2 im Stammverzeichnis
    Datei 1 in Verzeichnis 2
    Datei 2 in Verzeichnis 2
  Verzeichnis 3 im Stammverzeichnis
    Datei 1 im Stammverzeichnis
    Datei 2 im Stammverzeichnis
```


Damit hast du nun das Werkzeug, um in deinen riesigen Datenspeichern Ordnung halten zu können. Es liegt an dir, dieses Werkzeug sinnvoll zu benutzen! Verwende für deine Dateien und Verzeichnisse aussagekräftige Namen, die dir auch nach längerer Zeit noch sagen, worum es sich jeweils handelt. Wie das konkret funktioniert, erfährst du dann in Kapitel 6.2.5.

5.2 Schnittstellen

Kann man mit Computern auch Scherenschnitte machen? Oder was bedeutet Schnittstellen?

Eine Schnittstelle ist eine Stelle, an der man die Teile eines Systems – in diesem Fall eines Computersystems – auseinanderschneiden kann! Natürlich kann man alles irgendwo durchschneiden, wenn man nur das richtige Werkzeug dafür hat, aber hier ist natürlich gemeint, dass die Teile, die man auseinanderschneidet, dabei heil bleiben und dass man sie anschließend auch wieder zusammensetzen kann!

Kannst du dir an Hand dieser Erklärung denken, was gemeint ist? Im Prinzip jeder Stecker, den wir irgendwo einstöpseln können – und noch ein kleines bisschen mehr! Streng genommen ist also auch der Netzstecker, mit dem wir den Computer ans Stromnetz anschließen, eine Schnittstelle! Daran lässt sich auch ganz anschaulich das Problem von Schnittstellen erklären:

Als erstes muss der Stecker in die Steckdose hineinpassen! Das hört sich banal an, aber wenn du schon einmal im Ausland in Urlaub warst, hast du vielleicht gesehen, dass man dort andere Steckdosen hat als bei uns!

Wenn der Stecker dann drin steckt, muss sichergestellt sein, dass die Kabel auch alle richtig miteinander verbunden sind und der Strom auch durch dasjenige Kabel in den Computer fließt, das dafür gedacht ist!

Ja und zuguterletzt muss auch noch die Art und Stärke des Stroms stimmen! In Deutschland kommt aus den Steckdosen Wechselstrom mit 230Volt und 50Hz.

Volt ist die Einheit, mit der man elektrische Spannung misst, also die Kraft die den elektrischen Strom durch eine Leitung drückt. Die Frequenz 50 → Hertz bedeutet in diesem Fall, dass diese Kraft 50 Mal in der Sekunde die Richtung wechselt; deshalb heißt es Wechselstrom, aber das braucht uns hier nicht näher zu interessieren.

Wichtig an diesem Beispiel „Stromanschluss“ ist nur, dass du verstehst, dass jede Schnittstelle genau beschrieben, man sagt auch *spezifiziert* sein muss. In diesem Kapitel wird beschrieben, was man an einen Computer normalerweise alles anschließen kann. Falls dich das nicht so sehr interessiert, weil dein Computer schon fertig angeschlossen ist, hier nur ein kleiner, tröstlicher Hinweis:

Man kann so viel nicht falsch machen beim Anschließen der Geräte an den Computer: Die meisten Stecker passen nur dort, wo sie auch hingehören, und wenn es doch mal mehrere Möglichkeiten gibt (z.B. bei den Lautsprechern), dann gibt es oft farbliche Markierungen und man braucht den Stecker nur in die Buchse der gleichen Farbe zu stecken.

Manche der in den nachfolgenden Kapiteln beschriebenen Schnittstellen hat dein Computer vielleicht gar nicht. Immerhin erhältst du aber eine Vorstellung davon, was es alles gibt.

Auf den Bildern der Schnittstellen wird dir vielleicht auffallen, dass manchmal links und rechts der „Steckdose“ zwei Schraubenlöcher sind. Die Stecker haben dann an den Seiten Schrauben, die sich dort hineindrehen lassen. Auf diese Weise kannst (und solltest) du verhindern, dass ein gelockerter Stecker Probleme macht. – Und du solltest daran denken, wenn du versuchst die Stecker abzuziehen! Verwende niemals Gewalt! Sie könnten festgeschraubt sein.

5.2.1 Erweiterungskarten

Die Tatsache, dass man Computer für bestimmte Aufgaben programmieren kann, macht ihn so vielseitig einsetzbar. Aber auch die Möglichkeit, → Hardware für bestimmte Aufgaben einfach als Erweiterung einzubauen, trägt sehr dazu bei!

Für diesen Zweck bietet die → Hauptplatine sogenannte *Steckplätze*, in die man Erweiterungskarten hineinstecken kann, mit denen der Computer dann zusammenarbeitet.

Meistens muss man auch ein wenig Software → installieren, damit der Computer richtig mit der neuen Hardware arbeitet. Das gilt generell für jede angeschlossene Hardware. Solche kleinen Programme heißen *Treiber*, eine unglückliche Übersetzung des englischen Begriffs *Driver*; gemeint ist damit, dass diese Programme die Hardware sozusagen „antreiben“.

Die Steckplätze haben furchtbar viele Anschlüsse. Man kann aber trotzdem nicht viel falsch machen beim Einbau einer Erweiterungskarte. Ein großer Teil dieser Anschlüsse dient dazu, Daten – also wieder unsere geliebten Nullen und Einsen, die irgendetwas darstellen, wie Zahlen, Bilder, Texte, usw. – zwischen der Hauptplatine (und dort meistens dem Prozessor oder dem Arbeitsspeicher) und der Erweiterungskarte zu übertragen. Immer wenn auf Leitungen Daten übertragen werden, dann heißen diese Leitungen *Bus*. Das kann man sich leicht merken, wenn man sich das Gewusel von Nullen und Einsen als das Gedrängel in einem Omnibus vorstellt. Wenn man Daten besonders schnell übertragen will, macht man das gleichzeitig und synchron, das bedeutet im selben Takt, auf mehreren Leitungen gleichzeitig. Ein solcher Bus heißt dann ein *parallel* Bus. Stelle dir einen Omnibus mit mehreren Türen vor: Da geht das Ein-

und Aussteigen viel schneller als wenn alle Fahrgäste nacheinander durch eine Tür müssen.

Ein Bus mit nur einer Tür, durch die sich die Bits alle der Reihe nach durchschieben, heißt *serieller* Bus.

Es gibt verschiedene Arten von Steckplätzen für Erweiterungskarten; die wichtigsten sind der *ISA-Bus* (dessen Stecker sind auf der Hauptplatine meistens schwarz) und der *PCI-Bus* (mit weißen Steckerleisten).

PCI ist die Abkürzung von *Peripheral Component Interconnect*, was so viel heißt wie *Zusatzgeräteanschluss*.

ISA steht für *Industry Standard Architecture* und bedeutet übersetzt *Industriestandard-Bauweise*. Man merkt mal wieder, dass Abkürzungen mit drei Buchstaben oft dazu dienen, einfach einen Namen zu erzeugen, weil man einen braucht. Was er aussagt ist eigentlich egal, nur dass ein ausgeschriebener Name – noch dazu in deutscher Übersetzung – von jedem verstanden werden kann, und das ist natürlich uncool!

Außerdem gibt es auch noch Steckplätze für → Arbeitsspeicher.

Abbildung 27 zeigt ein → Mainboard mit verschiedenen Steckplätzen.

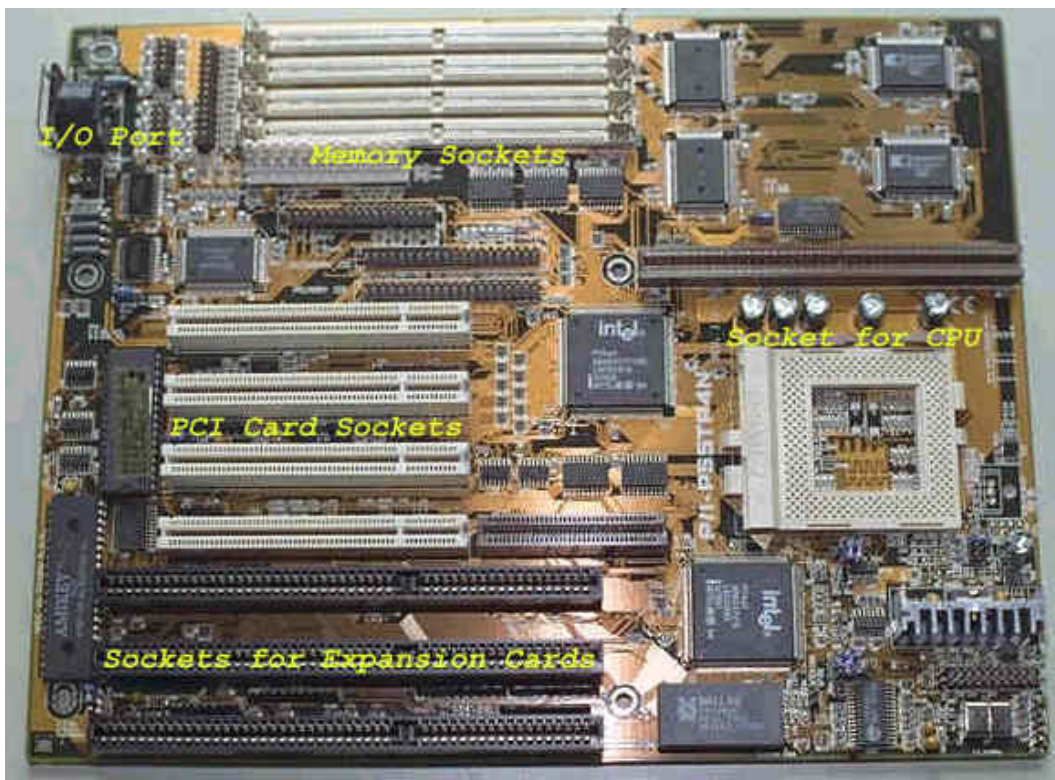


Abbildung 27: Erweiterungssteckplätze

Man kann die verschiedensten Erweiterungen einbauen: Mainboards, die keine Bausteine für das Erzeugen von Ton und/oder Grafik besitzen, erfordern sogar zwangsläufig, dass man dafür Erweiterungskarten benutzt.

Sogenannte *Soundkarten*, erzeugen heute nicht nur den Ton (Ton heißt auf englisch *sound*, sprich: *βaund*), man kann auch Musikgeräte oder Mikrofone daran anschließen und deren Tonausgabe vom Computer → digital aufnehmen lassen. Meist lässt sich auch noch ein *Joystick* daran anschließen, ein Steuerknüppel wie man ihn für bestimmte Spiele (z.B. Flugsimulator) braucht.

Grafikkarten erzeugen die Signale, die der → Monitor braucht, um das Bild richtig darstellen zu können.

Ein Modem, über das du in Kapitel 5.6 lesen kannst, wofür man es braucht, lässt sich meist auch in einem Erweiterungsslot (*slot* ist englisch für Schlitz) unterbringen. Ebenso Netzwerkkarten, mit deren Hilfe man Computer untereinander verbinden kann.

Man benutzt Steckplätze auch dafür, um andere Schnittstellen bereitzustellen; entweder, weil man von einem bestimmten Schnittstellentyp mehr braucht, oder weil man einen bestimmten Typ braucht der anders nicht verfügbar ist. Ich habe zum Beispiel eine Schnittstellenkarte in meinen Computer eingebaut, die mir eine *Firewire*-Schnittstelle zur Verfügung stellt, um meine Videokamera mit dem Computer verbinden zu können. Firewire (sprich: *feier-uai-er*, zu deutsch „Feuerdraht“) ist ein schneller → serieller Bus.

Die Steckplätze für Erweiterungskarten sind immer so angebracht, dass erforderliche Stecker, die die Erweiterungskarten nach außen bereitstellen müssen, an der Rückseite des PC zugänglich sind.

Das ganze gilt natürlich nur für Arbeitsplatzcomputer, also nicht für Laptops (tragbare Computer). Darin ist für Erweiterungskarten im herkömmlichen Sinne kein Platz. Sie verfügen statt dessen oft über eine PCMCIA-Schnittstelle.

5.2.2 USB



Die Abkürzung USB steht für Universal Serial Bus oder Universeller Serieller Bus. Diese Schnittstelle ist relativ neu, ziemlich schnell, und wirklich für den Anschluss sehr vieler Geräte geeignet: Heute werden fast alle → Drucker, → Scanner, → Digitalkameras und was man sonst noch so braucht, über den USB angeschlossen. Außerdem kann man wie schon erwähnt Speicherbausteine daran anschließen.

Abbildung 28: USB-Anschluss für zwei Geräte.

5.2.3 Plug & Play

Dass man bei der Computerei nur Englisch spricht, bist du wohl inzwischen gewohnt. Das Schlagwort Plug & Play (sprich: plag änd pley) bedeutet so viel wie Einstöpseln und Spielen. Man soll also sofort loslegen können, sobald man sein neues Gerät an den Computer angeschlossen hat – manchmal sogar während der Computer läuft!

Die meisten Geräte, die man heute kaufen kann unterstützen Plug & Play, bisweilen auch Plug 'n Play geschrieben, wobei das 'n für *and* (engl. und) steht.

Nicht immer werden die Geräte diesem Anspruch ganz gerecht, aber wenn man sich vor Augen hält, was alles passieren muss, damit ein Computer nicht nur erkennt, *dass* ein neues Gerät angeschlossen wurde, sondern auch *welches* das ist und *wie* er es ansteuern muss, dann funktioniert das schon beachtlich gut.

5.2.4 COM:

Mit diesem Kürzel, das vom englischen Wort *common* (gemeinsam) herrührt, wird auf PC's die „alte“ serielle Schnittstelle bezeichnet, die heute kaum noch benutzt wird. Früher waren daran meist → Modems oder die → Maus angeschlossen.

Die meist zwei COM-Schnittstellen eines PC arbeiten nach der RS-232-Spezifikation. Du erinnerst dich an das Beispiel mit der Steckdose für Strom, und dass wir gesagt haben, man muss immer genau festlegen, wie eine Schnittstelle aussehen und funktionieren soll. Die Festlegung für die COM-Schnittstellen heißt *RS-232*. Das ist ein sehr weit verbreiteter Standard, mit dem auch viele andere Geräte arbeiten, wie z.B. Messgeräte. Es ist also manchmal immer noch hilfreich, wenn man eine solche RS-232-Schnittstelle hat.

Man kann bei einer RS-232-Schnittstelle eine ganze Menge einstellen, um die verschiedenartigsten Geräte miteinander verbinden zu können. Zum Beispiel kann man auch sagen, wie viele Bits pro Sekunde übertragen werden sollen. Selbst einfachste Geräte schaffen fast immer 9600 *Baud*. Baud ist die Einheit für Datenübertragungsgeschwindigkeit und bedeutet ungefähr das selbe wie Bits pro Sekunde, jedenfalls solange man wie wir nur zwei verschiedene Zeichen verwendet. (Rate welche gemeint sind!
Richtig: Null und Eins!)



Abbildung 29:
zwei COM-Schnittstellen

5.2.5 LPT1:

Auch hierbei handelt es sich um eine ältere Schnittstelle. Sie wird oft auch *Parallelport* genannt. *Port* ist Englisch und bedeutet wörtlich Hafen (*airport* = Flughafen). Bei Computern ist damit ein Anschluss gemeint – entweder ein richtiger, also ein Stecker, oder ein Anschluss an ein Programm: Wenn Daten von einem Rechner zu einem anderen geschickt werden, muss immer dazu gesagt werden, welches Anwendungsprogramm die Daten bekommen soll; dazu benutzt man eine Portnummer.

Am Parallelport wurden früher meistens Drucker angeschlossen.

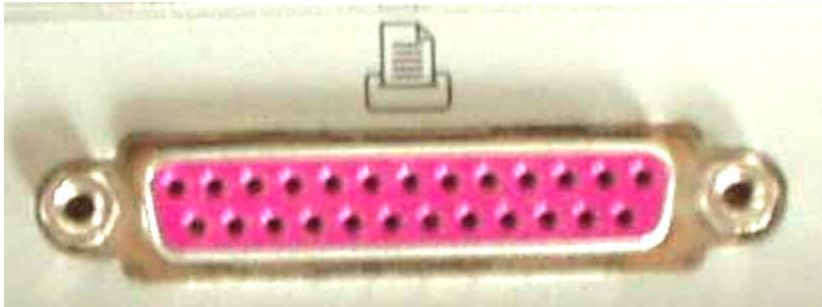


Abbildung 30: Parallele Schnittstelle

5.2.6 PS/2



So heißt die neuere Schnittstelle, an die man heute Tastatur und Maus anschließt.

Die Abkürzung bedeutet „Personal System 2“. So hat die Firma IBM einmal ein von ihr entwickeltes Computersystem benannt, das sich aber nicht durchgesetzt hat. Nur diese Schnittstellenspezifikation ist davon übrig geblieben!

Abbildung 31: PS/2-Schnittstelle für Maus (grün) und Tastatur (violett)

5.2.7 PCMCIA

Auch das ist wieder eine englische Abkürzung für „Personal Computer Memory Card International Association“ was so viel heißt wie Internationale Vereinigung für Computer Speicherkarten.

Eine Schnittstelle ist nur dann wirklich sinnvoll, wenn sich möglichst alle Hersteller, an die zugehörige Spezifikation halten. Als Nutzer eines Computers möchte man ja schließlich selbst entscheiden können, welchen → Monitor oder → Drucker man verwendet. Das geht aber nur dann, wenn man das Gerät auch

an den Computer anschließen kann, wenn also die Schnittstelle passt. Eine Firma, die zum Beispiel Computer herstellt, möchte natürlich eigentlich *nicht*, dass der Kunde jeden beliebigen Drucker (um bei diesem Beispiel zu bleiben) einer anderen Firma anschließen kann, besonders dann nicht, wenn diese Computerfirma gleichzeitig auch Drucker baut. Sie würde ja viel lieber nicht nur den Computer sondern zusätzlich auch den Drucker verkaufen! Also könnte sie auf die Idee kommen, die Schnittstelle zwischen Computer und Drucker so zu bauen, dass nur die eigenen Geräte aneinander passen! (Wenn zwei Geräte zueinander passen sagt man auch sie sind *kompatibel*.)

Ein Computer, der nur mit wenigen Druckern kompatibel ist, wird natürlich von weniger Leuten gekauft – das will die Computerfirma natürlich auch nicht!

Je nachdem wie die Verhältnisse so sind, wie viel Konkurrenz es am Markt gibt und welche Schnittstellen es vielleicht schon gibt, setzen sich manche Schnittstellen durch und andere nicht. Manchmal, ganz selten, passiert etwas tolles: Alle Firmen, die zum Beispiel Computer und Zubehör dafür bauen, merken irgendwann, dass man eine neue Schnittstellenspezifikation braucht, und setzen sich dann an einen Tisch und überlegen gemeinsam, wie man diese Schnittstelle spezifizieren sollte. Dann halten sich alle an diese Spezifikation und alle Geräte sind zueinander kompatibel. Jeder profitiert davon – auch und vor allem der Kunde!

Eine solche Geschichte war das auch mit der PCMCIA. 1989 haben sich verschiedene Hersteller zusammengetan und überlegt, wie man eine Schnittstelle definieren sollte, an der man Erweiterungen für tragbare Computer (Laptop) anschließen kann. Nach dieser Organisation wird die Schnittstelle heute benannt. Sie wird fast ausschließlich bei Laptops verwendet und man kann Arbeitsspeicher-Erweiterungen, → Festplatten, → Modems (auch für die Übertragung von Daten per Handy!) und alles mögliche andere daran anschließen!

5.2.8 Bildschirm

Der Bildschirm wird heute meist noch über die 15-polige VGA-Schnittstelle angeschlossen. VGA steht für Video Graphics Array, manchmal liest man auch Video Graphics Adapter, was beides mal wieder englisch ist und den Standard für die Umwandlung der → digitalen Bilddaten, die der Computer anzeigen will, in ein → analoges elektrisches Signal, aus dem dein Monitor ein Bild erzeugen kann. Der Anschluss für den Bildschirm kommt immer aus der Grafikkarte, wenn eine separat eingebaut ist.



Abbildung 32: Bildschirm-Anschluss

5.2.9 Netzwerk

Man kann Computer miteinander verbinden, so dass sie gegenseitig auf ihre Ressourcen (Festplatten, Drucker, usw.) zugreifen können. Insbesondere in Firmen, wo viele Leute mit Computern arbeiten, wird das gemacht, damit sie ihre Daten leichter untereinander austauschen können. Es gibt aber auch Spiele, die auf vernetzten Rechnern basieren.

Wenn viele Computer, die relativ nah beieinander stehen, miteinander verbunden sind, spricht man von einem lokalen Netzwerk oder in der englischen Abkürzung von einem LAN (*Local Area Network*).

Die gängige Schnittstellenspezifikation dafür heißt Ethernet (sprich: iesernet).

Auch das Internet ist ein Netzwerk von Computern, das allerdings weltumspannend ist; man sagt auch es ist ein WAN (englisch für *Wide Area Network*).

In Kapitel 8 liest du mehr über Netzwerke und das Internet.

Um deinen Computer mit weit entfernten Rechnern zu verbinden, benutzt du am besten die Telefonleitung. Die stöpselst du in ein → Modem oder, wenn du digitales Telefon benutzt, in eine → ISDN-Karte ein. Beides kann in deinem Rechner als → Erweiterungskarte eingebaut oder als externes Gerät angeschlossen sein.

5.2.10 Drahtlose Schnittstellen

Alle bisher vorgestellten Schnittstellen funktionierten immer so, dass man ein Kabel irgendwo 'reinstecken musste, so dass eine elektrische Verbindung entsteht. Manchmal ist das zu lästig und deshalb hat man Schnittstellen entwickelt, die ohne Kabel auskommen. Sie werden hier aber nur der Vollständigkeit halber erwähnt:

5.2.10.1 Infrarot-Schnittstelle

Euer Fernseher zuhause hat vermutlich eine Fernbedienung. Nach dem selben Prinzip arbeitet auch die Infrarot-Schnittstelle eines Computers: Die Information wird mit Hilfe von Licht übertragen, das allerdings eine „Farbe“ hat, die unser Auge nicht sehen kann. (Videokameras und Schlangen können diese Farbe sehen!)

Der Name des am weitesten verbreiteten Standards ist *IrDA*. Meist schließt man damit Drucker, Tastaturen, Mäuse oder Handys an.

5.2.10.2 Bluetooth-Schnittstelle

Eine ebenso aus dem alltäglichen Leben bekannte Möglichkeit des drahtlosen Informationsaustauschs ist Funk! Bluetooth (englisch für blauer Zahn, sprich: bluu tuuß) ist der Markenname der gängigen Schnittstellenspezifikation.

Viele moderne Mobiltelefone haben zum Beispiel eine Bluetooth-Schnittstelle, so dass man zum Beispiel Bilder oder das Adressbuch zwischen Handy und Computer austauschen kann.

5.2.11 Benutzerschnittstelle

Wie? Muss der Computerbenutzer sich auch irgendwo einstöpseln??? Natürlich nicht buchstäblich, aber so wie ein eingestöpseltes Gerät mit dem Computer über eine Schnittstelle Informationen austauscht, so tut das der Benutzer ja auch! Deshalb verwendet man den Begriff Schnittstelle auch für jedes Mittel, über das der Computer und sein Benutzer Informationen austauschen. Das sind zum Beispiel die Tastatur, die Maus, der Bildschirm, aber auch die Lautsprecher. Bei Programmen versteht man unter Benutzerschnittstelle vor allem, wie sich das Programm dem Benutzer auf dem Bildschirm präsentiert und wie es vom Benutzer bedient werden kann. Jedes Programm hat also eine eigene Benutzerschnittstelle! Man hob früher vor allem grafische Benutzerschnittstellen hervor (englisch *GUI, graphical user interface*), was heute allerdings selbstverständlich ist: Dass nämlich alle Bedienelemente wie Menüs, Buttons (Knöpfe auf die man klicken kann) auf dem Bildschirm – also grafisch – dargestellt werden und man sie mit der Maus betätigen kann.

Früher musste man Programme über Texteingaben bedienen, also Befehle eingeben, was dem Benutzer viel mehr Wissen abverlangt hat und viel umständlicher war. Dafür war der Aufwand der Programmierung und die Anforderungen an die Hardware viel geringer.

Ich erwähne die Benutzerschnittstelle hier vor allem deswegen, weil für sie genau die selben Anforderungen gelten wie im Kapitel 5.2 beschrieben: Die Art und Weise wie Informationen ausgetauscht, also Daten ein oder ausgegeben werden, muss genau festgelegt sein! Wenn du mit der Maus auf eine bestimmte Stelle des Bildschirms klickst, hat das eine genau definierte Bedeutung und es ist sehr wichtig, dass du diese Bedeutungen kennst, damit der Computer auch das macht, was du möchtest und es keine „Missverständnisse“ zwischen euch gibt!

In den nächsten beiden Unterkapiteln erkläre ich dir deshalb die zwei wesentlichen Teile der Benutzerschnittstelle, die du bedienen musst: Die Tastatur und die Maus.

5.2.11.1 Tastatur

In Abbildung 10 auf Seite 30 haben wir schon eine Tastatur gesehen, damit ich dir zeigen konnte, wie welche Taste genannt wird. In diesem Kapitel werden wir das vervollständigen. Ich gehe dabei von einer normalen PC-Tastatur mit 102 Tasten aus; auf einem Laptop sind die Tasten meist etwas anders angeordnet und einige fehlen auch um Platz zu sparen, und auch an anderen Rechnern sind manchmal Tastaturen dran, die ein wenig anders aussehen. Aber das muss uns nicht aus der Ruhe bringen, denn das Wesentliche ist überall gleich. Vorausgesetzt, dass an deinem Rechner alles richtig eingestellt ist (man muss ihm nämlich auch sagen, welche Tastatur er hat), gilt auf jeden Fall das Nutella-Prinzip: Es steht immer drauf, was drin ist! Das soll heißen, dass du dich im Zweifel am Namen einer Taste (dem was draufsteht) und nicht an ihrer Position auf einer der Abbildungen orientieren musst.

Auch wenn du vielleicht schon einen PC benutzt und die vorstehenden Experimente gemacht hast, fange ich noch einmal *ganz von vorne* an:

Wenn du auf eine Taste tippst, wird das Zeichen, das auf dieser Taste steht, in den Computer *einggegeben*. Rein in den Computer also – aber wohin? Für die Antwort denken wir wieder an unsere Modellcomputer zurück: Wovon hängt es ab, was der Computer macht? Vom Programm!

Welche Bedeutung dein Tastendruck für den Computer hat, hängt also davon ab, welches Programm der Computer gerade ausführt. Dabei gibt es zwei grundsätzlich unterschiedliche Möglichkeiten der Bedeutung:

Entweder ist das bzw. sind die Zeichen, die du eingibst, Daten, also irgendeine Information, die der Rechner weiterverarbeiten soll. Wenn er beispielsweise „1+3“ ausrechnen soll, musst du irgendwann einmal eine „1“ eingeben, und wahrscheinlich auch eine „3“.

Wenn man einfach nur Daten eingeben möchte, ist es meist wichtig, wo man sie hinschreibt. Programme, bei denen man etwas eingeben kann (zum Beispiel Texte in *Textverarbeitungsprogramme* oder *Editoren*, s. Kapitel 7.3.1), haben meistens eine *Einfügemarke* oder auf Englisch sagt man *Cursor* dazu, die bzw. der einem anzeigt, wo das Zeichen, das man als nächstes eingeben wird, eingefügt werden wird. Meist blinken diese Cursor immer ganz hektisch. (Die Blinkgeschwindigkeit kann man aber einstellen!)

Die zweite mögliche Bedeutung deines Tastendrucks (wenn es sich also nicht um Daten handelt) ist ein Befehl an den Computer, etwas bestimmtes zu tun.

Um dir das zu verdeutlichen, kannst du das folgende sehr einfache Experiment machen:

Experiment 4: Tastatureingaben als Befehle

1. Starte deinen Computer und warte bis du den Windows-Bildschirm siehst.
2. Du siehst einige Symbole auf deinem Bildschirm. Bestimmt ist eines dabei, das „Arbeitsplatz“ heißt und eines das mit „Eigene Dateien“ betitelt ist.
3. Wenn du nun die Taste [A] antippst, wirst du sehen, dass das Symbol „Arbeitsplatz“ anders dargestellt wird. Du hast es *ausgewählt*. Wenn mehrere Symbole mit „A“ beginnen, kann es sein, dass du [A] mehrmals drücken musst, und die mit „A“ beginnenden Symbole nacheinander ausgewählt werden.
4. Ausgewählt bedeutet, dass sich alle deine folgenden Eingaben auf dieses Symbol beziehen sollen. Wenn du nun zum Beispiel die Eingabetaste [↵] betätigst, wird der → Explorer geöffnet und zeigt den Arbeitsplatz an.

Du hast gesehen, dass kein „A“ auf dem Bildschirm erschienen ist, obwohl du die Taste [A] betätigt hast. Das Programm „Betriebssystem Windows“ – was genau das ist, erfährst du in Kapitel 6.2 – reagiert auf deine Eingabe, indem es den Befehl ausführt, das erste Symbol, dessen Name mit „A“ beginnt, auszuwählen.

Das Programm Word – ein Textverarbeitungsprogramm – hätte auf die selbe Eingabe damit reagiert, an die Stelle der Einfügemarke ein „a“ zu schreiben – und zwar ein kleines.

Obwohl die Reaktion deines Computers auf deine Tastendrücke vom Programm abhängen, das gerade abläuft, haben sich natürlich bestimmte Bedeutungen eingebürgert. So wie ein Auto auch immer mit einem Lenkrad gelenkt wird und die Reihenfolge der Pedale von rechts nach links Gas, Bremse, Kupplung bei allen Herstellern gleich ist, so sind auch die meisten Programmreaktionen auf Tastendrücke identisch. Deshalb kann ich sie dir hier beschreiben, sonst müsstest du in allen Programm-Handbüchern nachlesen, was nun was bedeutet! Trotzdem kann es natürlich im Einzelfall vorkommen, dass etwas anders ist als hier beschrieben. Du musst es einfach ausprobieren.

Buchstabentasten:

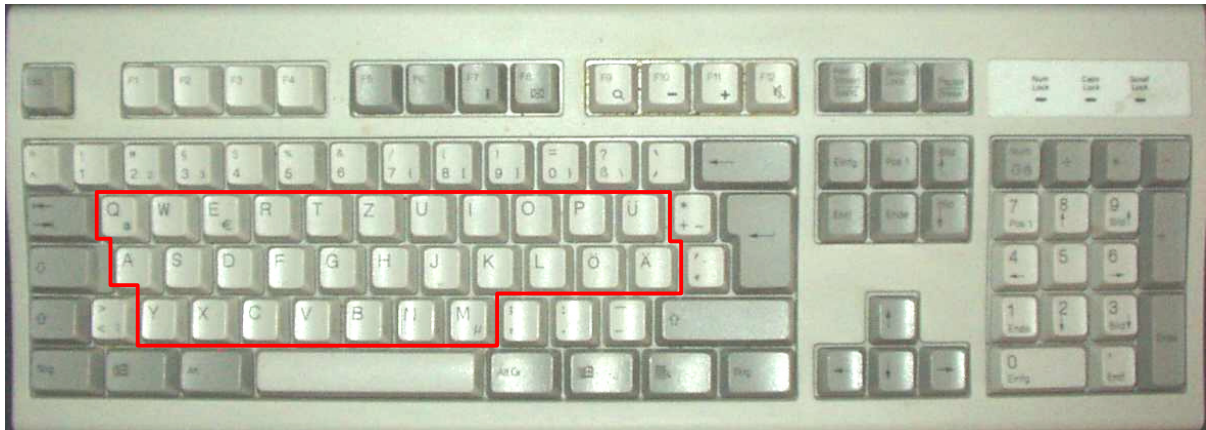


Abbildung 33: Die Buchstabentasten

Die kennt jeder! Trotzdem gibt es ein paar erwähnenswerte Dinge: Vielleicht hast du dich schon gewundert, warum die Tasten nicht in alphabetischer Reihenfolge und nicht alle ordentlich untereinander angeordnet sind. Man wollte bei der Festlegung dieser → Benutzerschnittstelle erreichen, dass man möglichst schnell schreiben kann, und die Tasten sind so angeordnet, dass man in der Sprache häufig vorkommende Buchstaben und Buchstabenkombinationen gut erreichen kann, wenn man im *Zehnfingersystem* schreibt, wenn man also alle (!) Finger zum Tippen benutzt. Wenn man fleißig übt, kann man das lernen! Und wenn man es richtig kann, schreibt man auf einer Computertastatur viel schneller als mit der Hand. Dabei schaut man dann nicht einmal mehr auf die Tasten (da liegen ohnehin die Hände drüber und man sieht gar nichts), sondern man liest gleich während des Schreibens am Bildschirm mit! Deshalb gibt es auf vielen Tastaturen auf der Taste [F] und der Taste [J] auch Markierungen, damit man fühlen kann, wo die Zeigefinger liegen sollen, wenn man beginnt, mit zehn Fingern zu schreiben.

Dies ist schon wieder ein Beispiel, wie wichtig solche Schnittstellenspezifikationen sind, auch für die Benutzerschnittstelle: Stell dir vor, jeder Computer hätte die Tasten unterschiedlich angeordnet! Es würde die Mühe gar nicht lohnen, das Zehn-Finger-Schreiben zu üben!

Trotzdem gibt es natürlich auch hierbei wieder Varianten: Weil die Buchstaben in den verschiedenen Sprachen unterschiedlich häufig vorkommen, gibt es je nach Land verschiedene Anordnungen. Ein wesentliche Unterschied ist, wo das [Z] und das [Y] sitzen, so dass Tastaturen häufig auch nach den ersten sechs Zeichen der ersten Reihe Buchstaben bezeichnet werden: eine *QWERTZ*-Tastatur ist die gebräuchliche deutsche Ausführung. Bei der *QWERTY*-Tastatur haben [Z] und [Y] die Plätze getauscht. Außerdem sind die Umlaute [Ä], [Ö], [Ü] und das [ß] anders belegt, weil man diese Zeichen im Englischen nicht braucht, und infolgedessen sind auch andere Sonderzeichen an anderer Stelle.

Wenn dein Rechner also mal andere Zeichen ausgibt, als du ihm über die Tastatur eingegeben hast, besonders wenn er statt einem z ein y schreibt und umgekehrt, dann ist deine Tastaturbelegung falsch eingestellt.

Wenn du dieses Buch zuende gelesen hast, kannst du das Problem leicht beheben, indem du unter Start/Einstellungen/Systemsteuerung „Tastatur“ aufrufst und das wieder richtig einstellst.

Leertaste



Abbildung 34: Die Leertaste

Die größte Taste! Und nichts drin!

Wirklich nichts? Doch, das Leerzeichen! Der Computer unterscheidet zwischen „Nichts“ und einem *Leerzeichen*! Das erkennt man schon daran, dass es dafür auch zwei verschiedenen → ASCII-Codes gibt. Erinnerst du dich an die Tabelle 6: ASCII-Code auf Seite 27? „Nichts“ hat den ASCII-Code Null und das Leerzeichen hat den ASCII-Code 32.

Wenn man ein wenig darüber philosophiert, kommt man tatsächlich zu dem Ergebnis, dass der Computer wirklich unterscheiden können muss, ob sein Herr und Meister – der Bediener – (noch) nichts eingegeben hat, oder ob er *wollte*, dass die Eingabe leer bleibt. Im ersten Fall, wurde der Bediener noch gar nicht gefragt und weiß demzufolge gar nicht, dass eine bestimmte Eingabe noch nicht vorgenommen wurde (vielleicht überlegt er aber auch noch), im zweiten Fall wurde er gefragt, wollte aber keine Eingabe tätigen. Auch bei Texten wird der Unterschied deutlich: Zwischen Worten steht nicht „nichts“ sondern ein Zwischenraum – das Leerzeichen eben.

Das sind Themen, über die man am besten dann nachdenken sollte, wenn man abends im Bett nicht einschlafen kann! Dann schläft man nämlich bestimmt sehr schnell ein! Für uns ist nur wichtig zu wissen, dass man mit der Riesentaste den Abstand zwischen zwei Worten erzeugt und dass der Computer diesen Abstand genauso behandelt wie jedes andere Zeichen.

Umschalttaste [⇧]

Es gibt große und kleine Buchstaben. Sie werden alle mit den selben Tasten geschrieben. Wenn man einen großen Buchstaben eingeben will, muss man dafür die Umschalttaste betätigen. Auf Englisch heißt diese Taste *Shift* (sprich: schift).



Abbildung 35: Umschalttasten

Wie du in Abbildung 35 sehen kannst, gibt es zwei Umschalttasten. Sie sind gleichwertig; du solltest sie immer so benutzen, dass die Hand mit der du den Buchstaben tippst, noch frei bleibt. Um also Beispielsweise ein „L“ zu tippen, würdest du die linke Umschalttaste benutzen, weil man das [L] mit der rechten Hand betätigt. Entsprechend würde man um ein „A“ zu schreiben, die rechte Umschalttaste benutzen.

Man drückt immer zuerst die Umschalttaste, *hält sie fest*, und tippt dann den Buchstaben. Wenn danach wieder ein kleiner Buchstabe folgen soll, lässt man die Umschalttaste wieder los; andernfalls kann man sie auch festhalten.

Feststelltaste [⇩]

Manchmal möchte man so viele große Buchstaben hintereinander schreiben, dass es unbequem wäre, die ganze Zeit die Umschalttaste gedrückt zu halten. Man könnte ja einen Krampf bekommen! Vor allem fehlt aber die Hand, die die Umschalttaste drückt, beim schnellen Zehnfingerschreiben. Deshalb gibt es die Feststelltaste: Wenn man die einmal tippt, dann ist es so, als würde man von da an die Umschalttaste die ganze Zeit festhalten. Dieser Zustand wird durch das mittlere der drei Lämpchen über dem → Ziffernblock angezeigt. Das geht so lange, bis man entweder die Umschalt- oder die Feststelltaste ein zweites Mal drückt. (Welche Variante zutrifft kann man unter Start / Einstellungen / Systemsteuerung „Tastatur“ einstellen. Probiere einfach aus, wie dein Rechner eingestellt ist!)



Abbildung 36: Feststelltaste

Ihren Namen hat die Feststelltaste daher, dass sie früher bei den mechanischen Schreibmaschinen wirklich die Umschalttaste in der gedrückten Stellung arretiert hat. Auf Englisch heißt sie *CapsLock*, ausgeschrieben *Capitals Locked*, was so viel heißt wie „Großbuchstaben eingerastet“.

Eigentlich ist die Feststelltaste wie vieles andere auch ein Relikt (so nennt man ein Überbleibsel, etwas das von etwas Vergangenen übrig geblieben ist) aus der Schreibmaschinenzeit, denn man könnte dem Computer viel einfacher durch einen Programmbefehl sagen, welche Buchstaben Großbuchstaben sein sollen. MAN KANN SOGAR EINSTELLEN, DASS ES GROßE UND KLEINE GROßBUCHSTABEN – SOGENANNT KAPITÄLCHEN – GEBEN SOLL, SO WIE IN DIESEM SATZ.

Aber das kam so: Das erste große Einsatzgebiet für Computer war die Textverarbeitung. Schreibmaschinen waren mit das Erste, was durch Computer überflüssig wurde. Die Leute, die statt einer Schreibmaschine einen Computer benutzen sollten, wollten aber möglichst wenig umlernen und noch viel weniger auf das bisschen Komfort verzichten, das die Schreibmaschine Ihnen schon bot. (Die → Benutzerschnittstelle sollte sich nicht ändern!) Also war das Wichtigste, wenn man einer Firma einen – oder noch besser: ganz viele – Computer verkaufen wollte, dass man ihn genauso bedienen konnte, wie eine Schreibmaschine. Deshalb musste es auch eine Feststelltaste geben.

Wenn man auf einer Schreibmaschine etwas unterstreichen wollte, ging das so, dass man zuerst den Text geschrieben hat, dann den Wagen³ wieder an das erste zu unterstreichende Zeichen positioniert hat, und dann jedes zu un-

³ Hast du schon einmal eine Schreibmaschine gesehen? Der „Wagen“ ist das bewegliche Teil, in das das Papier eingespannt wird, und das sich bei jedem getippten Zeichen eine Stelle weiter bewegt, damit der nächste Buchstabe rechts daneben gedruckt wird. Außerdem wird das Papier beim Zurückschieben des Wagens am Zeilenende um eine Zeile nach oben bewegt.

terstreichende Zeichen mit der → Umschalttaste und der Unterstrich-Taste [_] unterstrichen hat. Die erste Erleichterung, die elektrische Schreibmaschinen brachten, war, dass man die Unterstrich-Taste nicht für jedes Zeichen neu betätigen, sondern nur festhalten musste; dann wurde der Unterstrich so lange automatisch wiederholt, bis man die Tasten wieder losgelassen hat. Der Einfachheit halber funktionierte das mit allen Tasten.

Unterstreichen geht in einem Textverarbeitungsprogramm viel einfacher (auch damals schon!) und auch ohne das Unterstrichsymbol. Trotzdem gibt es sowohl das Symbol wie auch die Wiederholungsfunktion heute noch auf jedem Computer!

Wenn du in einem Textverarbeitungsprogramm die Taste [X] (oder irgend eine andere) einige Sekunden festhältst, erscheint „xx“ (bzw. das Zeichen, das *du* festgehalten hast)!

Ein ähnliches Relikt aus Schreibmaschinentagen ist, dass die Feststelltaste nicht nur auf die Buchstabentasten wirkt, sondern auf alle Tasten, auch auf die Zifferntasten, jedenfalls auf die, die es bei der Schreibmaschine auch schon gab (die über den Buchstaben). Eigentlich wäre es viel praktischer, wenn die Zifferntasten auch bei betätigter Feststelltaste Ziffern schreiben würden. Bei den mechanischen Schreibmaschinen ging das aber nicht, und deshalb wurde das bei Computern so beibehalten, damit die Leute sich nicht umgewöhnen mussten. So ist es bis heute geblieben, und es ist auch nicht mehr so wichtig, weil man ja einen zusätzlichen Ziffernblock hat.

[Strg]-Taste

Wir haben schon gelernt, dass Tastendrucke entweder die Eingabe von Daten oder von Befehlen sein kann, je nachdem, wie das → Programm geschrieben ist und an welcher Stelle der Ausführung es sich gerade befindet.

Was mache ich denn, wenn ich an ein und der selben Stelle des Programms sowohl Daten als auch Befehle eingeben können möchte? Nehmen wir also beispielsweise an, ich arbeite mit einer Textverarbeitung und gebe laufend Zeichen ein. Das sollen alles Daten sein und das Programm soll sie wie programmiert verarbeiten. Jetzt möchte ich dem Programm aber während der Dateneingabe auch Befehle geben können, beispielsweise soll es einen bestimmten Textabschnitt markieren, damit ich den woanders hinschieben kann.

Dazu muss ich die Möglichkeit haben, Dateneingaben von Befehlen zu unterscheiden. Dafür hat man die [Strg]-Tasten erfunden: „Strg“ ist die Abkürzung für „Steuerung“. Auf manchen Tastaturen steht auch „Ctrl“ für „Control“, was übersetzt Steuerung bedeutet. Die Tasten werden benutzt wie die Umschalttasten,

das heißt man drückt zuerst die [Strg]-Taste, hält sie fest und tippt dann die zugehörige zweite Taste.



Abbildung 37: Steuerungstasten

Nun kann ich während meiner Eingabe ein [P] tippen und das Textverarbeitungsprogramm wird es als „Dateneingabe Buchstabe P“ verstehen. Oder ich kann [Strg] + [P] tippen und damit den Befehl geben, den Text zu drucken (P von englisch *print*, drucken).

Wieder ist es natürlich vom Programm abhängig, welche Befehle es bereitstellt und mit welchen Tasten man sie erreichen kann! Die Steuerungstasten vergrößern nur die Möglichkeiten für den Programmierer.

Die linke und die rechte Steuerungstaste können gleichwertig benutzt werden; genau wie bei der Umschalttaste wird es davon abhängen, welche zweite Taste man drücken will.

[Alt]-Taste

Die [Alt]-Taste stellt eine weitere Möglichkeit der alternativen Tastenbelegung dar. Sie wird meist ähnlich verwendet wie die [Strg]-Taste, vor allem in Verbindung mit den → Funktionstasten.

Darüber hinaus hat sie meist die selbe Funktion wie die [F10]-Taste: Man kann mit ihr das Menü aufrufen. Wenn man zusätzlich eine Buchstabentaste drückt, wird das Menü oder der Befehl, in dessen Namen dieser Buchstabe unterstrichen ist, ausgewählt; gibt es keine unterstrichenen Buchstaben, wird der erste Befehl oder Menüpunkt ausgewählt, der mit diesem Buchstaben beginnt.



Abbildung 38: Die [Alt]-Taste

Über Menüs und Befehle erfährst du mehr im Kapitel 6.2.1.3. Du kannst aber, um dich schon jetzt damit vertraut zu machen, auch folgendes Experiment machen:

Experiment 5: Gebrauch der Alt-Taste

1. Starte deinen Computer und warte, bis du den Windows-Bildschirm siehst.
2. Drücke die Taste [F1].
3. Es erscheint die Windows-Hilfe
4. Drücke [Alt] + [I], [Alt] + [N], [Alt] + [S] oder [Alt] + [F] und du siehst, wie die Registerkarten Index, Inhalt, Suchen oder Favoriten angezeigt werden.

Sollte in deiner Windows-Version eine andere Hilfe erscheinen, suche nach unterstrichenen Buchstaben wie in Abbildung 39.

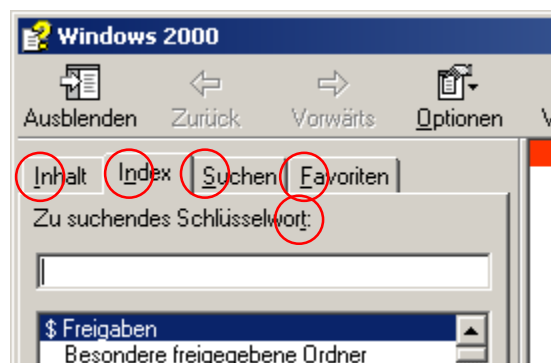


Abbildung 39: Unterstrichene Buchstaben für Benutzung mit der [Alt]-Taste

[AltGr]-Taste



Abbildung 40: Die [AltGr]-Taste

Vorsicht mit den Tasten [Alt] und [AltGr]! Im Gegensatz zu den jeweils doppelt vorhandenen Steuerungs- und Umschalttasten haben diese beiden Tasten unterschiedliche Bedeutung – ist ja auch klar: Es steht ja auch etwas unterschiedliches drauf! Bei den anderen steht jeweils das selbe drauf. Auch hier gilt das Nutella-Prinzip!

Die Taste [AltGr] funktioniert genau wie auch Umschalt-, Steuerungs- und Alt-Taste: Zuerst [AltGr] drücken und festhalten, dann eine weitere Taste tippen.

Es gibt einige festgelegte Zeichen, die man mit Hilfe der [AltGr]-Taste tippen muss, wie zum Beispiel

[AltGr]+[Q] ergibt @. Dieses Zeichen heißt das *at*-Zeichen. Es wird „ät“ ausgesprochen und bedeutet „bei“. Man braucht es vor allem, weil es ein vorgegebener Teil von e-Mail-Adressen ist, also von Adressen in der elektronischen Internetpost, über die du in Kapitel 8.3 noch etwas lesen wirst.

[AltGr]+[E] ergibt €, das Währungszeichen für den Euro.

[AltGr]+[M] ergibt μ , einen griechischen Buchstaben, mit dem man „Millionstel“ abkürzt so wie man das „m“ für Tausendstel benutzt: 1 μ m ist ein Millionstel Meter oder ein Tausendstel Millimeter.

[AltGr]+[B] ergibt \, den rückwärts gerichteten Schrägstrich oder auf Englisch *Backslash* (sprich: bäcksläsch). Dieses Zeichen braucht man, wenn man → Pfadangaben schreibt, als Trennzeichen zwischen den → Verzeichnisnamen. (Wenn du das jetzt noch nicht verstanden hast, macht nichts, dazu kommen wir noch!)

Und dann gibt es da noch die geschweiften und eckigen Klammern, die man in der gezeigten Reihenfolge { [] } mit [AltGr]+[7], [AltGr]+[8], [AltGr]+[9], [AltGr]+[0] schreibt.

Die Tasten [Strg], [Alt], [AltGr] und [⇧] (Umschalten) können auch in Kombinationen gleichzeitig verwendet werden! Beispiel aus dem → Textverarbeitungsprogramm Word:

- [→] bewegt die Einfügemarke *ein Zeichen* nach rechts.
- [⇧]+[→] bewegt die Einfügemarke *ein Zeichen* nach rechts und markiert das Zeichen.
- [Strg]+[→] bewegt die Einfügemarke *ein Wort* nach rechts.
- [⇧]+[Strg]+[→] bewegt die Einfügemarke *ein Wort* nach rechts und markiert das Wort.

Dabei gilt, dass die Reihenfolge, in der die jeweils verlangten Tasten [Strg], [Alt], [AltGr] und [⇧] betätigt werden, egal ist, sie müssen nur alle gedrückt *sein*, wenn die letzte Taste getippt wird, im Beispiel also die Pfeiltaste [→].

Klingt ganz schön kompliziert, was? Ist es aber nicht! Probier es bei Gelegenheit mal aus und du wirst verstehen, was gemeint ist!

Tabulatortaste [→]



Abbildung 41: Die Tabulatortaste

Die Tabulatortaste war ursprünglich (auf der Schreibmaschine) dafür vorgesehen, Tabellen oder mehrere Spalten schreiben zu können. Man kann mit dieser Taste bestimmte vorher festgelegte Stellen in einer Zeile, nämlich jeweils die Stellen, wo eine neue Tabellenspalte beginnt, anspringen. Diese Stellen nennt man die *Tabulatoren*.

Wenn man beispielsweise eine Liste mit Geburtstagen schreiben wollte, bei der Vornamen, die Nachnamen und Geburtstage übersichtlich untereinander stehen, ginge das so:

Name →	Vorname →	Geburtstag
Janßen →	Katharina →	27.11.1993
Janßen →	Alexander →	31.07.1996
Pimpelhuber →	Hans-Hermann →	21.02.1967

Immer wenn man an der Stelle war, wo hier die roten Pfeile stehen, musste man die Tabulatortaste betätigen. Dann schrieb die Schreibmaschine am Beginn der nächsten Spalte weiter.

In Textverarbeitungsprogrammen kann man das auch heute noch so machen, es gibt aber viel elegantere Methoden, Tabellen zu schreiben. Manchmal ist diese einfache aber immer noch ganz brauchbar, eben weil sie so einfach ist.

Inzwischen haben sich für die Tabulatortaste weitere, zeitgemäßere Verwendungen eingebürgert. Du erinnerst dich ja, dass ein Programm auf deine Tastatureingaben so reagiert, wie es der Programmierer programmiert hat:

Sehr oft wird einem auf dem Bildschirm ein Formular mit mehreren Feldern angeboten, das man ausfüllen muss. Du kennst das vielleicht auch schon, wenn du einen Computer hast, bei dem du dich anmelden musst.

Wenn man mit dem Ausfüllen des ersten Feldes fertig ist, muss man ja irgendwie zum zweiten Feld kommen. Das geht entweder mit der Maus, was aber umständlich ist, weil man die Hände ja gerade auf der Tastatur hat, oder es geht mit der Tabulatortaste.

Auf der Tabulatortaste sind übrigens zwei entgegengesetzte Pfeile abgebildet, weil man mit der Umschalttaste die Springrichtung ändern kann; mit [⇧]+[→] kommt man also in das vorhergehende statt in das nächste Feld.

Zifferntasten



Abbildung 42: Die Zifferntasten

Du hast zwei verschiedene Möglichkeiten, Ziffern einzugeben: entweder mit den Tasten über den Buchstaben, oder mit Hilfe des Ziffernblocks.

Die Tasten auf dem Ziffernblock sind doppelt belegt, das heißt du kannst die Bedeutung umschalten, und zwar indem du die Taste [NumLock] betätigst:



Abbildung 43: NumLock-Taste und Kontrolllämpchen

Die Tasten des Ziffernblocks schreiben nur dann Ziffern, wenn das linke der drei Kontrolllämpchen leuchtet. Andernfalls haben sie die Funktionen wie der → Sechserblock und die → Pfeiltasten.

Eingabetaste(n) [↵]

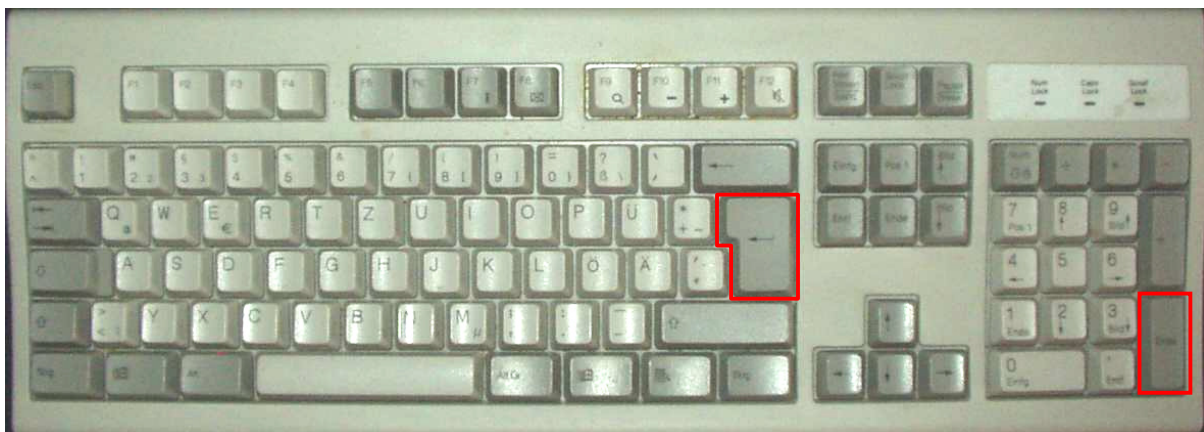


Abbildung 44: Eingabetasten

Diese beiden Tasten waren früher von Ihrer Bedeutung her unterschiedlich, heute spielt es praktisch keine Rolle mehr und du kannst sie gleichwertig benutzen:

Die große Taste neben dem [+] und dem [#] brauchte man ursprünglich, um eine neue Zeile zu beginnen; auf Englisch heißt sie *Return*. Daran kann man ihre Herkunft von der Schreibmaschine erkennen: Zu Deutsch bedeutet das „Rückkehr“. Wenn man sie betätigt hat, ist der Wagen, der bei der Schreibmaschine das Papier bewegt hat, an den Zeilenanfang *zurückgekehrt* und hat den Papierbogen eine Zeile höher bewegt, damit man eine neue Zeile beginnen

konnte. Diese Taste war ein echter Luxus elektrischer Schreibmaschinen! Bei mechanischen gab es einen großen Hebel, der die selbe Funktion erfüllte aber mit ziemlich viel Kraft bewegt werden musste, weil dadurch eine Feder gespannt wurde, die den Wagen beim Schreiben der nächsten Zeile Zeichen für Zeichen weiter bewegt hat. Auf Deutsch heißt die Taste dann auch „Wagenrückauftaste“ aber dieses Wort ist so unhandlich, dass wir wohl besser bei Return oder Eingabetaste bleiben!

Die andere Eingabetaste heißt auf Englisch *Enter*. Wenn du Piratenfilme magst, weißt du schon was das bedeutet:

Wenn Piraten ein Schiff erobern wollen, dann entern sie es! Sie fahren darauf zu, sammeln sich grimmig blickend an der Reling ihres Schiffes, das Enter-Messer zwischen den Zähnen, in jeder Hand einen Säbel, und sobald sie nah genug dran sind, springen sie auf das andere Schiff und erobern es – jedenfalls wenn es im Film die Guten sind, die gewinnen sollen.

Auf deinem Computer passiert genau das selbe: Du gibst zum Beispiel Daten in ein Formular ein (die Daten sammeln sich an der Reling). Wenn du damit fertig bist, gibst du den Daten den Befehl zum Entern des Computers, indem du die Taste [Enter] betätigst! Dann entern die Daten deinen Computer und erobern ihn – jedenfalls wenn es gute Daten waren.

Spaß beiseite: Mit dem Drücken der Entertaste sagte man früher einem Programm, dass die eingegebenen Daten nun verarbeitet werden sollen. Auf deutsch heißt die Taste deshalb auch *Datenfreigabe*. Auch ziemlich unhandlich!

Zusammengefasst: Meistens haben die Eingabetasten [Return] und [Enter] identische Bedeutung. Man sagt damit, dass eine Eingabe fertig ist und die eingegebenen Daten verarbeitet werden sollen. In einem Textverarbeitungsprogramm erzeugt man mit den Eingabetasten einen neuen Absatz.

Zeichentasten

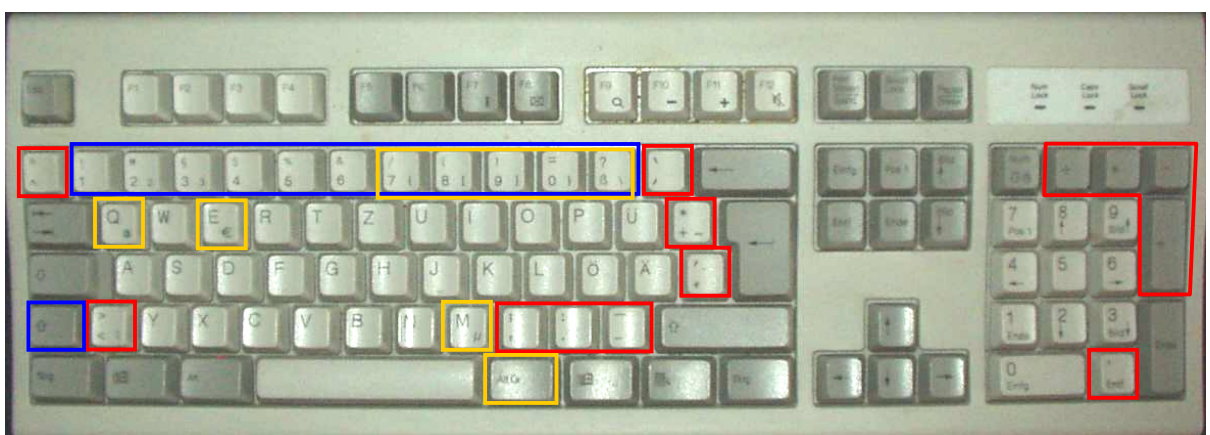


Abbildung 45: Zeichentasten

Die meisten Zeichen erreicht man mit der Umschalttaste und einer Zifferntaste (in Abbildung 45 **blau** eingerahmt). Einige Tasten sind nur mit Sonderzeichen belegt, zum Beispiel [,] (**rot** eingerahmt in Abbildung 45).

Es gibt ein paar Besonderheiten:

Sonderzeichen, die rechts auf einer Taste stehen, wie das „@“ beim [Q], erreicht man nicht über die Umschalttaste, denn das ergibt ja das große Q, sondern mit der Taste → [AltGr]. Diese Tasten sind in Abbildung 45 **gelb** gekennzeichnet.

Einige Sonderzeichen sind dafür gedacht, in Verbindung mit Vokalen wie dem „ä“ oder „e“ eingesetzt zu werden. Das sind so genannte Akzente, die einem sagen, wie dieser Buchstabe an dieser Stelle ausgesprochen wird. Zum Beispiel in der französischen Sprache werden diese Zeichen oft benutzt.

Wie kriegt man diese Zeichen nun auf diese Buchstaben? Zuerst die Akzenttaste tippen und dann den Buchstaben. Beispiele:

[´] und danach [E] ergibt „é“.

[´] und danach [⇧] + [E] ergibt „É“.

[⇧] + [`] und danach [A] ergibt „à“.

[^] und danach [O] ergibt ô.

Das funktioniert für alle gültigen Kombinationen von Akzent und Buchstabe. Tippt man nach dem Akzent einen Buchstaben, den es mit diesem Akzent nicht gibt, werden die beiden Zeichen nacheinander gedruckt wie zum Beispiel hier: „^m“.

[Esc]-Taste



Abbildung 46: Die Escape-Taste

„Esc“ steht für das englische Wort *Escape* (sprich: iskäip) und das bedeutet Flucht. Die ursprüngliche Idee dieser Taste, die auch heute noch bei den meis-

ten Programmen funktioniert, war, einen Ausweg aus folgender Situation anzubieten:

Ein Programm tut etwas, das ich nicht erwarte. Zum Beispiel verlangt es eine Eingabe von mir, die ich nicht machen kann, weil ich die Information vielleicht gerade nicht habe. Ich habe offensichtlich einen falschen Befehl eingegeben, und weiß nun nicht, was ich tun soll.

Das Betätigen der Taste [Esc] führt dann dazu, dass man wieder an die Ausgangsposition zurück kommt, ohne etwas verändert zu haben.

Manchmal kann man mit [Esc] auch ein Programm beenden oder eine Berechnung, die zu viel Zeit in Anspruch nimmt, unterbrechen.

Speziell im letzten Fall ist ein *versehentliches Betätigen* der [Esc]-Taste natürlich ärgerlich. Deshalb ist sie ganz weit oben außen angebracht, damit man nicht so schnell versehentlich darauf drückt. So manch einem, der seine Aktenordner auf der Tastatur liest, hat aber selbst diese Vorsichtsmaßnahme nicht geholfen.

Funktionstasten



Abbildung 47: Die Funktionstasten

Was ist das nun wieder?! Nichts weiter als die konsequente Ausnutzung der Tatsache, dass die Reaktion des Computers auf einen bestimmten Tastendruck vom Programm abhängt. Man hat durch die meistens zwölf Funktionstasten eine weitere Möglichkeit geschaffen, bestimmte Funktionen, die ein Programm ausführen kann, direkt über eine Taste – eben eine der Funktionstasten – aufrufen zu können.

Welche Funktion sich hinter welcher Taste verbirgt, muss man in der Bedienungsanleitung des jeweiligen Programms nachlesen. Es gibt jedoch ein paar Belegungen, die fast überall gleich sind:

[F1] ruft die Hilfefunktion auf, mit der man sich bei der Programmbedienung helfen lassen kann.

[Alt] + [F4] beendet ein Programm.

[F10] Menü auswählen

Probiere die Funktionstasten in folgendem Experiment aus!

Experiment 6: Funktionstasten

1. Starte deinen Computer und warte, bis du den Windows-Bildschirm siehst.
2. Drücke die Taste [F1].
3. Es erscheint die Windows-Hilfe
4. Wenn du magst kannst du ja mal in ein paar Themen schmökern. Klicke dazu auf die Registerkarte „Index“ oder „Inhalt“. Klicke in der linken Spalte zweimal schnell hintereinander auf ein Thema, das dich interessiert. Rechts wird dir der Hilfetext angezeigt.
Wahrscheinlich hilft dir die Hilfe jetzt nicht besonders, weil sie etwas schwierig geschrieben ist, aber zumindest weißt du, wo du später im Zweifel mal eine Information suchen kannst.
5. Betätige die Taste [Alt], halte sie fest und drücke dann [F4].
(Das meine ich mit der Schreibweise [Alt] + [F4].)
Dadurch wird das Hilfeprogramm beendet.


Schon hast du gelernt, wie im Betriebssystem Windows zwei Funktionstasten verwendet werden.

Startmenü-Taste



Abbildung 48: Die Startmenü-Tasten

Diese Taste [⇧] und die → Kontextmenütaste gibt es erst seit ungefähr zehn Jahren. Damals hat die Firma Microsoft Tastaturen herausgebracht, die diese Tasten enthielten, um die dazugehörigen besonderen Funktionen ihres damals neuen → Betriebssystems Windows 95 herauszuheben.

Wenn du die Startmenütaste [⇧] tippst, passiert genau das selbe, als wenn du mit der Maus auf den Knopf , ganz unten links auf dem Windows-Bildschirm klickst. Was genau das ist, erfährst du in Kapitel 6.2.1.2!

Kontextmenü-Taste



Abbildung 49: Kontextmenütaste

Mit der Kontextmenütaste erreichst du das selbe wie mit einem Klick auf die *rechte* Maustaste; du rufst das Kontextmenü des gerade ausgewählten Gegenstandes auf. Mehr dazu in Kapitel 6.2.1.5.

Dreierblock



Abbildung 50: Dreierblock der Tastatur

Drucken / Print Screen

Diese Taste war einmal dafür gedacht, das was gerade auf dem Bildschirm angezeigt wird, zu drucken. Auf Englisch nennt man einen solchen Ausdruck auch *Hardcopy*. Beim Betriebssystem Windows wird mit der Taste [Drucken] der Bildschirm in die → Zwischenablage (ein Bereich des Arbeitsspeichers) kopiert. Man kann das dann mit Hilfe eines Grafikprogramms ausdrucken oder anders weiterverarbeiten. Die englische Beschriftung der Taste „Print Screen“ (sprich: print skrien) bedeutet „Drucke Bildschirm“; komischerweise sind auch manche deutsche Tastaturen so beschriftet.

Wenn man [Alt]+[Drucken] betätigt, wird nicht der gesamte Bildschirm sondern nur das gerade → aktive Fenster in die Zwischenablage kopiert.

Rollen / Scroll lock

Mit dieser Taste kann man in manchen Programmen einstellen, dass nicht die → Einfügemarke sich nach oben und unten bewegt, sondern dass sich der Bildschirminhalt unter der Einfügemarke bewegt. Ich persönlich habe sie noch nie benutzt!

Pause / Break

Mit dieser Taste können bei manchen Programmen langwierige Berechnungen unter- oder abgebrochen werden.

Sechserblock



Abbildung 51: Sechserblock der Tastatur

Einf

Stell dir vor, du schreibst einen Text und möchtest mitten drin einen Satz einfügen. Dann soll ja der Text, der schon da steht, weitergeschoben werden, damit man ihn nicht noch einmal schreiben muss.

Es kann aber auch sein, dass man einen Satz ersetzen möchte, also mit den neuen Buchstaben quasi über die alten drüber schreiben möchte, so wie du mit Tintenkiller alte Buchstaben wegmachst und anschließend darüber schreibst.

Mit der Taste [Einf], was die Abkürzung für *Einfügen* ist, kannst du zwischen diesen beiden Möglichkeiten umschalten. Entweder schreibst du im sogenannten *Einfügemodus*, dann wird der alte Text behalten, oder im *Überschreibmodus*, das ist die Geschichte mit dem Tintenkiller. Meistens wird dir irgendwie angezeigt, in welcher Betriebsart du dich gerade befindest. Manchmal hat die Einfügemarke unterschiedliche Formen oder in Word wird am unteren Rand des → Fensters ein „ÜB“ angezeigt, wenn du dich im Überschreibmodus befindest.

Mit [↑]+[Einf] kannst du das was sich in der → Zwischenablage befindet an der Stelle einfügen, an der die Einfügemarke steht.

Entf

Entf steht für *Entfernen*. Damit löscht man das was man zuvor → markiert hat. Wenn man nichts markiert hat, löscht man das Zeichen, das rechts von der Einfügemarke steht.

Mit der Tastenkombination [↑]+[Entf] löscht man die Markierung und verschiebt sie gleichzeitig in die → Zwischenablage, so dass man sie an anderer Stelle wieder → einfügen kann.

Pos 1

Pos 1 steht für Position Eins. Damit ist der Beginn einer Zeile oder eines Dokuments gemeint. Wenn du [Pos1] drückst, springt die Einfügemarke an den Anfang der Zeile, wenn du [Strg]+[Pos1] drückst, an den Beginn des Dokuments.

Beides funktioniert auch in Verbindung mit der Umschalttaste, nur das dabei der Bereich von der aktuellen Position bis zum Zeilenanfang (bei [↑]+[Pos1]) bzw. bis zum Dokumentanfang (bei [↑]+[Strg]+[Pos1]) markiert wird.

Ende

[Ende] macht das selbe wie [Pos1], nur mit dem Zeilen- bzw. Dokument-*ende* statt *-anfang*.

Bild ↑

Mit dieser Taste kann man den oben nicht angezeigten Fensterinhalt in den Anzeigebereich des Fensters holen. Man blättert sozusagen seitenweise rückwärts durch ein angezeigtes Dokument. Wenn man dasselbe mit der Maus machen wollte, würde man den → Rollbalken benutzen.

Auf manchen Tastaturen steht auch [PgUp] für Page Up (sprich: paitsch ap), Seite nach oben.

Bild ↓

Hiermit wird der unten nicht angezeigte Teil eines Dokuments in die Anzeige geholt, man blättert also sozusagen seitenweise vorwärts durch ein Dokument. Auf Englisch heißt diese Taste [PgDn] für *Page Down* (sprich: paitsch daun).

Pfeiltasten



Abbildung 52: Die Pfeiltasten

Die Pfeiltasten, auch Cursortasten genannt, sind sozusagen der Ersatz für die Maus auf der Tastatur. Hiermit kann man die Einfügemarke (oder in manchen Programmen auch irgendein anderes Objekt auf dem Bildschirm) in die jeweils auf den Tasten angegebene Richtung bewegen. Wenn man die Einfügemarke in einem Text bewegen will, geht das mit den Cursortasten oft einfacher als mit der Maus, weil die Einfügemarke von alleine immer genau in eine Zeile (bei *auf/ab*) bzw. genau zwischen zwei Zeichen (bei *links/rechts*) springt. Außerdem kann man durch Drücken von [Strg]+[→] wortweise springen. Drückt man zusätzlich zur Pfeiltaste die Umschalttaste, wird der Text über den man sich be-

wegt → markiert. [↑]+[Strg]+[→] markiert also das Wort rechts der Einfügemarke.

5.2.11.2 Maus

Um es gleich zu sagen: Eigentlich braucht man keine Maus! Man kann normalerweise – abgesehen vielleicht von Grafikeingaben oder Spielen – alles auch mit der Tastatur machen, und wenn man etwas Übung hat sogar schneller als mit der Maus!

Trotzdem ist die Maus eine tolle Erfindung und erleichtert vieles vor allem wenn man kein so erfahrener Bediener ist.



Abbildung 53: Die Maus

Die Maus macht zusätzlich zur Tastatur die gesamte Bildschirmfläche zur Eingabemöglichkeit! Natürlich nicht die Maus alleine! Entscheidend ist wieder das → Programm, das dir auf dem Bildschirm Dinge zeigt, die du mit der Maus anklicken kannst, was dann das Programm wiederum als Befehl oder Dateneingabe verstehen kann. Aber alles hübsch der Reihe nach!

Auf dem Bildschirm befindet sich ein *Mauszeiger*. Das ist meist ein kleiner Pfeil, man kann aber auch andere Symbole dafür einstellen und manche Programme zeigen ihre eigenen Mauszeigersymbole.

Diesen Mauszeiger kann man auf dem Bildschirm bewegen: Wenn man die Maus nach links bewegt, bewegt sich der Mauszeiger nach links. Du darfst raten oder ausprobieren, was der Mauszeiger macht, wenn man die Maus nach oben, unten oder rechts bewegt!

Der Mauszeiger bewegt sich dabei immer *über* den Dingen, die auf dem Bildschirm dargestellt werden.

Dann hat eine Maus noch mindestens zwei Tasten, manchmal auch eine dritte Taste oder ein oder sogar zwei kleine Rädchen, eins zwischen den Tasten, das zweite dort wo der Daumen sitzt.

Mit den Tasten kann man folgende Aktionen machen (zu den Rädchen kommen wir später):

- Einmal auf eine der beiden Tasten klicken
- Zweimal schnell hintereinander auf die linke Taste klicken
- Auf beide Tasten gleichzeitig klicken
- Eine der beiden Tasten klicken, festhalten und zwischenzeitlich die Maus bewegen

Natürlich kann man sich noch mehr Sachen ausdenken, die man mit den Maustasten anstellen könnte, wie zum Beispiel abwechselnd links-rechts-links klicken

oder so etwas, aber nur die Dinge, die ich genannt habe, sind normalerweise in der → Benutzerschnittstelle mit irgendwelchen Bedeutungen versehen. Und das sind:

Einmal linke Taste klicken

Damit wird normalerweise der „Gegenstand“, über dem sich der Mauszeiger gerade befindet, ausgewählt (um später damit weitere Aktionen durchführen zu können), oder es wird eine Aktion damit ausgelöst. Wenn auf dem Bildschirm zum Beispiel ein Schaltknopf dargestellt ist, bedeutet „einmal mit der linken Taste darauf klicken“ so viel wie, den Schaltknopf zu betätigen.

Einmal rechte Taste klicken

Beim → Betriebssystem Windows wird durch diese Aktion das Kontextmenü aufgerufen, das wir in Kapitel 6.2.1.5 besprechen werden.

Andere Programme können diese Taste mit anderen Funktionen belegen. Viele Zeichenprogramme benutzen die rechte Maustaste wie die linke, nur dass man mit einer anderen Farbe malt, wenn man die rechte Taste benutzt.

Zweimal schnell hintereinander linke Taste klicken

Damit löst man meist eine Aktion des Programms aus. Wenn man beispielsweise auf dem → Windows-Desktop ein Symbol doppelt anklickt, wird das zugehörige Programm gestartet.

Die Zeit die höchstens zwischen den zwei Klicks liegen darf, damit es noch ein Doppelklick ist, kann man unter Start / Einstellungen / Systemsteuerung / Maus einstellen. Dort kann man auch die Geschwindigkeit verändern, mit der sich der Mauszeiger bewegt.

Beide Tasten gleichzeitig klicken

Mit dieser Aktion kann man schon begonnene drücken-festhalten-bewegen-Aktionen (s.u.) abbrechen. Sie kann aber von verschiedenen Programmen unterschiedlich benutzt werden.

Wenn die Maus drei Tasten hat, bedeutet die mittlere so viel wie die beiden anderen gleichzeitig zu drücken.

„Gleichzeitig“ bedeutet nur, dass die zweite Taste gedrückt werden muss, bevor man die erste loslässt. Beginn und Ende des Tastendrückens können jeweils unterschiedlich sein.

Linke Taste klicken, festhalten, Maus bewegen

Damit werden meist „Gegenstände“ bewegt. Zum Beispiel kannst du die Symbole auf der Windows-Oberfläche anordnen, indem du sie auf diese Weise verschiebst.

Rechte Taste klicken, festhalten, Maus bewegen

Diese Aktion ist identisch wie das Bewegen mit der linken Taste, nur dass man beim Loslassen noch einmal ein → Kontextmenü angezeigt bekommt, das einem mehrere Möglichkeiten gibt. Wenn man also nicht genau weiß, was eine linke-Taste-festhalten-bewegen-Aktion bewirkt, sollte man das selbe lieber mit der rechten Taste machen.

Eine weitere Aktion bezieht sich auf den Mauszeiger allein, ohne das Betätigen irgendwelcher Tasten. Damit kann man nämlich zeigen, das heißt an einer Stelle stehen bleiben. Je nachdem, wie die → Benutzerschnittstelle des Programms festgelegt ist, kann auch das schon „auswählen“ bedeuten! Bei Windows kann man das einstellen, ob man das so haben will.

Das eventuell vorhandene Rädchen zwischen den Tasten ersetzt die → Rollbalken eines Fensters: Man kann damit den Inhalt eines Fensters nach oben und unten rollen, um auch die gerade nicht angezeigten Teile zu sehen. Ein eventuell vorhandenes zweites Rädchen rollt den Fensterinhalt nach links und rechts.

Auch wenn sich das Ganze jetzt ziemlich abstrakt und schwierig liest, wirst du feststellen, dass die Bedienung eines Programms mit der Maus in der Regel sehr einfach ist. Gute Programme kann man *intuitiv* bedienen, das bedeutet, dass du ohne irgend etwas über das Programm gelesen zu haben, allein durch die Darstellung auf dem Bildschirm weißt, wie du es bedienen musst.

Das klappt deswegen, weil einem die Maus ziemlich schnell so vorkommt wie eine verlängerte Hand. Mit der Maus etwas auf dem Bildschirm anzuklicken ist genauso einfach, wie mit dem Finger darauf zu tippen.

Wie funktioniert eine Maus?

Es gibt zwei verschiedene Arten von Mäusen: Optische Mäuse, die mit einem Lichtstrahl messen, wie weit und in welche Richtung man die Maus bewegt. Sie haben den Vorteil, dass sie unempfindlicher gegen Schmutz sind, und den Nachteil, dass sie teurer sind als die zweite, immer noch am weitesten verbreitete Art von Mäusen, die mechanischen mit Kugel:

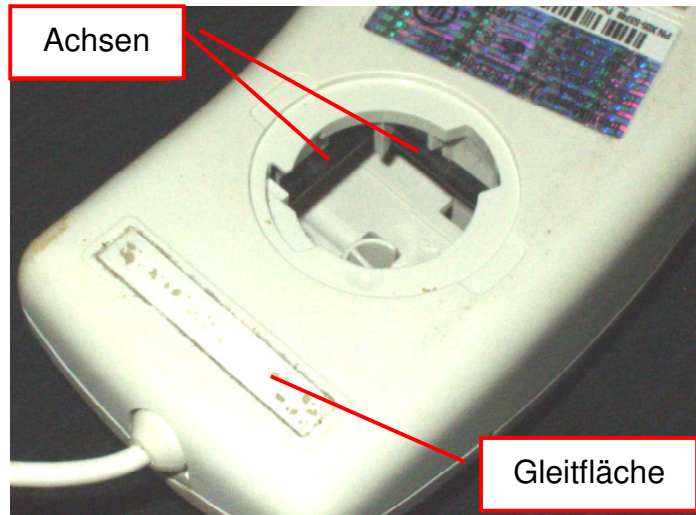
Der Vorläufer der Maus war der sogenannte *Trackball*, was – wer hätte das gedacht – Englisch ist und so viel heißt wie Spurball. Das war eine Kugel in einer Fassung, die man in alle Richtungen drehen konnte, um den Mauszeiger – par-

don den Trackballzeiger zu bewegen. Irgendein Schlaukopf kam auf die Idee, dass man den Trackball auch umdrehen und hin und her schieben könnte. Durch die Bewegung würde die Kugel bewegt werden. So ist das bis heute geblieben: Wenn du deine Maus einmal auf den Rücken legst, wirst du sehen, dass sie unten ein Loch hat, aus dem eine Kugel herauschaut. Den Ring um dieses Loch kann man bei meiner Maus gegen den Uhrzeigersinn drehen, dann kann man den Ring abnehmen und die Kugel herausnehmen. Bei anderen Mäusen kann der Öffnungsmechanismus anders sein.



Ring

Kugel



Achsen

Gleitfläche

Abbildung 54: Maus von unten gesehen

Abbildung 55: Maus von innen

Innen siehst du dann zwei dünne Achsen. Wenn du eine dieser Achsen hin und her drehst, bewegt sich der Mauszeiger am Bildschirm entweder auf und ab oder hin und her. Bei der anderen Achse macht er das jeweils andere.

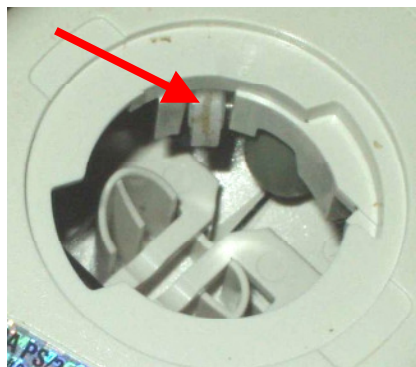


Abbildung 56: Andruckrolle der Maus für ihre Kugel

Außerdem siehst du ein kleines Rädchen (Pfeil in Abbildung 56), das federnd gelagert ist.

Jetzt kannst du dir vorstellen, wie die Maus funktioniert: Die Kugel wird von dem federnden Rädchen immer gegen die beiden Achsen gedrückt. Wenn man die Maus bewegt, rollt die Kugel. Damit das besonders gut klappt, ist die Kugel mit Gummi beschichtet und aus möglichst schwerem Material. Durch das Rollen dreht die Kugel die beiden Achsen in der Maus. Die Drehungen der Achse werden in der Maus gezählt, indem an den Achsen kleine Flügerrädchen sitzen, die sich mitdrehen und beim Drehen Licht-

schranken unterbrechen. Bei jedem Unterbrechen der Lichtschranke wird ein kleiner Impuls an den Computer gesendet. Der merkt daran, dass die Maus sich bewegt hat, und bewegt den Mauszeiger dementsprechend auf dem Bildschirm. Es gibt also ein kleines → Programm in deinem → Arbeitsspeicher, das die Mauspulse zählt, den Mauszeiger bewegt und auch die Tastenklicks registriert. (Unter den Tasten sitzen einfach Schalter, die beim Drücken der Taste schließen, also Strom durchlassen, und beim Loslassen wieder öffnen.)

Da du die Maus gerade geöffnet hast, solltest du sie auch gleich reinigen! Auf den Abbildungen oben kannst du erkennen, dass zumindest meine Maus es nötig hatte! Man erkennt grauen Schmutz auf der Andruckrolle und den Gleitflächen; auf den Achsen sieht man nur deshalb keinen, weil die Achsen schwarz sind!

Spätestens wenn der Mauszeiger sich nicht mehr gleichmäßig bewegt, solltest du den Schmutz von den Achsen, dem Andrückrädchen und den Gleitflächen entfernen. Natürlich solltest du dabei keine Gewalt anwenden, aber du kannst ihn vorsichtig mit dem Fingernagel abkratzen.

Wem das Reinigen zu umständlich ist, der kann auch eine optische Maus an seinem Computer installieren: Die hat keine Kugel, sondern eine kleine Lampe und einen lichtempfindlichen Sensor, der Helligkeitsschwankungen bei Bewegung der Maus registriert.

Ein Tipp zur Mausbedienung:

Manche Computerneulinge fallen fast vom Stuhl, wenn sie die Maus bewegen, weil sie mit dem Mauszeiger noch weiter nach rechts wollen, aber die Maus schon so weit rechts liegt. Da du jetzt weißt, wie eine Maus funktioniert, kannst du das besser: Man kann die Maus einfach hochheben und an einer anderen Stelle wieder absetzen. Wenn die Maus nicht auf einer Unterlage verschoben wird, bewegt sie den Mauszeiger nicht, weil die Kugel nicht rollt!

5.3 Bildschirm

Der Bildschirm ist neben der Zentraleinheit die wichtigste Hardware, die du brauchst, denn die meisten Ein- und Ausgaben laufen über ihn.

Man unterscheidet im Prinzip zwei Typen von Bildschirmen: Röhrenbildschirme, die so ähnlich aussehen und auch so arbeiten wie Fernseher, und Flachbildschirme. Letztere gibt es in verschiedenen Technologien, eine heißt TFT, was für „Thin Film Transistor“ steht. Man spricht auch oft von LCD-Bildschirmen. LCD bedeutet „Liquid Crystal Display“, zu deutsch *Flüssigkristall-Bildschirm*. Warum die so heißen und wie sie genau funktionieren, soll uns hier nicht weiter interessieren.

Wichtig für dich als Nutzer ist eigentlich nur, was du schon im Experiment ausprobiert hast: Wie viele Bildpunkte und wie viele Farben mit welcher Frequenz (Bilder pro Sekunde) dargestellt werden können. Das hängt aber nicht nur vom Monitor sondern vor allem auch von der → Grafikkarte ab, so dass man immer beides berücksichtigen muss.

Wenn dein Bild flimmert, solltest du die Bildwiederholfrequenz vergrößern. Das geht auch über das Fenster aus Experiment 2 auf Seite 38, indem du unter Nummer 4 auf den Knopf „Erweitert“ klickst. Sollte man die Wiederholrate nicht mehr vergrößern können, kannst du zuvor die Auflösung verringern, dann geht es vielleicht.

Meist kann man auch am Bildschirm selbst noch einiges einstellen, um ein optimales Bild zu erhalten: Dazu gehört zum Beispiel Helligkeit und Kontrast des Bildes. Man kann aber auch die verschiedensten Arten von Verzerrungen korrigieren. Wenn du wissen willst wie das alles geht, musst du die Bedienungsanleitung deines Monitors lesen oder es dir von jemandem zeigen lassen, der sich mit dem Monitor auskennt.

5.4 Beamer

Das ist eine Sonderform des Bildschirms. Wörtlich übersetzt bedeutet dieses englische Wort *Strahler*. Damit kann man das Bildschirmbild an eine Wand projizieren, so dass viele Leute es gleichzeitig sehen können. Das ist sehr hilfreich, wenn man zum Beispiel einen Vortrag halten möchte und dazu Bilder oder eine Präsentation zeigen möchte.

5.5 Drucker

Eine der ersten wichtigen Aufgaben von Computern war der Ersatz der Schreibmaschine. Mit der Erfindung von → Internet und → e-Mail (elektronische Post) kam die Illusion auf, man könnte auf das Drucken irgendwann ganz verzichten. Es hat dazu viele Versuche gegeben, und zwar aus gutem Grund: Etwas, das ich direkt in meinem Computer als Nachricht erhalte, also zum Beispiel eine e-Mail oder eine → Datei auf einer → CD, kann ich sofort mit dem Computer weiter verarbeiten! Erhalte ich etwas auf Papier, muss ich die darauf geschriebene Information erst wieder mühsam in meinen Rechner hineinbringen, um weiter mit ihr arbeiten zu können.

Trotzdem ist der Papierverbrauch seit dem Siegeszug der Computer gestiegen statt gesunken! Das liegt zum einen daran, dass es so herrlich einfach ist, etwas aus dem Computer zu Papier zu bringen – es kostet nur einen Mausklick. Zum anderen ist etwas, das auf Papier geschrieben steht, den Menschen ein-

fach sympathischer! Man nimmt gerne ein Buch mit ins Bett, um es zu lesen, aber keinen Computer!

Aus dieser Vorrede erkennst du sicher schon, wofür ein Drucker da ist: Man braucht ihn, wenn man Dinge, die im Computer zum Beispiel am Bildschirm dargestellt werden, auf Papier haben möchte. Was auf dem Papier erscheinen soll, muss natürlich irgendwie vom Computer zum Drucker gelangen. Das geht über eine → Schnittstelle, von der wir schon so viel gehört haben, also einfach gesagt ein Kabel. Früher war das meistens die parallele Schnittstelle (siehe Abschnitt 5.2.5); heute werden die meisten Drucker über die USB-Schnittstelle (siehe Abschnitt 5.2.2) angeschlossen.

Was man an einen Drucker senden möchte, ist entweder Text oder Bilder. Aus dem Kapitel über Nullen und Einsen weißt du, dass man zum Darstellen von Text sehr viel weniger Bits braucht als für Grafik, wenn man einen Code wie den ASCII-Code aus Tabelle 6 benutzt. Einzige Voraussetzung ist, dass der Drucker auch weiß, wie das Zeichen aussieht, das er drucken soll, wenn er einen bestimmten ASCII-Code empfängt, er muss also eine Schrift „eingebaut“ haben oder vielleicht sogar mehrere, unter denen man auswählen kann.

*Eine Schriftart nennt man auch einen **Font**. Arial, Times, Courier sind die Namen bekannter Fonts, aber es gibt noch viel, viel mehr.*

Mehr über Schriftarten erfährst du im Kapitel 7.3.1.

Früher hat man beim Drucken streng zwischen Bildern und Grafiken unterschieden, damit man nicht so viele Daten übertragen musste. Heute ist die Datenübertragung kein so großes Problem mehr und man überträgt auch Text als Grafikdaten. Das hat den Vorteil, dass die Schriftarten auf dem Computer gespeichert sein können. Dort kann man sie viel einfacher um weitere Schriftarten ergänzen; außerdem weiß der Computer dann genau, wie groß jeder Buchstabe ist. Das ist nämlich Voraussetzung dafür, dass ein Textverarbeitungsprogramm schon beim Schreiben den Text so anzeigen kann, wie er später auf dem Papier erscheint. Diese Fähigkeit nennt man auch *WYSIWYG*. Das ist die englische Abkürzung für „What you see is what you get“ und bedeutet so viel wie: „Du bekommst genau das ausgedruckt, was du auf dem Bildschirm siehst!“

Außer dem was gedruckt werden soll, muss noch etwas anderes an den Drucker übertragen werden: Der Drucker muss ja beispielsweise wissen, wenn er ein neues Blatt beginnen oder irgendwelche seiner Einstellungen ändern soll (zum Beispiel Umschalten von Hochformat auf Querformat).

Umgekehrt wäre es ganz schön, wenn dem Benutzer am Bildschirm angezeigt werden würde, wenn der Drucker irgendein Problem hat, zum Beispiel wenn kein Papier mehr drin ist.

Damit das alles funktioniert, braucht man ein kleines Programm auf dem Computer, in dem steht, wie der Drucker gesteuert werden muss, und das die Meldungen des Druckers in verständliche Meldungen für den Benutzer umwandeln kann. Dieses Programm weiß zum Beispiel auch, auf welchen Bereichen des Papiers der Drucker überhaupt nur drucken kann; oben und unten braucht er nämlich oft einen kleinen Rand, wo er nicht drucken kann, weil er dort das Papier festhalten muss.

*Ein solches Programm, das den Drucker steuert, heißt **Druckertreiber**. Wenn du einen neuen Drucker anschließt, musst du meistens auch einen Druckertreiber auf deine Festplatte kopieren. Wie das geht steht im Handbuch des Druckers.*

Wenn die Daten nun endlich durch das Kabel oder die → Infrarotschnittstelle beim Drucker angekommen sind, müssen sie ja noch zu Papier gebracht werden. Je nachdem auf welche Weise das geschieht, unterscheidet man verschiedene Druckertypen: Die zwei gängigsten sind Laserdrucker und Tintenstrahldrucker. Beide gibt es in Ausführungen für Schwarz-Weiß- und für Farbdruk.

Laserdrucker können recht schnell und preiswert drucken. Dafür sind sie in der Anschaffung so teuer, dass sie sich für Otto Normalverbraucher normalerweise nicht lohnen; das gilt insbesondere für Farb-Laserdrucker.

Tintenstrahldrucker sind auch als Farbdruker sehr preiswert. Dafür sind die Patronen, die die Tinte enthalten, sehr teuer, wodurch die Kosten pro gedruckte Seite ziemlich groß sind. Das Drucken dauert auch um einiges länger als mit einem Laserdrucker. Trotzdem kann man mit ihnen brillante Bilder drucken, vor allem wenn man Spezialpapier und Spezialtinte verwendet!

Druckgeschwindigkeiten werden in Seiten pro Minute angegeben. Man muss aber erstens darauf achten, dass die Geschwindigkeiten für Schwarzweiß- und Farbdruk unterschiedlich sind und zweitens, dass die Geschwindigkeit auch davon abhängen kann, wie die Seite bedruckt ist.

Generelle Qualitätsmerkmale für Drucker sind ihre Auflösung und ihr Speicher:

Auflösung meint genau wie beim Bildschirm die Anzahl Punkte, die er nebeneinander darstellen kann, nur dass man das beim Drucker nicht auf die Bildschirmbreite bezieht sondern auf einen Zoll (= 2,54cm). Die Einheit in der die

Auflösung angegeben wird, heißt demzufolge *dpi*, was die Abkürzung ist für *dots per inch*, also Punkte pro Zoll. Eigentlich kann man mit 600dpi schon sehr ordentliche Ausdrücke machen, aber die meisten Drucker, auch preiswerte, können heute sehr viel mehr! In dieser Hinsicht kann man also kaum falsch kaufen.

Drucker haben genau wie der Computer selbst Arbeitsspeicher, weil das Drucken sehr viel länger dauert als das Übertragen dessen, was gedruckt werden soll, vom Computer zum Drucker. Die Druckdaten werden deshalb in einem Speicher im Drucker zwischengespeichert. Je größer der Speicher des Druckers ist, desto eher braucht sich der Computer nicht mehr um den Druckvorgang zu kümmern.

5.6 Verbindung mit anderen Computern

Im Kapitel 5.2.9 hast du schon gelesen, dass du deinen Computer über kurze oder lange Entfernungen mit anderen Computern verbinden kannst. Dazu brauchst du ein Gerät, das die Daten die du übertragen willst für die Übertragungstrecke aufbereitet, und das die Daten sendet und empfängt.

Je nachdem über welche Leitung du deinen Computer verbindest, brauchst du unterschiedliche Geräte. Alle haben zwei Schnittstellen: Mit der einen werden sie an deinen Computer angeschlossen; bei eingebauten Geräten ist das meist eine → PCI-Schnittstelle. An die andere kommt das Netzkabel; das ist im einfachsten Fall die Telefonleitung!

Wenn du einen ganz normalen → analogen Telefonanschluss benutzt, heißt das Gerät *Modem*. Das ist ein Kunstwort aus Modulator/Demodulator und bedeutet, dass beim Senden die digitalen Daten in Töne umgewandelt (moduliert) werden und beim Empfangen Töne wieder in digitale Daten zurückverwandelt (demoduliert) werden. Das muss man machen, weil die Telefonleitung ja eigentlich dafür gedacht und konstruiert ist, um Töne, nämlich deine Stimme, zu übertragen.

In Wirklichkeit wird natürlich nichts in Töne verwandelt, denn auch die würde man nicht über eine elektrische Leitung transportieren können, sondern ein Modem wandelt deine Daten in das, was dein Telefon aus Tönen machen würde: Elektrischen Strom mit der gleichen → Frequenz wie die Töne.

Wenn du ein moderneres, digitales Telefon, also ISDN⁴ oder gar einen DSL⁵-Anschluss benutzt, brauchst du im ersten Fall eine ISDN-Karte, und im zweiten eine normale Netzwerkkarte (mit der du auch zwei Computer per → Ethernet miteinander verbinden kannst) und einen sogenannten Splitter, an den dein Computer, das Telefon und andere Geräte, die die Telefonleitung benutzen, angeschlossen werden.

Falls du dicht beieinander stehende Computer verbinden willst, benutzt du natürlich keine Telefonleitung sondern ein Netzkabel. Das schließt du an eine Netzwerkkarte an.

5.7 Die Augen des Computers

5.7.1 Scanner

Im Kapitel 3.5 hast du gesehen, dass es ziemlich mühselig ist, ein Bild Pixel für Pixel zu malen! Natürlich gibt es Grafikprogramme, mit denen man trotzdem ziemlich schnell schöne Bilder zaubern kann, aber es wäre doch eine echte Hilfe, wenn man Bilder, die es schon auf Papier gibt, irgendwie in den Computer bekommen würde! Man könnte sie dort vielfältig weiterverarbeiten: Die Bilder verändern (noch schöner machen), in Texte einfügen, mit der elektronischen Post fast kostenlos in alle Welt verschicken, wieder ausdrucken (also vervielfältigen), usw.

Ein Gerät, mit dem man Papiervorlagen in → digitaler Form in den Computer bekommt, mit dem man also *digitalisieren* kann, heißt *Scanner* (sprich: skän-ner). Das ist mal wieder Englisch und bedeutet *Abtaster*. Früher gab es Scanner, die die Papiervorlage durchgezogen haben oder die man mit der Hand über die Vorlage fahren musste. Heute gibt es fast nur noch *Flachbettscanner*, bei denen man die Vorlage auf eine Glasscheibe legt.

Vielleicht hast du – zum Beispiel in deiner Schule – schon einmal einen Kopierer gesehen. Das ist nichts anderes als ein Flachbettscanner mit eingebautem Laserdrucker.

Du weißt inzwischen, dass digitale Bilder aus Punkten bestehen. Man sagt auch, sie sind *gerastert*. Der Scanner muss also die Vorlage an jedem Punkt abtasten – daher der Name! – und nachsehen, welche Farbe (bzw. Helligkeit,

⁴ ISDN steht für Integrated Services Digital Network. Das ist ein Name für den digitalen Telefonstandard.

⁵ DSL ist die Abkürzung für Digital Subscriber Line. Dabei handelt es sich um einen sehr schnellen digitalen Datenübertragungsstandard für Telefonleitungen.

wenn es um Schwarzweiß geht) die Vorlage an der betreffenden Stelle hat. Den Bitwert, der zu dieser Farbe gehört, sendet der Scanner an den Computer. Damit das ganze schneller geht, macht er das Abtasten mit ganz vielen Punkten auf einmal, nämlich immer mit einer ganzen Zeile von Punkten. Die Sensoren, die er dazu braucht, sitzen auf einem Schlitten, der unter der Glasscheibe langsam an der Vorlage vorbei fährt und sie auch beleuchtet.

Die Qualität eines Scanners misst sich unter anderem wieder daran, welche Auflösung er maximal erreicht, das heißt, wie feinmaschig er das Netz der Punkte, an denen er abtastet, machen kann. Wie bei Druckern misst man das wieder in $\rightarrow dpi$.

Oft ist die Auflösung eines Scanners in den zwei Richtungen des Bildes unterschiedlich.

Natürlich kann man auf einen Scanner auch eine Vorlage mit Text legen. Aber nachdem du jetzt weißt, wie der Scanner arbeitet und du dich an Kapitel 3.4 und 3.5 erinnerst, und daran, auf welche unterschiedlichen Arten Texte und Bilder im Computer dargestellt werden, leuchtet dir sicherlich ein, dass der Text, wenn er vom Scanner in den Computer gelangt ist, erst mal als *Bild* gespeichert wird, oder? Der Scanner hat ja erst mal nur Punkte abgetastet und Farbe gemessen; was die Farbpunkte darstellen, ein Auto, einen Baum, ein Haus oder einen Buchstaben, das kann er nicht erkennen!

Trotzdem ist es manchmal hilfreich, wenn ein Text auch als Text gespeichert wird. Erstens braucht er dann viel weniger Speicherplatz, und zweitens möchte man den Text ja vielleicht verändern, irgendwo einen Satz einfügen oder die vielen Rechtschreibfehler korrigieren.

Zur Lösung dieses Problems gibt es Programme, die in einem Bild Buchstaben finden, und in der Lage sind, diese wieder als Text zu speichern. Auf Englisch heißen solche *Texterkennungsprogramme OCR-Software*. OCR steht dabei für Online Character Recognition (sprich etwa: onlein kerekter rekognischen), was so viel heißt wie Buchstabenerkennung während des Betriebes.

5.7.2 Digitalkamera

Ein Trend – um nicht zu sagen Boom – in neuerer Zeit sind Digitalkameras: Mit Ihnen macht man Fotos wie mit einem Fotoapparat, nur dass die Bilder nicht chemisch auf einem Film belichtet und gespeichert werden, sondern direkt in digitaler Form erzeugt und im Fotoapparat gespeichert werden. Dazu hat eine Digitalkamera einen ähnlichen Sensor wie ein Scanner, nur dass der nicht nur eine Zeile abtastet sondern eine *Fläche*! Nämlich die Fläche, die früher auf dem Film als Bild belichtet worden wäre.

Der große Vorteil daran ist, dass die Bilder sofort verfügbar sind; man muss sie nicht erst entwickeln lassen sondern kann sie sich gleich ansehen und nötigen-

falls auch wieder löschen um die Aufnahme zu wiederholen, falls sie nichts geworden ist. (Allerdings ist damit auch die Spannung weggefallen, was denn wohl alles auf dem Film drauf war, den man zum Entwickeln gebracht hat!) Außerdem liegen die Fotos digital vor und können ohne Umweg über einen Scanner direkt in den Computer übertragen werden (meist über die → USB-Schnittstelle).

Auch bei Digitalkameras ist ein Qualitätsmerkmal wieder die Auflösung, die hier in Pixeln gemessen wird: Eine heute übliche Digitalkamera hat zum Beispiel fünf Mega-Pixel; das bedeutet, dass auf der Fläche, auf die das Bild fällt, fünf Millionen Sensoren sitzen, die den Farbwert des Bildes an ihrer jeweiligen Stelle messen! Dabei sind diese Sensoren so klein, dass die Fläche, auf die das Bild fällt, um einiges kleiner sein kann als bei einem herkömmlichen Fotoapparat. Das wiederum erlaubt eine kleinere und billigere *Optik*, so nennt man die Gesamtheit der Linsen, die dafür sorgt, dass ein scharfes Bild auf der Sensorfläche bzw. dem Film entsteht.

Die Speicher, auf denen eine Digitalkamera die Bilder speichert, arbeiten ähnlich wie die in Kapitel 5.1.3.3 beschriebenen.

Es gibt auch digitale Videokameras. Da bei den üblichen 25 Bildern pro Sekunde, aus denen ein Film besteht (vgl. Seite 34!), die Datenmenge sehr schnell sehr groß wird, speichert man Filme auf Magnetbändern. Außerdem ist deshalb die Auflösung der Sensoren nicht so groß, denn – du erinnerst dich – je mehr Pixel ein Bild hat, desto mehr Speicherplatz brauche ich ja auch dafür!

Heute können fast alle Kameras zumindest behelfsmäßig auch das jeweils andere: Mit digitalen Videokameras kann man auch Fotos machen und mit digitalen Fotoapparaten kann man auch – allerdings nur sehr kurze – Filme drehen.

6 Bedienung

Nach so viel grauer Theorie kommen wir endlich zu dem, was du eigentlich wissen willst, nämlich wie du deinen Computer bedienen musst, damit er das tut, was du gerne möchtest!

Im Prinzip könnten dieses und die nächsten zwei Hauptkapitel komplett als **Experiment** ausgewiesen sein, denn natürlich sollst du alles, was hier beschrieben wird, auch gleich ausprobieren!

Wir gehen davon aus, dass dein Rechner richtig angeschlossen und betriebsbereit ist. Du musst ihn nur noch einschalten.

Alle Beschreibungen versuche ich möglichst allgemein zu halten. Wo das nicht geht, beziehen sie sich auf das → Betriebssystem Windows 2000 professional.

6.1 Booten

Was nach dem Einschalten von ganz alleine passiert, nennt man „booten“ (sprich: buuten).

„Boot“ ist das englische Wort für Stiefel; die hatten früher einen sogenannten „bootstrap“, einen Lederstreifen, an dem man ziehen konnte, um sie einfacher anzuziehen. Daraus hat sich dann die Bedeutung von bootstrap in „etwas beginnen“ verwandelt, denn wenn man seine Stiefel anzieht, beginnt man je meistens etwas – zum Beispiel eine Wanderung. Später wurde das Wort dann wieder auf boot verkürzt.

Was bedeutet *anfangen* für einen Computer?

Erinnere dich an unsere Modellrechner: Es muss eine Schalterstellung, also einen Befehl geben, den der Computer ausführt, wenn er eingeschaltet wird. Diese Befehle stehen im → BIOS und sorgen zum Beispiel dafür, dass der Computer seinen Arbeitsspeicher findet und „weiß“ wie groß der ist, und dass er die Laufwerke findet, vor allem das Laufwerk und die Stelle auf diesem Laufwerk, an der steht, was er tun soll, sobald er das BIOS abgearbeitet hat.

„Wissen“ bedeutet bei einem Computer, dass eines seiner → Register mit der entsprechenden Information gefüllt ist. Informationen, wo etwas steht, heißen bei einem Computer *Adresse*. Genauso ist es, wenn du jemandem sagst, wo du wohnst, dann nennst du auch deine Adresse. Adressen können Nummern eines Speicherplatzes im Arbeitsspeicher aber auch Zahlen sein, mit denen bestimmte Geräte angesprochen werden, und die über einen → Bus übertragen werden.

Die Befehle, die der Rechner nach dem BIOS beim Booten benötigt, stehen meistens auf der Festplatte, können aber auch auf einer Diskette stehen. In je-

dem Fall ist für diese Befehle ein besonderer Bereich reserviert, den man *Bootsektor* nennt.

Nun könnte es ja sein, dass das Laufwerk, von dem der Rechner booten soll – eben meistens die Festplatte – kaputt ist. Dann könnte der Rechner nicht mehr starten. Man kann deshalb im BIOS eine Reihenfolge der Laufwerke einstellen, von denen der Rechner versuchen soll, zu starten. Wenn dann die Festplatte kaputt ist, kann man eine Diskette, CD oder DVD einlegen, von der aus der Rechner starten kann. – Voraussetzung dafür ist allerdings, dass man eine solche Startdiskette oder CD hat! Neueren Rechnern ist meistens eine CD oder DVD beigelegt; da steht etwas von „Recovery“ drauf, was Englisch ist und „Wiederherstellen“ bedeutet. Für ältere Geräte kann man sich selbst eine startfähige Diskette anlegen. Das geht ganz einfach:

Wähle den Menüpunkt

Start/Programme/Zubehör/Systemprogramme/Sicherung

Klicke auf der Registerkarte „Willkommen“ den Punkt „Notfalldiskette“ und folge den Anweisungen auf dem Bildschirm. Alles was du sonst noch brauchst, ist eine 3,5“-Diskette.

Die fertigestellte Notfalldiskette solltest du gut aufheben und im Zweifelsfall – nämlich dann wenn dein Rechner mal nicht mehr startet – wiederfinden!

Normalerweise braucht man im BIOS keine Änderungen vorzunehmen und sollte es auch nicht tun, wenn man nicht ganz genau weiß, was man macht! Nur um dir eine Vorstellung davon zu geben, was alles darin eingestellt werden kann, kannst du folgendes Experiment durchführen:

Experiment 7: BIOS

- 1.) Starte deinen Rechner
- 2.) Achte auf den Text, der auf dem Bildschirm erscheint
- 3.) Es erscheint eine Textzeile, in der (auf Englisch!) steht, welche Taste man für **Setup** drücken soll. Wenn du diese Zeile nicht findest, probiere entweder die Taste [F2] in der obersten Zeile der Tastatur oder die Lösch taste [Entf], die manchmal auch mit [Del] beschriftet ist.
- 4.) Drücke die Taste noch während die Textzeilen angezeigt werden.

- 5.) Du gelangst in das BIOS-Programm, wo man einige Einstellungen vornehmen kann. Die Menüs und Einstellungen sind jedoch meist in Englisch; lasse dir also ggf. übersetzen, was du nicht verstehst.
- 6.) In dem Programm wird dir auch angezeigt, welche Taste du für welche Aktion brauchst. Sieh dir die Einstellungen an und staune ein wenig. Verändere nichts ohne genau zu wissen, was du tust! Lass dir ggf. helfen!
- 7.) Verlasse das BIOS mit dem dort angegebenen Befehl, meistens die Taste [Esc].

Wir haben gesagt, aus dem BIOS lernt der Rechner unter anderem, welche Laufwerke er zur Verfügung hat. Du als Benutzer wirst dem Computer auch bei vielen Gelegenheiten sagen müssen, auf welches Laufwerk sich dein Befehl bezieht, und umgekehrt muss der Computer ein Laufwerk benennen können, um dir Informationen darüber zu geben. Die Vereinbarung darüber, wie welches Laufwerk heißt, ist ein Teil der → Benutzerschnittstelle.

In den Anfangstagen des Personal Computers (PC) hatte man nur selten einen solchen Luxus wie Festplatten. Damit man nicht so oft Disketten wechseln musste, hatte man dafür meist zwei Diskettenlaufwerke, eins für das Betriebssystem und eins für Programme und Daten, mit denen man gerade arbeiten musste. Auch zum Kopieren waren zwei Diskettenlaufwerke praktisch. Damals hat man festgelegt, dass die Laufwerke mit Buchstaben durchnummeriert werden und dass die ersten beiden Buchstaben *A:* und *B:* für die wichtigen, immer vorhandenen zwei Diskettenlaufwerke sind. Wenn jemand eine Festplatte hatte, konnte er sich mit Sicherheit auch zwei Diskettenlaufwerke leisten. Deshalb hieß die Festplatte immer „*C:*“. So ist es bis heute geblieben, nur dass man wenn überhaupt nur noch ein Diskettenlaufwerk hat. Deshalb fehlt in der Nummerierung der Laufwerke der Buchstabe *B:* und manchmal sogar das *A:*! Der Doppelpunkt kennzeichnet eine Laufwerksangabe. Verzeichnisse beginnen immer mit einem *Back-Slash* (sprich bäcksläsch). Das ist Englisch und bedeutet Rückwärtsschrägstrich. Das ist dieses Zeichen „\“, das du mit [AltGr]+[ß] erzeugen kannst.

Alle weiteren Laufwerke wie CD- oder DVD-Laufwerk oder auch ein Speicher an der → USB-Schnittstelle werden automatisch mit *D:*, *E:*, *F:* usw. durchnummeriert.

Wenn der Bootvorgang beendet ist, erscheint auf deinem Bildschirm die Benutzeroberfläche (die → Benutzerschnittstelle) des Betriebssystems, um das es im nächsten Kapitel geht.

6.2 Betriebssystem

*Das **Betriebssystem** ist das → Programm, das du für den Betrieb deines Rechners brauchst. Es ist die → Schnittstelle zwischen den Ressourcen des Computers und dem Benutzer oder anderen Programmen.*

***Ressourcen** – in klassischem deutsch würde man dazu „Betriebsmittel“ sagen – sind der Prozessor, der Arbeitsspeicher, alle Laufwerke und an den Computer angeschlossene Geräte wie Bildschirm oder Drucker.*

Betrieb des Rechners heißt zunächst einmal, → Programme zu starten und „laufen zu lassen“. Diese Programme steuerst du dann aber auch durch Eingaben mit der Maus oder der Tastatur, sie lesen oder schreiben Daten von CDs, Disketten oder Festplatten oder zeigen dir etwas auf dem Bildschirm an. Alle diese Möglichkeiten stellt das Betriebssystem den anderen Programmen und damit auch dir zur Verfügung.

Das Betriebssystem selbst ist auch ein Programm! Manchmal sind die Grenzen zwischen Betriebssystem und den anderen sogenannten *Anwendungsprogrammen*, fließend: Wenn du zum Beispiel eine Bilddatei von einer CD auf deine Festplatte kopieren willst, stellt dir das Betriebssystem die Möglichkeiten dafür zur Verfügung, ohne dass du ein extra Anwendungsprogramm dafür starten musst.

Egal wie wir es nun nennen, eines ist *allen* Programmen gemeinsam: Wenn der Computer sie ausführen soll, müssen sie im Arbeitsspeicher stehen, denn nur von dort kann der Prozessor den Code für den nächsten auszuführenden Befehl in sein → Register laden!

Das Betriebssystem wurde beim → Booten in den Arbeitsspeicher kopiert; das hast du in Kapitel 6.1 gelesen.

*Den Vorgang, Programme oder Daten von einem Datenträger (Festplatte, CD, usw.) in den Arbeitsspeicher zu kopieren nennt man auch **laden**.*


Wenn man davon spricht, ein Programm auszuführen, ist darin eingeschlossen, dass es zunächst geladen wird, denn anders kann man es nicht starten.

Ein Programm zu starten bedeutet, den Befehlszähler – du erinnerst dich: Das ist das → Register in der → CPU, das die Adresse der Speicherzelle enthält, in der der nächste auszuführende Befehl steht – auf den ersten Befehl des Programms zu stellen.

Wenn wir nun ein Anwendungsprogramm ausführen wollen, um zum Beispiel ein Bild zu bearbeiten, dann müssen wir als erstes dieses Bildbearbeitungsprogramm in den Arbeitsspeicher laden. Wie geht das? Dazu muss uns das Betriebssystem die Möglichkeit geben!

Bei alten Betriebssystemen wie zum Beispiel DOS (englisch für *Disk Operating System*, Disketten-Betriebssystem), passierte das, indem man den Namen der Datei, die das Programm enthielt, und das Verzeichnis, in dem sie zu finden war, eintippte. An der Endung des Dateinamens *.exe* (oder *.com*) erkannte der Computer, dass es sich um ein Programm handelte, das er ausführen soll.

Experiment 8: Programme starten durch Texteingaben

- 1.) Starte deinen Rechner und warte, bis die Windows-Oberfläche erscheint.
- 2.) Klicke auf den Start-Knopf  unten links oder tippe auf die Taste [Start].
- 3.) Klicke auf den Menüpunkt „Ausführen“.
- 4.) Schreibe in dem sich öffnenden kleinen Fenster hinter das Wort „Öffnen“ den Programmnamen „cmd.exe“ (das *.exe* kannst du auch weglassen) und betätige die Eingabetaste [↵].
Der Dateiname *cmd* steht für „command“, was natürlich Englisch ist und „Befehl“ bedeutet.
- 5.) Es öffnet sich ein Fenster, das fast genauso aussieht wie früher das Betriebssystem DOS. (Windows enthält nämlich noch das alte Betriebssystem, damit man alte Programme aus dieser Zeit noch laufen lassen kann.) Früher gab es nur keine Fenster sondern der ganze Bildschirm war so wie jetzt das Fenster.
- 6.) Du siehst jetzt wahrscheinlich „C:\>“ (das kann man auch anders einstellen). Damit zeigt dir der Computer, in welchem Verzeichnis er die Programme, deren Dateinamen du eingibst, suchen wird, und du kannst daran sehen, dass er auf deine Eingabe wartet, und nicht mit

irgend etwas anderem - also der Ausführung eines Programms - beschäftigt ist.

Man nennt dieses Zeichen das *Prompt*.

7.) Tippe hinter das Prompt (nur da kann man in diesem Fenster schreiben) den Dateinamen „edit.com“. Dadurch wird ein Editor-Programm geladen und ausgeführt.

Dieses kannst du mit [Alt] [D] [B] wieder beenden.

8.) Wenn du etwas eintippst, das kein gültiger Dateiname ist, wird die Meldung ausgegeben, dass der Befehl nicht gefunden werden konnte.

9.) Selbst zum Beenden dieses DOS-Programms brauchst du einen Textbefehl: Tippe „exit“ ein. (Du kannst aber auch auf das Kreuz oben rechts in der Fensterecke klicken.)

Schon mit Nummer 4. des Experiments hast du gesehen, dass man Programme durch Eingabe ihres Dateinamens starten kann, denn auch cmd ist nichts anderes als ein Programm! Genau wie *edit*, das du danach gestartet hast.

Du kannst dir vorstellen, dass es ziemlich mühsam ist, immer genau zu wissen, wie denn das Programm heißt, das man gerade ausführen will und vor allem wo es steht; auch das Tippen ist ein wenig lästig. Deswegen waren damals Computer eher etwas für Freaks. Damit das nicht so blieb, und die ganze Sache etwas komfortabler wurde, hat man sich etwas Pfiffiges einfallen lassen:

Diese gute Idee ist, dass man alles was das Betriebssystem wissen muss, um ein Programm zu starten, also vor allem den Dateinamen und die → Pfadangabe (also wo die Datei steht), in einer anderen kleinen Datei speichert.

*Solche kleinen Dateien, in denen nur Hinweise auf andere Dateien (Programm- oder Datendateien) stehen, nennt man **Verknüpfungen**.*

Dem Betriebssystem hat man beigebracht (= man hat es so programmiert), dass das Doppelklicken auf eine solche Verknüpfung das selbe bedeutet, als wenn der Benutzer den Inhalt der Datei eingetippt hätte. Aber damit sind wir schon mitten in der Benutzerschnittstelle:

6.2.1 Windows Benutzerschnittstelle

Sobald dein Rechner fertig → gebootet hat, siehst du die Benutzerschnittstelle des Betriebssystems, in unserem Fall heißt es „Windows“ und ist von der Firma Microsoft.

Windows (sprich: windous) ist übrigens Englisch und bedeutet „Fenster“. Der Name kommt daher, dass die Bildschirmdarstellung eines Programms nicht den ganzen Bildschirm in Anspruch nehmen muss, sondern in einem Rahmen, einem Fenster eben, angezeigt werden kann. Dieses Fenster kann man größer und kleiner machen und auch verschieben. Das ist ganz praktisch, weil Computer heute auch mehrere Programme gleichzeitig ausführen können; dazu später mehr.

Die Benutzerschnittstelle von Windows selbst besteht im wesentlichen aus drei Teilen:


- dem Desktop
- dem Startmenü
- der Taskleiste

6.2.1.1 Der Desktop

Der auffälligste Teil von Windows – oder vielleicht auch der am wenigsten auffällige, weil man den Wald vor lauter Bäumen nicht sieht, ist der *Desktop*. Du darfst raten, welche Sprache das ist! Es bedeutet Schreibtischoberfläche und mit ein wenig Fantasie kann man im Symbol für den Desktop eine Schreibunterlage, einen Notizblock und einen Stift erkennen:



Abbildung 57: Symbol für den Desktop

Dieses Symbol findest du oft ganz unten links auf dem Bildschirm neben dem Knopf . Wenn du darauf klickst, erscheint immer wieder diese „Benutzeroberfläche“ des Betriebssystems, auch wenn zuvor die eines anderen Programms zu sehen war.

Normalerweise – wenn dein Computer nicht schon beim Start automatisch andere Programme startet (auch das kann man nämlich einstellen!) – wird dir der Windows-Desktop auch angezeigt, sobald der Bootvorgang beendet ist.

Windows selbst wird also nicht wie andere Programme in einem Fenster ausgeführt, sondern bildet quasi die Wand in der sich die Fenster befinden.

Auf dem Desktop siehst du ein paar *Symbole*, auf Englisch nennt man sie *Icons* (sprich: eikens). Wenn du auf ein solches Symbol doppelklickst (vgl. Kapitel 5.2.11.2 über die Funktionen der Maus!), wird das Programm, das von diesem Symbol repräsentiert wird, gestartet.



6.2.1.2 Startmenü

Ein Menü kennst du vielleicht aus einem Restaurant: Es ist die „Karte“ auf der steht, aus welchen Speisen und Getränken man wählen kann. Beim Italiener stehen dort jede Menge verschiedene Pizzen und Salate, beim Griechen die verschiedensten Fleischgerichte usw. Je nachdem worauf du Appetit hast, wählst du aus dem Menü aus und sagst dem Kellner dann, was du essen möchtest.

Ein Menü auf einem Computer ist auch zum Auswählen da: Man wählt Dinge aus, die der Computer tun soll, Programme oder Befehle. (Befehle sind eigentlich auch nichts anderes als Programme, nur dass sie Teil eines größeren Programms sind.)

Menüs im Restaurant sind manchmal ziemlich lang; die Liste der Gerichte und Getränke wird dann einfach auf mehrere Seiten geschrieben. Damit es übersichtlich bleibt, werden die verschiedenen Dinge in Gruppen mit Überschriften zusammengefasst, also zum Beispiel: Vorspeisen, Salate, Fisch, Fleisch, Kindergerichte, Desserts, Warme Getränke, Kalte Getränke, usw.

Genauso wird das beim Computer auch gemacht! Nur wird hier, damit es *noch* übersichtlicher wird, erst mal nur die Überschrift angezeigt, und erst wenn du sagst, dass du ein Kindergericht haben möchtest, wird dir gezeigt, welche es gibt.

In Abbildung 58 wird dir ein Startmenü gezeigt. So etwas erscheint, wenn du unten links mit der Maus auf den Knopf  klickst, oder wenn du die → Startmenü-Taste  (s. Kapitel 5.2.11.1, Seite 98) betätigst. „Ein“ Startmenü, weil es bei dir - jedenfalls teilweise - ganz anders aussehen kann! Das hängt erstens davon ab, welche Programme alle auf der Festplatte deines Rechners stehen und geladen werden können, und zweitens kann man das Aussehen des Menüs, also die Gruppen und in welcher Gruppe welcher Eintrag steht, selbst verändern. (Das geht mit der Mausfunktion Klicken-Festhalten-Bewegen.)

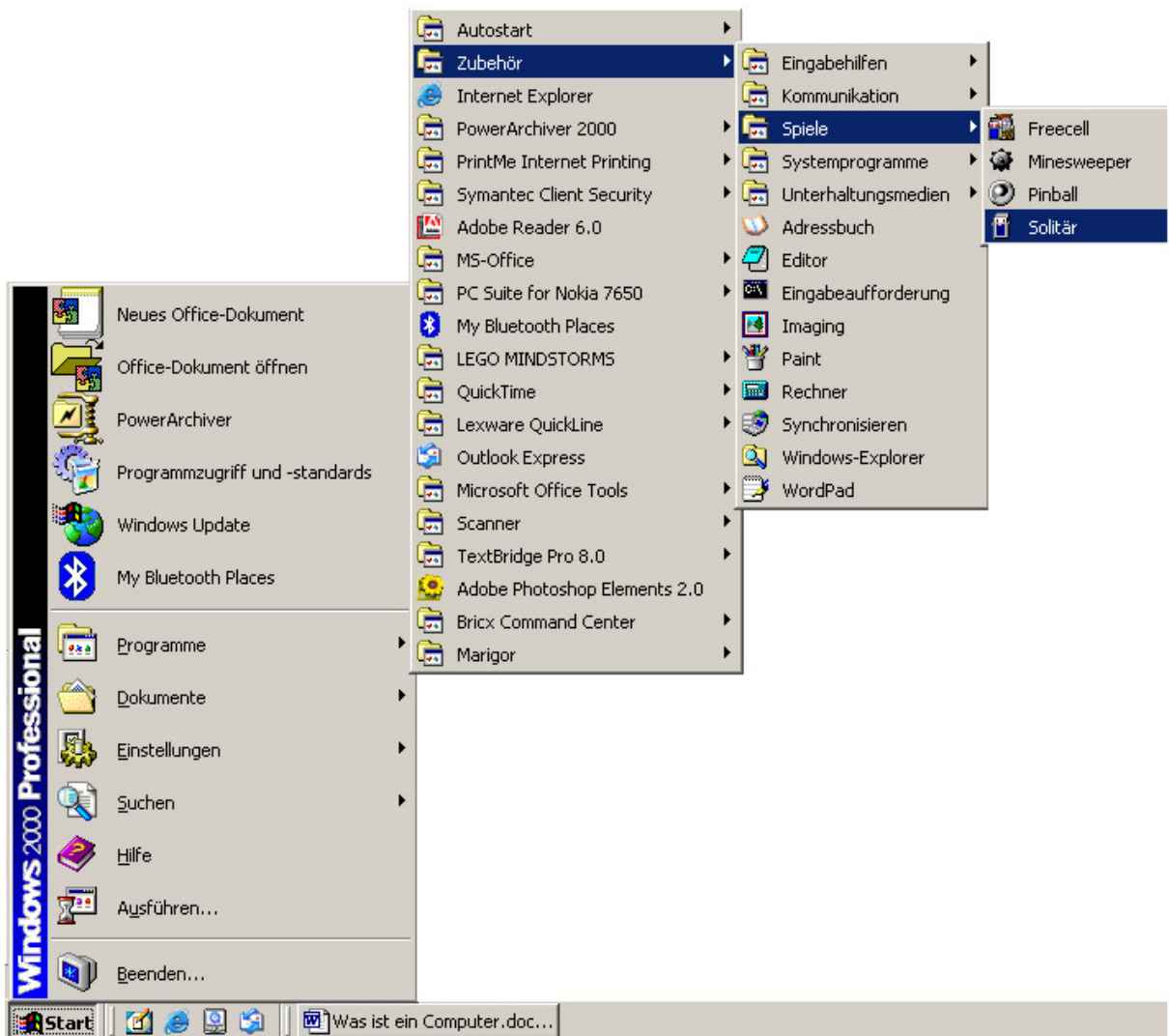


Abbildung 58: Ein Startmenü

In dem gezeigten Startmenü sind schon verschiedene Untermenüs geöffnet. Immer wenn neben dem Titel des Menüeintrags ein kleiner Pfeil steht, öffnet sich ein Untermenü, sobald du darauf klickst. In dem Untermenü funktioniert das dann genauso weiter, so lange bis du das Programm, das du in den Arbeitsspeicher laden möchtest, gefunden und darauf geklickt hast.

Versuche einmal das gezeigte Programm (oder ein anderes, das du vielleicht schon kennst) auf deinem Rechner zu finden und zu starten! (Du kannst es später mit der Tastenkombination [Alt]+[F4] wieder beenden.)

Das Startmenü hat außer „Programme“ noch andere Menüpunkte: Unter „Dokumente“ werden – automatisch! – die Dateien aufgelistet, an denen man zuletzt gearbeitet hat (klappt allerdings nicht immer!). Wenn man auf einen Eintrag klickt, wird das Anwendungsprogramm geladen, das man braucht, um an dieser



Datei zu arbeiten, und anschließend wird diese Datei geladen. Weißt du noch, woher der Computer weiß, welches Anwendungsprogramm er braucht, um eine bestimmte Datei zu bearbeiten? Wenn nicht solltest du noch mal in Kapitel 5.1.3.4.2 auf Seite 68 nachlesen!

Wenn dir die Sortierung der Menüpunkte nicht gefällt, kannst du sie ändern, indem du sie mit der Maus verschiebst (klicken, festhalten, ziehen, loslassen; vergleiche Abschnitt 5.2.11.2)

6.2.1.3 Menüs, Symbole und Verknüpfungen

Wie hängt nun das Startmenü und die Symbole auf dem Desktop mit der erwähnten genialen Erfindung der Verknüpfungen zusammen?

Die Verbindung ist die Programmierung von Windows:

Es gibt ein → Verzeichnis auf der Festplatte, in dem stehen Verknüpfungen für alle Programme, die im Startmenü erscheinen sollen! Die Untermenüs, die jeweils eine neue Auswahl öffnen, sind nichts anderes als → Unterverzeichnisse dieses Startmenüordners. Windows ist so programmiert, dass jedes Mal wenn du auf den Start-Knopf  klickst oder du die Windows-Taste  betätigst, die Datei- und Verzeichnisnamen im Startmenüordner gelesen werden und daraus das Menü zusammengebastelt wird.

Genauso verhält es sich mit dem Desktop: Auch dafür gibt es ein Verzeichnis, das beim Start von Windows gelesen wird, und für jede Verknüpfungsdatei, die darin steht, wird ein Symbol auf dem Desktop angezeigt. (Wo die Bilddatei für dieses Symbol steht, wird übrigens auch in der Verknüpfungsdatei gespeichert!)

Auch der Menüpunkt „Dokumente“ wird aus einem Verzeichnis zusammengebaut; dieses Verzeichnis heißt „Recent“ (englisch für „kürzlich“, sprich rießnt) und enthält die Verknüpfungen für die zuletzt bearbeiteten Dateien. Wann immer man eine Datei benutzt oder neu erzeugt, die in diesem Verzeichnis noch nicht oder nicht mehr steht, erzeugt Windows eine neue Verknüpfung mit dieser Datei im Recent-Verzeichnis und löscht die älteste darin enthaltene Verknüpfung, damit der Ordner nicht „überläuft“.

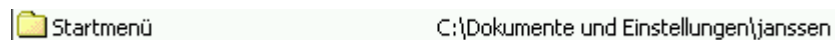
Wenn du wissen willst, wo genau sich die Startmenü-, Recent- und Desktop-Ordner befinden, führe folgendes Experiment durch:

Experiment 9: Verknüpfungen finden

- 1.) Starte deinen Rechner und warte, bis die Windows-Oberfläche erscheint.
- 2.) Klicke mit der **rechten** Maustaste auf „Arbeitsplatz“.
Wenn du keine Maus hast, oder um es auszuprobieren, kannst du auch so oft [A] tippen, bis „Arbeitsplatz“ markiert ist (vergleiche

Experiment 4 auf Seite 83!) und dann die → Kontextmenü-Taste (siehe Seite 99) drücken.

- 3.) Wähle den Menüpunkt „Suchen...“, indem du mit der Maus darauf klickst oder die Pfeiltasten tippst, bis er hinterlegt ist, und dann die Eingabetaste [↵] drückst.
- 4.) Es öffnet sich ein Fenster „Suchergebnisse“.
Klicke in das Feld unter „Nach folgenden Dateien oder Ordnern suchen:“ (wenn die Einfügemarke nicht ohnehin schon dort steht) und gib „Startmenü“ ein. Betätige die Eingabetaste [↵] oder klicke auf „Jetzt suchen“.
- 5.) Nun durchsucht Windows alle Laufwerke, Verzeichnisse und Unterverzeichnisse auf deinem Rechner - wirklich alle! - nach Dateien oder Ordnern, in deren Namen „Startmenü“ vorkommt. Das kann eine Weile dauern!
Am unteren Rand des Fensters wird dir das Verzeichnis angezeigt, in dem gerade gesucht wird.
Sobald etwas gefunden wird, wird es im rechten Teil des Fensters angezeigt. Wenn du nicht mehr weitersuchen willst, kannst du mit [Esc] die Suche abbrechen.
- 6.) Am Ende der Suche steht mindestens ein Eintrag in der Liste der Suchergebnisse. Wenn auf deinem Rechner mehr als ein Benutzer verwaltet wird, können es auch mehrere sein. Der Eintrag kann zum Beispiel so aussehen:



Startmenü C:\Dokumente und Einstellungen\janssen

- 7.) Doppelklicke auf den Eintrag; wenn es mehrere sind, nimm den, der zu deinem Anmeldenamen passt. Es öffnet sich ein Fenster, in dem du einen Eintrag deines Startmenüs als Unterverzeichnis wiederfindest, nämlich „Programme“. (Die übrigen Einträge des Startmenüs werden aus anderen Verzeichnissen zusammengebaut.) Wenn du auch auf „Programme“ doppelklickst, wirst du die gleichen Einträge sehen, die auch erscheinen, wenn du im Startmenü auf „Programme“ klickst.
- 8.) Wiederhole Nummer 4. und tippe jetzt „Desktop“ ein. In dem Verzeichnis, das du findest, müsstest du, wenn du darauf doppelklickst, die selben Einträge finden, die auch auf deinem Desktop angezeigt werden.

9.) Wiederhole Nummer 4. und tippe jetzt „Recent“ ein. In dem Verzeichnis, das du findest, müsstest du, wenn du darauf doppelklickst, die selben Einträge finden, die auch unter dem Menüpunkt „Dokumente“ angezeigt werden.

Nun wollen wir uns eine Verknüpfung mal aus der Nähe ansehen:

Klicke mit der *rechten* Maustaste auf einen Startmenüeintrag, mit dem man ein Programm starten kann. Im Beispiel in Abbildung 58 habe ich das mit dem Spiel „Solitär“ gemacht. Du kannst aber auch auf eine Verknüpfung klicken, die du beim Suchen der Ordner in [Experiment 9](#) gefunden hast; es soll nur eine Datei und kein Verzeichnis sein. Du kannst Verknüpfungen immer daran erkennen, dass die Symbole unten links in der Ecke einen kleinen Pfeil haben, wie in folgendem Beispiel zu sehen:



Abbildung 60: Beispiel für ein Verknüpfungssymbol

Wenn du jedenfalls mit der rechten Maustaste auf eine Verknüpfung geklickt hast, erscheint das Kontextmenü, über das du in 6.2.1.5 noch mehr erfahren wirst:

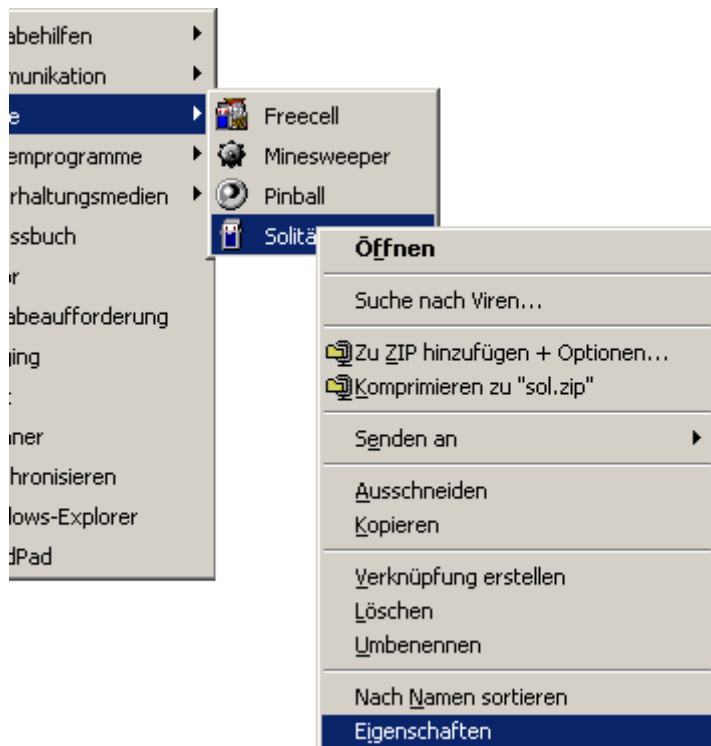
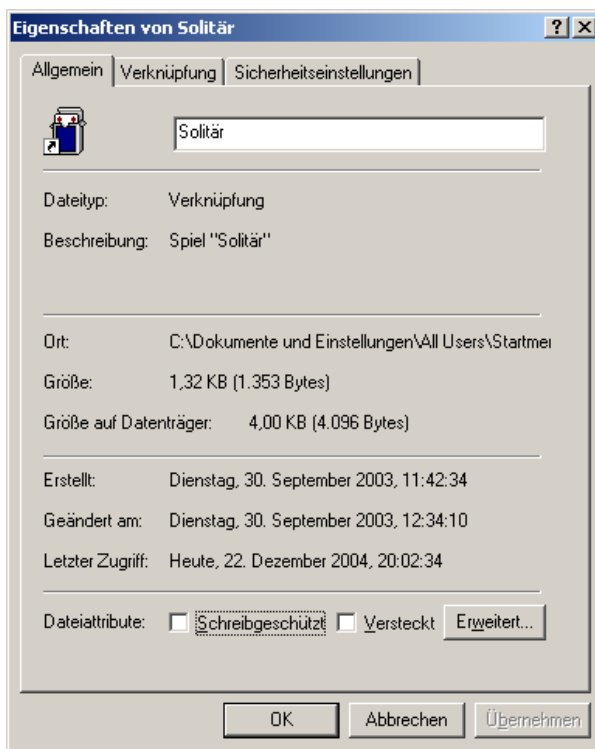


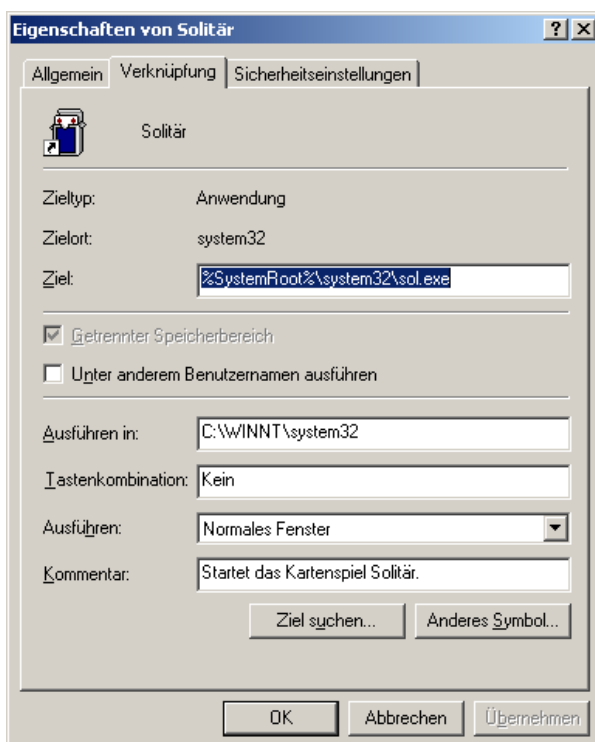
Abbildung 61: Kontextmenü eines Startmenüeintrages aufrufen



Wir kümmern uns zunächst nicht darum, was in diesem Menü alles steht, sondern wählen nur den untersten Eintrag „Eigenschaften“. Damit wird uns in diesem Fall angezeigt, was in dieser Verknüpfung alles eingestellt ist.

Es öffnet sich ein neues Fenster, von dem uns die beiden Registerkarten „Allgemein“ und „Verknüpfung“ interessieren, wie es in Abbildung 62 gezeigt wird:

Auf der Registerkarte „Allgemein“ steht alles, was sich auf die Verknüpfungsdatei selbst bezieht, also zuerst einmal der Name, wie er als Menüeintrag erscheint. Er ist identisch mit dem Dateinamen der Verknüpfungsdatei, darunter steht der Dateityp der Verknüpfungsdatei im Klartext (vergleiche 5.1.3.4.2!). Es folgen der Ort, wo die Verknüpfungsdatei zu finden ist, wie groß sie ist und noch ein paar andere Details.



Die Registerkarte „Verknüpfung“ sagt uns, was alles in der Verknüpfungsdatei gespeichert ist und sich auf das Programm bezieht, das mit Hilfe der Verknüpfung gestartet werden soll: Zuerst steht da, dass es sich um eine Anwendung handelt, der Dateityp also .exe lautet. (Wenn du die Eigenschaften einer Verknüpfung aus dem Ordner *Recent* aufgerufen hast, wird hier ein anderer Dateityp stehen!)

Dann steht dort, wo die Datei zu finden ist, also genau das, was wir

Abbildung 62: Eigenschaften einer Verknüpfung

im DOS-Fenster eingetippt haben, um das Programm zu starten. Das komische %SystemRoot% bedeutet dabei nur den Laufwerksbuchstaben des Laufwerks von dem Windows geladen wurde.

„Ausführen in“ erlaubt uns, analog zum Prompt des DOS-Fensters ein Verzeichnis einzustellen, das das Programm „sieht“, wenn es etwa eine Datei speichert oder lädt, ohne ein Laufwerk und einen Pfad anzugeben.

Unter Tastenkombination kann man eine Tastenkombination einstellen, mit der man das Programm starten kann, das heißt, statt die Verknüpfung auf dem Desktop anzuklicken, kannst du die Tastenkombination benutzen. Das kann für häufig genutzte Verknüpfungen ganz praktisch sein, weil man nicht immer zum Desktop zurückkehren muss, um ein Programm aufzurufen. Der Nachteil ist, dass man sich die Tastenkombination merken muss!

Solche Tastenkombinationen beginnen immer mit [Strg]+[Alt]+ und man kann eine weitere Taste ergänzen.

„Ausführen“ bestimmt, wie das Fenster aussieht, in dem das Programm läuft, wenn es über diese Verknüpfung geladen wird. Was die Möglichkeiten bedeuten, erfährst du in Kapitel 6.2.2.1.2.

Da es sich bei Verknüpfungen um eigenständige Dateien handelt, kannst du dir vorstellen, dass die Angaben, die darin stehen, auch einmal falsch sein können; nämlich dann, wenn die Programmdatei umbenannt oder in ein anderes Verzeichnis verschoben oder sogar vollständig gelöscht worden ist, nachdem die Verknüpfung erstellt wurde. Falls so etwas einmal vorkommen sollte, kann man mit dem Knopf „Ziel suchen“ die Programmdatei wieder suchen und die Einträge korrigieren.

Normalerweise werden die Verknüpfungen, die man zum Laden und Starten der Programme braucht, beim → Installieren erstellt.

Du kannst aber auch sehr leicht selbst eine erstellen! Nehmen wir an, du hast eine Datei, die du immer wieder benötigst, zum Beispiel ein Tagebuch, das du jeden Tag aufrufen willst, um es fortzusetzen. Du brauchst nur mit der *rechten* Maustaste auf eine freie Fläche des Desktop zu klicken, also das Kontextmenü des Desktop aufrufen, und unter „Neu“ den Unterpunkt „Verknüpfung“ auswählen. Es erscheint ein kleiner Assistent, mit dessen Hilfe du die Datei, auf die die Verknüpfung verweisen soll (das *Ziel* der Verknüpfung), suchen kannst. Anschließend kannst du mit dem Kontextmenü der Verknüpfung wie oben schon beschrieben die Verknüpfung nach deinen Vorstellungen anpassen. Probiere es ruhig einmal aus!

Es gibt noch weitere Möglichkeiten, Verknüpfungen zu erzeugen: Wenn man mit der rechten Maustaste eine Datei aus einem Fenster auf den Desktop zieht, erscheint wieder ein → Kontextmenü, das eine Auswahl „Verknüpfung hier

erstellen“ enthält. Wenn man darauf klickt, wird auf dem Desktop eine Verknüpfung mit der gezogenen Datei erstellt.

Wenn du eine Verknüpfung später nicht mehr brauchst, kannst du sie auch bedenkenlos wieder löschen. Die Programm- oder Datendatei, auf die die Verknüpfung verweist, bleibt ja bestehen. Klicke dazu mit der *rechten* Maustaste auf das Verknüpfungssymbol und wähle aus dem Kontextmenü „Löschen“. Achte aber immer darauf, dass das Symbol auch den Pfeil zeigt, der Verknüpfungen kennzeichnet (vergleiche Abbildung 60), damit du nicht versehentlich doch einmal eine Datei löschst! Um solche Verwechslungen zu vermeiden, sollte man in den Menü- und Desktopverzeichnissen niemals irgendwelche Dateien speichern, sondern *ausschließlich* Verknüpfungen!

6.2.1.4 Taskleiste

Die Leiste, die standardmäßig ganz unten am Bildschirm angezeigt wird, heißt *Taskleiste*. Task bedeutet im Englischen so viel wie „Aufgabe“ und daher kommt auch die Bezeichnung: Im größten Teil der Taskleiste werden dir die Aufgaben angezeigt, an denen der Computer gerade arbeitet. Eine Aufgabe, an der ein Computer arbeitet ist natürlich – ein Programm!

Du kannst die Taskleiste auch an jede andere Kante deines Bildschirms verlegen, indem du einfach mit der linken Maustaste in einen freien Bereich klickst, die Taste festhältst und währenddessen die Maus an die gewünschte Kante bewegst; die Taskleiste kommt mit!

Platzierst du die Maus an der Oberkante der Taskleiste, so dass der Mauszeiger ein Doppelpfeil wird, kannst du die Taskleiste dicker machen. Das ist ganz hilfreich, wenn einmal so viel darin steht, dass man die Bezeichnungen nicht mehr lesen kann.

Die Taskleiste hat auch ein → Kontextmenü: Klicke mit der rechten Maustaste in einen freien Bereich, um es aufzurufen.


Der Menüpunkt „Alle Fenster minimieren“ ist gleichbedeutend mit dem Klick auf das Desktopsymbol.

Mit dem Menüpunkt „Eigenschaften“ rufst du ein Fenster auf, in dem du das Verhalten der Taskleiste einstellen kannst. Eine interessante Option ist dabei „Automatisch im Hintergrund“. Sie bedeutet, dass die Taskleiste nur dann angezeigt wird, wenn sich die Maus an der Bildschirmkante befindet.

In Abbildung 63 siehst du die Taskleiste meines Laptops, auf dem ich gerade diesen Text schreibe. Ich habe in dem → Textverarbeitungsprogramm „Word“ die Datei „Die Geheimnisse eines Computers.doc“ geöffnet und außerdem das Programm „Microsoft Photoeditor“ geladen (das brauchte ich nämlich, um Abbildung 63 auf die richtige Größe zu bringen).



Abbildung 63: Beispiel einer Taskleiste

Ganz links siehst du den Knopf  für das Startmenü, über das wir in Abschnitt 6.2.1.2 schon gesprochen haben. Dann kommt der sogenannte Schnellstart-Bereich, in dem man auch Verknüpfungen ablegen kann. Dort steht bei mir das Desktopsymbol aus 6.2.1.1 und Verknüpfungen mit den Programmen „Internet Explorer“, „Quick Time“ und „Microsoft Outlook Express“.

Am rechten Ende der Leiste stehen Symbole für verschiedene Programmteile des Betriebssystems (Batterieüberwachung, Lautstärkeinstellung, Uhr) oder auch zusätzlich installierte Programme (Antivirenprogramm, Handyverbindung, Scansoftware), die in diesem „System Tray“ (Englisch für „System Schublade“) genannten Bereich ein Symbol anlegen.

An deinem Rechner kann der Inhalt der Taskleiste natürlich anders aussehen, je nachdem wie er eingestellt ist.

Noch mal zurück zu dem größten Teil der Taskleiste, den gerade ausgeführten Programmen: Alle dort angezeigten Programme und die von ihnen zum Bearbeiten geöffneten Dateien befinden sich im Arbeitsspeicher! Du kannst auf die Schaltfläche eines Programms klicken und bekommst sofort dessen Benutzeroberfläche angezeigt, ohne dass es erst langwierig geladen werden müsste. (Mit der Tastenkombination [Alt]+[→] kannst du auch zwischen den Fenstern wechseln.)

Ja mehr noch: Während du die Oberfläche eines Programms siehst und beispielsweise Eingaben machst, wird an den anderen Programmen weiter gearbeitet! Während also zum Beispiel in einem Programm eine Berechnung ausgeführt wird, kannst du in einem anderen Programm etwas anderes machen!

Wie können wir das mit dem in Einklang bringen, was wir über Programme gelernt haben, dass sie nämlich festlegen, was in welcher Reihenfolge getan wird?! Wie kann auf einmal die Reihenfolge der Aktionen geändert werden, nur weil ein weiteres Programm geladen wird?

Die Antwort ist, dass das Programm „Betriebssystem“ die Abarbeitung der Anwendungsprogramme so steuert, dass immer der Reihe nach ein Befehl des einen, dann einer des nächsten, dann einer des übernächsten usw. abgearbeitet wird, und wenn alle Programme einmal dran waren, geht die Reihe vorne wieder los. Die Programme werden also nicht wirklich gleichzeitig bearbeitet, sondern nur quasi-gleichzeitig, aber weil das so schnell geht, merken wir das nicht.

Diese Steuerung passiert einfach dadurch, dass das Betriebssystem sich für jedes laufende Programm merkt, wo es im Arbeitsspeicher steht, und welcher

Befehl als nächstes auszuführen ist. Den Befehl des Programms, das gerade an der Reihe ist, kopiert es in den Befehlszähler des → Prozessors.

*Ein Betriebssystem, das mehrere Programme quasi gleichzeitig ablaufen lassen kann, nennt man ein **Multitasking**-System.*

Das ist eine tolle Sache, oder? Allerdings sollte man, je nach Leistungsfähigkeit des Rechners, diese Möglichkeiten nicht über Gebühr beanspruchen, denn sonst kann es passieren, dass der Rechner unnötig langsam wird. Der Grund ist, dass der Arbeitsspeicher in einem Rechner ja begrenzt ist. Das macht zunächst nichts, weil Windows, ohne dass du es merkst, auch dieses Problem geschickt umgeht:

Wenn Programme oder Daten in den Arbeitsspeicher geladen werden sollen, für die nicht mehr genügend Platz darin vorhanden ist, so kopiert Windows die gerade nicht benötigten Teile des Arbeitsspeichers auf die Festplatte und holt sie erst dann wieder zurück, wenn diese Speicheradressen von einem Programm angesprochen werden, wenn es also daraus lesen oder dorthin schreiben will.

*Den Bereich auf der Festplatte, den Windows zum Zwischenspeichern des Arbeitsspeicherinhalts benutzt nennt man **Virtueller Arbeitsspeicher**.*

*Dabei handelt es sich auch um eine Datei, das **Swapfile** (engl. swap = überschwappen; sprich: suopp).*

Virtuell bedeutet so viel wie „nur in der Vorstellung vorhanden“.

Die Größe dieses Bereichs kann man sogar einstellen, und zwar unter Start/Einstellungen/Systemsteuerung/System, Registerkarte „Erweitert“, Schaltfläche „Systemleistungsoptionen“. So lange alles läuft, sollte man daran allerdings nicht herumspielen!

Dieser Vorgang passiert, ohne dass du etwas davon merkst, außer dass er Zeit kostet.

6.2.1.5 Kontextmenü

Das Kontextmenü haben wir schon oft angesprochen und auch schon benutzt. Es ist eine feine Sache, denn damit schlägt Windows dir vor, was du mit einem angeklickten Objekt alles machen kannst; aber schön der Reihe nach:

Man ruft das Kontextmenü eines Objektes auf, indem man mit der *rechten* Maustaste darauf klickt, oder indem man das Objekt auswählt (hervorgehobene Darstellung) und dann die Kontextmenütaste auf der Tastatur tippt (siehe 5.2.11.1 auf Seite 99). Ein Objekt kann dabei alles mögliche sein: Angefangen beim Startmenü-Knopf, über die Schaltflächen in der Taskleiste, Verknüpfungen auf dem Desktop oder in einem Menü, freie Flächen des Desktops, oder Dinge, die in einem Fenster angezeigt werden. Alle diese Dinge haben ein Kontextmenü!

Kontext ist ein Fremdwort für *Zusammenhang* oder *Umfeld*. Das Menü heißt deshalb so, weil sein Inhalt, also die Auswahlmöglichkeiten, die es dir anbietet, davon abhängen, was du angeklickt hast.

Nachdem du das Kontextmenü geöffnet hast, kannst du die Auswahl daraus übrigens wahlweise mit der linken oder rechten Maustaste betätigen.

Solltest du ein Kontextmenü einmal versehentlich geöffnet haben, keine Panik: Klicke einfach irgendwo außerhalb des Menüs oder betätige die Taste [Esc] (siehe Abschnitt 5.2.11.1 auf Seite 96), und das Menü verschwindet wieder.

Zum Üben kannst du eine Erweiterung von [Experiment 4](#) machen (siehe Seite 83): Führe die vier genannten Schritte aus. Füge als fünften Schritt das Betätigen der Kontextmenütaste aus.

Damit hast du das Kontextmenü des Arbeitsplatzes aufgerufen, was natürlich genauso gut mit einem rechten Mausklick auf das Arbeitsplatzsymbol funktioniert hätte.

Was du erhältst sollte so aussehen wie in [Abbildung 67](#). In den anderen [Abbildungen](#) siehst du Kontextmenüs anderer Objekte:

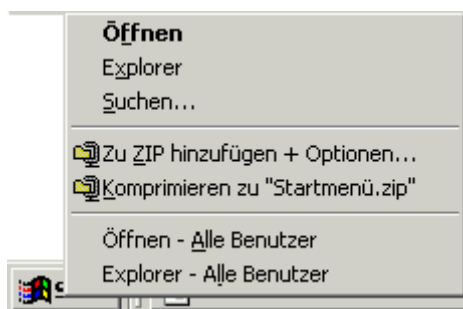


Abbildung 63: Kontextmenü des Startmenüs

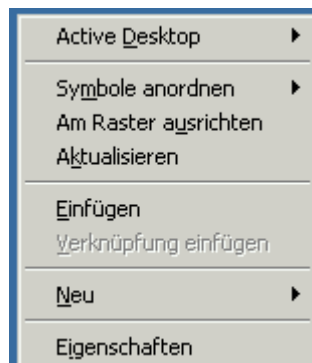


Abbildung 64: Kontextmenü des Desktops

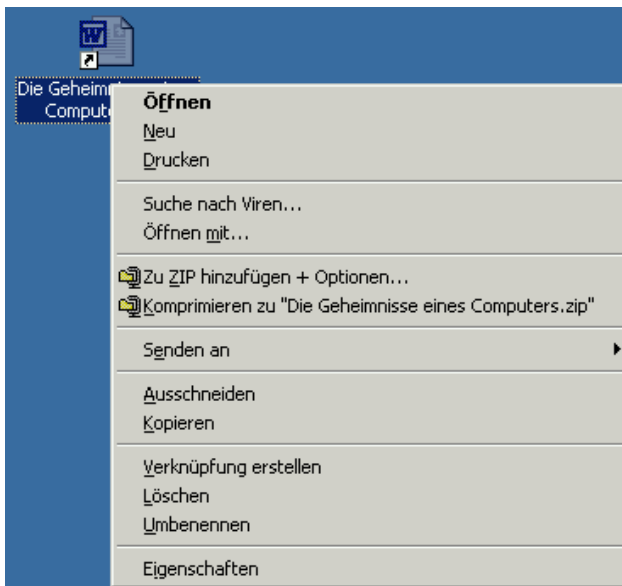


Abbildung 65: Kontextmenü der Verknüpfung mit der Word-Datei dieses Buches

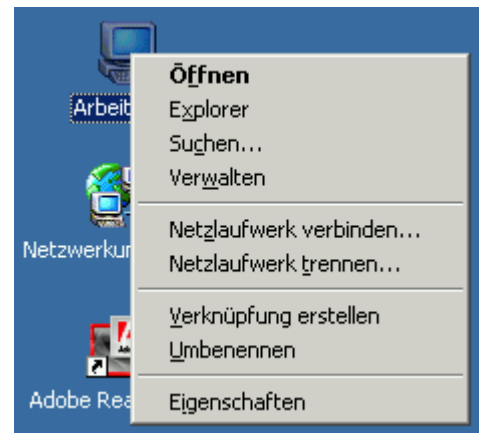


Abbildung 67: Kontextmenü von „Arbeitsplatz“

Das Wesentliche was du daran sehen sollst ist, dass die Menüs unterschiedlich sind, je nachdem worauf man klickt.

Vielleicht ist dir aufgefallen, dass es immer einen Menüeintrag gibt, der fett gedruckt ist. Das ist die Aktion, die ausgeführt wird, wenn du auf das Objekt mit der linken Maustaste doppelklickst.

Sehr häufig heißt dieser fett gedruckte Eintrag „Öffnen“ oder zumindest gibt es einen Eintrag der so heißt.

Öffnen bedeutet, dass sowohl die Datei als auch das Programm, das man braucht um sie zu bearbeiten, in den Arbeitsspeicher geladen werden. Das Programm wird ausgeführt und du kannst mit dem angeklickten Objekt darin arbeiten.

Weitere Einträge, die man oft findet, heißen „Ausschneiden“, „Kopieren“ und „Einfügen“. Ihre Bedeutung lernst du in Kapitel 6.2.3.

Bisher haben wir über Kontextmenüs von Objekten gesprochen, also von irgendwelchen Dingen, die du auf dem Bildschirm siehst und anklicken kannst. Es gibt aber auch Kontextmenüs von *Aktionen*.

Erinnerst du dich an Kapitel 5.2.11.2, wo etwas über die Bedienung der Maus stand? Dort haben wir schon darüber gesprochen, dass man ein Objekt mit der

rechten Maustaste anklicken, die Taste festhalten und dann das Objekt bewegen kann.

Das Kontextmenü in Abbildung 67 habe ich beispielsweise erzeugt, indem ich eine Verknüpfung mit der rechten Maustaste auf dem Desktop verschoben habe.

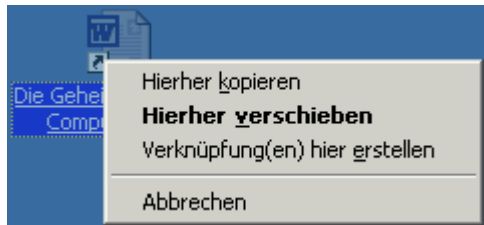


Abbildung 67: Kontextmenü einer Aktion

„Verschieben“ bedeutet hier, dass die Verknüpfung an einer anderen Stelle des Desktops dargestellt wird. Man kann eine Datei aber auch von einem Ordner in einen anderen verschieben, was nichts anderes heißt, dass sie zuerst in den neuen Ordner kopiert und, wenn das erfolgreich war, im alten Ordner gelöscht wird.

Dieser Eintrag ist fett gedruckt, wird also ausgeführt, wenn du statt mit der rechten mit der linken Maustaste ziehst.

„Kopieren“ bedeutet, dass die gleiche Datei an der Stelle wo sich der Mauszeiger beim Loslassen befindet, eine Kopie der Datei erstellt wird.

„Verknüpfung erstellen“ haben wir in Kapitel 6.2.1.3 schon geübt.

Wenn du „Abbrechen“ wählst, passiert gar nichts.

Die Aktionen „Kopieren“ und „Verknüpfung erstellen“ kannst du auch ohne Kontextmenü erreichen: Wenn du während des Verschiebens mit der linken Maustaste die [Strg]-Taste drückst, bedeutet das „Kopieren“ und wird dir durch ein kleines Plus am Mauszeiger angezeigt. Wenn du statt dessen die [Alt]-Taste betätigst, bedeutet das „Verknüpfung erstellen“ und wird dir durch den Verknüpfungspfeil (vergleiche Abbildung 60) angezeigt.

Du brauchst dir diese ganzen Möglichkeiten nicht merken! Es soll dir nur zeigen, dass es für fast alles immer mehrere Möglichkeiten gibt und für die Dinge, die du öfter tust, solltest du dir deine Lieblingsmethode aussuchen und merken. Das kommt aber ganz von alleine: Wenn dir irgend etwas zu umständlich erscheint, forsche ein wenig nach und du wirst wahrscheinlich feststellen, dass es einen einfacheren Weg gibt.

Die rechte Maustaste ist in allen Fällen, in denen man nicht mehr genau weiß, wie etwas geht, eine gute Wahl. Das Kontextmenü enthält in der Regel das Gesuchte!

6.2.2 Fenster

Fenster sind der Namensgeber für das Betriebssystem Windows. Allerdings war Windows nicht das erste Betriebssystem, das diese Technik verwendet hat.

Als man Betriebssysteme erfunden hatte, die mehrere Programme quasi gleichzeitig ablaufen lassen konnten, brauchte man eine Möglichkeit, dem Benutzer auch mehrere Benutzeroberflächen der Programme auf einem Bildschirm darstellen zu können.

Dazu wurden die Fenster erfunden: Sie sind für das Programm quasi der Bildschirm, auf dem Sie ihre Bedienoberfläche darstellen müssen.

Man kann Fenster größer und kleiner machen, auch ganz groß (gesamter Bildschirm) und ganz klein (nur Symbol in der Taskleiste), man kann sie verschieben und nebeneinander oder übereinander darstellen lassen. Und wie ein richtiges Fenster kann man sie natürlich auch öffnen oder schließen, was gleichbedeutend ist mit dem → Öffnen einer Datei oder eines Programms bzw. dem Beenden eines Programms.

Alle diese Funktionen werden den Anwendungsprogrammen vom Betriebssystem zur Verfügung gestellt. Deshalb sehen die Fenster für alle Programme gleich aus und haben die gleichen Bedienelemente. Allerdings gibt es auch Programme, die ihre eigenen Fenster programmiert haben oder sogar den gesamten Bildschirm für sich beanspruchen.

Manche Programme können mehrere Dateien gleichzeitig für die Bearbeitung geöffnet haben. Diese Programme haben in ihrem Fenster meistens eine Fläche, auf der sie wiederum Fenster darstellen, nämlich für jede geöffnete Datei eins. Diese Dateifenster funktionieren genauso wie die Programmfenster.

In Abbildung 68 siehst du beispielhaft das Fenster der Anwendung Microsoft Word, mit der ich dieses Buch geschrieben habe.

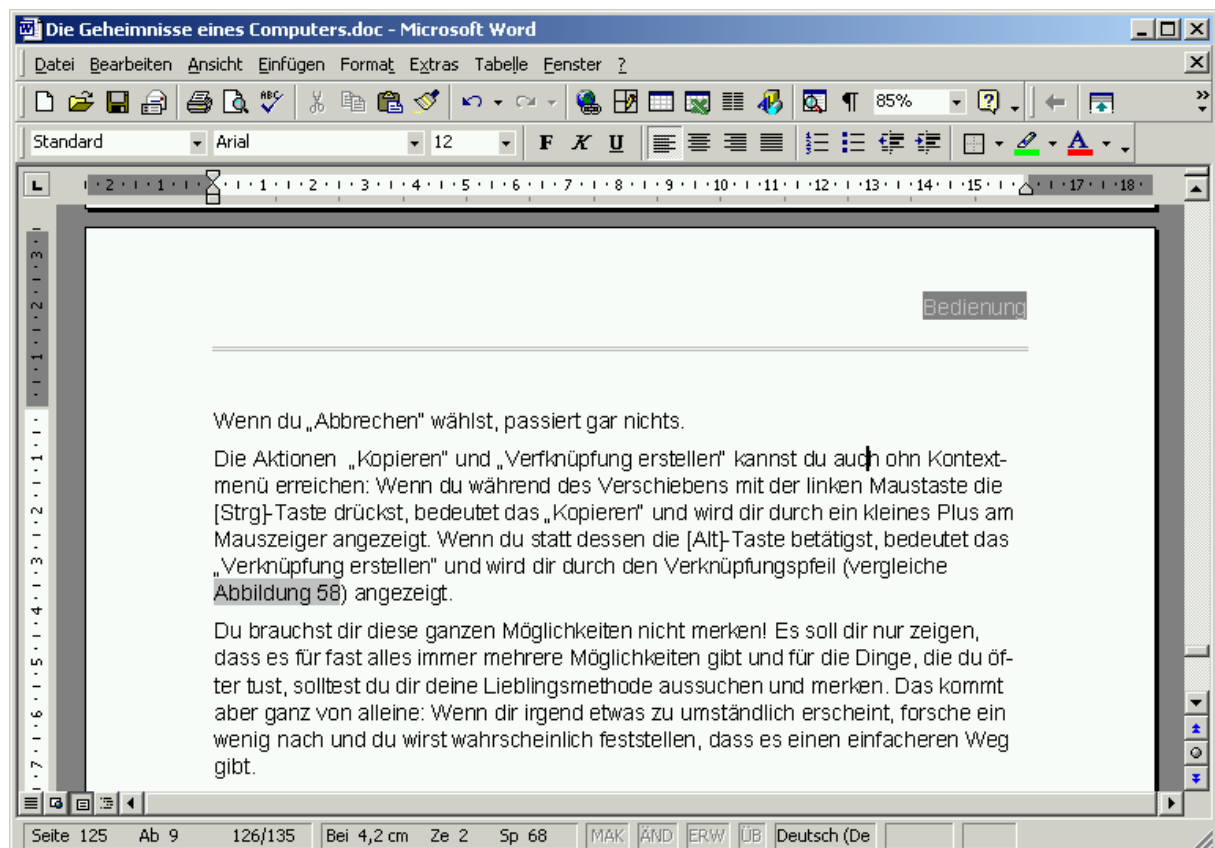
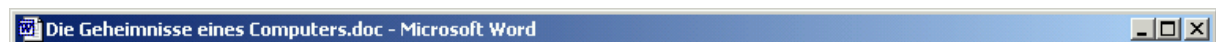


Abbildung 68: Beispiel für ein Fenster

In den folgenden Kapiteln werden die einzelnen Teile und Funktionen von Fenstern an diesem Beispiel erklärt:

6.2.2.1 Titelleiste

Beginnen wir oben:



Die Oberkante des Fensters enthält einen Titel. In der Regel ist das der Name des Programms und/oder der mit dem Programm geöffneten Datei. Dieser Titel, oder zumindest der Anfang davon, erscheint auch auf dem Symbol in der Taskleiste, das zu diesem Fenster gehört.

Wenn man mehrere Programme gleichzeitig ausführt, muss der Computer wissen, für welches der Programme irgendwelche Eingaben gedacht sind, die du vielleicht machst. Deshalb ist immer ein Fenster das sogenannte *aktive Fenster*. Alle Mausklicks und Tastatureingaben werden an das Programm geleitet, das im aktiven Fenster dargestellt wird. Du erkennst das aktive Fenster daran, dass

seine Titelleiste in einer anderen Farbe (standardmäßig blau, man kann das aber ändern) dargestellt wird. Außerdem erscheint das Symbol des aktiven Fensters in der Taskleiste wie ein eingedrückter Knopf. In der Regel ist das aktive Fenster auch automatisch vorne (wird also bei überlappenden Fenstern komplett dargestellt und überdeckt ggf. andere Fenster). Es gibt immer höchstens *ein* aktives Fenster; alle anderen sind inaktiv!

Wenn du ein Fenster an der Titelleiste mit der Maus „anfasst“, also mit der linken Maustaste darauf klickst und die Taste festhältst, dann kannst du das Fenster verschieben. Es hängt sozusagen an deinem Mauszeiger.

6.2.2.1.1 Systemmenü

Links in der Titelleiste gibt es ein kleines Symbol. Meistens sieht das genauso aus wie das Symbol, das auch im Menü und bei Verknüpfungen für das Programm dargestellt wird. Wenn du darauf klickst, öffnet sich das sogenannte *Systemmenü*.

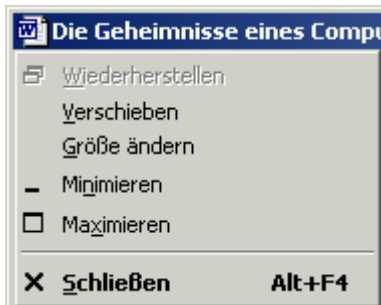


Abbildung 69: Systemmenü

Abbildung 69 zeigt das Systemmenü. Es enthält die selben Möglichkeiten wie das Schaltflächen-trio⁶ im nächsten Kapitel (siehe dort) und man kann die Größe und Position des Fensters ändern. Da man die Position auch mit der Titelleiste ändern kann und auch die Größe auf komfortablere Weise geändert werden kann (siehe 6.2.2.2), braucht man dieses Menü fast nie! Es existiert eigentlich nur noch aus historischen Gründen, weil die komfortableren Möglichkeiten

erst später eingeführt wurden.

Das Kontextmenü einer Schaltfläche in der Taskleiste hat übrigens genau die selben Einträge wie das Systemmenü.

6.2.2.1.2 Schaltflächen-Trio

Am rechten Rand der Titelleiste befinden sich drei Schaltflächen. Sie sehen aus wie eine der beiden Möglichkeiten in Abbildung 70. Welche von beiden angezeigt wird, hängt davon ab, ob das Fenster seine maximale Größe hat oder nicht.

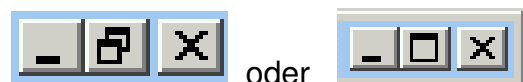


Abbildung 70: Schaltflächen-Trios
Minimieren, Wiederherstellen /
Maximieren, Schließen

⁶ Ein Trio sind *drei*, die zusammengehören, so wie ein Paar zwei sind. Wenn drei Musiker zusammenspielen, nennt man sie ein Trio.

Mit der linken Schaltfläche *Minimieren*, kann man das Fenster so verschwinden lassen, dass nur noch das dazugehörige Symbol in der Taskleiste angezeigt wird. Der Strich auf der Schaltfläche soll das Taskleistensymbol darstellen.

Das Programm läuft trotzdem weiter, wenn man ein Fenster minimiert! Man kann seine Bedienoberfläche einfach wieder anzeigen, indem man auf sein Symbol in der Taskleiste klickt. Das Fenster erscheint wieder in dem dann aktuellen Zustand (meistens der selbe wie beim Minimieren, aber wenn das Programm zwischenzeitlich weitergerechnet hat, kann das auch anders aussehen). Es gibt sogar Programme, die ihr Fenster bei bestimmten Ereignissen selbst wieder vergrößern und zum aktiven Fenster machen, zum Beispiel weil sie eine Eingabe erwarten oder eine wichtige Ausgabe erfolgt. Bei einem → e-Mail-Programm kann das zum Beispiel der Fall sein, wenn eine neue Nachricht eingetroffen ist.

Wenn das Fenster seine maximale Größe hat, also den gesamten Bildschirm einnimmt, erscheint die linke Variante des Trios, die in der Mitte das Symbol für *Wiederherstellen* anzeigt. Wenn man darauf klickt, bekommt das Fenster wieder seine zuvor eingestellte Größe. Das Symbol zeigt zwei gleichzeitig dargestellte Fenster.

Sobald ein Fenster wiederhergestellt ist, wechselt das Schaltflächentrio in die andere Darstellung und in der Mitte wird nun das Symbol für *Maximieren* angezeigt. Es ist nicht schwer zu erraten, was passiert, wenn man darauf klickt, oder? Probiere es aus, wenn du nicht darauf kommst!

Das rechte Symbol heißt *Schließen*. Damit *beendest* du das Programm! Das bedeutet, der Bereich des Arbeitsspeichers, in dem das Programm stand, wird wieder freigegeben. Außerdem muss Windows dieses Programm nun nicht mehr ständig mit ausführen.

Bevor ein Programm geschlossen werden kann, müssen auch alle Dateien, die damit bearbeitet wurden, geschlossen werden, denn sonst bleiben sie im Arbeitsspeicher stehen, werden aber von keinem Programm mehr benutzt.

*Eine Datei zu **schließen** bedeutet, sie aus dem Arbeitsspeicher zu lösen!*

Vielleicht haben wir aber eine Datei verändert, während sie im Arbeitsspeicher stand!?! Dann wollen wir diese Änderungen normalerweise natürlich vorher auf die Festplatte (oder einen anderen *nicht flüchtigen* Speicher) übernehmen, denn sonst sind sie ja verloren!

Die meisten Programme merken sich, ob du eine Datei, die du mit ihnen geöffnet hast, verändert hast, und wenn du das Programm schließt, fragen sie dann

vorher, ob du diese Änderungen speichern möchtest. Es erscheint ein Fenster wie in Abbildung 71 zu sehen:



Abbildung 71: Sicherheitsabfrage beim Schließen eines Programms

Diese Fenster sollte man immer aufmerksam lesen und darüber nachdenken! Nicht einfach wegeklicken! Man vertut sich sonst sehr leicht, besonders, wenn mehrere Dateien gleichzeitig geöffnet sind!

Wenn du auf *Ja* klickst (oder einfach die Eingabetaste [↵] betätigst, denn dieser Knopf hat den → Auswahlrahmen!), wird die Datei *unter dem in dem Fenster genannten Namen* gespeichert.

ACHTUNG! Auch das kann manchmal nicht das sein was du möchtest, denn vielleicht möchtest du eine Datei unter einem *anderen Namen* speichern, damit die alte Version erhalten bleibt.

Klicke in diesem Fall auf *Abbrechen*. Damit wird der Vorgang, das Programm zu beenden, abgebrochen, und du kannst normal weiterarbeiten, also auch die Datei unter einem anderen Namen speichern.

Manchmal sieht man unter dem Schaltflächentrio der Titelleiste noch mal das gleiche. Dies ist dann der Fall, wenn ein Programm mehrere Dateien gleichzeitig zur Bearbeitung geöffnet haben kann. Das untere Trio bezieht sich dann immer auf die in diesem Fenster angezeigte Datei (zum Beispiel ein Text in einem Textverarbeitungsprogramm), nicht auf das Programm. Funktionieren tun diese Schaltflächen nach dem selben Schema.

6.2.2.2 Fensterrahmen

Wenn ein Fenster nicht die Maximalgröße hat, wird es von einer dünnen Linie eingerahmt. Sobald du den Mauszeiger genau auf diese Linie bewegst, verwandelt er sich in einen kleinen Doppelpfeil. Wenn du nun klickst (und festhältst), kannst du die Größe des Fensters durch Ziehen mit der Maus verändern. Wenn du den linken oder rechten Rahmen dazu benutzt, kannst du die Breite des Fensters verändern, wenn du den oberen oder unteren Rahmen benutzt die Höhe, und wenn du den Mauszeiger auf eine Ecke des Rahmens platzierst, kannst du beides gleichzeitig ändern!

6.2.2.3 Statuszeile

An der Unterkante des Fensters zeigen viele Programme eine Statuszeile, wie du sie beispielhaft in Abbildung 72 siehst:



Abbildung 72: Beispiel für eine Statuszeile

Status ist – nein nicht Englisch sondern Latein! Es bedeutet *Zustand*. Das Textverarbeitungsprogramm Word zeigt dort zum Beispiel an, an welcher Stelle des Dokuments sich die → Einfügemarke gerade befindet.

6.2.2.4 Rollbalken

Manchmal, sehr oft sogar, will ein Programm auf seiner Benutzeroberfläche mehr darstellen als in das ihm zugedachte Fenster passt.

Natürlich könnte man beim Verkleinern des Fensters den Inhalt gleich mit verkleinern, damit immer noch alles hineinpasst, aber man soll es ja auch noch lesen können! Deshalb hat man die *Rollbalken* erfunden. Wenn man sich vornehm ausdrücken will, kann man auch *Bildlaufleiste* sagen oder auf Englisch *scroll bar*.

Sobald der Fensterinhalt zu breit wird, erscheint am unteren bzw. wenn er zu hoch wird am rechten Rand ein schmaler Streifen mit kleinen Pfeilen am Ende und einem mehr oder weniger großen erhabenen wirkenden Teil, den ich hier einmal *Schieber* nennen will.

An der Bildlaufleiste kannst du sehen, welcher Teil der Gesamtdarstellung dir im Fenster gezeigt wird: Der Schieber repräsentiert nämlich nach seiner Lage *und* nach seiner Größe den Teil, den du tatsächlich im Fenster siehst, und die Gesamtlänge der Bildlaufleiste entspricht der Gesamtheit dessen was es zu sehen gibt.

Schau dir Abbildung 68 bzw. die hier daraus entnommenen Rollbalken in Abbildung 73 noch einmal an: Der Schieber im unteren (dem waagerechten) Rollbalken ist so groß wie die ganze Bildlaufleiste. Das bedeutet, dass es links und rechts nichts mehr zu sehen gibt; und in der Tat siehst du im Fenster die komplette Breite der beschriebenen Seite. Hingegen ist der Schieber in der vertikalen Bildlaufleiste (also der am rechten Rand) sehr klein. Das bedeutet, dass es über und unter dem angezeigten Teil des Dokuments noch eine Menge zu sehen gibt, und zwar nach oben hin mehr als nach unten, weil der Schieber ziemlich unten steht.

Du kannst den Schieber mit der Maus bewegen (klicken, festhalten und ziehen) und veränderst damit den Teil, den du mit deinem Fenster betrachtest.



Wenn du oberhalb des Schiebers in die Bildlaufleiste klickst, wird das Fenster seitenweise nach oben bewegt, so dass du den Teil siehst, der genau oberhalb dessen steht, was du vorher gesehen hast. Genauso geht das natürlich, wenn du unter bzw. links oder rechts des Schiebers klickst.

Wenn du auf die kleinen Pfeilsymbole an den äußeren Enden der Rollbalken klickst, wird das Fenster jeweils ein kleines Stück in die entsprechende Richtung bewegt. Wenn du diese kleinen Tasten festhältst, beginnt das Bild zu rollen.



Abbildung 73: Rollbalken

6.2.2.5 Menüs

Darüber was ein Menü ist, haben wir schon in den Kapiteln 6.2.1.2, 6.2.1.3 und 6.2.1.5 gesprochen: Eine Auswahl an Möglichkeiten, die einem möglichst übersichtlich angeboten wird. Fast alle Programme bieten die Dinge, die man mit ihnen tun kann auch in einem Menü an:

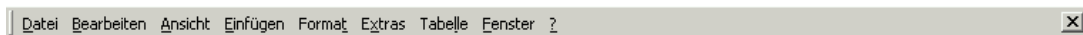


Abbildung 74: Beispiel für eine Menüleiste

Wie schon zuvor beschrieben, werden auch hier die vielen Befehle zu Gruppen zusammengefasst und damit man sich nicht jedes Mal neu an eine bestimmte Systematik gewöhnen muss, sind die Menüs der meisten Programme zumindest teilweise ähnlich, weil es Dinge gibt, die man mit jedem Programm tun können muss. Dazu zählt zum Beispiel das Öffnen der Datei, die man mit dem Programm bearbeiten will: Mit einem Grafikprogramm wird man eine Bilddatei und mit einer Textverarbeitung eine Textdatei bearbeiten wollen, aber man muss sie auf jeden Fall von der Festplatte in den Arbeitsspeicher kopieren (= öffnen) und möglichst auch wieder speichern können. Alle Befehle zu diesem Thema findet man fast immer unter einem Menü mit dem Namen *Datei* und das steht fast immer ganz links in der Menüleiste eines Programmfensters.

Bearbeiten ist auch ein Standard-Menüpunkt. Darunter befinden sich zumindest die Befehle Ausschneiden, Kopieren und Einfügen, die im Kapitel 6.2.3 über die Zwischenablage erklärt werden.

Mit den Befehlen unter dem Menü *Ansicht* kann man immer zwischen verschiedenen Darstellungsarten einer Datei wählen, die ein Programm so bietet. Zum Beispiel kann man in Grafikprogrammen darunter meistens die Vergrößerung einstellen, mit der das Bild dargestellt wird. Manchmal kann man darunter sogar auswählen, ob und welche Symbolleisten dargestellt werden. Mehr darüber im nächsten Abschnitt:

6.2.2.6 Symbolleisten

Menüs mit vielen Untermenüs können manchmal ziemlich umständlich sein! Wenn du dich für einen Befehl, den du häufiger brauchst, erst durch mehrere Untermenüs hangeln musst, kann das ziemlich lästig werden. Außerdem sind die Geschmäcker sehr verschieden: Es gibt Leute, die lesen lieber, und welche die Bilder schöner finden. Aus diesem Grund hat man Symbolleisten erfunden:



Abbildung 75: Beispiel für eine Symbolleiste (hier: Word)

Darin werden kleine Bildchen gezeigt, auf die man klicken kann, um einen bestimmten Programmbefehl auszuführen.

Mit dem Bild einer Diskette wird zum Beispiel *symbolisiert*, dass man damit speichern kann, das Bild eines Druckers steht für den Befehl „Drucken“.

Weil die Bildchen auf dem Bildschirm aussehen wie ein Schalter, den man hineindrückt, nennt man die Symbole auch *Schaltflächen*. Im Englischen, das man ja bei Computerleuten auch manchmal hört, wird eine solche Schaltfläche *Icon* (sprich: eiken) genannt.

Wenn du mit dem Mauszeiger über ein Symbol fährst, *ohne darauf zu klicken*, erscheint meist ein kleines gelbes Fenster, das dir den Befehl nennt, der mit diesem Symbol ausgeführt werden kann. Diese Fenster heißen *Quick Info* und man kann sie auch abschalten (Menü *Extras/Optionen*, Registerkarte *Ansicht*).

6.2.2.7 Tastaturkürzel

Dieser Abschnitt beschreibt gar keine Funktion eines Fensters – allenfalls ein paar unscheinbare Unterstriche. Es geht darum, wie man die Programmfunktionen, die man sonst mit Menüs und Schaltflächen unter Zuhilfenahme der Maus aufruft, auch mit der Tastatur erreichen kann:

Mit der Taste [Alt] oder der Taste [F10] kannst du das Menü sozusagen „einschalten“ oder besser gesagt: Du kannst damit von *Dateneingabe* auf *Befehlseingabe* *umschalten*. Du erkennst das daran, dass die Menüs wie eine Schaltfläche aussehen. (Sollte dir das versehentlich passieren, brauchst du nur die [Esc]-Taste drücken und alles ist wie vorher.)

Mit den → Pfeiltasten kannst du dich dann im Menü bewegen und die Eingabetaste wählt den Befehl aus, den du brauchst.

Statt der Pfeiltasten kannst du auch eine Buchstabentaste betätigen und zwar jeweils diejenige, die in dem Menü, das du anwählen willst, unterstrichen ist. Wenn es keine unterstrichenen Buchstaben gibt, versuche es mit den Anfangsbuchstaben des Menüs!

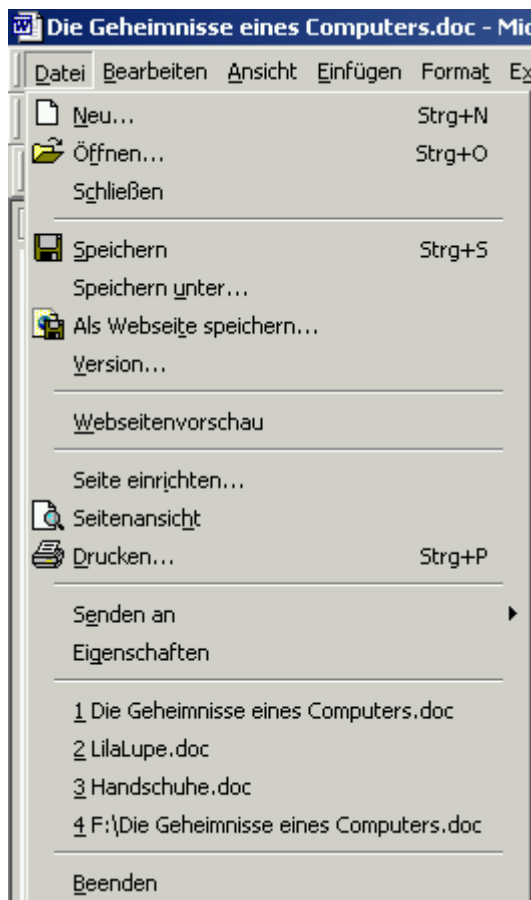


Abbildung 76: Befehle in einem Menü mit zugehörigen Symbolen und Tastaturkürzeln

Und dann gibt es noch eine dritte Möglichkeit: Für häufig benutzte Befehle, stellen die meisten Programme Tastenkombinationen bereit. Auf Englisch nennt man sie *Shortcuts*, was so viel heißt wie Abkürzung, weil es gegenüber dem langwierigen Weg durch die Menüs eine Abkürzung zu dem gewünschten Befehl ist.

Wenn du dir das Dateimenü in Abbildung 76 ansiehst, erkennst du links neben einigen Befehlen das Symbol, mit dem dieser Befehl aus der Symbolleiste aufgerufen werden kann, und rechts von manchen Befehlen steht eine Tastenkombination. Wenn du in Word also zum Beispiel [Strg]+[P] drückst, bedeutet es das selbe als wenn du im Menü Datei den Befehl Drucken mit der Maus anklickst.

Welche Tastaturkürzel es gibt, hängt vom jeweiligen Programm ab! Das ist keine Funktion des Betriebssystems mehr (gehört streng genommen also nicht mal in dieses Kapitel!).

Am Beispiel des Befehls „Drucken“ will ich noch einmal alle Möglichkeiten auflisten, die man normalerweise hat, um einen Befehl in einem Programm aufzurufen:

- Du kannst mit der Maus auf die Schaltfläche mit dem Druckersymbol klicken.
- Du kannst mit der Maus erst auf das Menü „Datei“ und dann auf den Befehl „Drucken“ klicken
- Du kannst die Tasten [Alt] [D] [D] drücken.
- Du kannst [Strg]+[P] drücken.

Auf einen kleinen Unterschied zwischen der ersten und den anderen Möglichkeiten will ich noch hinweisen: Wenn du die Symbolleiste benutzt, wird sofort gedruckt. Die anderen Möglichkeiten führen dich über das Menü und wenn du in Abbildung 76 genau hinsiehst, kannst du sehen, dass das Menü gar nicht „Drucken“ sondern „Drucken...“ heißt! Sind die drei Punkte denn so wichtig? Sie sagen dir, dass der Befehl nicht sofort ausgeführt wird, sondern dass zuerst noch ein Fenster aufgeht, in dem du etwas einstellen kannst. In diesem Beispiel

kannst du in diesem Fenster den Drucker auswählen, sagen welche Teile wie oft in welcher Reihenfolge gedruckt werden sollen usw.

6.2.3 Zwischenablage

Auf Englisch heißt die Zwischenablage *Clipboard*, was so viel bedeutet wie Klemmbrett. An diesem Bild lässt sich ihre Funktion auch ganz gut beschreiben:

An einem Klemmbrett kann man Notizen oder andere Zettel festmachen, die man zwar im Moment nicht braucht, aber später wieder verwenden will.

So ähnlich funktioniert die Zwischenablage deines Computers. Du kannst in fast allen Programmen wie zum Beispiel Textverarbeitungen, Grafikprogrammen, → Internetbrowsern, usw. Teile dessen, was du gerade bearbeitest, quasi an ein Klemmbrett hängen, um sie später in der selben Anwendung *oder einer anderen (!)* wieder zu benutzen.

Dazu musst du nur folgende Schritte durchführen:

1. Markiere den Teil, den du an das Klemmbrett hängen willst.
2. Hänge den Teil ans Klemmbrett. Dafür gibt es zwei Möglichkeiten:
 - a. Schneide den Teil an der ursprünglichen Stelle aus. Das bedeutet das selbe wie löschen! Ganz verloren ist er aber nicht, weil er ja am Klemmbrett hängt.
Ausschneiden geht mit dem Menüpunkt *Bearbeiten/Ausschneiden* oder mit den Tasten [Strg]+[X] oder [⌘]+[Entf]
 - b. Kopiere den Teil in die Zwischenanlage. Dann bleibt er an der ursprünglichen Stelle erhalten und hängt zusätzlich am Klemmbrett. Das geht mit dem Menüpunkt *Bearbeiten/Kopieren* oder den Tastenkombinationen [Strg]+[C] oder [Strg]+[Einf]
3. Bewege die Einfügemarke dorthin, wo du das Teil, das am Klemmbrett hängt, einfügen möchtest. Das kann auch in einer anderen Anwendung sein!
4. Füge das Teil dort ein, indem du *Bearbeiten/Einfügen* wählst, oder die Tasten [Strg]+[V] oder [⌘]+[Einf] betätigst.

Die Geschichte mit dem Klemmbrett ist natürlich nur eine bildliche Vorstellung! In Wirklichkeit wird das Teil, das du in die Zwischenablage übernommen hast, von Windows in einen separaten Teil des Arbeitsspeichers geschrieben bzw. von dort wieder geholt.

Bei jedem neuen Zwischenspeichern wird der alte Inhalt der Zwischenablage überschrieben, auch dann, wenn du das darin befindliche Teil noch nicht wieder verwendet hast! (Manche Programme bieten auch Zwischenablagen mit mehre-

ren Speicherplätzen, standardmäßig hat Windows aber nur eine. Für meinen Geschmack genügt das auch!)

Wegen der Erklärung, dass die Zwischenablage ein Stück Arbeitsspeicher ist, weißt du natürlich schon, dass die Zwischenablage auch gelöscht wird, wenn du den Computer ausschaltest!

Es gibt Programme, die fragen dich beim Beenden, ob große Elemente, die du in die Zwischenablage kopiert hast, dort bleiben sollen. Damit soll vermieden werden, dass man unnötig Arbeitsspeicher belegt hält.

6.2.4 Arbeitsplatz

Auf deinem Windows-Desktop findest du ein Symbol, mit dem wir schon in einigen Experimenten gearbeitet haben:



Abbildung 77: Desktop-Symbol für den Arbeitsplatz

Es sieht fast genauso aus wie die anderen → Verknüpfungen auf dem Desktop, es ist aber keine. Es ist eine Funktion des Betriebssystems, ähnlich wie der Knopf für das Startmenü. Du kannst den Unterschied erkennen, wenn du dir die *Eigenschaften* von Arbeitsplatz ansiehst (rechte Maustaste!): Dort gibt es eine ganze Menge über deinen Computer zu erfahren, auf das ich jetzt hier gar nicht näher eingehen will. Ich will dir damit nur zeigen, dass das Eigenschaftsfenster von „Arbeitsplatz“ ganz anders aussieht als das einer Verknüpfung.

Der Arbeitsplatz repräsentiert – deinen Arbeitsplatz! Irgendwie logisch, oder? Gemeint ist damit dein gesamter Computer. Wenn du auf den Arbeitsplatz doppelklickst, wird ein Programm geöffnet, mit dem du auf alles, was zu deinem Computer gehört, zugreifen kannst, also auf alle Laufwerke und auf die *Systemsteuerung*, unter der alle anderen → Ressourcen deines Rechners wie Maus, Bildschirm, Drucker, Scanner usw. zusammengefasst sind.


Der Arbeitsplatz begegnet dir auch an anderer Stelle: Kannst du dich an die verschachtelten Kisten aus Kapitel 5.1.3.4.3 erinnern? (Wenn nicht, solltest du ab Seite 70 noch einmal nachlesen!) Der Arbeitsplatz ist die *äußerste* Kiste! Wenn du irgendwo eine Datei suchst, steht an oberster Stelle deines Verzeichnisbaums, wie man die verschachtelten Kisten ja auch nennt, immer der Arbeitsplatz. Er ist der Stamm des Dateibaums.

Das Programm, das du durch Doppelklick auf Arbeitsplatz öffnest heißt *Explorer* und wird im nächsten Kapitel erklärt.

6.2.5 Explorer

Der Explorer – oder genauer heißt es Windows-Explorer – ist das Programm, mit dem du die LEGO®-Steine in die verschachtelten Kisten sortierst – äh, ich meine natürlich Dateien in die Ordner verschiebst. Vom Explorer bekommst du neue Kisten – äh, Ordner, wenn du welche brauchst und du kannst mit seiner Hilfe Kisten öffnen und nachsehen was drin ist, und noch viel mehr! Alles was du mit einer Datei anstellen kannst, außer ihren Inhalt zu bearbeiten – denn dazu hast du ja die anderen Anwendungsprogramme – machst du mit dem Explorer! Die Einzelheiten erfährst du in den folgenden Abschnitten.

Starten kannst du den Explorer auf viele verschiedene Weisen:

- Doppelklick auf Arbeitsplatz
- Doppelklick auf ein beliebiges Ordnersymbol auf dem Desktop (z.B. *Eigene Dateien*)
- Doppelklick auf *Papierkorb*
- Start/Ausführen und „Explorer“ eingeben
- Kontextmenü (rechte Maustaste) des Start-Buttons 

6.2.5.1 Öffnen

Wenn du den Arbeitsplatz geöffnet hast (Doppelklick darauf), siehst du im Explorerfenster alle Laufwerke deines Rechners. Du kannst nachsehen, welche Dateien und Verzeichnisse sich auf einem Laufwerk befinden, indem du mit der linken Maustaste darauf doppelklickst.



Im Explorerfenster wird dann der Inhalt dieses Laufwerks angezeigt.



(Es kann auch sein, dass der Inhalt in einem neuen Fenster angezeigt wird. Diese Einstellung sollte man aber ändern, man macht sonst irgendwann nichts anderes mehr als Explorerfenster zu schließen! Das geht im Menü *Extras/Ordneroptionen*, Registerkarte *Allgemein* unter *Ordner durchsuchen*.)

Wenn du auf ein Verzeichnis in diesem neuen Fenster doppelklickst, passiert das gleiche wieder: Im Explorerfenster erscheint der Inhalt dieses Ordners. So kannst du dich Kiste für Kiste – äh, Ordner für Ordner durchhangeln. Ein Doppelklick auf einen Ordner ist gleichbedeutend mit dem Öffnen einer LEGO-Kiste aus dem Beispiel.



Noch einfacher wird das Navigieren in den ganzen Verzeichnissen, wenn du in der Symbolleiste des Explorers auf „Ordner“ (siehe links) klickst oder das Menü *Ansicht / Explorer-Leiste / Ordner* wählst:

Im linken Teil des Fensters, der *Explorer-Leiste* heißt, wird dir nun die Verzeichnisstruktur als Baum dargestellt ähnlich wie schon in Kapitel 5.1.3.4.3 beschrieben. Durch Klick auf das kleine Plusymbol   neben einem Laufwerk

oder Verzeichnis kannst du dessen Unterverzeichnisse anzeigen lassen. Ein Klick auf das dann erscheinende Minuszeichen   blendet sie für bessere Übersichtlichkeit wieder aus.

Ein Klick auf den Verzeichnisnamen selbst zeigt den gesamten Inhalt (Unterverzeichnisse *und* Dateien) im rechten Explorerfenster an.

Nicht nur Verzeichnisse, auch Dateien kannst du öffnen. Dazu muss natürlich zuerst die Anwendung geladen werden, mit der man die Datei bearbeiten kann. Wenn du auf eine Datei doppelklickst, wird zuerst das für diesen → Dateityp vorgesehene Anwendungsprogramm in den Arbeitsspeicher geladen und dann die Datei. Wenn du dich an Kapitel 5.1.3.4.2 erinnerst, weißt du noch, dass der Computer an Hand der Dateiendung weiß, welches Anwendungsprogramm er laden soll. Diese Zuordnung kannst du im Explorer übrigens auch ändern: Im Menü *Extras/Ordneroptionen*, Registerkarte *Dateitypen* sind alle Zuordnungen aufgelistet und durch Anklicken von *Ändern* kann man die gerade markierte Zuordnung verändern.

Wenn man eine Datei nur ausnahmsweise mit einer anderen als der voreingestellten Anwendung bearbeiten möchte, kann man → im Kontextmenü dieser Datei den Punkt *Öffnen mit ...* wählen. (Falls der nicht erscheinen sollte, muss man das Kontextmenü bei gedrückter Umschalttaste [↑] aufrufen.)

6.2.5.2 Sortieren

Obwohl wir mit der Möglichkeit Ordner und Unterordner zu verwenden ein tolles Hilfsmittel haben, um Ordnung zu schaffen, kommt es vor, dass in einem Verzeichnis sehr viele Dateien oder Unterverzeichnisse stehen (ein Hilfsmittel *schafft* eben keine Ordnung, es unterstützt einen nur bei dieser lästigen Arbeit).

Aus diesem Grund wäre es doch ganz hilfreich, wenn man die vielen Ordner und Dateien alphabetisch auflisten könnte.

Manchmal möchte man auch wissen, an welcher Version einer Datei man zuletzt gearbeitet hat, welche also die aktuellste ist.

In diesen Fällen ist es sehr hilfreich, wenn man die Ansicht der Dateien sortieren kann. Das geht so:

1. Wähle im Explorer-Fenster das Menü *Ansicht/Details*. Dadurch werden alle Ordner und Dateien untereinander in einer Liste dargestellt.
2. Diese Liste hat Überschriften: Dateiname, Größe, Typ, Geändert, usw. (Mit dem Menüpunkt *Ansicht / Spalten auswählen ...* kannst du diese Überschriften übrigens auch ändern!).

Dateiname	Größe	Typ	Geändert ▾
 Die Geheimnisse eines Computers.doc	12.409 KB	Microsoft Word-Dokument	02.02.2005 23:56

Abbildung 78: *Detailansicht in einem Explorer-Fenster*

Neben einer dieser Überschriften findest du einen kleinen Pfeil. Der bedeutet, dass die Liste nach dieser Spalte sortiert ist! In Abbildung 78 ist das die Spalte „Geändert“, in der steht, wann die Datei zuletzt gespeichert worden ist.

3. Wenn du auf eine der Überschriften klickst, wird die Liste nach der Spalte, auf die du geklickt hast, umsortiert! Klickst du auf die Spalte, nach der bereits sortiert ist (die mit dem Pfeil), wird in der umgekehrten Reihenfolge sortiert. Wenn also in Abbildung 78 die neueste Datei oben steht, würde bei erneutem Klicken auf *Geändert* die älteste Datei zuoberst in der Liste stehen. Du erkennst das daran, dass der kleine Pfeil dann andersherum zeigt.

6.2.5.3 Suchen

Wenn selbst unsere vielen, ordentlich angelegten Unterordner, unsere sorgfältig ausgedachten Dateinamen und die sortierten Ansichten des Explorers nicht mehr helfen, eine Datei wiederzufinden, dann haben wir noch eine letzte, mächtige Chance, mit der wir jede Datei aufspüren können, die sich irgendwo auf unserem Computer befindet, sofern wir wenigstens *irgend etwas* darüber wissen! Das geht so:



In der → Symbolleiste des Explorers findest du eine Schaltfläche *Suchen*; die sieht so aus wie das nebenstehende Bildchen zeigt.

Wenn du darauf klickst, wird dir in der Explorer-Leiste eine Eingabemöglichkeit für deine Suchkriterien geöffnet.

Das selbe erreichst du auch, wenn du das Menü *Ansicht / Explorer-Leiste / Suchen* wählst! Da es noch mehr Menüpunkte unter *Ansicht / Explorer-Leiste* gibt, hast du dir bestimmt schon gedacht, dass man sich da auch andere Dinge anzeigen lassen kann?! Das stimmt! Über den Punkt „Ordner“ haben wir schon gesprochen, die anderen kannst du selbst ausprobieren. Zurück zum Suchen:

Abbildung 79 zeigt das Suchfenster. Fangen wir wieder oben an:

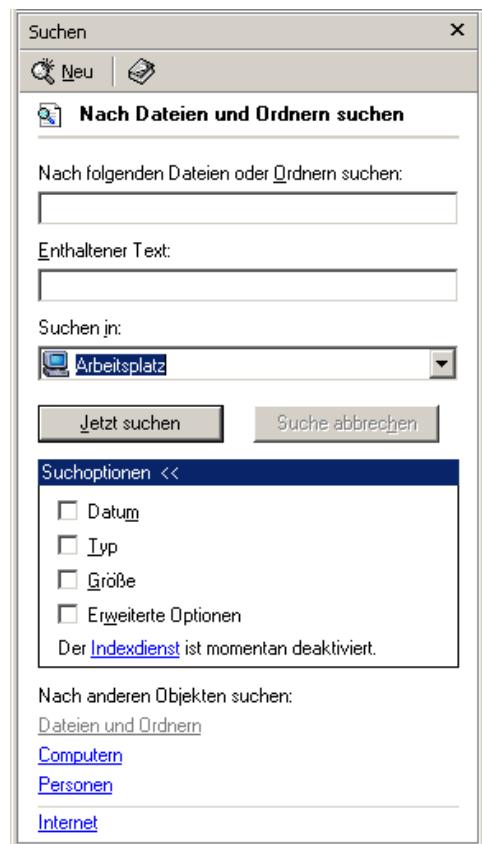


Abbildung 79: Suchen im Explorer

Der Titel *Suchen* ist selbsterklärend. Das Kreuz bedeutet das selbe wie bei anderen Fenstern auch. Damit kannst du die Explorer-Leiste schließen.

Mit „*Neu*“ kann man bereits eingegebene Suchkriterien löschen, um nach etwas anderem zu suchen und das Buch mit dem Fragezeichen ruft einen Hilfetext zu diesem Fenster auf.

Darunter gibt es ein Eingabefeld *Nach folgenden Dateien oder Ordnern suchen.*; in dem du den Dateinamen der Datei, die du suchst, oder auch nur Teile davon eintragen kannst. Wenn du beispielsweise die Datei „Die Geheimnisse eines Computers.doc“ suchst, aber nur noch weißt, dass im Titel irgendetwas mit „Geheimnis“ vorkam, kannst du in dieses Feld „Geheimnis“ eintippen und dann wird die Datei gefunden – und auch jede andere, die dieses Wort im Dateinamen hat. Natürlich werden umso mehr Dateien gefunden, je kleiner der Teil ist, der dir noch einfällt! Wenn du nur noch weißt, dass im Dateinamen ein „a“ vorkommt und du „a“ in dieses Feld eintippst, wird jede Datei gefunden, die irgendwo im Dateinamen ein „a“ hat, und das sind ziemlich viele!

Falls du nicht mal mehr weißt, wie die gesuchte Datei ungefähr heißt, sondern nur, dass ein bestimmter Text in der Datei steht, dann benutzt du das nächste Eingabefeld, über dem *Enthaltener Text:* steht. Darin trägst du den Text ein, von dem du weißt, dass er in der Datei vorkommt, die du suchst. Wenn ich beispielsweise nach der Datei suchen würde, die den Text dieses Buches enthält, und mich genau an diesen Absatz hier erinnern würde, könnte ich in dieses Feld eingeben „*Wenn ich beispielsweise nach der Datei suchen würde*“.

Früher oder später würde der Explorer in der Liste der Suchergebnisse die Datei „*Die Geheimnisse eines Computers.doc*“ auflisten.

Auch hier gilt natürlich wieder, dass umso weniger Dateien als Ergebnis angezeigt werden, je länger der Suchtext ist. Im obigen Beispiel gibt es ziemlich sicher nur eine Datei, die genau diesen Satz enthält.

Diese Art der Suche kann sehr lange dauern, weil der Computer eine Datei nach der anderen in den Arbeitsspeicher laden und nach dem Suchtext durchsuchen muss!

Ein weiterer Nachteil ist, dass man damit natürlich nur Dateien finden kann, die *Text* enthalten! Bilder kann man mit dieser Methode nicht finden!

Im dritten Eingabefeld „*Suchen in:*“ kann man die Verzeichnisse, die durchsucht werden, einschränken. Dadurch wird die Suche schneller. Wenn man zum Beispiel weiß, dass die gesuchte Datei sich nur auf der Diskette A:\ befinden kann, braucht man ja nicht auch die Festplatten abzusuchen. Steht in diesem Feld „*Arbeitsplatz*“, so werden *alle* Laufwerke des Computers durchsucht!

Dieses Feld ist ein → Dropdown-Feld, zu erkennen an dem kleinen Pfeil am rechten Rand, mit dessen Hilfe es aufgeklappt wird. Darin sind alle Laufwerke

und ein paar wichtige Verzeichnisse enthalten. Wenn du in einem Verzeichnis suchen möchtest, das in der Auswahl nicht verfügbar ist, dann musst du „Durchsuchen...“ auswählen; dadurch öffnet sich ein Fenster, in dem du das zu durchsuchende Verzeichnis festlegen kannst.

Falls die Suche nach Dateiname oder enthaltenem Text dir nicht weiterhilft, kannst du durch Klick auf „Datum“ gezielt nach Dateien suchen, die in einem bestimmten Zeitraum gespeichert wurden. Durch Auswahl von „Typ“ kannst du sagen welchen Dateityp die Datei hat, die du suchst, und mit „Größe“ kann man die Suche auf Dateien bestimmter Größe beschränken. Unter den „Erweiterten Optionen“ kannst du einstellen, ob Groß-/Kleinschreibung bei der Suche berücksichtigt werden soll oder nicht (ist der Haken gesetzt, wird die Datei „test.doc“ nicht gefunden, wenn du nach „Test.doc“ suchst!), und ob außer dem gewählten Verzeichnis auch die darin enthaltenen Unterverzeichnisse nach der Datei durchsucht werden sollen.

Wenn du mehrere Suchkriterien angibst, also zum Beispiel

„Dateiname enthält: *Geheimnis*, enthaltener Text: *Computer*, zuletzt gespeichert: *heute*, Größe: *mehr als 1000kByte*“

dann werden nur Dateien gefunden, die *alle* Kriterien erfüllen!

6.2.5.4 Umbenennen

Manchmal will man den Namen einer Datei ändern, zum Beispiel weil man den Inhalt verändert hat und der Name nicht mehr richtig passt, oder einfach weil man eine bessere Idee hat, wie man den Inhalt treffender bezeichnen kann.

Das geht im Explorer ganz einfach, und wieder hast du mehrere Möglichkeiten:

Die bequemste ist, das Kontextmenü der Datei, die du umbenennen willst, aufzurufen (mit der rechten Maustaste auf die Datei klicken), und den Menüpunkt Umbenennen anzuklicken. Die nächste Möglichkeit ist, die Datei zu markieren (einmal mit der linken Maustaste anklicken oder mit den Pfeiltasten die Markierung dorthin bewegen) und im Explorer das Menü *Datei/Umbenennen* aufrufen. Noch eine Alternative ist das Drücken der Taste [F2] sobald die fragliche Datei markiert ist.

In jedem Fall erscheint der Dateiname dann in einem Kästchen, in das du den neuen Namen einfach reinschreiben kannst. Der alte Name ist markiert und wird gelöscht, sobald du etwas Neues tippst. Wenn du Teile des alten Namens behalten möchtest, kannst du erst mit den Pfeiltasten die Einfügemarke dorthin bewegen, wo du den Dateinamen ändern möchtest und dann deine Änderung vornehmen.

Sobald du mit Ändern fertig bist, betätige die Eingabetaste [↵].

Auf zwei Dinge solltest du achten, bzw. dich nicht erschrecken, wenn der Computer für dich darauf achtet:

Dateinamen müssen immer eindeutig sein! Das bedeutet, dass es in einem Ordner keine zwei Dateien mit demselben Namen geben darf. Wenn du nun den Namen einer Datei so änderst, dass er identisch ist mit dem einer anderen Datei in diesem Ordner, wird der Rechner dir das sagen und die Änderung nicht zulassen. In diesem Fall musst du dir einen anderen Namen ausdenken oder zuerst die andere Datei umbenennen.

Wenn du dich noch an Kapitel 5.1.3.4.2 erinnerst, dann weißt du noch, dass in einem Dateinamen die Zeichen nach dem letzten Punkt dem Computer sagen, mit welchem Programm die Nullen und Einsen, die in dieser Datei stehen, bearbeitet werden können; diese Zeichen geben den Typ der Datei an. Wenn du diese Zeichen änderst, kann es sein, dass der Computer nicht mehr weiß, in welcher Anwendung er diese Datei öffnen soll. Du *kannst* sie ändern, aber du solltest wissen, was es bedeutet. Damit du eine solche Änderung nicht versehentlich vornimmst, erscheint in diesem Fall eine Warnung.

6.2.5.5 Löschen

Das Schöne am Computer im Vergleich zur Schreibmaschine ist, dass man alles was man mal geschrieben (oder gemalt oder gerechnet oder programmiert) hat, speichern kann. Noch Jahre später kann man solche Dinge wieder hervorkramen, wenn man etwas Ähnliches noch einmal machen will, und kann die alte Arbeit weiterverwenden.

Das führt dazu, dass die Speicher, also vor allem die Festplatte, langsam aber sicher immer voller werden. Schon aus diesem Grund muss man Dateien manchmal auch löschen, um Platz zu schaffen für Neues. (Die Funktionen zum Sortieren nach Datum oder Größe aus Kapitel 6.2.5.2 sind dabei sehr hilfreich!)

Es gibt aber auch andere Gründe, Dateien zu löschen, zum Beispiel weil definitiv Unsinn drin steht oder weil man verhindern möchte, dass andere den Inhalt zu sehen bekommen. Für den letzten Fall gibt es allerdings auch andere Methoden: Manche Betriebssysteme verwalten für jede einzelne Datei Zugriffsrechte, oder man kann Dateien auch verschlüsseln.

Wie dem auch sei: Löschen funktioniert bei Windows wie im richtigen Leben: Man kann etwas in den Papierkorb werfen! Und das tollste: Wie im richtigen Leben kann man es auch wieder herausholen, wenn man sich geirrt hat und es doch noch braucht – allerdings nur, wenn die Müllabfuhr nicht zwischenzeitlich schon da war!

Wie für fast alles andere gibt es auch für das Löschen mehrerer Möglichkeiten:

1. Die einfachste Variante geht wieder über das → Kontextmenü: Suche die Datei, die du löschen willst, (unter Umständen mit Hilfe Suchfunktion o-

der der Ordneranzeige in der Explorer-Leiste) und klicke mit der rechten Maustaste darauf (vergleiche Abbildung 65); Klicke auf „Löschen“.

2. Markiere die Datei (einmal mit der linken Taste anklicken) und drücke die Taste [Entf].
3. Markiere die Datei und wähle das Menü Datei / Löschen.

Egal für welche Variante du dich entscheidest, erscheint ein Fenster wie in Abbildung 80:



Abbildung 80: Löschen in Papierkorb bestätigen

Ausnahme: Wenn sich die Datei, die du löschen möchtest, auf einer Diskette oder einem anderen → Wechseldatenträger befindet, erscheint die Frage wie in Abbildung 82, die später noch erklärt wird.

Was ist das für ein ominöser Papierkorb?

Du hast ihn wahrscheinlich schon gesehen: Auf dem → Desktop wird dir das Symbol angezeigt:

Je nachdem, ob etwas drin ist, sieht es unterschiedlich aus:

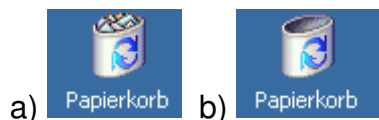


Abbildung 81: Papierkorb a) gefüllt, b) leer

Der Papierkorb ist nichts anderes als ein besonderes Verzeichnis, also ein Teil der Festplatte! Allerdings hat dieses Verzeichnis eine besondere Eigenschaft: Du kannst nämlich festlegen, welche maximale Größe es hat! Bisher haben wir nie über die Größe von Verzeichnissen gesprochen; das war auch gar nicht nötig, denn sie sind immer so groß wie das was drin ist (insofern unterscheiden sie sich von den LEGO-Kisten, mit denen ich sie verglichen habe)!

Beim Papierkorb ist das anders: Er hat eine festgelegte Größe. Immer wenn du eine Datei löscht, wird sie nicht wirklich gelöscht, sondern in das Verzeichnis *Papierkorb* → verschoben. Ist die maximale Größe des Papierkorbs erreicht, muss Platz geschaffen werden; das geschieht, indem die Dateien, die sich dann

am längsten im Papierkorb befinden, endgültig gelöscht werden – so viele wie nötig und ohne dass noch einmal groß nachgefragt wird! Die Müllabfuhr fragt schließlich auch nicht mehr nach, ob sie die Mülltonne wirklich leeren soll, die an der Straße bereit steht.

Der Papierkorb ist sozusagen eine kleine Sicherheit gegen versehentliches Löschen. Man soll sich aber trotzdem keinesfalls darauf verlassen, denn man weiß nie, wann eine Datei dann doch endgültig gelöscht wird.

Die Größe des Papierkorbs kannst du einstellen, indem du im → Kontextmenü seine *Eigenschaften* aufrufst.

Dort kannst du auch einen Haken setzen, ob du den Papierkorb verwenden willst, oder ob du Dateien lieber sofort endgültig löschen willst. Außerdem kannst du einen Haken dafür setzen, ob du die Abfrage aus Abbildung 82 bei jedem Löschvorgang sehen willst.



Abbildung 82: Endgültiges Löschen bestätigen

6.2.5.6 Kopieren

Eine der tollsten Eigenschaften von → digital gespeicherter Information ist, dass man sie mit Hilfe des Computers sehr leicht vervielfältigen kann!

Nehmen wir an, du hättest auf deinem Computer ein wunderschönes Pferdebild, das du dir als Hintergrund auf dem Desktop anzeigen lässt. (Wie das geht? Finde es heraus! Es ist eine *Eigenschaft* deines Desktops!) Deine Freundin sieht dieses Bild und möchte es auch haben. Ohne Computer müsste man es mühsam abmalen oder durchpausen.

Mit dem Computer kannst du diese Bilddatei aber einfach kopieren, und sie sieht exakt so aus wie das Original! Ohne jede Abweichung und ohne Qualitätsverlust! Wenn du dich daran erinnerst, wie Bilder im Computer gespeichert werden (siehe Kapitel 3.5!), wird dir sehr schnell klar warum: Eine Folge von Nullen und Einsen – nichts anderes ist ja eine Datei! – zu lesen und in der selben Reihenfolge woanders wieder hin zu schreiben ist sehr einfach, so einfach

dass das sogar die Maschine „Computer“ kann – sehr schnell und fehlerlos! Damit ist auch klar, dass das ganze nicht nur mit Bildern sondern mit *allen* Dateien geht.

Obwohl das Kopieren digitaler Informationen sehr einfach ist, bedeutet das nicht, dass es auch immer erlaubt ist!

Auch Informationen oder Daten wie Texte, Bilder, Musik, Programme usw. können jemandem gehören! Meistens muss man demjenigen, dem sie gehören, etwas dafür bezahlen, wenn man sie benutzen will! Zumindest muss man fragen, ob man sie benutzen darf.

Für Leute, die davon leben, Programme zu schreiben, ist es ein großes Problem, dass sich Dateien – also auch Programme – so leicht kopieren lassen, denn es gibt natürlich viele, die lieber einfach etwas kopieren – obwohl das verboten ist – als dafür Geld zu bezahlen. Deshalb lassen sich Programmierer oft raffinierte Tricks einfallen, damit man ihre Programme nicht mehr ohne weiteres kopieren kann.

Ansonsten kann man aber jede Datei kopieren, und dafür gibt es mal wieder eine Reihe von Möglichkeiten:

Kopieren bedeutet ja nichts anderes, als eine zweite Datei mit dem selben Inhalt zu erzeugen. Man braucht also immer eine *Quelle* (die Datei, die kopiert werden soll) und ein *Ziel* (die neu erzeugte Datei).

Am einfachsten kopiert man im Explorer, indem man die → Ordneransicht in der Explorerleiste einschaltet, und einfach die Quelldatei mit der Maus dorthin zieht, wo die Kopie nachher stehen soll. Wenn Quelle und Ziel auf unterschiedlichen Laufwerken sind, wird mit dieser Mausektion kopiert; sind sie auf dem selben Laufwerk, wird die Datei verschoben (s. nächstes Kapitel); in diesem Fall musst du zusätzlich die [Strg]-Taste drücken um sie zu kopieren.

Wenn du dir das alles nicht merken kannst oder willst, benutze beim Ziehen einfach die rechte statt der linken Maustaste und du erhältst das Kontextmenü aus Abbildung 67 auf Seite 134.

Ein weiterer Weg, wie man kopieren kann, wurde schon im Kapitel 6.2.3 Zwischenablage beschrieben: Die Befehle Ausschneiden, Kopieren und Einfügen der Zwischenablage funktionieren im Explorer auch mit ganzen Dateien! Allerdings werden dabei nicht die Dateien mit ihrem kompletten Inhalt im Arbeitsspeicher kopiert, sondern der Explorer merkt sich nur den Pfad- und Datei-

namen und kopiert die gesamte Datei dann erst beim Einfügen Befehl auf den Datenträger.

6.2.5.7 Verschieben

Wenn man eine Datei in einem anderen Verzeichnis stehen haben möchte als dort wo sie gerade ist, kann man sie in den Ordner, in dem sie künftig stehen soll, kopieren und sie dann an der alten Stelle löschen. Diese beiden Bearbeitungsschritte erledigt der Explorer auf einmal, wenn man den Befehl *Verschieben* benutzt. Er taucht ebenfalls im Kontextmenü auf, wenn man eine Datei mit der rechten Maustaste zieht.

6.2.5.8 Verknüpfungen erstellen

Über Verknüpfungen haben wir schon in Kapitel 6.2.1.3 gesprochen, dort vor allem im Zusammenhang mit Menüs. Verknüpfungen kann man aber überall erstellen! Beispielsweise könnte es sein, dass eine Datei bei deiner Sortierlogik – denke an das Beispiel mit den LEGO-Kisten! - in zwei Verzeichnisse einsortiert werden könnte. Trotzdem wäre es Unsinn, diese Datei zweimal auf deinem Rechner zu speichern, vor allem dann, wenn du sie auch noch regelmäßig verändern willst! Dann müsstest du nämlich jedes Mal zwei Dateien ändern! Statt dessen legst du einfach in einem der beiden Verzeichnisse eine Verknüpfung an.

Das geht genauso wie das Kopieren und Verschieben mit der Maus: Am einfachsten durch Ziehen mit der rechten Maustaste. Wenn du eine Datei mit der linken Maustaste von einem Ordner in einen anderen ziehst, musst du dabei die [Alt]-Taste gedrückt halten, um eine Verknüpfung zu erstellen. dabei erscheint das Verknüpfungssymbol an deinem Mauszeiger.

6.2.5.9 Markieren

Bei den Erklärungen in den vorangegangenen Kapiteln war öfters von *Markieren* die Rede: Zum Beispiel solltest du Dateien, die du umbenennen oder löschen wolltest, markieren und dann den entsprechenden Menüpunkt auswählen.

Das ist ein Grundsatz beim Explorer und den meisten anderen Programmen:

Bevor du die Aktion (Umbenennen, Löschen, ...) auswählst, musst du das Objekt, also das Ding, mit dem die Aktion ausgeführt werden soll, auswählen. Dazu markiert man es.

Einzelne Objekte markiert man, indem man einmal mit der linken Maustaste darauf klickt. Sobald man auf ein anderes Objekt klickt, wird die alte Markierung entfernt.

Häufig will man aber mehrere Sachen gleichzeitig markieren. Dazu gibt es mehrere Möglichkeiten:

Wenn man beim Klicken die [Strg]-Taste gedrückt hält, wird die zuvor getätigte Markierung nicht gelöscht. Dadurch kann man mehrere Objekte einzeln anwählen.

Hält man statt dessen die Umschalttaste [⇧] gedrückt, werden alle Objekte, die sich zwischen dem ersten und dem letzten Klick befinden, markiert.

Oft kann man mit der Maus auch ein Rechteck aufziehen (links klicken, Taste festhalten, Maus bewegen), dann wird alles was sich in diesem Rechteck befindet markiert. Allerdings darf man dabei kein Objekt direkt anklicken, sondern man muss daneben klicken.

Mit dem Menü *Bearbeiten / Alles markieren* oder der Tastenkombination [Strg]+[A] werden alle Objekte in der aktuellen Anzeige markiert.

Spiele ein wenig damit! Entdecke die Möglichkeiten!

6.2.6 Hilfe

Wenn du trotz der vielen Bücher, die du über Computer gelesen hast, einmal nicht weißt, wie etwas funktioniert, dann kannst du dir vom Computer selbst helfen lassen!

Sowohl für das Betriebssystem Windows und den Explorer als auch für fast jedes Anwendungsprogramm gibt es eine Hilfefunktion, die – wenn sich die Programmierer an die Richtlinien für die → Benutzerschnittstelle gehalten haben – mit der Taste [F1] aufgerufen wird. Außerdem gibt es einen Menüpunkt „?“, mit dem man die Hilfefunktion erreicht.

In den Hilfe-Programmen kann man meistens suchen oder in den Kapiteln ähnlich wie in einem Verzeichnisbaum navigieren.

Bevor man lange herumprobiert oder gar entnervt aufgibt, lohnt sich auf jeden Fall ein Blick in die Hilfetexte! Sie sind zwar manchmal etwas hochgestochen geschrieben und man hat das Gefühl, genauso schlau zu sein wie vorher, aber meistens findet man wenigstens einen Hinweis, der einem beim weiteren Probieren hilft!

Lass dich auch nicht von der Länge der Texte abschrecken! Wer solch ein dickes Buch wie dieses hier lesen kann, für den ist ein Windows-Hilfetext ein Klacks!

Man muss sich immer vor Augen halten, dass Dinge etwas mehr Zeit und Aufwand erfordern, wenn man sie zum ersten Mal macht! Du kannst nicht erwarten, dass neue Dinge gleich beim ersten Mal klappen; es ist völlig normal, wenn du ein wenig Zeit brauchst, bis du Neues richtig verstanden hast! Sobald du es aber verstanden (und vielleicht ein wenig geübt) hast, erledigst du deine Arbeit mit Hilfe des Gelernten viel, viel schneller und holst so den Aufwand für das Lernen wieder herein! Außerdem macht es einfach auch Spaß herauszufinden, wie bestimmte Dinge funktionieren! Manchmal muss man den Computer und seine Programme auch als Rätselaufgabe verstehen!

7 Software

Nun sind wir – endlich – bei dem angelangt, was Computer so universell einsetzbar und für fast alles (außer Zimmer aufräumen) nutzbar macht: Die Programme.

Es gibt sie für jeden nur denkbaren Zweck, und wenn es für irgendwas mal kein Programm gibt, dann gibt es Programme, um Programme dafür zu schreiben, aber das hatten wir ja alles schon.

*Oft nennt man Programme, die einem bestimmten Zweck dienen, als **Abgrenzung zum Betriebssystem Anwendungsprogramme oder Anwendungen.***

In diesem Kapitel erfährst du grundsätzliche Dinge zum Arbeiten mit Anwendungsprogrammen sowie einige Kapitel zu den wichtigsten Anwendungen wie zum Beispiel Textverarbeitung.

7.1 Installieren von Anwendungsprogrammen

Wenn du dir ein Anwendungsprogramm gekauft (oder aus dem → Internet ein kostenloses Programm heruntergeladen) hast, kannst du so noch nicht damit arbeiten, denn du hast nur eine CD, auf der alle Dateien, die das Programm benötigt, enthalten sind. In aller Regel musst du deine Software erst *installieren*.

Damit sind verschiedene Dinge gemeint:

Es beginnt mit so banalen Dingen, dass du deine Anwendung natürlich gerne mit dem Startmenü aufrufen können möchtest; es muss also eine Verknüpfung erstellt werden.

Meist gehören zu einer Anwendungssoftware eine ganze Reihe von Dateien, die in verschiedenen Verzeichnissen stehen müssen, damit die Software richtig läuft. Manchmal müssen auch Verzeichnisse neu angelegt werden, die das Programm braucht, um bestimmte Dinge zu speichern. Dies können zum Beispiel die Einstellungen der verschiedenen Benutzer oder Spielstände sein.

Je nachdem welchen Computer du hast, werden vielleicht auch nur bestimmte Dateien benötigt, weil es verschiedene Versionen geben kann. Oder du möchtest aus Platzgründen nicht alle Funktionen, die die Software zu bieten hat, auf deiner Festplatte haben.

Um alle diese Dinge kümmert sich ein kleines Programm!

Es heißt fast immer „*setup.exe*“, manchmal auch „*install.exe*“. Wenn die CD nicht ohnehin automatisch startet und dieses Programm aufruft, dann kannst du es im Explorer suchen und darauf doppelklicken oder sie über das *Startmenü / Einstellungen / Systemsteuerung / Software* und dann *Neue Programme hinzufügen* aufrufen.

Dieses Programm erledigt die Installation und fragt meist auch einige deiner Wünsche ab: Zum Beispiel wie das Verzeichnis heißen soll, in das die Programmdateien kopiert werden oder ob auch ein Symbol zum Starten des Programms auf dem Desktop erzeugt werden soll. Bei gekaufter Software ist auch immer eine *Lizenzvereinbarung* dabei, in der steht, wie du das Programm verwenden, und ob du es kopieren darfst.

Du solltest immer aufmerksam lesen was dort steht und die entsprechenden Eingaben machen.

Meistens gibt es auf der CD auch eine Datei mit dem Namen „*Readme.txt*“ oder „*Readme.doc*“. Manchmal heißt sie auch „*Liesmich.txt*“, denn das ist die deutsche Übersetzung dieses englischen Dateinamens und sagt auch schon aus, was du damit tun sollst. In solchen Dateien stehen Informationen zu dem Programm, zum Beispiel wenn es bekannte Fehler gibt oder dieses Programm nicht gleichzeitig mit anderen betrieben werden kann.

Bei sehr einfachen Programmen, die keine Hilfefunktion über die [F1]-Taste anbieten, ist manchmal auch eine separate Anleitungsdatei mit auf der CD. Die solltest du natürlich auch lesen oder wenigstens überfliegen, so dass du weißt, welche Art von Information du dort im Zweifelsfall findest.

7.2 Laden von Programmen und Daten

Was ein *Programm* ist, hast du ja schon gelernt: Es sind die Befehle, die der Computer ausführen soll. Damit er das tun kann, muss das Programm in den Arbeitsspeicher geladen werden. Das passiert, wenn du seinen Startmenü-Eintrag anklickst.

Im Gegensatz dazu sind *Daten* die Dinge, mit denen der Computer etwas tun soll, also zum Beispiel ein Text, ein Bild oder einfach Zahlen, mit denen der Computer rechnen soll. Die Daten müssen **auch** im Arbeitsspeicher stehen, wenn der Rechner etwas damit tun soll! Allerdings ist das nur dann sinnvoll, wenn das Programm, mit dem du die Daten bearbeiten willst, auch schon im Arbeitsspeicher steht.

Wenn du im Explorer auf eine Datendatei doppelklickst lädt der Computer deshalb immer auch das Anwendungsprogramm gleich mit, das du zum Bearbeiten dieser Daten benötigst. Welches das ist, erkennt er am → Dateityp.

Ein Programm hingegen kannst du auch aufrufen, ohne dabei Daten gleich mit zu laden, denn es kann ja auch sein, dass du neue Daten erzeugen und in einer neuen Datei speichern möchtest.

7.3 Anwendungsprogramme

7.3.1 Textverarbeitung

Mit einem Textverarbeitungsprogramm kann man – wer hätte das gedacht! – Text verarbeiten! Aber was bedeutet das?

Zuerst einmal bedeutet es *schreiben!* Den Text, den du gerade liest, habe ich auf der Tastatur meines Computers eingetippt. Während ich das getan habe, hat mir der Computer auf dem Bildschirm gezeigt, wie der Text später auf dem Papier aussehen würde (→ WYSIWYG). Das ist alles? Nein!

Zum Beispiel hat sich der Computer von ganz alleine darum gekümmert, wann eine neue Zeile begonnen werden muss! Selbst wenn ich irgendwo nachträglich ein Wort eingefügt oder gelöscht habe, hat der Computer den Text dieses Absatzes automatisch so neu angeordnet, dass an den richtigen Stellen eine neue Zeile beginnt. Dabei hat er sogar die Silbentrennung berücksichtigt, ohne dass ich daran einen Gedanken verschwenden musste!

Selbstverständlich ging auch die Seitennummerierung vollautomatisch, ohne dass ich auf jeder Seite die Zahl von Hand hätte eintippen müssen!

Außerdem hat der Rechner – noch während ich getippt habe! – überprüft, ob ich Rechtschreibfehler gemacht habe! (Trotzdem wirst du den einen oder anderen Fehler gefunden haben, weil die Rechtschreibprüfung nicht alle Fehler finden *kann*.)

Ich konnte dem Computer sagen, in welcher Schriftart mein Text erscheinen soll und welche Worte **fett**, *kursiv* oder unterstrichen dargestellt werden sollen.

Das Inhaltsverzeichnis hat der Rechner automatisch erstellt, indem ich nur gesagt habe, was eine Überschrift ist. Die Überschriften wurden automatisch nummeriert und diese Nummerierung auch jedes Mal aktualisiert, wenn ich irgendwo ein Kapitel eingefügt oder gelöscht habe!

Auch die Nummerierung der Abbildungen, Tabellen und Experimente hat der Computer ganz allein immer richtig gemacht. Sogar die Verweise auf andere Kapitel – wenn ich also zum Beispiel geschrieben habe, dass du im Kapitel 7.3.1 etwas über Textverarbeitung erfährst – wurden immer automatisch geändert, wenn sich an der Nummerierung etwas geändert hat!

Sogar eine Reihe von Zeichnungen habe ich mit dem Textverarbeitungsprogramm angefertigt, zum Beispiel Abbildung 1 bis Abbildung 9. Ich habe auch

eingeben können, an welcher Stelle ein Bild erscheinen soll, das dann aus einer Datei in den Text hineinkopiert wurde.

Du siehst: Eine Textverarbeitung kann einem sehr viel Arbeit abnehmen, um die man sich beim Schreiben mit einer Schreibmaschine selbst kümmern müsste. Dazu kommt noch der Vorteil, dass der Text auf der Festplatte gespeichert ist, und dadurch auch nachträglich beliebig geändert werden oder beliebig oft gedruckt werden kann – zumindest wenn man genügend Papier hat.

Selbst wenn du nicht unbedingt ein ganzes Buch schreiben willst, solltest du die Vorteile eines Textverarbeitungsprogramms nutzen; es lohnt sich schon für einen Brief, eine Einladung, einen Stundenplan oder ein Referat! Deshalb will ich dir im Folgenden ein wenig über das bekannteste Programm dieser Art erzählen, es heißt Microsoft Word.

7.3.1.1 Aller Anfang ist leicht!

Wenn du Word gestartet hast, siehst du meist ein Bild, das so ähnlich aussieht wie das Beispiel für Fenster in Abbildung 68 auf Seite 136.

Solltest du statt des „Papiers“ nur eine graue Fläche sehen, wähle das Menü *Datei/Neu...* um eine neue Textdatei zu erzeugen.

Nun kannst du anfangen zu schreiben!

Wenn man ein umfangreicheres Dokument schreiben möchte, ist es sinnvoll, sich zuerst eine Gliederung zu überlegen und die Überschriften auch schon hinzuschreiben. Man kann sie ja später beliebig ändern oder verschieben. Man kann auch Stichworte unter den Überschriften notieren, damit man später weiß, was man in einem bestimmten Abschnitt ausdrücken wollte. Auch die Stichworte lassen sich ja leicht wieder löschen, wenn man sie abgearbeitet hat.

Unter technischen Gesichtspunkten kann man ansonsten einfach drauf los schreiben – das sollte man sogar! Ein häufiger Anfängerfehler in der Textverarbeitung ist nämlich, dass man darauf achtet, wann die Zeile zuende ist, um dort mit der → Eingabetaste [↵] einen Zeilenumbruch, also eine neue Zeile, zu erzwingen. Das sollte man aber unbedingt dem Computer überlassen! Der kann das viel besser! Die Eingabetaste ist *nur* dazu da, neue *Absätze* zu erzeugen.

Einen Absatz braucht man, wenn man nach einigen Sätzen einen neuen Gedanken beginnt. Dann fängt man wieder mit einer neuen Zeile an, auch wenn die vorangegangene noch gar nicht zu Ende war. Man kann auch einstellen, dass vor oder nach einem Absatz immer noch ein kleiner Abstand gehalten wird, wie in diesem Buch der Fall.

Was ist nun so schlimm daran, wenn man aus jeder Zeile einen eigenen Absatz macht? Nun, erstens ist es unnötig Arbeit. Zweitens aber passiert etwas ganz Schreckliches, wenn man in einem solchen Text eine Änderung vornimmt. Das unten stehende Beispiel soll dir das verdeutlichen.

Ursprünglicher Text, an jedem Zeilenende eine Absatzmarkierung:

Bei diesem Text wird jede Zeile mit einer Absatzmarke beendet. Das führt dazu, dass bei einer Änderung des Textes Absätze an Stellen entstehen, wo sie gar nicht hingehören! Die automatische Silbentrennung funktioniert dabei auch nicht, weil die Worte wegen des Absatzzeichens nicht erkannt werden können.

Geänderter Text (**fett gedruckt**):

Bei diesem Text wird jede Zeile mit einer Absatzmarke beendet. Das führt dazu, dass bei einer **solchen** Änderung des Textes Absätze an Stellen entstehen, wo sie gar nicht hingehören! Die automatische Silbentrennung funktioniert dabei auch nicht, weil die Worte wegen des **falschen** Absatzzeichens nicht erkannt werden können.

Das gemeine an diesem Fehler ist, dass es bei einem längeren Text unheimlich viel Arbeit ist, alle falschen Absatzzeichen wieder zu löschen! Also: Für Zeilenumbrüche ist der Computer zuständig! Wir tippen einfach drauf los!

Es gibt noch einen weiteren typischen Anfängerfehler, der uns am Lostippen hindert: Das Einstellen von Rändern oder Textinzügen mit Hilfe von Leerzeichen. Manchmal möchte man, dass ein Absatz links mehr Rand hat, oder dass – zum Beispiel bei einer Aufzählung – die erste Zeile etwas nach links herausragt.

Wenn man nicht weiß, wie man das macht, behilft man sich mit Leerzeichen, und das wird dann wieder sehr unschön, wenn man den Text ändern muss!

Ein Beispiel:

falsch	richtig
<p>1. Diese Aufzählung soll so aussehen, dass die Ziffern immer links 'rausstehen.</p> <p>2. Dazu habe ich an den gelb markierten Stellen Leerzeichen eingefügt.</p> <p>3. Das klappt zu allem Überfluss nur, wenn ich an jedem Zeilenende auch einen Zeilenwechsel eingebe</p>	<p>1. Am einfachsten macht man so etwas sowieso mit der Nummerierungsfunktion.</p> <p>2. Aber auch die verwendet die Absatzformatierung „Einzug“ und „hängender Einzug“, um das richtig hinzukriegen.</p> <p>3. Man kann aber schreiben, ohne sich um Zeilenwechsel oder Trennung zu kümmern.</p>
<p>1. Diese Aufzählung diesmal mit eingefügtem Text soll so aussehen, dass nur die Ziffern immer links 'rausstehen.</p> <p>2. Dazu habe ich an den hier immer noch gelb markierten Stellen Leerzeichen eingefügt.</p> <p>3. Das klappt zu allem Überfluss jetzt gar nicht mehr nur, wenn ich an jedem Zeilenende auch einen Zeilenwechsel eingebe</p>	<p>1. Am einfachsten macht man so etwas sowieso mit der genialen Nummerierungsfunktion.</p> <p>2. Aber auch die verwendet die Absatzformatierung „Einzug“ und „hängender Einzug“, um das wirklich und wahrhaftig richtig hinzukriegen.</p> <p>3. Man kann aber schreiben, ohne sich um Zeilenwechsel oder Trennung zu kümmern. Es kommt immer automatisch richtig hin!</p>

Tabelle 10: Einzüge mit Leerzeichen

Tabelle 10 zeigt dir, was passiert, wenn man einen Text ändert, bei dem die Einzüge (Ränder) mit Leerzeichen eingestellt wurden. Im zweiten Teil der Tabelle ist der neue Text fett gedruckt.

Wie man das macht? Das lernst du im Kapitel 7.3.1.5.

Jetzt merke dir nur:

*Keine Zeilenwechsel eingeben, außer am Ende eines Absatzes!
 Nie Leerzeichen an den Beginn einer Zeile setzen!
 Immer einfach drauf los schreiben!*

Und was machen wir mit Satzzeichen, wie Kommas, Punkten, Klammern, Fragezeichen etc.? Kommen Leerzeichen davor oder dahinter – oder davor und

dahinter? Am besten überlegen wir uns, wo die Satzzeichen jeweils stehen sollen, wenn eine neue Zeile beginnt: Immer am Ende der Zeile! Keine Zeile soll mit einem Komma oder Punkt anfangen. Word erzeugt neue Zeilen immer zwischen zwei Wörtern, also *nach* einem Leerzeichen. Wenn das Komma also am Ende der Zeile stehen bleiben soll, muss *dahinter* ein Leerzeichen stehen, *davor* darf *kein* Leerzeichen stehen. Genauso verhält es sich mit den anderen Satzzeichen – außer: Öffnende Klammern ([{, denn die sollen ja im Zweifelsfall auch am Beginn der neuen Zeile stehen. Deshalb macht man davor ein Leerzeichen, aber keins dahinter.

Alles halb so wild. Schau dir einfach an, wie es in diesem Buch gemacht ist. Für den Anfang ist es vielleicht auch noch gar nicht sooo wichtig, sich um solche Details zu kümmern.

7.3.1.2 Rangordnung der Gestaltung

Wenn wir mit dem Drauflosschreiben fertig sind, wenn also der eigentliche Text eingetippt ist, dann können wir uns Gedanken darüber machen, wie dieser Text erscheinen soll:

Wie groß sollen die Seitenränder sein?

Welche Schriftart und –größe?

Welchen Abstand sollen die Zeilen und Zeichen voneinander haben?

Wo sollen Tabellen oder Bilder platziert werden?

Alle diese Fragen können wir uns jetzt stellen!

Um zu verstehen, wie das geht, müssen wir als erstes wissen, *was* man alles gestalten kann! Nehmen wir als Beispiel den Buchstaben „a“:

a a a	Verschiedene <i>Schriftarten</i> : Arial, Times, Courier
a a a	Verschiedene <i>Schriftgrößen</i> : 11, 14, 16 Punkte
a a a	Verschiedene <i>Farben</i>
a a a	Verschiedene <i>Textauszeichnungen</i> : Fett, kursiv, unterstrichen, ...
a a a	... durchgestrichen, doppelt durchgestrichen, schattiert, ...
a a a	... Outline, Relief, Gravur, ...
A ^a a	... Kapitälchen, hochgestellt, tiefgestellt

Tabelle 11: Zeichenformatierungen

Tabelle 11 zeigt dir verschiedene Möglichkeiten, wie du in Word einen Buchstaben darstellen kannst – und diese Tabelle ist noch nicht einmal vollständig!

Unsere Ausgangsfrage lautete aber (noch) nicht, *wie* man gestalten kann, sondern *was*! Wir haben in der Tabelle den Buchstaben „a“ gestaltet. Natürlich hätten wir auch „b“, „x“, „1“ oder jedes beliebige andere Zeichen nehmen können. Wir haben also ein *Zeichen* gestaltet! Alle gezeigten Darstellungsarten sind Eigenschaften eines Zeichens.

*Man nennt solche verschiedenen Darstellungsarten auch **Format** oder **Formatierung**. Im Beispiel handelt es sich um Zeichen-Formatierungen.*

Das Zeichen ist die kleinste Einheit in einem Text. Dann kommen Wörter und dann Sätze. Kann man sich Eigenschaften ausdenken, die sich auf Wörter oder Sätze beziehen?

Mir fällt nur eine ein: Die Sprache! Auch die kann man in Word einstellen, denn das ist für die Rechtschreib- und Grammatikprüfung wichtig. Mit der Formatierung (also dem Aussehen des Textes) hat die Sprache allerdings nichts zu tun.

Die nächstgrößere Einheit, der man Format-Eigenschaften zuweisen kann, ist deshalb der Absatz:

Ausrichtung eines Absatzes:	Einzüge und Abstände:
Dieser Absatz ist linksbündig ausgerichtet, das bedeutet, dass die linke Kante durchgängig ist, während an der rechten Seite die Zeilen enden, wie es gerade auskommt.	In diesem Absatz ist die erste Zeile anders formatiert als der Rest des Absatzes, weil sie ein Stück eingerückt ist.
Umgekehrt werden bei einem rechtsbündig ausgerichteten Absatz die Zeilen so verschoben, dass rechts eine glatte Kante entsteht. Links- und rechtsbündige Ausrichtung nennt man auch Flattersatz, im Gegensatz zum Blocksatz.	Man kann die erste Zeile auch nach außen überstehen lassen oder die rechten und linken Kanten des Absatzes verschieben. Man nennt das rechter und linker <i>Einzug</i> .
Beim Blocksatz richtet Word die Worte so aus, dass beide Kanten des Absatzes ausgerichtet sind. Die unterschiedlichen Längen der Zeilen werden durch verschiedenen Abstände zwischen den Worten ausgeglichen.	Diese beiden Absätze haben verschiedene Zeilenabstände! Dieser hier hat doppelte Zeilenhöhe. In diesem Absatz ist der Zeilenabstand einfache Zeilenhöhe.
Die vierte Möglichkeit einen Absatz auszurichten ist „zentriert“, also sozusagen Flattersatz an beiden Enden.	

Tabelle 12: Absatzformat Ausrichtungen, Einzüge und Abstände

Tabelle 12 zeigt dir verschiedene Beispiele, welche Eigenschaften (Formatierungen) man Absätzen zuweisen kann. Die Tabelle zeigt Ausrichtungen, Einzüge und Abstände, es gibt aber auch noch Absatzformatierungen, die sich nicht so ohne weiteres zeigen lassen: Zum Beispiel kann man einstellen, dass innerhalb eines Absatzes kein Seitenwechsel erfolgen darf, oder dass vor dem Absatz auf jeden Fall eine neue Seite beginnen soll. Du erkennst, dass das wirklich Eigenschaften sind, die man nicht einem Zeichen zuweisen kann.

Wenn du die Anzeige von Sonderzeichen eingeschaltet hast, kannst du das Zeichen für einen Absatzwechsel überall dort sehen, wo du die → Eingabetaste [↵] gedrückt hast. Es sieht so aus: ¶ Du kannst dir vorstellen, dass alle Formatierungen eines Absatzes in diesem Zeichen gespeichert werden. Wenn du es löscht, gehen auch die Formatierungen verloren; es gelten dann diejenigen des Absatzes, zu denen der Text dann gehört.

Gibt es noch größere Einheiten, denen man Eigenschaften zuerkennen kann? Selbstverständlich: Die Seite! Man muss dem Textverarbeitungsprogramm doch sagen können wie groß das Papier ist, auf dem man drucken möchte, und ob der Text darauf quer oder längs erscheinen soll! Auch wie viel Rand man oben, unten, links und rechts lassen will, ist wichtig – nicht nur weil Lehrer immer viel Wert darauf legen, damit sie ausreichend Platz für ihre Korrekturen haben. Wenn man ein Dokument beispielsweise zusammenheften möchte oder sogar ein Buch daraus werden soll, ist es ganz gut, wenn man am linken Rand etwas mehr Platz lässt.

In diesem Buch findest du in der Kopfzeile jeder Seite die Überschrift des Hauptkapitels, zu dem sie gehört. Ich habe mir nicht die Mühe gemacht, das von Hand auf jeder Seite einzutragen, sondern das ist eine Kopfzeile. Kopfzeilen werden innerhalb eines *Abschnitts* auf jeder Seite gleich gedruckt. Damit haben wir schon das nächst höhere Element, dem wir Formate zuweisen können.

Neben den Kopf- und Fußzeilen ist ein Merkmal von Abschnitten die Anzahl

Spalten, in denen der Text dargestellt wird. Man kann mit Word nämlich auch wie in

einer Zeitung schreiben!

*Innerhalb eines **Abschnitts** haben alle Seiten das gleiche Format. Insofern gehören das Seiten- und Abschnittsformat zusammen.*

Es gibt aber auch fortlaufende Abschnittswechsel, die ohne Seitenwechsel auskommen. Damit kann man zum Beispiel die Spaltenzahl verändern.



Über dem Abschnitt kommt nur noch das Dokument selbst. Auch ihm sind Eigenschaften zugeordnet, die aber nichts mit der Gestaltung des Textes zu tun haben; zum Beispiel der Dateiname, wer es geschrieben hat usw.

7.3.1.3 Die Funktionen von Word

Nachdem wir nun die Hierarchie in einem Dokument verstanden haben – eigentlich muss man dabei gar nichts verstehen, sondern man muss sie nur zur Kenntnis nehmen wie einen Busfahrplan – sollten wir einfach mal alle Funktionen durchgehen, die uns das Textverarbeitungsprogramm Word so bietet. Darüber allein schreiben andere Leute Bücher, die noch viel dicker sind als dieses hier; deshalb fällt diese Zusammenfassung in Tabelle 13 recht kurz aus. Einige Befehle, die du erst mal nicht brauchst, sind gar nicht enthalten. Dafür folgen anschließend noch ein paar Erklärungen, die dir helfen, die Arbeitsweise von Word zu verstehen.







Der Sinn der Tabelle besteht vor allem darin, dass du siehst, was es alles gibt, dass du ein wenig damit herumspielst, ausprobierst und vielleicht in der Hilfe ([F1]-Taste!) darüber nachliest, und dass du dich dann daran erinnerst, wenn du es wirklich mal brauchst. (Sich erst dann mit einer Funktion zu beschäftigen, wenn man sie braucht, genügt nicht, denn erstens fällt einem gar nicht ein, dass es sie gibt, und zweitens hat man dann andere Dinge im Kopf und möchte nur möglichst schnell das Problem lösen, das man gerade hat.)

Die Befehle sind in der Reihenfolge ihres Auftauchens im Menü aufgeführt.






Menü	Tastenkürzel ⁷	Symbol	Beschreibung
<u>D</u>atei	[Alt] [D] [F10] [D]		Beinhaltet Befehle, die sich auf das gesamte Dokument oder die Datei beziehen.
<u>N</u> eu...	[Strg]+[N] [Alt] [D] [N] [F10] [D] [N]		Damit erzeugst du ein neues Dokument. Wie bei jedem Menüpunkt, dem drei Punkte folgen, öffnet sich ein weiteres Fenster. Darin kannst du eine → <i>Vorlage</i> für dein neues Dokument auswählen. Wenn du die erste Tastenkombination oder das Symbol verwendest, wird die Standardvorlage <i>Normal.dot</i> verwendet.
Öffnen...	[Strg]+[O] [Alt] [D] [F] [F10] [D] [F]		Damit kannst du eine Datei, die du früher mal gespeichert hast, öffnen. Natürlich sollte es eine Datei sein, die Word auch bearbeiten kann. Wenn du den Menüpunkt wählst, öffnet sich ein Fenster, in dem du die Datei auswählen kannst.

⁷ Beachte: Wenn ein „+“ zwischen den Tasten steht, musst du sie gleichzeitig drücken, sonst nacheinander.

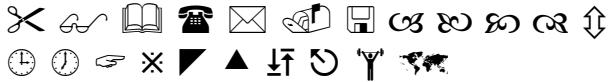
Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Schließen	[Alt] [D] [C] [F10] [D] [C]		Damit schließt du das Dokument, das im gerade aktiven Fenster angezeigt wird. Falls der aktuelle Bearbeitungsstand noch nicht gespeichert ist, wirst du gefragt, ob du das tun möchtest. Wenn du das nicht tust (auf „Nein“ klickst), gehen deine Änderungen an dem Dokument verloren, die du seit dem letzten Speichern gemacht hast!
Speichern	[Strg]+[S] [Alt] [D] [S] [F10] [D] [S]		Damit speicherst du deine Arbeit unter dem Dateinamen, unter dem du sie auch geöffnet hast. Falls du ein neues Dokument das erste Mal speicherst, geht automatisch ein Fenster auf, in dem du Speicherort und Dateinamen eingeben musst.
Speichern unter...	[Alt] [D] [U] [F10] [D] [U]		Mit diesem Befehl kannst du eine Datei unter einem anderen Namen speichern. Die Datei mit dem alten Namen bleibt so bestehen, wie sie zuletzt gespeichert wurde; die neue Datei enthält den aktuellen Stand. In einem Fenster kannst du wieder Speicherort und neuen Dateinamen angeben.
Seite einrichten...	[Alt] [D] [I] [F10] [D] [I]		Hiermit kannst du einstellen, wie groß das Papier ist, auf dem du schreibst, ob du es hochkant oder quer benutzt, wie groß die Seitenränder sind und einiges mehr.
Seitenansicht	[Alt] [D] [H] [F10] [D] [H]		Mit diesem Befehl kannst du dir das Dokument so am Bildschirm anzeigen lassen, wie es später auch gedruckt werden wird. Das ist ganz hilfreich in den wenigen Fällen, wo → WYSIWYG nicht ganz funktioniert.
Drucken...	[Strg]+[P] [Alt] [D] [D] [F10] [D] [D]		Dieser Befehl ist selbsterklärend: Damit kann man das Dokument drucken. Benutzt du das Menü oder eine Tastenkombination, öffnet sich ein Fenster, in dem du eine Reihe von Einstellungen für den Druck vornehmen kannst („...“ hinter dem Menüeintrag!). Benutzt du die Schaltfläche, wird sofort auf dem eingestellten Standarddrucker gedruckt.
1, 2, 3, 4	[Alt] [D] [1] [F10] [D] [1]		Mit diesen Befehlen kannst du die zuletzt in Word bearbeiteten Dateien öffnen, ohne lange suchen zu müssen, wo die stehen; Word merkt sich das für dich. Unter Extras/Optionen Registerkarte Allgemein kannst du die Anzahl Dateien, die Word sich merkt verstellen (max. 9).
Beenden	[Alt]+[F4] [Alt] [D] [B] [F10] [D] [B]		Es werden alle Dateien geschlossen und Word wird beendet. (Daten und Programm werden aus dem Arbeitsspeicher entfernt.) Sind Dateien noch nicht gespeichert, gibt es eine Abfrage, ob das noch geschehen soll.

Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Bearbeiten	[Alt] [B] [F10] [B]		Befehle für die Zwischenablage, Markierungen und Suchfunktionen.
<u>R</u> ückgängig	[Strg] + [Z] [Alt] [B] [R] [F10] [B] [R]		Dies ist ein sehr praktischer Befehl: Damit kannst du deine letzte Aktion – oder auch mehrere Aktionen – in Word rückgängig machen und den alten Zustand wieder herstellen. Es ist also gar nicht so schlimm, wenn man mal etwas falsch gemacht hat; meistens kann man den Schaden mit diesem Befehl wieder beheben. Klickst du auf den kleinen Pfeil rechts neben dem Symbol, klappt eine Liste deiner letzten Aktionen auf und du kannst wählen, bis wohin du alles wieder rückgängig machen willst.
<u>W</u> iederherstellen			Mit dem Symbol daneben kannst du rückgängig gemachte Aktionen doch wieder ausführen. Auch dabei erzeugt der kleine Pfeil rechts eine Liste von Aktionen, die wieder hergestellt werden kann.
<u>W</u> iederholen	[Strg]+[Y] [Alt] [B] [W] [F10] [B] [W]	 ⁸	Wenn du an mehreren Stellen eines Dokuments das selbe tun willst, dann kannst du mit diesem Befehl deine jeweils letzte Aktion an der neuen Position erneut durchführen.
<u>A</u> usschneiden	[Strg]+[X] [Alt] [B] [U] [F10] [B] [U]		
<u>K</u> opieren	[Strg]+[C] [Alt] [B] [K] [F10] [B] [K]		Diese Befehle beziehen sich auf die Zwischenablage wie in Kapitel 6.2.3 beschrieben.
<u>E</u> infügen	[Strg]+[V] [Alt] [B] [I] [F10] [B] [I]		
Markierung <u>l</u> öschen	[Entf] [Alt] [B] [C] [F10] [B] [C]		Dieser Befehl löscht den markierten Teil des Textes.
<u>S</u> uchen...	[Strg]+[F] [Alt] [B] [S] [F10] [B] [S]		Damit kann man im Dokument bestimmte Textstellen suchen. Es ist auch möglich, Text mit bestimmten Formatierungen zu suchen.

⁸ Diese Schaltfläche ist nicht standardmäßig in der Symbolleiste enthalten. Wenn man sie haben möchte, muss man sie mit Hilfe des Befehls Extras/Anpassen einfügen.



Menü	Tastenkürzel ⁷	Symbol	Beschreibung
<u>E</u> rsetzen...	[Strg]+[H] [Alt] [B] [E] [F10] [B] [E]		Dieser Befehl ersetzt jedes Vorkommen eines bestimmten Textes durch einen anderen, entweder vollautomatisch oder mit Bestätigung für jeden Einzelfall. Wenn du also eine Geschichte geschrieben hast, in der der Held <i>Franz</i> heißt, und du fändest es besser, dass er <i>Lutz</i> heißt, kannst du das mit diesem Befehl innerhalb von Sekunden ändern!
<u>A</u>nsicht	[Alt] [A] [F10] [A]		Alle Befehle, die etwas mit der Darstellung des Dokuments am Bildschirm zu tun haben.
<u>N</u> ormal	[Alt] [A] [N] [F10] [A] [N]	 ⁹	Stellt das Dokument bei der Bearbeitung ohne → WYSIWYG dar. Empfehlenswert für langsame Rechner.
<u>W</u> eblayout	[Alt] [A] [W] [F10] [A] [W]		Stellt das Dokument bei der Bearbeitung so dar, wie es auf einer Internetseite aussehen würde.
Seitenlay <u>o</u> ut	[Alt] [A] [O] [F10] [A] [O]		Stellt das Dokument bei der Bearbeitung so dar, wie es auf dem Papier entsprechend der Seiteneinstellungen aussieht. Annähernd → WYSIWYG
<u>G</u> liederung	[Alt] [A] [G] [F10] [A] [G]		Stellt nur die Überschriften bis zu einer einstellbaren Gliederungsebene dar. Sehr praktisch, wenn man Kapitel umgruppieren will; man kann sie im Ganzen verschieben.
<u>S</u> ymbolleisten ►	[Alt] [A] [S] [F10] [A] [S]		Symbolleisten fassen Schaltflächen für Befehle thematisch zusammen. Mit diesem Befehl kann man auswählen, zu welchen Themen man Symbolleisten anzeigen lassen möchte.
<u>L</u> ineal	[Alt] [A] [L] [F10] [A] [L]		Damit kann man die über und links neben dem Text angezeigten Lineale ein- und ausblenden.
<u>K</u> opf- und Fußzeile	[Alt] [A] [K] [F10] [A] [K]		Kopf- und Fußzeilen sind Bereiche am oberen bzw. unteren Seitenrand, die auf allen Seiten eines Abschnitts identisch sind. Mit diesem Befehl kann man diesen Bereich bearbeiten.
Ganzer <u>B</u> ildschirm	[Alt] [A] [B] [F10] [A] [B]		Mit diesem Befehl kannst du sämtliche Symbol- und Menüleisten und alle anderen Fensterelemente ausblenden, um mehr Text auf dem Bildschirm zu sehen. Mit der Taste [Esc] erhältst du die normale Ansicht zurück.

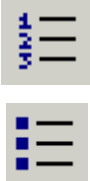



⁹ Die vier Schaltflächen für die verschiedenen Ansichten befinden sich im Fenster unten links neben dem Rollbalken


Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Zoom...	[Alt] [A] [Z] [F10] [A] [Z]	151%	Hiermit stellst du ein, wie groß dein Dokument <i>am Bildschirm</i> dargestellt wird. Diese Einstellung hat nichts damit zu tun, wie groß nachher der Ausdruck ist!
Einfügen	[Alt] [E] [F10] [E]		Befehle für das Einfügen besonderer Elemente in das Dokument.
Manueller Wechsel...	[Alt] [E] [W] [F10] [E] [W]		Hiermit kannst du Seiten- oder Abschnittswechsel einfügen.
Seitenzahlen...	[Alt] [E] [S] [F10] [E] [S]		Dieser Befehl fügt die Seitenzahl in die Kopf- oder Fußzeile ein.
Datum und Uhrzeit...	[Alt] [E] [U] [F10] [E] [U]		An der Einfügemarke wird das aktuelle Datum und/oder die Uhrzeit eingefügt. Das Format kannst du wählen. Du kannst es sogar für jeden Druck automatisch aktualisieren lassen.
Feld...	[Alt] [E] [F] [F10] [E] [F]		Felder können Texte enthalten, die Word automatisch berechnet. Das aktualisierte Datum ist nur ein Beispiel. Du kannst auch alle möglichen Daten des Dokuments in Feldern anzeigen lassen. Automatisch erstellte Inhaltsverzeichnisse sind auch Felder.
Symbol...	[Alt] [E] [O] [F10] [E] [O]		Damit lassen sich Zeichen einfügen, die du über die Tastatur nicht erreichen kannst. Einige Beispiele: 
Fußnote...	[Alt] [E] [N] [F10] [E] [N]		Fußnoten sind Bemerkungen am unteren Rand der Seite oder am Ende des Dokuments. Mit dieser Funktion werden sie automatisch richtig durchnummeriert und immer auf der richtigen Seite angezeigt! ¹⁰
Beschriftung...	[Alt] [E] [E] [F10] [E] [E]		Wenn du dir die Abbildungen und Tabellen in diesem Buch betrachtest, siehst du, dass sie alle mit einer durchnummerierten Unterschrift versehen sind. Diese Unterschriften wurden mit dem Befehl Beschriftung eingefügt. Auf diese Weise kann man einen → Querverweis in den Text einfügen, der immer automatisch aktualisiert wird, auch wenn man später noch Beschriftungen ergänzt. Auch die Funktion zum Erzeugen von Verzeichnissen greift auf Beschriftungen zurück.

¹⁰ Dies ist ein Beispiel für eine Fußnote: Im Text steht die hochgestellte Nummer. Die Anmerkung dazu erscheint automatisch am unteren Rand der richtigen Seite.

So etwas verwendet man, um zusätzliche Informationen zu geben, die nicht unmittelbar zum Text gehören.

Menü	Tastenkürzel ⁷	Symbol	Beschreibung
<u>Q</u> uerverweis...	[Alt] [E] [Q] [F10] [E] [Q]		Wenn man sich im Text auf eine Abbildung, ein Kapitel oder eine Tabelle mit Hilfe der laufenden Nummer bezieht, verwendet man diesen Befehl. Er stellt sicher, dass diese Verweise automatisch aktualisiert werden, wenn sich die Nummer des Bildes etc. geändert hat!
<u>I</u> ndex und Verzeichnisse...	[Alt] [E] [I] [F10] [E] [I]		Hiermit lassen sich Inhaltsverzeichnisse automatisch erstellen, wenn man für die Überschriften die richtigen → Formatvorlagen verwendet hat. Auch Abbildungsverzeichnisse sind möglich, wenn man → Beschriftungen benutzt hat. Der Index in Kapitel 12 wurde auch mit dieser Funktion erstellt.
<u>G</u> rafik ►	[Alt] [E] [G] [F10] [E] [G]		Damit kannst du verschiedene Arten von Bildern einfügen; entweder aus bestehenden Dateien oder du holst es direkt vom → Scanner. WordArt ist ein kleines Programm mit dem man Text gestalten kann: 
Text <u>f</u> eld	[Alt] [E] [X] [F10] [E] [X]		Wenn du diesen Befehl wählst, wird die Maus zu einem Fadenkreuz und du kannst eine Box aufziehen, die du wie eine Grafik frei verschieben, mit Rahmen und Hintergrund versehen, und außerdem mit Text füllen kannst.  Die Schaltfläche siehst du nur, wenn du die Symbolleiste „Zeichnen“ eingeschaltet hast.
<u>D</u> atei...	[Alt] [E] [D] [F10] [E] [D]		Wenn du einen Text bereits in einer anderen Datei gespeichert hast und ihn jetzt wieder verwenden willst, kannst du den Inhalt einer Textdatei mit diesem Befehl einfügen.
<u>O</u> bjekt...	[Alt] [E] [B] [F10] [E] [B]		Mit diesem Befehl kannst du Daten einfügen, die du mit anderen Programmen erstellt hast.
<u>T</u> extmarke...	[Alt] [E] [T] [F10] [E] [T]		Damit markierst du eine beliebige Stelle im Text und gibst dieser Stelle einen Namen. So kannst du dich in einem → Querverweis darauf beziehen.
Format	[Alt] [T] [F10] [T]		Befehle, mit denen sich das Aussehen des Textes verändern lässt.
<u>Z</u> eichen...	[Alt] [T] [Z] [F10] [T] [Z]		Damit kannst du alle Zeichenformatierungen einstellen, von denen du schon einige Beispiele in Tabelle 11 gesehen hast.
<u>A</u> bsatz...	[Alt] [T] [A] [F10] [T] [A]		Die in Tabelle 12 vorgestellten Absatzformate lassen sich mit diesem Befehl einstellen.

Menü	Tastenkürzel ⁷	Symbol	Beschreibung
<u>N</u> ummerierung und Aufzählungszeichen...	[Alt] [T] [N] [F10] [T] [N]		Mit dieser Funktion kann man 1. Nummerierungen 2. Aufzählungspunkte erzeugen. Vorteile: <ul style="list-style-type: none"> • Verschiedene Gliederungsebenen möglich • Automatische Korrektur bei nachträglichem Einfügen So sieht das dann aus. Es sind natürlich auch andere Darstellungen möglich!
<u>R</u> ahmen und Schattierung...	[Alt] [T] [R] [F10] [T] [R]		Damit lassen sich Tabellenzellen, Absätze, → Textfelder usw. mit Umrandungen und Hintergründen versehen wie in diesem Beispiel.
<u>S</u> palten...	[Alt] [T] [P] [F10] [T] [P]		Damit kann man Schreiben wie in einer Zeitung. Die Spalteneinstellungen gelten jeweils für einen → Abschnitt.
<u>T</u> abstopp...	[Alt] [T] [P] [F10] [T] [P]		Mit Hilfe dieses Befehls lassen sich Tabstopps festlegen, die beim Betätigen der → Tabulatortaste angesprungen werden. Einfacher und schneller geht das aber mit Hilfe des → Lineals!
<u>I</u> nitial...	[Alt] [T] [I] [F10] [T] [I]		M it diesem Befehl kann man den ersten Buchstaben eines Absatzes ganz groß machen, so wie das in alten Büchern gemacht wurde.
<u>T</u> extrichtung...	[Alt] [T] [X] [F10] [T] [X]		Damit stellst du ein, in welcher Richtung der Text eines Textfeldes oder einer Tabellenzelle gedruckt werden soll. Die Schaltfläche erscheint nur in der Symbolleiste „Textfeld“.
<u>G</u> roß-/ <u>K</u> lein-schreibung...	[Alt] [T] [K] [F10] [T] [K]		Erlaubt das Umwandeln von Groß- in Kleinbuchstaben und umgekehrt mit verschiedenen Auswahlmöglichkeiten.
<u>F</u> ormatvorlagen...	[Alt] [T] [V] [F10] [T] [V]		Du kannst dem Aussehen eines Absatzes einen Namen geben und es speichern. Später kannst du dann anderen Absätzen das selbe Format geben. Ein solches gespeichertes Aussehen nennt man Formatvorlage. Es gibt eine Reihe von vordefinierten Vorlagen z.B. für Überschriften.
<u>O</u> bjekt...	[Alt] [T] [O] [F10] [T] [O]		Hiermit kannst du für ein → Objekt verschiedene Einstellungen vornehmen. Unter anderem, ob es vor oder hinter dem Text erscheint.

Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Extras	[Alt] [X] [F10] [X]		Eine Sammlung von Werkzeugen und Einstellmöglichkeiten für das Programm Word.
<u>R</u> echtschreibung und Grammatik...	[F7] [Alt] [X] [R] [F10] [X] [R]		Hiermit kannst du Word überprüfen lassen, ob du alles richtig geschrieben hast. Unter den → Optionen kannst du auch einstellen, dass dir falsche Wörter schon während der Eingabe markiert werden. Warum du dich auf diese Funktion nicht verlassen kannst, erklärt dir Kapitel 7.3.1.7.
<u>S</u> prache ►	[Alt] [X] [S] [F10] [X] [S]		Dieser Menüpunkt enthält drei weitere interessante Untermenüs (zu erkennen an dem kleinen Pfeil hinter dem Menüeintrag).
<u>S</u> prache bestimmen...	[Alt] [X] [S] [S] [F10] [X] [S] [S]		Damit sagst du Word, in welcher Sprache dein Text geschrieben ist, damit es für die Rechtschreibprüfung das richtige Wörterbuch benutzt.
<u>T</u> hesaurus...	[↑] + [F7] [Alt] [X] [S] [T] [F10] [X] [S] [T]		Dieses Ding, das sich anhört wie ein urzeitlicher Dinosaurier, ist eine tolle Sache! Sicher hat dir dein Deutschlehrer schon mal gesagt, man solle ein Wort nicht so oft hintereinander benutzen. Manchmal fällt einem aber kein anderes Wort mit der selben Bedeutung (das nennt man ein <i>Synonym</i>) ein. Da hilft dir der Thesaurus: Es ist ein Synonymwörterbuch. Markiere das Wort, das durch ein anderes mit der gleichen Bedeutung ersetzen willst und drücke [↑] + [F7].
<u>S</u> ilbentrennung...	[Alt] [X] [S] [I] [F10] [X] [S] [I]		Word kann auch Wörter am Zeilenende automatisch trennen! Mit diesem Befehl schaltest du das an und aus. Für einzelne Absätze kannst du das im Absatzformat verbieten.
<u>W</u> örter zählen...	[Alt] [X] [W] [F10] [X] [W]		Der Befehl macht was er sagt! Er zählt die Wörter in deinem Text. Bei Prüfungsaufgaben ist so etwas manchmal wichtig.
Auto <u>Z</u> usammenfassen...	[Alt] [X] [Z] [F10] [X] [Z]		Wenn man lange Texte schreibt, stellt man manchmal eine Zusammenfassung voran, um den Leuten, die nicht alles lesen können, eine Hilfe zu geben. Diese Funktion versucht, eine solche Zusammenfassung automatisch zu erzeugen.
Auto <u>K</u> orrektur...	[Alt] [X] [E] [F10] [X] [E]		Hier kannst du Abkürzungen einstellen, die während der Eingabe automatisch in die Langform umgewandelt werden sollen. Viele Sonderzeichen sind schon voreingestellt.

Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Änderungen verfolgen ►	[Alt] [X] [V] [F10] [X] [V] Statuszeile: ÄND		Diese Funktion ist sehr praktisch, wenn man mit mehreren an einem Text arbeitet: Man bekommt einen Text von jemand anderem und überarbeitet ihn. Zuvor doppelklickt man auf das Feld „ÄND“ in der Statuszeile. Dann werden alle Änderungen markiert und Word merkt sich, was von wem geändert wurde. Nachher kann man die Änderungen dann einzeln durchgehen und annehmen oder rückgängig machen.
Dokumente zusammenführen...	[Alt] [X] [T] [F10] [X] [T]		Wenn verschiedene Leute an einzelnen Teilen eines Dokuments arbeiten, kann man diese Teile hiermit verwalten und zusammenführen.
Dokument schützen...	[Alt] [X] [M] [F10] [X] [M]		Hiermit kannst du ein Dokument gegen unbeabsichtigte Veränderung schützen (nicht jedoch gegen unbefugte Weiterverwendung des Textes!).
Seriendruck...	[Alt] [X] [D] [F10] [X] [D]		Mit dieser Funktion kann man viele gleiche Texte erzeugen (z.B. Briefe oder Einladungen) die sich nur in ganz wenigen Punkten (z.B. Name und Adresse) unterscheiden; für Letztere braucht man nur eine Tabelle in Word oder Excel.
Makro ►	[Alt] [X] [K] [F10] [X] [K]		Ein Makro ist nichts anderes als ein → Programm! Nur dass man es hier ganz leicht schreiben kann: Man kann nämlich Aktionen, die man in Word ausführt, aufzeichnen und sie später wiederholen!
Anpassen...	[Alt] [X] [A] [F10] [X] [A]		Mit diesem Befehl kann man sich alle Menüs, Tastenkürzel und Symbolleisten nach eigenem Geschmack umgestalten. (Ich habe auch schon Word-Menüs in bayerischer Mundart gesehen!)
Optionen...	[Alt] [X] [O] [F10] [X] [O]		Hier kann man alle möglichen Einstellungen für Word vornehmen. Sie sind thematisch in Registerkarten einsortiert. Beispiel: Wie viele zuletzt geöffnete Dateien im Menü <i>Datei</i> erscheinen.
Tabelle	[Alt] [L] [F10] [L]		Alles was man mit einer Tabelle machen kann – zum Beispiel einen Stundenplan!.
Tabelle zeichnen	[Alt] [L] [C] [F10] [L] [C]		Damit kann man die Tabellenzellen einzeln von Hand zeichnen. Einfacher und schneller geht der Befehl → <i>Zellen einfügen / Tabelle</i> .
Zellen einfügen ►	[Alt] [L] [G] [F10] [L] [G]		Dieser Menüeintrag fasst alle Befehle zusammen, mit denen man Tabellenzellen einfügen kann: Einzeln, zeilen- oder spaltenweise oder eine ganze Tabelle.
Tabelle	[Alt] [L] [G] [T] [F10] [L] [G] [T]		Damit kannst du eine Tabelle erstellen, indem du nur angibst, wie viele Zeilen und Spalten sie haben soll (man kann das später problemlos ändern!).


Menü	Tastenkürzel ⁷	Symbol	Beschreibung
Zellen verbinden	[Alt] [L] [D] [F10] [L] [D]		Damit kannst du aus mehreren Zellen einer Tabelle (die du zuvor markieren musst) eine einzige machen.
Zellen teilen...	[Alt] [L] [T] [F10] [L] [T]		Damit kannst du aus einer Tabellenzelle mehrere machen.
Tabelle teilen	[Alt] [L] [I] [F10] [L] [I]		Damit werden aus einer Tabelle zwei gemacht und zwar an der Stelle, an der die Einfügemarke steht. Man kann dann zwischen den Tabellen wieder normal Text eingeben.
Überschriftenzeilen wiederholen	[Alt] [L] [I] [F10] [L] [I]		Wenn sich eine Tabelle über mehrere Seiten erstreckt (wie diese hier), kannst du die Überschriftenzeile(n) der Tabelle markieren und diese Funktion wählen. Diese Zeile(n) tauchen dann auf jeder Seite wieder neu am Kopf der Tabelle auf.
Umwandeln ►	[Alt] [L] [U] [F10] [L] [U]		Damit kannst du nachträglich eine Tabelle erzeugen, wenn ihr Inhalt (z.B. durch → Tabulatoren getrennt) schon da steht. Oder du kannst eine Tabelle entfernen, so dass nur noch ihr Inhalt übrig bleibt.
Sortieren...	[Alt] [L] [S] [F10] [L] [S]		Mit dieser praktischen Funktion kannst du deine Tabelle sortieren: Die Zeilen werden automatisch umsortiert, wenn du das Sortierkriterium eingegeben hast, also z.B. ob du eine Liste nach Namen oder nach Vornamen sortieren willst. Die Schaltfläche erscheint nur in der Symbolleiste „Tabelle und Rahmen“.
Formel...	[Alt] [L] [O] [F10] [L] [O]		Mit diesem Befehl kannst du Word einfache Berechnungen in deiner Tabelle durchführen lassen. Für größere Sachen empfiehlt sich aber → Excel.
Tabelleneigenschaften...	[Alt] [L] [E] [F10] [L] [E]		Du kannst eine Tabelle genauso frei platzieren wie eine Grafik und kannst den Text drumherum fließen lassen. Dieses und noch mehr stellst du mit diesem Befehl ein.
Fenster	[Alt] [F] [F10] [F]		Man kann mit Word mehrere Dokumente gleichzeitig laden, die in verschiedenen Fenstern dargestellt werden. Man kann auch ein Dokument in zwei Fenstern darstellen, wenn man an zwei verschiedenen Stellen arbeitet. Unter diesem Menü finden sich die Befehle dafür.
?	[F1] [Alt] [ß] [F10] [ß]		Hier findest du Hilfe zu Word.

Tabelle 13: Menübefehle von Word

Das waren jetzt fast zehn Seiten Tabelle mit Word-Funktionen – und es waren nicht einmal alle! Trotzdem weißt du immer noch nicht, *wie* man den einen oder anderen Befehl im einzelnen anwenden muss.

Ich kann dich nur ermuntern, nach und nach alles auszuprobieren, von dem du glaubst, dass du es gebrauchen kannst. Wenn dir mal etwas nicht klar ist, probiere systematisch, lies in der Hilfe nach oder frage jemanden.

Wenn ein Befehl in grau dargestellt ist, dann kann man ihn nicht auswählen. Das kann zum Beispiel daran liegen, dass du erst etwas markieren musst, auf das du den Befehl anwenden willst, oder daran, dass der Befehl sich auf das Ausgewählte nicht anwenden lässt; Tabellenbefehle zum Beispiel kann man eben nur auf Tabellen anwenden und nicht auf normalen Text.

7.3.1.4 Seitenansicht

Wenn du den Befehl Seitenansicht aufrufst, wird ein neues Fenster geöffnet, in dem das Dokument so dargestellt wird, wie es später auf dem Drucker erscheinen wird. Du musst es später mit der Schaltfläche „Schließen“ wieder zumachen, um wieder Text eingeben zu können.

Wenn du mit dem Aussehen zufrieden bist, kannst du mit Hilfe des Drucker-symbols, das es auch in diesem Fenster gibt, direkt drucken.

Eine interessante Möglichkeit gibt es in diesem Fenster, wegen der ich diesen Abschnitt eingefügt habe: Manchmal hat man einen Text geschrieben und muss am Ende für wenige Zeilen noch einmal eine neue Seite beginnen. Das sieht nicht sehr schön aus. Hier kommt die nebenstehende Schaltfläche ins Spiel: Wenn du sie anklickst, versucht Word, den selben Text auf einer Seite weniger unterzubringen. Dazu wird die Schrift ein klein wenig verkleinert.



7.3.1.5 Das Lineal, Einzüge und Tabulatoren

In Kapitel 7.3.1.1 habe ich versprochen, zu zeigen, wie man Einzüge einstellen kann:

Am oberen Rand des Fensters siehst du das sogenannte *Lineal*¹¹. Neben der Zentimeterskala sieht man links und rechts komische kleine Dreiecke, in Abbildung 83 rot eingekreist:

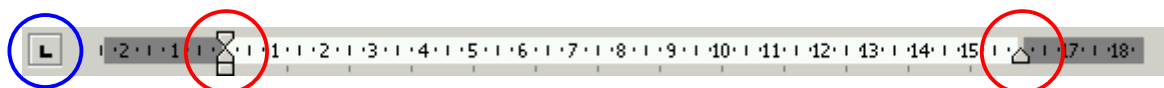


Abbildung 83: *Lineal mit Einzugsreglern*

¹¹ Falls das Lineal nicht angezeigt wird, kannst du das im Menü *Ansicht / Lineal* ändern.

Damit stellt man die Ränder eines Absatzes wie folgt ein:

Mit dem rechten Dreieck stellt man den rechten Rand ein. Man kann es mit der Maus hin und her schieben.



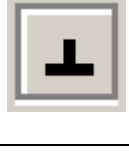
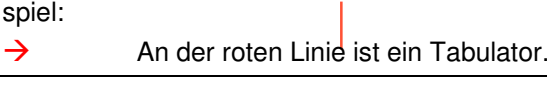

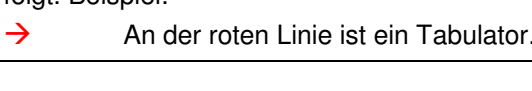
Bei den linken Reglern ist das obere Dreieck für die Randeinstellung der ersten Zeile zuständig und das untere für den Rand des restlichen Absatzes.

Wenn man den Regler an dem Rechteck unten anfasst, verstellt man beide Dreiecksregler gleichzeitig, ohne ihren Abstand zueinander zu verändern.

Diese Einstellungen gelten nur für den Absatz, in dem sich die Einfügemarke befindet! Will man die Ränder für alle Seiten eines Abschnitts ändern, geht das auch direkt im Lineal: Dazu muss man den Mauszeiger auf den Übergang zwischen weißem und dunkelgrauen Bereich des Lineals bewegen, bis ein kleiner Doppelpfeil erscheint. (Wenn die Absatzeinzugsregler im Weg sind, kann man sie kurz beiseite schieben.) Durch Klicken und Ziehen mit der Maus kann man dann den Seitenrand verändern. Wenn dir das zu schwierig ist, kannst du die selben Einstellungen natürlich auch über das Menü *Datei / Seite einrichten...* vornehmen.

Noch etwas kann man direkt im Lineal einstellen: Die Tabulatoren.

Das sind die Stellen, die man mit der Tabulatortaste „anspringen“ kann wie auf Seite 92 beschrieben. Zuerst wählt man dazu mit dem kleinen Feld links des Lineals (in Abbildung 83 blau eingekreist) die Art des Tabulators aus, indem man so oft darauf klickt, bis das gewünschte Symbol erscheint. Tabelle 14 zeigt dir, welche Arten es gibt. Dann klickt man im Lineal an die Stelle, wo die Einfügemarke hinspringen soll, wenn man die Tabulatortaste betätigt.

Symbol	Bedeutung
	<i>linksbündiger</i> Tabulator: Das Zeichen markiert die <i>linke</i> Kante des Textes, der nach dem Tabulatorzeichen folgt. Beispiel: → 
	<i>zentrierter</i> Tabulator: Das Zeichen Markiert die <i>Mitte</i> des Textes, der auf das Tabulatorzeichen folgt. Beispiel: → 
	<i>rechtsbündiger</i> Tabulator: Das Zeichen markiert die <i>rechte</i> Kante des Textes, der nach dem Tabulatorzeichen folgt. Beispiel: → 


Symbol	Bedeutung
	<p><i>Dezimaltabulator:</i></p> <p>Das Zeichen markiert die Stelle, an der das Dezimalkomma steht, wenn man Zahlen so untereinander schreiben will, dass sie sich leicht addieren lassen. Beispiel:</p> <p>→ 123,54</p> <p>→ 1,2345</p> <p>→ 4567,1</p> <p>An der roten Linie steht der Dezimaltabulator. Der Text, d.h. die Zahlen nach dem Tabulatorzeichen werden so lange rechtsbündig geschrieben, bis das Komma kommt, dann geht es linksbündig weiter.</p>

Tabelle 14: Verschiedene Arten von Tabulatoren

7.3.1.6 Neue Dateien von Vorlagen

Wenn du von Hand einen Brief schreibst, nimmst du dazu meist ein leeres, weißes Papier. Wenn du einer guten Freundin schreibst, ist es aber vielleicht ein besonders hübsches Briefpapier, auf das Verzierungen aufgedruckt sind. Manchmal füllst du vielleicht ein Formular aus, auf dem schon Text steht, und wo du nur in bestimmte Felder etwas einträgst. Wenn du einen formellen Brief schreibst, kommt oben dein Name und deine Anschrift, Ort und Datum – der so genannte Briefkopf – hin. Leute, die solche Briefe öfter schreiben, lassen sich Papier drucken, auf dem dieser Briefkopf schon drauf steht, damit sie ihn nicht jedes Mal neu schreiben müssen.

Du siehst: Auch wenn man etwas Neues schreiben will, kann es vorkommen, dass man das auf Papier tun möchte, auf dem schon etwas steht. In Word geht das auch so: Man kann beim Erstellen eines neuen Dokumentes auswählen, welches Papier, bzw. in Word heißt das welche *Vorlage* man verwenden möchte. Solche Vorlagen kann man sogar selbst erstellen!

Sobald du das Menü *Datei / Neu...* auswählst, bekommst du alle vorhandenen Vorlagen angezeigt. Wählst du eine aus, wird alles was darin steht in deine neue Datei hineinkopiert. (Wenn du auf das in Tabelle 13 gezeigte Symbol für diesen Befehl verwendest, geht das nicht; damit ruft man die Standardvorlage auf.)

Am besten verstehst du den Vorgang, wenn du selbst einmal eine Vorlage erstellst, zum Beispiel einen Briefkopf:

Klicke auf die Schaltfläche für die Standardvorlage und schreibe auf die leere Seite deinen Namen und deine Adresse. Wenn du willst, kannst du ja auch dein Lieblingsbild einfügen. Dann wähle das Menü *Datei / Speichern unter...* und wähle im untersten Eingabefeld den Dateityp *Dokumentvorlage (*.dot)* aus. Wenn du gut aufpasst, siehst du, dass sich darüber auch das Verzeichnis geändert hat: Word hat automatisch in das Standard-Verzeichnis für Vorlagen ge-

wechselt, das du übrigens unter *Extras / Optionen*, Registerkarte *Speicherort für Dateien* selbst einstellen kannst. Gib noch einen Dateinamen ein, zum Beispiel „Briefkopf“ und klicke auf die Schaltfläche *Speichern*.

Nun kannst du die Vorlagendatei schließen. Wenn du jetzt *Datei / Neu...* wählst, steht in der Liste der Vorlagen auch dein Briefkopf drin und du kannst einen neuen Brief schreiben, in dem alles schon drinsteht, was du in der Vorlage gespeichert hattest.

Du kannst für das Fenster, in dem du die Vorlage auswählst, sogar eigene Registerkarten erzeugen! Dazu brauchst du nur im Standard-Vorlagenverzeichnis ein Unterverzeichnis anzulegen.

Spaßes halber kannst du dir ja auch die Vorlagen mal ansehen, die mit Word schon mitgeliefert werden. Da ist sogar eine dabei, mit der man Kalender automatisch erstellen kann!

7.3.1.7 Rechtschreibprüfung

Ein paar Worte wie die Rechtschreibprüfung funktioniert, damit du weißt, dass du dich nicht blind darauf verlassen kannst:

Bei jedem Wort, das du schreibst, sieht Word in einem Wörterbuch – einer Datei, in der fast alle Wörter der deutschen Sprache drin stehen – nach, ob es das Wort darin gibt. Falls nicht, behauptet Word, das Wort sei falsch geschrieben, und schlägt dir im Kontextmenü sogar vor, wie es richtig geschrieben werden könnte.

Das erste Problem dabei ist, dass man im Deutschen viele Hauptworte (Nomen) zusammensetzen kann. Damit nicht auch alle zusammengesetzten Wörter im Wörterbuch gespeichert werden müssen, erkennt Word unter bestimmten Bedingungen auch zwei aneinander geschriebene Worte als richtig an. Willst du zum Beispiel „Staubsauger“ schreiben, und tippst statt dessen „Schraubsauger“ ein, wird Word den Fehler nicht finden, weil es „Schraub“ als Befehlsform von „schrauben“ kennt, und „Sauger“ sowieso!

Auch wenn du ganze Worte vergisst, kann Word das nicht merken, weil es ja den *Sinn* deiner Worte nicht kennt! „*Word ist ein olles Programm!*“ geht genauso als fehlerfrei durch wie „*Word ist ein tolles Programm!*“

Immerhin ist Word in der Lage, Wörter zu lernen: Wenn es ein Wort nicht kennt und als falsch geschrieben kennzeichnet, kannst du im Kontextmenü (oder im Fenster der Rechtschreibprüfung *Hinzufügen* wählen, und das Wort wird zum Wörterbuch hinzugefügt und künftig als richtig erkannt. (ACHTUNG! Achte darauf, dass das Wort, das du zum Wörterbuch hinzufügst, wirklich richtig geschrieben ist!)

Die Rechtschreibprüfung ist also eine gute Hilfe, um den einen oder anderen Fehler zu finden; sie kann aber keine Fehlerfreiheit garantieren. Dafür bist DU selbst verantwortlich!

Mit der Grammatikprüfung ist es ähnlich: Deutsche Grammatik ist so komplex, dass eine wirklich gute Überprüfung kaum möglich ist. Probiere selbst aus, ob und wie gut dir diese Funktion hilft.

7.3.2 Tabellenkalkulation

Tabellen hatten wir doch schon in Word!? Was soll das nun wieder? In Word hast du gesehen, dass Tabellen so wichtig sind, dass es für sie sogar einen eigenen Hauptmenüeintrag gibt. Tabellen sind sehr übersichtlich; man kann mit Ihnen einfach und anschaulich Zusammenhänge darstellen; viel besser als mit langatmigen Texten!

Es gibt aber noch eine zweite Worthälfte: *Kalkulation*. Das bedeutet übersetzt nichts anderes als *Berechnung*. Es geht also um Rechnen in Tabellen, und das geht so:

Nehmen wir an, ihr seid drei Kinder und jeder hat einen Sack voller Süßigkeiten. Mit einer Tabellenkalkulation könntet ihr auflisten, wer von welcher Süßigkeit wie viel hat, wer wovon am meisten und am wenigsten hat, und wie viel ihr insgesamt habt. Zugegeben ein etwas weit her geholtes Beispiel (welche Kinder haben schon so viele Süßigkeiten, dass sie das aufschreiben müssen?!), aber du kannst daran leicht verstehen, wie eine Tabellenkalkulation funktioniert.

Name	Schokolade	Kaugummi	Bonbons	Gesamt
Katharina	2	6	24	32
Alexander	1	1	3	5
Moritz	3	5	10	18
Gesamt	6	12	37	55

Tabelle 15: Beispiel für Tabellenkalkulation

Das Beispiel in Tabelle 15 ist noch nicht sehr berauschend, aber vielleicht siehst du das anders, wenn du erfährst, dass ich keine einzige der Summen in der Zeile bzw. Spalte „Gesamt“ selbst ausgerechnet habe! Das hat das Tabellenkalkulationsprogramm „Microsoft Excel“ für mich erledigt!

Und Excel kann noch viel kompliziertere Sachen rechnen als Summen!

Außerdem kann man damit solche Sachverhalte auch grafisch darstellen. Für das obige Beispiel könnte das so aussehen:

Die Süßigkeiten unserer Kinder

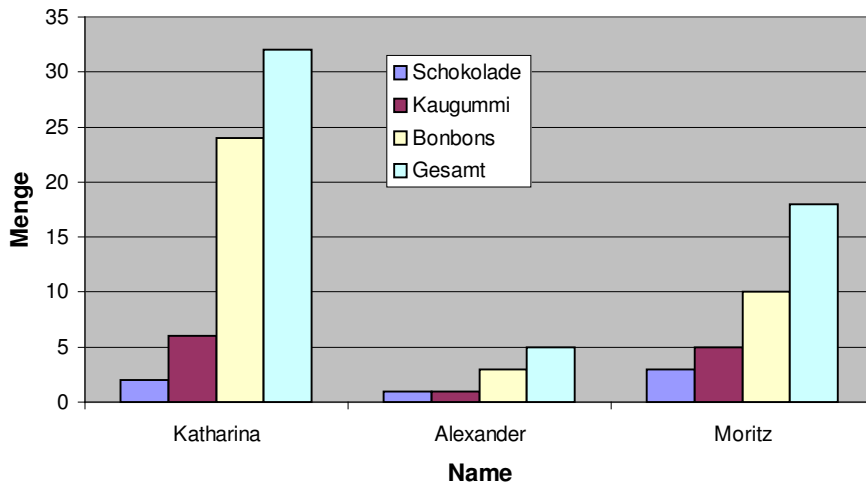


Abbildung 84: Säulendiagramm in Excel

Oder so:

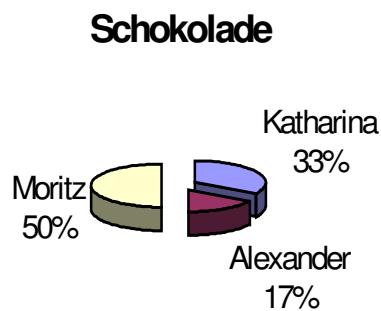


Abbildung 85: Tortendiagramm in Excel

Diese Bilder lassen sich in Excel mit wenigen Mausklicks zeichnen, wenn man die Daten, also die dargestellte Information, einmal eingegeben hat. Das haben wir mit Tabelle 15 bereits getan.

So viel als Appetithappen, was man alles machen kann. Nun dazu, wie das geht:

Wenn du das Programm *Excel* startest, siehst du schon eine Tabelle vor dir. Falls nicht, klicke auf die Schaltfläche für eine neue Datei (die sieht genauso aus wie in *Word*) oder wähle das Menü *Datei / Neu...!* Das sieht dann ungefähr so aus wie in Abbildung 86:

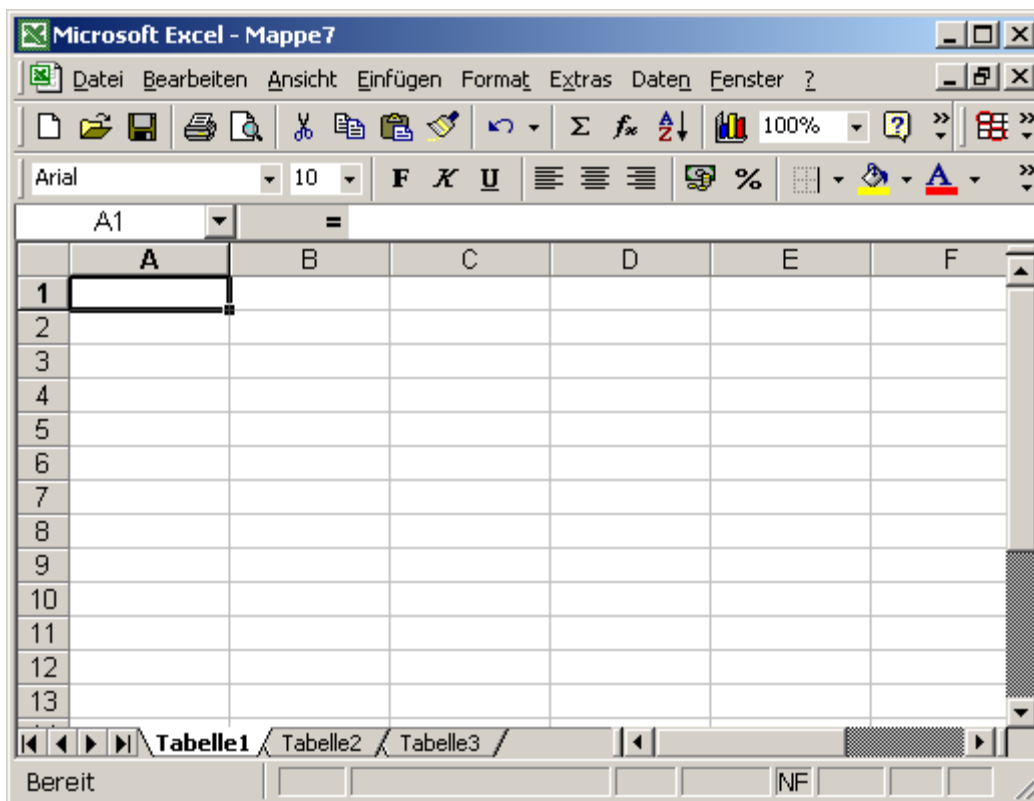


Abbildung 86: Fenster des Tabellenkalkulations-Programms Microsoft Excel

Die Schaltflächen in den Symbolleisten kommen uns schon ziemlich bekannt vor, und auch die ersten Menüpunkte *Datei*, *Bearbeiten*, *Ansicht*, *Fenster* und *?* kennen wir schon! Darauf werde ich in diesem Kapitel auch nicht mehr weiter eingehen, denn im Großen und Ganzen funktioniert das wirklich alles genauso wie in Word!

Neu hingegen ist die Tabelle, deren Spalten mit Buchstaben und deren Zeilen mit Zahlen beschriftet sind.

7.3.2.1 Excel rechnen lassen

In die Zellen dieser Tabelle kann man nun etwas eintragen, wie wir das in Tabelle 15 schon getan haben. Darin fehlen allerdings noch die Buchstaben, mit denen die Spalten und die Zahlen, mit denen die Zeilen angesprochen werden. Wenn wir die selbe Tabelle in Excel ansehen, sieht das Ganze aus wie in Abbildung 87; ich habe eine Zelle nachträglich gelb hinterlegt, weil ich dazu gleich etwas erklären will. Es ist die Zelle in Zeile 5 und Spalte E, man sagt auch die Zelle *E5*. Außerdem habe ich einen Bereich rot eingerahmt.

The screenshot shows the Microsoft Excel interface with a table containing the following data:

	A	B	C	D	E
1	Name	Schokolade	Kaugummi	Bonbons	Gesamt
2	Katharina	2	6	24	32
3	Alexander	1	1	3	5
4	Moritz	3	5	10	18
5	Gesamt	6	12	37	55
6					

The formula bar above the table shows the formula for cell E5: `=SUMME(B2:D4)`. The status bar at the bottom indicates 'Bereit' and 'NF'.

Abbildung 87: Excel-Tabelle

Über der Tabelle siehst du eine Zeile vor der ein dickes Gleichheitszeichen steht. Wir nennen sie mal *Bearbeitungszeile*, weil sie ja irgend einen Namen haben muss. Dort wird dir immer angezeigt, was in der Tabellenzelle steht, auf der sich gerade die Markierung (der dicke Rahmen um die Zelle) befindet. – Warum das denn? Sieht man das denn nicht in der Tabellenzelle selbst? Nicht immer!

Wenn du einfach irgendwelche Daten in eine Tabellenzelle einträgst – das sind im Beispiel die Namen, die Überschriften und die Zahlen, wie viel jeder wovon hat – dann steht in der Zelle das selbe wie in der Bearbeitungszeile.

In der Einführung habe ich dir aber schon erzählt, dass Excel die Summen in Spalte E und Zeile 5 für mich berechnet hat, das bedeutet, dass ich die Zahlen, die ich dort stehen sehe, nicht selbst eingegeben habe. Statt dessen habe ich eingegeben, was und wie Excel rechnen soll. Man nennt eine solche Berechnungsvorschrift eine *Formel*. In der Bearbeitungszeile siehst du immer die Formel und in der Tabelle steht das Ergebnis.

Eine Formel erkennt Excel immer daran, dass sie mit einem Gleichheitszeichen beginnt. Dahinter kommt dann, was Excel rechnen soll.

Falls du einmal ein Gleichheitszeichen in der Tabelle stehen haben möchtest, musst du den gesamten Zelleninhalt in Anführungszeichen setzen. Das ist für Excel das Signal, dass es sich um Text handelt.

In der Zelle E5 steht: `=SUMME(B2:D4)`.

Das Wort „SUMME“ ist der Name einer sogenannten *Funktion*. Das kannst du dir vorstellen wie ein kleines Programmteil, in dem steht, wie gerechnet werden soll. In den Klammern hinter dem Namen der Funktion stehen immer die Dinge, mit denen gerechnet werden soll; die heißen *Parameter* oder *Argument*.

Die Funktion SUMME berechnet die Summe der Zahlen, die in den Zellen stehen, die als Parameter angegeben sind. *B2:D4* bedeutet dabei das Rechteck, das von den Zellen B2 und D4 aufgespannt wird; das ist der Bereich, der in Abbildung 87 rot eingerahmt ist.

Du könntest in die Bearbeitungszeile auch folgendes hineinschreiben und bekommst das selbe Ergebnis:

$=B2+B3+B4+C2+C3+C4+D2+D3+D4$

Kannst du sagen, was in Zelle B5 stehen muss? Die Antwort ist:

$=SUMME(B2:B4)$

weil in B5 ja stehen soll, wie viel Schokolade alle Kinder zusammen haben.

In Zelle E2 steht $=SUMME(B2:D2)$ weil man dort sehen soll, wie viele Teile Süßigkeiten Katharina insgesamt hat.

Der Bereich über den eine Summe gebildet werden soll, muss nicht zusammenhängend sein; du kannst auch einzelne Zellen durch Semikolons getrennt angeben oder sogar gemischte Angaben machen wie

$=SUMME(A5; B3; C1:D4; E:E)$

“E:E” ist dabei die Bezeichnung für die gesamte Spalte E.

Noch eine kleine Bemerkung, die du vielleicht ausprobieren solltest, um richtig beeindruckt zu sein: Stell dir vor du hast eine Tabelle, in der mehrere Zellen mit Hilfe von Formeln berechnet werden. Darin sind womöglich sogar berechnete Zellen Argumente für andere berechnete Zellen. Nun änderst du die Daten, mit denen die Berechnung gemacht wurde! *Alle Berechnungen werden automatisch mit den neuen Daten neu durchgeführt und das geht so schnell, dass du ganz genau hinsehen musst, um es überhaupt mitzukriegen!*

Es ist übrigens eine gute Angewohnheit, Tabellen in Excel immer mit einer Überschriftenzeile zu versehen und die Daten und Formeln durchgehend ohne Leerzeilen einzutragen, wie wir das auch in Tabelle 15 getan haben. Excel kann dann viele Dinge automatisch erkennen und die Steuerung des Programms vereinfacht sich, weil Excel dann oft von alleine richtig auswählt, mit welchen Daten du gerade etwas machen möchtest.

7.3.2.2 Funktionen

Du möchtest etwas anderes berechnen als eine Summe? Eine Differenz, einen Mittelwert, ein Maximum? Kein Problem! Die Installation von Excel auf meinem


Rechner stellt etwa 350 Funktionen bereit – darunter sollte die zu finden sein, die du brauchst, zumal eine Formel ja auch mehr als eine Funktion enthalten kann. Und falls das noch nicht reicht, kann man neue Funktionen programmieren!

Hier nur ein paar Beispiele, was es noch so gibt:

Funktion	Ergebnis	Bedeutung
=BININDEZ(1001)	9	Wandelt eine binäre Zahl in eine Dezimalzahl um. Für jede andere Umrechnung zwischen binär, dezimal, hexadezimal und oktal (8er-System) gibt es auch eine Funktion!
=CODE("A")	65	Liefert den ASCII-Code eines Buchstabens, vergleiche Tabelle 6!
=FAKULTÄT(4)	24	Liefert die Fakultät einer Zahl, hier: $1 \times 2 \times 3 \times 4$
=GGT(24;36;132)	12	Berechnet den größten gemeinsamen Teiler der genannten Zahlen
=LÄNGE("Hallo Welt!")	11	Berechnet die Länge eines Textes, (hier inkl. Leerzeichen!)
=POTENZ(2;3)	8	Berechnet die Potenz, hier: $2^3 = 2 \times 2 \times 2$
=RANG(A2;A:A)		Berechnet welchen Platz der Inhalt der Zelle A2 in der Spalte A belegt, also der wievieltgrößte Wert es ist.
=REST(14;3)	2	Berechnet den Rest einer Division (Geteiltaufgabe), (hier: 14 geteilt durch 3 ergibt 4, Rest 2)
=RÖMISCH(26)	XXVI	Wandelt eine arabische in eine römische Zahl um

Tabelle 16: Excel-Funktionen

Wahrscheinlich ist es überflüssig zu erwähnen, dass Excel natürlich auch ganz normal plus, minus, mal und geteilt rechnen kann. Auch die trigonometrischen Funktionen Sinus, Cosinus, Tangens usw. sowie jede Menge Statistik-Funktionen sind selbstredend auch enthalten. Wenn du noch nicht weißt, was das ist, freue dich darauf, dass Excel dir für den gesamten Rest deiner Schul- und vielleicht Studienzeit mit diesen Dingen wird helfen können – allerdings nur beim Rechnen! Verstehen wirst du das alles immer noch selber müssen!

Wenn du eine Funktion verwenden willst und nicht genau weißt, wie sie heißt, oder welche Parameter man in welcher Reihenfolge angeben muss, dann klicke auf die Schaltfläche  oder wähle das Menü Einfügen / Funktion...

Es öffnet sich dann ein Fenster, in dem dir die Funktionen thematisch sortiert mit einer kurzen Hilfe angeboten werden. Hast du eine Funktion ausgewählt, kannst du mit Hilfe des Funktionsassistenten, der sich dann öffnet, alle notwendigen Parameter eingeben. Abbildung 88 zeigt dir den Funktionsassistenten am Beispiel der Funktion ZEICHEN(), die sozusagen die Umkehrung der in Tabelle

16 aufgeführten Funktion CODE() ist: Sie liefert dir zu einem ASCII-Code das zugehörige Zeichen.

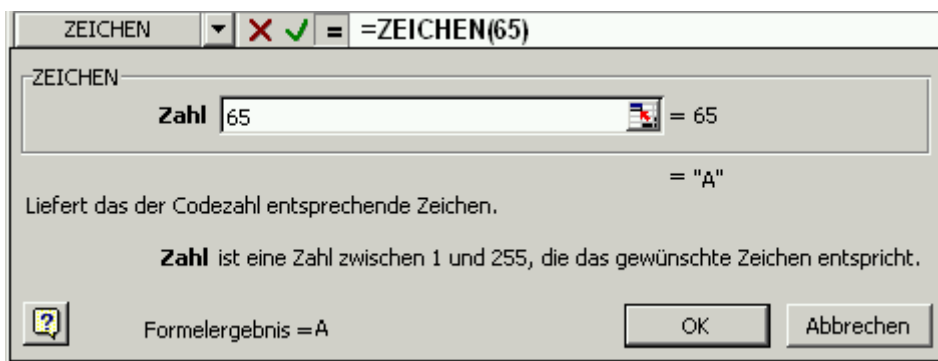



Abbildung 88: Funktionsassistent in Excel

Der Funktionsassistent trägt für dich das Gleichheitszeichen, den Funktionsnamen und die obligatorischen runden Klammern in die Bearbeitungszeile ein. In seinem Fenster ist für jeden anzugebenden Parameter eine Zeile vorgesehen; wenn die Einfügemarke darauf steht, wird der Parameter fett gedruckt und eine Erklärung zu seiner Bedeutung angegeben.

Außerdem berechnet der Funktionsassistent auch immer das Ergebnis der Funktion sowie der gesamten Formel in der Bearbeitungszeile, so dass du schon während der Eingabe einer Formel überprüfen kannst, ob vernünftige Ergebnisse herauskommen.

Wenn du als Parameter den Wert einer anderen Zelle oder eines Zellbereiches angeben willst, kannst du das wie zuvor schon beschrieben tun. Der Funktionsassistent hilft dir aber sogar auch dabei: Du kannst einfach auf die Zelle(n) klicken, die als Parameter übernommen werden sollen, und der Assistent trägt sie für dich ein! Dabei kannst du sogar Tabellen- und Dateiübergreifend klicken! Falls das Fenster des Funktionsassistenten die Zelle überdecken sollte, die du anklicken möchtest, kannst du mit dem Symbol  am Ende der Eingabezeile das Fenster so lange wegklappen.


7.3.2.3 Diagramme

Man sagt: „Ein Bild sagt mehr als 1000 Worte!“ Und das stimmt! Der Mensch ist so konstruiert, dass er Informationen aus Bildern sehr viel schneller erfassen und aufnehmen kann als aus Zahlen. Deshalb sollte man Zusammenhänge, die in einer Tabelle mit Zahlen dargestellt werden, möglichst auch bildlich darstellen. Dazu braucht man Diagramme. In der Einführung zu Excel habe ich schon ein paar gezeigt. Du kannst es daran ausprobieren: Beantworte die folgenden Fragen einmal an Hand von Tabelle 16 und einmal mit Hilfe von Abbildung 84 bzw. Abbildung 85:

1. „Wer hat die meisten Süßigkeiten?“

2. "Wer hat mehr Schokolade, Moritz oder Katharina?"

Zugegeben: Die Frage „Wie viele Bonbons hat Alexander?“ lässt sich schneller aus der Tabelle ablesen, aber meistens sind es die sogenannten *Tendenzaussagen*, also Antworten auf *Vergleiche*, die einen interessieren und die man vor allem auch anderen präsentieren will.

Wie erstellt man also ein solches Diagramm? Positioniere die Einfügemarke in die Tabelle, deren Daten du grafisch darstellen möchtest und wähle das Menü *Einfügen / Diagramm* oder klicke auf die Schaltfläche . Excel startet dann einen Assistenten, mit dessen Hilfe du ein Diagramm erzeugen kannst, ohne dass ich hier noch viel dazu schreiben müsste! Okay, ein paar grundlegende Sachen will ich noch erklären, aber den Rest lernst du wirklich am schnellsten, wenn du ein wenig mit Diagrammen herumspielst!

Das erste, wonach dich der Diagrammassistent fragt, ist der Diagrammtyp: Abbildung 84 ist ein Säulendiagramm und Abbildung 85 ein Kreis- oder Tortendiagramm. Es gibt noch eine Reihe weiterer Typen, zu denen der Assistent dir immer auch Beispiele zeigt, so dass ich die hier nicht aufführen muss. Ich will nur auf einen gänzlich unterschiedlichen Diagrammtyp eingehen: In den Beispielen oben wurde in den Diagrammen immer nur eine Größe zur Zeit durch genaue Messwerte (Zahlen) dargestellt, nämlich die Süßigkeiten. Die andere Größe in dem Diagramm, die Namen der Kinder, war nicht mit Zahlen auszu-drücken, nur mit Namen. Solche Unterteilungen nennt man *Rubriken*. Du erkennst Rubriken immer daran, dass es für die Aussage des Diagramms keine Rolle spielt, in welcher Reihenfolge oder in welchem Abstand die Rubriken dargestellt werden. (Excel nimmt sie so wie sie kommen aus der Tabelle.)

Im Gegensatz dazu kann es sein, dass man zwei voneinander abhängige Größen, die beide durch Zahlen bestimmt werden, in einem Diagramm darstellen möchte:

Stell dir vor, du hättest eine Fahrradtour gemacht, und dir zu verschiedenen Uhrzeiten aufgeschrieben, wie weit du schon gefahren bist, um später daraus ablesen zu können, wie schnell du auf den einzelnen Teilstrecken warst. Eine solche Tabelle könnte etwa so aussehen wie Tabelle 17:

Zeitpunkt	Strecke [km]	Bemerkungen
8:20	0,0	Abfahrt
8:30	2,9	
9:00	11,5	
10:00	29,1	steiler Berg voraus!
10:15	30,5	Berg zuende, Pause!
10:50	30,5	Pause zuende
11:20	44,3	
14:00	60,0	Endlich wieder daheim!

Sowohl die Zeit als auch die jeweils zurückgelegte Strecke werden mit Zahlen bestimmt.

Tabelle 17: Uhrzeiten und zurückgelegte Strecken einer Fahrradtour

Wenn du das als Diagramm darstellst, könnte es so aussehen:

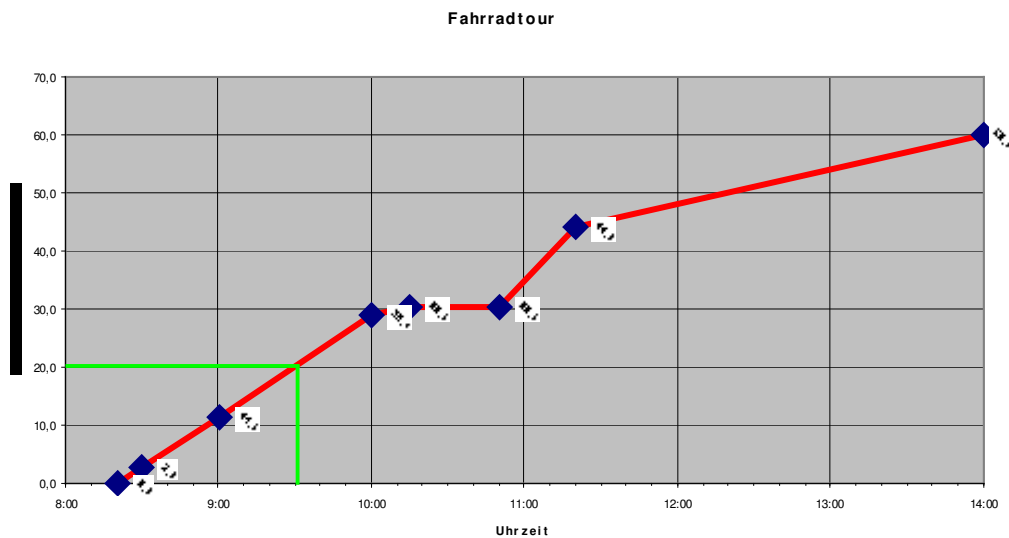


Abbildung 89: Weg-Zeit-Diagramm einer Fahrradtour

Was sagt dir dieses Diagramm? Du kannst daraus ablesen, wann du losgefahren bist; das ist nämlich die Uhrzeit, bei der die Kurve beginnt. Naja, aber das steht ja auch in der Tabelle! Du kannst auch sehen, wann du wie lange Pause gemacht hast, nämlich während des Zeitraums, in dem sich der Streckenwert nicht ändert. Du kannst sogar ablesen, dass du um 09:30 Uhr *ungefähr* eine Strecke von 20km hinter dir hattest. Du hast zwar zu dieser Zeit keinen Streckenwert notiert, aber wenn man annimmt – und das tut Excel, indem es zwischen zwei Messpunkten eine gerade Linie zeichnet – dass du zwischen zwei Messungen mit gleichbleibender Geschwindigkeit gefahren bist, dann stimmt diese Aussage! Dazu brauchst du nur auf der Zeitachse von 09:30 Uhr bis zur roten Linie senkrecht hochzugehen und von dort nach links waagerecht zur Streckenachse, wie ich es (nachträglich!) im Diagramm mit den grünen Linien angedeutet habe.

Kannst du auch sehen, mit welcher Geschwindigkeit du gefahren bist? Ja! Zumindest durchschnittliche Geschwindigkeiten kann man aus dem Diagramm ablesen. Was bedeutet Geschwindigkeit? Wenn man eine große Strecke in kurzer Zeit schafft, hat man eine große Geschwindigkeit, wenn man für eine kleine Strecke eine lange Zeit braucht, ist man langsam, also die Geschwindigkeit klein. Wie sieht dieser Sachverhalt im Diagramm aus? Eine große Strecke entspricht einem großen Stück auf der senkrechten Achse, eine kleine Zeit einem kurzen Stück auf der waagerechten Achse. Wenn also die rote Kurve steil ist, warst du schnell, wenn sie flach ist, warst du langsam!

Du kannst also, ohne die genauen Zahlen zu kennen, aus dem Diagramm sofort sehen, dass du vor der Pause sehr langsam warst (weil es dort den Berg

hinauf ging), und nach der Pause – ausgeruht – sogar schneller als zu Beginn deiner Tour. Das letzte Teilstück warst du wieder langsamer, vielleicht hattest du Gegenwind!?!)

Auch bei diesen Diagrammen kommt es also wieder darauf an, sehr schnell vergleichende Aussagen ablesen zu können, ohne dass einen die genauen Zahlenwerte interessieren.

Der Diagrammtyp, mit dem man solche Bilder erstellen lässt, heißt *Punkt (XY)*. Dieser Name kommt daher, dass man die beiden Richtungen, in denen die Größen (hier Uhrzeit und Strecke) aufgetragen werden, oft auch als x- und y-Achse bezeichnet, oder – wenn man ganz hochtrabend klingen will – als *Abzisse* und *Ordinate*.

In deiner Tabelle gehört zu jedem Zeitpunkt *genau ein* Wert für die zurückgelegte Strecke. Die Zeitpunkte, zu denen du die Streckenwerte aufgeschrieben hast, hat Excel als kleine blaue Karos in das Diagramm gezeichnet, die es dann mit der roten Linie verbunden hat.

Deine Messungen haben nicht in regelmäßigen Abständen stattgefunden, und Excel hat die Abstände entsprechend auf der Zeitachse auch unterschiedlich eingetragen.

Das ist die Besonderheit und das Wichtige bei diesem Diagrammtyp!

Wenn alle Zeitpunkte in dem selben Abstand auf der x-Achse aufgetragen worden wären (so wie in den Beispielen zuvor die Namen der Kinder), dann wäre die Aussage des Diagramms verfälscht worden: Du hättest falsche Geschwindigkeitsinformationen erhalten!

Nach der Eingabe des Diagrammtyps musst du angeben, welche Daten in dem Diagramm dargestellt werden sollen. Meist hat Excel das schon richtig vorausgewählt, du kannst es aber auf die selbe Weise wie beim Funktionsassistenten einfach durch Klicken auf die Zelle bzw. den Zellbereich eingeben. Dabei kannst du sogar gleich mehrere Kurven auf einmal in ein Diagramm zeichnen lassen.

Danach gibst du die ganzen Beschriftungen ein und bestimmst das Aussehen des Diagramms. Zuguterletzt entscheidest du noch, ob das Diagramm als eigenes Blatt dargestellt werden oder auf einem Tabellenblatt erscheinen soll – fertig!

7.3.3 Datenbank

Das Thema Datenbanksoftware werde ich nur sehr oberflächlich behandeln, weil man zu diesem Werkzeug normalerweise erst im fortgeschrittenen Stadium

des Computerhobbys greift. Andererseits ist *Datenbank* ein häufig verwendeter Begriff und es kann nicht schaden, wenn man eine Vorstellung davon hat, was er bedeutet. Es geht in diesem Kapitel also mehr um Grundsätzliches als um die Bedienung eines bestimmten Softwareprodukts.

Du könntest auf die Idee kommen, auf deinem Computer eine Liste zu führen, wem deiner Freunde du welches deiner Bücher geliehen hast.

Dazu könntest du zum Beispiel eine solche Tabelle anfertigen:

Buchtitel	ausgeliehen am	ausgeliehen an			Rückgabe erbeten bis	Rückgabe erfolgt am
		Name	Vorname	Telefon		

Tabelle 18: Bücherausleihe

So etwas könntest du in Excel machen und es handelt sich dabei schon um eine einfache Datenbank:

Jeder Ausleihvorgang wäre in dieser Tabelle eine Zeile. Das ist das Grundprinzip einer Datenbank: Sie beschreibt in Tabellen gleichartige Objekte, in diesem Fall Ausleihvorgänge. Die Eigenschaften jedes Objekts werden in Spalten und die Objekte selbst in Zeilen dargestellt. Das bedeutet nichts anderes als dass in diesem Beispiel bei allen Ausleihvorgängen der Buchtitel in der ersten Spalte steht, das Ausleihdatum in der zweiten usw. und dass nicht der eine Ausleihvorgang so und der nächste anders beschrieben wird.

*Bei Datenbanken nennt man die Spalten auch **Felder**. Eine komplett ausgefüllte Zeile heißt **Datensatz** oder auf englisch **Record**.*

Wenn du diese Tabelle ausfüllst, wirst du sehen, dass du die Buchtitel der beliebtesten Bücher immer wieder neu schreiben musst, weil es ja sein kann, dass du ein Buch nacheinander an mehrere Freunde ausleihst. Außerdem kann es sein, dass die selbe Person heute das eine Buch und in zwei Monaten ein anderes ausleiht. Du musst, um einen vollständigen Datensatz jedes Ausleihvorgangs zu erzeugen, den Namen jedes Mal wieder eintragen.

Das ist nicht nur lästig, sondern auch fehleranfällig! Nehmen wir an, du möchtest eine Aufstellung machen, welche Bücher ein bestimmter Freund schon alle bei dir ausgeliehen hat. Wenn du den Namen mehrmals eingetippt hast, kann es sein, dass du dich vertippt hast oder du mal den richtigen und mal den Spitznamen eingetragen hast. Bei der Auswertung würden die Datensätze mit anders geschriebenem Namen nicht mit auftauchen – oder du müsstest sie mit viel Mühe manuell herausuchen.

Ein weiterer Nachteil der einfachen Tabelle aus dem Beispiel: Wenn du wissen willst, ob du ein bestimmtes Buch verliehen hast – weil du es vielleicht in dei-

nem Regal nicht finden kannst – musst du die ganze Tabelle nach Einträgen für dieses Buch durchsuchen und nachsehen, ob ein Datensatz ohne Rückgabedatum existiert.

Wenn man sich hochtrabend ausdrücken will, kann man auch sagen, diese Datenbank enthält *redundante* Information. Redundanz heißt, dass etwas mehrfach vorhanden ist; im Beispiel der Datenbank wird eine Person (oder ein Buch) in mehreren Datensätzen mit der selben Information beschrieben.

Redundanz ist nicht grundsätzlich etwas Schlimmes: An deinem Fahrrad zum Beispiel gibt es zwei Bremsen, obwohl du auch mit einer zum Stillstand kommen könntest. Trotzdem will man diese Redundanz, für den Fall, dass mal eine Bremse ausfällt.

Bei Datenbanken möchte man *keine* Redundanz! Aus den oben beschriebenen Gründen möchte man, dass jede Information nur ein einziges Mal in der Datenbank enthalten ist.

Eine Datenbank, die diese Bedingung erfüllt, nennt man eine *relationale* Datenbank. Relation ist das lateinische Wort für „Beziehung“. Der Name kommt daher, wie man die Sache mit der Redundanz löst:

Man schreibt alles nur ein Mal in die Datenbank: Jedes Buch, jede Person und jeden Ausleihvorgang. Dazu hast du eine Tabelle mit allen Personen, die jemals etwas bei dir ausgeliehen haben; in dieser Tabelle steht jede Person aber nur *ein Mal*. Außerdem hast du eine Tabelle mit allen deinen Büchern. Für die Ausleihvorgänge legst du eine dritte Tabelle an, in der stehen die Termine für Ausleihe und Rückgabe. Außerdem steht darin ein Verweis auf die Büchertabelle, aus der hervorgeht, welches Buch in der Büchertabelle ausgeliehen wurde. Genauso gibt es einen Verweis auf die Personentabelle, der sagt welche Person aus der Personentabelle das Buch ausgeliehen hat. Auf diese Weise hast du also *Beziehungen* zwischen den Tabellen hergestellt, man kann auch sagen zwischen den drei verschiedenen Objekten Buch, Person und Ausleihe.

Tabelle 19 zeigt dir die neue *relationale* Datenbankstruktur:

Bücher		Personen			
Nr.	Titel	Nr.	Name	Vorname	Telefon

Ausleihen					
Nr.	Buch Nr.	ausgeliehen am	an Person Nr.	Rückgabe erbeten bis	Rückgabe erfolgt am

Tabelle 19: Tabellen einer relationalen Datenbank

Vielleicht ist dir aufgefallen, dass es in den Tabellen jetzt Felder gibt, die du vorher nicht brauchtest: „Nr.“

Die habe ich eingeführt, damit wir die Beziehungen zwischen den Tabellen herstellen können. Natürlich könnte man das auch über eins der existierenden Felder, zum Beispiel den Namen der Person. Aber es könnte ja sein, dass es zwei Personen mit dem selben Namen gibt (Geschwister!), die man nur durch den Vornamen unterscheiden kann. Dann wäre der Verweis von der Ausleihetabelle auf die Personentabelle nicht mehr eindeutig und das darf nicht sein – dann würden nämlich Fehler auftreten. Bei relationalen Datenbanken führt man deshalb der Einfachheit halber immer ein Feld ein, in dem man die Datensätze einfach durchnummeriert; das ist immer eindeutig!

Stell dir vor, einer deiner Freunde, der Bücher ausgeliehen hat, zieht um und seine Telefonnummer ändert sich. Im ersten Beispiel (Tabelle 18) müsstest du die ganze Tabelle nach Einträgen durchsuchen, in denen diese Person vorkommt und die Telefonnummer korrigieren. In der relationalen Datenbank (Tabelle 19) änderst du *nur den einen* Eintrag in der Personenliste und siehst bei allen Ausleihvorgängen die richtige Telefonnummer.

Tabellen haben wir doch schon gehabt! Excel ist doch eine *Tabellenkalkulation*! Ist das also das selbe wie ein Datenbankprogramm? Nein!

Man kann mit Excel die relationale Datenbankstruktur wie in Tabelle 19 erstellen und sie auch benutzen. Aber schon das Eintragen der Daten wird sehr umständlich. Ein Datenbankprogramm macht das viel einfacher und kann auch noch viel mehr!

Du kannst damit zum Beispiel Eingabeformulare erzeugen, so dass du dir beim Eingeben der Ausleihvorgänge gar keine Gedanken mehr darüber zu machen brauchst, welche Information du in welche Tabelle schreiben musst. Außerdem passt ein Datenbankprogramm auf, dass ein Feld, über das Beziehungen hergestellt werden, wirklich eindeutig ist und bleibt. Es sorgt dafür, dass in ein Datumsfeld nur Daten, in ein Nummernfeld nur Zahlen usw. eingegeben werden (das kann Excel allerdings auch!) und vieles mehr!

Das Datenbankprogramm, das zusammen mit Word und Excel im sogenannten Officepaket verkauft wird, heißt Access (sprich: äckseß).

Das Beispiel mit der Bücherausleihe war schon ein wenig an den Haaren herbeigezogen, und normalerweise braucht man zuhause kein Datenbankprogramm. Trotzdem ist es ganz hilfreich zu verstehen, wie so etwas funktioniert und wofür man es einsetzen kann. Dir begegnet so etwas öfter als du vielleicht denkst: Eine Stadtbücherei macht genau das selbe wie wir in unserem Beispiel! Hinter dem Internet-Auktionshaus ebay verbirgt sich nichts anderes als eine riesige Datenbank mit allen Auktionen. Jedes Geschäft hat eine Datenbank über Vorräte, Verkäufe, Einnahmen und Ausgaben.

7.3.4 Präsentation

Musstest du in der Schule schon einmal ein Referat halten, also deinen Mitschülern zu einem bestimmten Thema etwas erzählen? Oder musstest du schon einmal bei einem Referat zuhören? Auf jeden Fall wirst du im Unterricht bei deinen Lehrern schon zugehört haben. Welche Referate oder Vorträge haben dir am besten gefallen und vor allem bei welchen hast du am meisten behalten? Ich wette, es waren diejenigen, bei denen es nicht nur etwas zu hören sondern auch etwas zu *sehen* gab! Bei denen du die Hauptaussagen nicht nur gehört hast sondern sie auch lesen konntest; noch besser: bei denen dir Bilder oder sogar Filme das Thema veranschaulicht haben. Es ist erwiesen, dass man mit dem Auge sehr viel mehr Informationen aufnehmen kann als mit dem Ohr! Daher auch der Spruch in Kapitel 7.3.2.3 „Ein Bild sagt mehr als 1000 Worte!“ Aber selbst geschriebene Worte eignen sich besser als gesprochene, selbst wenn es kein Bild dazu gibt. Übertroffen wird das Sehen nur noch von *Ausprobieren*; Dinge die du *tust*, behältst du am besten im Gedächtnis!

Warum erzähle ich dir das? Weil der Computer dir auch dabei helfen kann, anderen etwas zu erklären. Das Programm *Powerpoint* (sprich: pauerpeunt) gibt dir die Möglichkeit, Texte, Bilder, → Animationen und Filme am Bildschirm, oder mit Hilfe eines → Beamers auch an einer Leinwand wie eine Diaschau darzustellen.

Die „Dias“, die du erzeugen kannst, heißen dabei *Folie*. Das kommt daher, dass man früher für den selben Zweck Tageslichtprojektoren benutzt hat. Auf Neudeutsch sagt man auch oft *Overhead*-Projektoren, weil sie über den Kopf des Vortragenden hinweg, der ja zum Publikum schauen soll, an die Wand in dessen Rücken projizieren. Wie auch immer man die Geräte nennt: Das was sie projizieren sollten, war auf eine Folie geschrieben, damit es von einer Lampe durchleuchtet werden konnte. Wie schon bei den Schreibmaschinen und der Textverarbeitung hat man bei Einführung des Computers und seiner Programme die Bezeichnungen beibehalten, damit die Leute nicht so viel umdenken mussten.

Die Menüs in Powerpoint werden dir vertraut vorkommen, oder wenn du sie durchschaust – oder noch besser: durchprobierst, wirst du sie leicht verstehen.

Beginne eine Präsentation mit dem Menü Datei / Neu.... Dort kannst du aus *Entwurfsvorlagen* eine auswählen und so sehr schnell zu einer ansehnlichen Präsentation gelangen. Mit dem Menü Einfügen / Neue Folie erzeugst du die nächste Seite deines Vortrages. „Abspielen“ kannst du deine Show mit Hilfe des Befehls *Bildschirmpräsentation* / Vorführen oder der Taste [F5].

Die Elemente, die aus der Entwurfsvorlage stammen und auf jeder deiner Folien auftauchen, kannst du mit dem Befehl Ansicht / Master / Folienmaster verändern und zwar für alle Folien gleichzeitig!

8 Netzwerke

Bestimmt hast du schon den Begriff *Internet* gehört! In diesem Kapitel will ich dir kurz erklären, was sich hinter Computernetzwerken und ganz besonders dem größten davon – dem Internet eben – verbirgt.

Die Grundidee ist einfach: In den vorangegangenen Kapiteln haben wir uns überlegt, wie ein Computer arbeitet und was er alles dafür braucht. Wir haben gesehen, dass der eigentliche Rechner (die → CPU) über → Busse, also Leitungen, mit verschiedenen Geräten wie dem Arbeitsspeicher, der Festplatte oder einem Drucker verbunden werden muss, damit die Daten bearbeitet, gespeichert oder gedruckt werden können.

Wenn nun bestimmte Daten auf zwei oder mehreren Computern verfügbar sein sollen, oder wenn mehrere Computer den selben Drucker oder den selben Scanner oder das selbe Laufwerk benutzen können sollen, dann liegt doch eigentlich nichts näher, als beide Computer mit einem Kabel zu verbinden und die entsprechenden Daten über dieses Kabel zu übertragen. Das ist ein *Netzwerk*! Der Name kommt daher, dass wenn man das mit mehreren Computern macht (und dann macht das erst richtig Spaß!), die Kabel den Maschen eines Netzes und die Computer seinen Knoten gleichen.

Natürlich wird es im Detail dann wieder nicht ganz so einfach, denn das Kabel ist eine → *Schnittstelle*, bei der man dafür sorgen muss, dass die Geräte an beiden Enden zueinander passen und sich gegenseitig „verstehen“. Das fängt bei den Steckern an und geht weiter über die Kabel, die bei manchem Standard zum Beispiel eine bestimmte Länge nicht überschreiten dürfen. Die Art und Weise, wie die Daten – du erinnerst dich: die Nullen und Einsen – dann auf dem Kabel dargestellt werden, und wer wann Daten senden oder empfangen soll, sind weitere Festlegungen, an die sich alle Geräte des Netzwerks halten müssen. Man nennt diese Festlegungen auch das *Netzwerkprotokoll*. Ein sehr häufig verwendetes Protokoll, mit dem auch das Internet arbeitet, heißt TCP/IP (sprich: ti-βi-pi ei-pi, Transmission Control Protocol / Internet Protocol, Übertragungssteuerungsprotokoll).

Als die Idee geboren wurde, Computer mit Kabeln zu Netzwerken zu verbinden, war auch irgendwann der Wunsch da, dies auch über weite Strecken tun zu können. Da lange Kabel erstens teuer und zweitens nicht so einfach zu verlegen sind, hat man dafür welche genommen, die es schon gab: Die Telefonleitungen. Die Idee war nicht schlecht und deshalb machen wir das heute auch noch so!

Das Telefon und seine Leitungen war nie dafür gedacht, Computer miteinander zu verbinden und Daten zu übertragen; es sollte nur in elektrische Signale umgewandelte Sprache übertragen können. Deshalb und wegen der großen Ent-

fernungen gibt es einige Einschränkungen hinsichtlich der Geschwindigkeit, mit der man die Daten darüber übertragen kann.

Wenn man nur innerhalb einer Firma oder in einer Schule auf einem begrenzten Gelände mit wenigen Gebäuden ein Netzwerk aufbauen will, dann benutzt man deshalb eine andere Technik und verlegt Kabel (oder sogar Lichtwellenleiter), die größere Geschwindigkeiten erlauben. Man spart sich dadurch auch das lästige Wählen, weil ohnehin immer alle Teilnehmer (Computer) miteinander verbunden sind.

*Solche Netzwerke nennt man **LAN**. Das steht für *Local Area Network*, also Netzwerk in einem örtlich begrenzten Gebiet.*

So wie man die Telefontechnik (also zum Beispiel *ISDN*) für Netzwerke über große Entfernungen – man spricht auch von *Wide Area Network* oder *WAN* – benutzt, so heißt die Technik, die bei LANs meist zum Einsatz kommt, *Ethernet* (sprich: iesernet). Das schreibe ich nur, damit du das mal gehört bzw. gelesen hast.

Wenn du also nun an einem Computer arbeitest, der mit einem Netzwerk verbunden ist, dann kannst du alle Daten und Geräte, die andere Rechner in diesem Netzwerk bereitstellen, von deinem Rechner aus benutzen. Es gibt sogar Spiele, die darauf ausgelegt sind, in einem Netzwerk von mehreren Spielern als Akteuren gespielt zu werden.

Das Internet ist ein besonderes Netzwerk: Zunächst einmal ist es das größte überhaupt! Es ist weltumspannend! Außerdem gibt es zwei wichtige Anwendungen: Zum einen den weltweiten, schnellen Versand von elektronischer Post und zum anderen die weltweit abrufbare Präsentation beliebiger Informationen.

Das erste kennst du unter dem Schlagwort *e-Mail*, was nichts anderes bedeutet als elektronische Post. Im Kapitel 8.3 liest du mehr darüber.

Das zweite kennt man als *Homepage* (sprich: houmpeytsch, zu deutsch „Heimatseite“) oder Internetseiten. Dabei handelt es sich um nichts anderes als → Dateien, die auf Rechnern im Internet gespeichert sind, und die man über das Netz auf den eigenen Rechner laden kann. Auf diese Weise werden im Internet unglaubliche Mengen von Informationen bereitgestellt: Zu allen möglichen und unmöglichen Themen bieten interessierte Leute Texte, Bilder, Filme und andere Daten an.

Das funktioniert deshalb so gut, weil sich für das → Dateiformat dieser Seiten ein Standard durchgesetzt hat, der mit einfachsten Mitteln, nämlich mit → ASCII-Text, umfangreiche Gestaltungsmöglichkeiten bietet.

Eine dieser Möglichkeiten, für die das Internet so berühmt geworden ist, ist der sogenannte *Hypertext* (sprich: heipertext): Dabei kann man durch Klicken auf besonders markierte Stellen im Text, sogenannte *Hyperlinks*, weitere Informationen abrufen, die irgendwo anders im Internet stehen können.

Als Leser dieser Informationen „bewegt“ man sich so von einer Datei zur nächsten, und alle können auf den verschiedensten, über die ganze Welt verstreuten Computern stehen! Für diesen Vorgang hat sich der Begriff *surfen* (sprich: sör-fen, englisch für „wellenreiten“) eingebürgert, weil man sich wie ein Wellenreiter auf einer Informationswelle durch die Welt bewegt.

Auf diese Weise werden Unmengen von Informationen im Internet miteinander verknüpft und erlauben es so, zu einem bestimmten Thema sehr schnell einen umfassenden Überblick zu bekommen.

*Der Standard – das Dateiformat – in dem Internetseiten geschrieben werden heißt **html**. Das steht für Hypertext Markup Language.*

8.1 HTML

Dem Schreiben von Internetseiten haftet der Nimbus des Schwierigen und des nur wenigen Eingeweihten Vorbehaltenen an. Häufig wird davon gesprochen, Internetseiten würden „programmiert“. Das ist meistens Unsinn! Man kann Internetseiten schreiben, wie man einen Brief schreibt! Natürlich sind die Möglichkeiten vielfältig und man muss gegebenenfalls lernen, wie man sie nutzt, aber das Grundprinzip ist einfach und soll hier kurz erläutert werden, damit du dich nicht von diesem Nimbus der Internetschreiberlinge ins Boxhorn jagen lässt.

Im Kapitel 7.3.1 hast du schon eine Menge über Formatierung gelernt, also darüber welche verschiedenen Möglichkeiten es in dem Programm Word gibt, den geschriebenen Text darzustellen. Genau die gleichen Möglichkeiten bietet dir auch html. Der einzige Unterschied besteht darin, dass die Befehle, mit denen du den Text formatierst, im Klartext gespeichert werden. Damit das Programm, das später die html-Datei anzeigen soll (siehe Kapitel 8.2), den eigentlichen Text von den Befehlen unterscheiden kann, werden alle <Befehle> in spitze Klammern gesetzt. Auf Englisch heißt ein solcher Klammerausdruck auch *tag* (sprich: täg).

Experiment 10: Wir schreiben eine Internetseite

1. Wähle aus dem Startmenü „Ausführen“.
2. Tippe „notepad“ ein; damit startest du den Texteditor.
3. Schreibe folgenden Text in das Fenster des Editors:


```
<p>
Fett schreiben geht so: <b>fett</b><br>
Kursiv schreiben so: <i>kursiv</i><br>
Unterstreichen so: <u>unterstrichen</u><br>
Und alles zusammen so: <b><i><u>fett, kur-
siv und unterstrichen!</u></i></b><br>
Auch <font color="#FF0000">rote
Schrift</font> geht und ganz viele andere
Sachen!
</p>
```
4. Wähle in Notepad das Menü „Datei / Speichern unter...“ und trage einen Dateinamen mit der Endung „.html“ ein, zum Beispiel „Mein-Test.html“.
5. Beende den Editor Notepad und suche im Explorer die gerade gespeicherte Datei.
6. Doppelklicke darauf und der Internet-Explorer (oder dein voreingestellter Browser) zeigt dir deine erste Internetseite an!

Ein paar Erklärungen dazu:

Fast immer haben Befehle einen Beginn und ein Ende: Zum Beispiel bedeutet `` „beginne mit Fettdruck“ und `` „Ende des Fettdrucks“. Dabei hat der Ende-Befehl immer den vorangestellten Schrägstrich. Hier die Bedeutung der verwendeten Befehle:

Befehl	Bedeutung
p	Beginn und Ende eines Absatzes (englisch: <i>paragraph</i>)
br	Neue Zeile beginnen (englisch: <i>break</i>), hierbei gibt es kein Ende-Tag
b	Beginn und Ende Fettdruck (englisch: <i>bold</i>)
i	Beginn und Ende Kursivschrift (englisch: <i>italic</i>)
u	Beginn und Ende Unterstreichen (englisch: <i>underline</i>)
font	Beginn und Ende Schriftart-Formatierung (englisch: <i>font</i> = Schriftart)

Tabelle 20: Einige html-Befehle

Beim *font*-Befehl ist dir sicherlich aufgefallen, dass in der Klammer des Befehls noch mehr stand als der Befehl selbst. Solche Parameter sind bei vielen Befehlen möglich. In diesem Fall wird damit die Farbe der Schrift (englisch: *color*, sprich: *kaller*) festgelegt. Alle Farben werden am Bildschirm aus rot, grün und blau zusammengemischt. Wie viel von jeder Farbe genommen werden soll, sagst du mit Hilfe einer Zahl, die von 0 bis 255 (oder → hexadezimal geschrieben von 00H bis FFH) reichen kann. Wenn du von jeder Farbe so viel wie möglich nimmst (255 rot, 255 grün, 255 blau), entsteht weiß, wenn du gar keine Farbe nimmst (0 rot, 0 grün, 0 blau), entsteht schwarz.

Das #-Zeichen kennzeichnet hier die hexadezimale Darstellung:

#000000 bedeutet „schwarz“, #FFFFFF bedeutet „weiß“ (alle Farben in voller Helligkeit), #2A2A2A bedeutet wie jeder Code, bei dem die drei Zahlenwerte gleich sind, grau.

#FF0000 bedeutet „kräftiges rot“, #610000 ein dunkleres rot.

#00FF00 bedeutet kräftiges grün, #0000FF bedeutet kräftiges blau und #FF00FF ist violett (rot und blau gemischt)!

Wenn du Spaß daran hast, spiele ein wenig mit den Farben herum! Du hast $256 \times 256 \times 256 = 16.777.216$ verschiedene zur Auswahl!

Das ganze sieht ziemlich umständlich aus, aber schwierig ist es nicht! Du solltest mit diesem Kapitel vor allem verstehen, dass Internetseiten nur Zeichen enthalten, die auch in Tabelle 6: ASCII-Code vorkommen! Das Kunststück, aus diesem Text eine Seite zu zaubern, wie du sie dann am Bildschirm siehst, das leistet das Darstellungsprogramm, der Browser.

Wenn es dir zu umständlich ist, die html-Befehle zu lernen, kannst du auch eine Word-Datei im html-Format speichern! Dazu musst du nur in Word das Menü *Datei / Speichern unter...* auswählen und in dem dann erscheinenden Fenster unter Dateityp „Webseite (*.htm; *.html)“ auswählen. Allerdings enthält der von Word erzeugte html-Code sehr viele Angaben, die du gar nicht brauchst.

8.2 Der Browser „Internet Explorer“

Der Internet Explorer ist ein solches Browser-Programm, mit dessen Hilfe du dir Internetseiten ansehen kannst; es handelt sich also um eine Anwendungssoftware und hätte demnach genauso gut im Abschnitt 7.3 besprochen werden können.

Da es hilfreich für das Verständnis ist, wenn du zuvor etwas über Netzwerke und html gelesen hast, kommt er aber erst jetzt.

Du hast schon gehört, dass der Internet Explorer aus html-Textdateien die Bildschirmdarstellung erzeugen muss, die du siehst. Man nennt diesen Vorgang auch *interpretieren* der html-Befehle.

Bevor er das kann, müssen die Seiten aber erst mal auf deinen PC kommen! Wir haben ja schon gesagt, dass Internetseiten auf irgendeinem Rechner irgendwo in der Welt stehen können, und du weißt auch schon, dass dein Computer nur Daten bearbeiten kann, die in seinem → Arbeitsspeicher stehen. Der Internet-Explorer hilft dir dabei, dass sie dort hin kommen:

8.2.1 Adressen

Dazu musst du als erstes einmal sagen, welche Seite du haben möchtest, oder bildlich gesprochen, *wohin du gehen möchtest* im Internet. Wie im richtigen Leben, nennt man das wohin man geht eine *Adresse*. Die trägt man in das Eingabefeld des Internet Explorers ein, vor dem „Adresse“ steht – ganz logisch, oder?

Und wie sieht so eine Adresse aus? Sicherlich tragen wir dort nicht ein, in welchem Land, welcher Stadt, welcher Straße und welchem Haus der Rechner steht, auf dem sich die Datei befindet, die wir sehen möchten!?

Eine Internetadresse beginnt mit einer Angabe zum → Protokoll, das man verwenden möchte, damit der besuchte Computer weiß, was man von ihm will. HTML-Seiten holt man mit dem Protokoll *http*, das steht für **H**ypertext **T**ransfer **P**rotokoll. Da wir mit dem Internet Explorer aber eigentlich immer nur html-Seiten übertragen wollen, können wir das auch weglassen; der Internet Explorer ergänzt es dann automatisch für uns. Manchmal siehst du auch, dass als Protokoll *https* dasteht. Das zusätzliche „s“ steht für *secure* und bedeutet, dass die Seite vor dem Übertragen verschlüsselt wird. In der → Statuszeile unten wird das mit einem kleinen gelben Vorhängeschloss angezeigt.

Die Angabe des Protokolls wird gefolgt von einem Doppelpunkt und zwei Schrägstrichen. Danach kommt ein weltweit eindeutiger (!) Name für den Computer, den wir besuchen wollen. Meistens ist dieser Name dreigeteilt und die Teile durch Punkte voneinander getrennt. Der erste Teil lautet meist *www*, das steht für *world wide web* (sprich: *wörld weid web*), also weltweites Netz und meint das Internet. Dann kommt ein Namensteil, der meist etwas über die Inhalte auf diesem Computer aussagen soll, oder wem er gehört, und dann kommt die sogenannte *Top Level Domain*. Das ist entweder ein Länderkürzel oder andere Kürzel wie *com* für kommerzielle Anbieter (Firmen), *org* für Organisationen oder *mil* für das amerikanische Militär. Rechner mit der Top Level Domain *de* werden in Deutschland verwaltet.

Eine vollständige Internetadresse, in diesem Fall die meiner privaten Homepage, sieht also so aus: <http://www.Regenbogenquadrat.de>

Fällt dir etwas ein, was daran eigentlich noch fehlt? Der obigen Beschreibung entsprechend führt diese Adresse doch erst mal nur zu einem Computer. Wir wollen aber doch eine Datei ansehen! Woher weiß der besuchte Computer, welche Datei wir sehen wollen? Ganz einfach: Es gibt einen Standard-Dateinamen, den der besuchte Computer immer dann übermittelt, wenn du nicht nach einer speziellen Datei fragst. Das ist ganz praktisch, weil man ja gar nicht so unbedingt weiß, wie die Datei heißt, die man ansehen möchte. Solltest du trotzdem einmal genau wissen, welche Datei du sehen möchtest, dann kannst du den Dateinamen und gegebenenfalls davor auch noch Verzeichnisnamen jeweils durch Schrägstriche getrennt angeben:

<http://www.Regenbogenquadrat.de/lego/kompressor.htm> zeigt dir beispielsweise die Seite, die ich über meinen aus LEGO gebauten Kompressor im Internet habe.

Bei der Angabe von Internetadressen ist es egal, ob man sie groß oder klein schreibt; du kannst genauso gut www.mijan.de oder WWW.MIJAN.DE oder wWw.mIjAn.dE eingeben. Allerdings kann es sein, dass der Computer, der sich hinter der Adresse verbirgt in seinem → Betriebssystem Groß- und Kleinschreibung unterscheidet (sehr oft benutzen diese Rechner UNIX oder LINUX), dann müssen mindestens die Verzeichnis- und Dateinamen das berücksichtigen! Meistens wird dann alles klein geschrieben.

Sobald du eine gültige Adresse eingegeben hast, versucht der Internet Explorer, die entsprechende Seite aus dem Internet zu laden. Dazu muss dein Rechner mit dem Internet verbunden sein (zu Hause geht das meistens über die Telefonleitung) und oft muss man dafür ein Kennwort eingeben. Die entsprechenden Einstellungen solltest du dir machen lassen, darauf kann ich hier jetzt nicht eingehen.

Streng genommen adressieren sich die Computer im Internet untereinander nicht direkt mit der Internetadresse, denn dann müsste ja auch der Rechner, mit dem du dich über die Telefonleitung ins Internet verbindest, einen solchen Namen bekommen. In Wirklichkeit besteht die Adresse *aller* Rechner im Internet, also auch derjenigen, die sich nur ab und zu einwählen, aus vier Zahlen zwischen Null und 255, die durch Punkte voneinander getrennt sind. Zum Beispiel 205.122.12.240

Eine solche Zahlenkombination heißt *IP-Adresse*. Wenn du in deinem Browser eine Internetadresse eingibst, fragt dein Rechner erst einen bestimmten Rechner im Internet, dessen Adresse er kennt, welche IP-Adresse zu diesem Namen gehört, und schickt dann erst die Anfrage nach der von dir gewünschten Seite an diese IP-Adresse. Das macht man so umständlich, weil Computer besser mit Zahlen und Menschen besser mit Namen umgehen können.

8.2.2 Surfen

Die Internetseite, die dir im Browser angezeigt wird, enthält Stellen, auf die du klicken kannst, um andere Seiten aufzurufen. Das können markierte (meist unterstrichene) Textstellen, Bilder oder Schaltflächen wie auf der Benutzeroberfläche eines Programms sein. Immer wenn der Mauszeiger sich über einer Stelle mit dieser Möglichkeit befindet, verwandelt sich der Pfeil in eine Hand mit ausgestrecktem Zeigefinger.

Wenn du klickst, wird die zu diesem *Link* gehörende Seite aufgerufen, meistens in demselben Fenster, in dem die vorangegangene Seite stand. Klickst du mit der rechten Maustaste auf den Link, gibt es eine Auswahl „In neuem Fenster öffnen“, so kannst du die alte geladene Seite behalten und die neue in einem anderen Fenster ansehen.

Mit den Schaltflächen „Zurück“ und „Vorwärts“ kannst du in den Seiten, die du mit diesem Fenster schon angesehen hast, blättern. Gefällt dir eine Seite so gut, dass du sie dir auch später immer mal wieder ansehen wirst, kannst du sie zu deinen *Favoriten* hinzufügen: Klicke dazu auf das Menü *Favoriten / Zu Favoriten hinzufügen...* Dadurch wird im Favoritenmenü ein Eintrag erzeugt, mit dem du die Seite später wieder aufrufen kannst.

Mit dem Menü *Ansicht / Quelltext* kannst du dir die html-Befehle ansehen, mit denen die gerade betrachtete Seite erzeugt wird. Das ist beim Lernen von html sehr hilfreich, wenn man wissen will, wie etwas, das man auf einer Homepage gesehen hat, mit html geschrieben werden kann.

8.2.3 Suchmaschinen

Oft – oder besser: meistens weiß man gar keine Internetadresse! Man sucht Informationen zu einem bestimmten Thema und weiß gar nicht, wo man die findet! Deshalb gibt es im Internet Suchmaschinen und Kataloge. Das sind Internetseiten, auf denen man Suchbegriffe eingeben kann, und man bekommt dann eine Liste von Links zu Seiten, in denen diese Suchbegriffe vorkommen.

Die am einfachsten zu bedienende Suchmaschine, die dazu noch die besten Treffer liefert, erreichst du unter der Adresse <http://www.Google.de> (sprich: guugl).

8.3 E-Mail

E-Mail (sprich: imeyl) ist die Kurzform von *electronic mail*, was nichts anderes heißt als „elektronische Post“. Natürlich kann man damit keine Weihnachtspakete versenden, aber es ist ein hervorragender Ersatz für die Briefpost! E-Mail hat nämlich zwei bis drei unschlagbare Vorteile:

- 1.) Sie ist quasi kostenlos! Man zahlt nur die Verbindungskosten ins Internet (Telefon).
- 2.) Sie ist unglaublich schnell! Selbst zu meinem Nachbarn kann ich einen Brief nicht so schnell hintragen, wie eine e-Mail ihn erreichen kann, und nach Hintertupfingen, Amerika, Afrika, Australien oder sonst wo in der Welt dauert es kaum länger! Dieser Umstand hat der herkömmlichen Briefpost – die in Deutschland ja auch schon ziemlich schnell ist! – den Spitznamen Snail-Mail eingebracht, was nicht nur ein schönes Wortspiel sondern auch die englische Übersetzung für Schneckenpost ist.
- 3.) Man braucht die Post, die man ja ohnehin meist am Computer schreibt, nicht mehr umständlich ausdrucken, in ein Kuvert stecken, frankieren und zum Briefkasten bringen, sondern man kann sie unmittelbar vom Arbeitsplatz versenden. Über den einfachen Brief hinaus kann man auch jede Datei (Bilder, Präsentationen, Tabellenkalkulationen usw.) als sogenannten Anhang einer Mail versenden.
Der Empfänger hat alles wieder in elektronischer Form vorliegen, so dass er es unglaublich einfach weiterverarbeiten, also zum Beispiel an jemand anderen weiterschicken kann.
Dieser Komfort macht sich ganz besonders dann bemerkbar, wenn man den selben Brief an sehr viele Leute schicken will.

Wie nutzt man also diese verlockenden Vorteile? Wie kann man e-Mails versenden und empfangen?

Nun, als erstes brauchst du dazu eine e-Mail-Adresse und einen Computer im Internet, der deine elektronische Post entgegennimmt, so lange dein Rechner nicht mit dem Internet verbunden ist. Diesen Computer kannst du dir vorstellen wie ein Postamt, das die ankommende Post in Postfächer einsortiert. Die Empfänger der Post kommen zu diesem Postamt und holen ihre Post aus dem Postfach.

Ein solches Postfach auf einem Computer, der permanent mit dem Internet verbunden ist, bekommst du von einem *Provider* (sprich: *proweider*, englisch für „Bereitsteller“). Viele Mobilfunkunternehmen bieten ihren Kunden gleichzeitig ein e-Mail-Konto an; auch wenn du von einer Firma Speicherplatz im Internet für eine Homepage anmietest, ist darin fast immer ein e-Mail-Konto enthalten. Es gibt aber auch Firmen, die dir so etwas kostenlos anbieten, weil Sie dadurch Werbung machen können. Du musst dich nur dazu anmelden. Eine seriöse Möglichkeit dazu findest du unter www.web.de.

Der Internetname dieses Postfach-Computers bildet den einen Teil deiner e-Mail-Adresse, nämlich den hinteren. Der vordere Teil ist ein Name für dein e-

Mail-Konto (dein Postfach), der auf diesem Computer eindeutig sein muss. Dazwischen steht das berühmte Zeichen „@“ (sprich: ät, englisch „bei“).

Früher war es einmal üblich, Vor- und Nachname durch einen Punkt getrennt für das e-Mail-Konto zu benutzen, aber die Zahl der Internet- und e-Mail-Nutzer ist so groß geworden, dass man das nicht mehr durchhalten kann. Beispiele für e-Mail-Adressen sind: Michael.Janssen@t-online.de; webmaster@MiJan.de; janssenmichael@web.de usw. Besonders originelle Varianten für e-Mail-Adressen (wenn man seinen Namen schon nicht mehr nehmen kann, weil es diese Adresse schon gibt) finde ich bildliche Bezeichnungen, die man sich gut merken kann wie zum Beispiel BuntesPony@provider.de oder RoterStein@provider.de usw. Welchen Namen du genau benutzt, verinbarst du mit deinem Provider.

Abbildung 90 Soll dir verdeutlichen, wie eine e-Mail durch das Internet befördert wird:

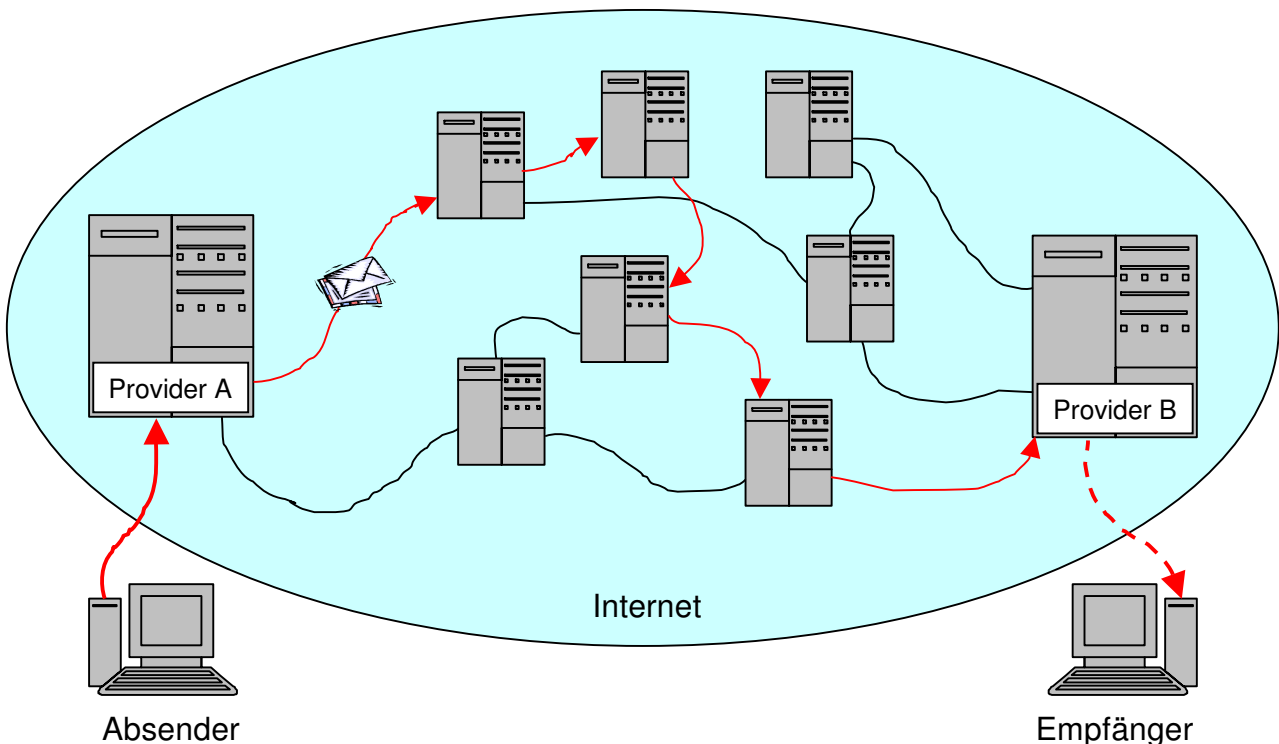


Abbildung 90: Weg einer e-Mail durchs Internet

Du als Absender schreibst auf deinem PC mit einem e-Mail-Programm einen Brief.

Dann sagst du deinem e-Mail-Programm, es soll deine Mail beim Rechner deines Providers (im Bild Provider A) abliefern. Dazu verbindet sich das e-Mail-Programm mit dem Internet und meldet sich mit deinen Benutzerinformationen (Benutzername und Kennwort) beim Provider A an. Dadurch weiß der Rechner des Providers, wer du bist und welches e-Mail-Konto dir gehört. Dann wird dein

Brief zu deinem Provider übertragen. Mit Hilfe des hinteren Teils der e-Mail-Adresse gelangt die Mail dann über viele willkürliche Zwischenstationen, die davon abhängen, welche Verbindungen gerade am besten funktionieren, zum Rechner des Empfänger-Providers (Provider B). Dieser weiß an Hand des ersten Teils der e-Mail-Adresse, für wen der Brief ist. Sobald sich der Empfänger das nächste Mal mit seinem e-Mail-Provider verbindet, wird sie dann auf dessen Rechner übertragen. Viele Provider bieten den (kostenpflichtigen!) Service, dich per SMS auf dein Handy zu informieren, wenn eine neue e-Mail eingetroffen ist. Manche Handys können sogar die komplette e-Mail abrufen.

Es gibt zwei verschiedene Methoden, sich mit dem Provider-Rechner zu verbinden: Bei der herkömmlichen Methode hat man auf dem eigenen Rechner ein kleines → Anwendungsprogramm, mit dem man e-Mails liest, schreibt, empfängt, sendet und verwaltet. Windows beinhaltet sogar ein solches Programm: Es heißt „Outlook Express“ (sprich: autluck). Bei dem zuvor schon beschriebenen Office-Paket, zu dem auch Word, Excel und Powerpoint gehören, ist sogar eine sehr leistungsfähige Variante dabei, die auch einen Kalender und viele Bürofunktionen beinhaltet, und nur „Outlook“ heißt. Diese Programme holen deine Post vom Rechner des Providers ab und bringen die zu versendende Post auch dorthin. Die → Protokolle, die dafür verwendet werden heißen SMTP (simple mail transfer protocol, einfaches Post-Übertragungsprotokoll, für das Versenden) und POP3 (post office protocol, Postamtprotokoll, für das Empfangen).

Es gibt aber auch Anbieter (Provider), die – meist zusätzlich zur oben genannten Methode – den Zugriff auf das e-Mail-Konto über eine Internetseite, also mit dem http-Protokoll erlauben. Dann braucht man zum Senden und Empfangen nur noch einen Browser, kein spezielles e-Mail-Programm mehr. Man spricht dann auch von einer *Web-Schnittstelle*. Ein Vorteil dieser Methode ist, dass man, egal wo man sich gerade in der Welt befindet, nur irgendeinen Rechner mit Internetzugang braucht, um seine e-Mails abzurufen. Nachteilig ist, dass die Post – auch die gesendete – dazu permanent im Internet gespeichert werden muss und dort ist der Platz meist begrenzt.

Für welche Methode du dich entscheidest, bleibt deinem persönlichen Geschmack überlassen.

So umständlich wie sich das jetzt hier anhört ist das alles gar nicht! Wenn einmal alles eingestellt ist, passiert das meiste von dem was ich hier erzählt habe, ohne dein Zutun. Dann ist das alles ganz einfach!

8.4 Gefahren des Internets

Oft hört man auch negative Berichte über das Internet: Sei es dass ein → Virus sich ungeheuer schnell verbreitet und großen Schaden allein dadurch anrichtet,

dass Millionen von Menschen von der Arbeit abgehalten werden, oder sei es dass Kriminelle das Internet für ihre Machenschaften nutzen, zum Beispiel indem sie unerlaubt mit Musik, Filmen oder Verbrechensbildern handeln oder indem sie Gewalt verherrlichende Texte und Bilder auf Ihren Internetseiten veröffentlichen, oder seien es sonst irgendwelche Betrügereien.

Man sollte das nicht verharmlosen; zwar finden ähnliche Delikte auch mit Hilfe anderer Medien statt, im Internet hat das Ganze aber deshalb eine besondere Qualität, weil eine viel größere Anzahl Menschen erreicht wird und weil die Strafverfolgung sehr viel schwieriger ist! Das Internet umspannt schließlich die ganze Welt und ein Gauner muss nicht unbedingt selbst in Deutschland sitzen, auch wenn seine Tat per Internet in Deutschland stattfindet.

Trotzdem ist es falsch und auch völlig wirkungslos, deshalb das Internet an sich zu verdammen! Wichtig ist nur, dass man sich der Gefahren bewusst ist und sich an bestimmte Regeln hält, um nicht betroffen zu werden. Straßenverkehr ist schließlich auch gefährlich; trotzdem bewegen wir uns täglich darin, weil wir die Chance eines Unglücks durch Regeln auf ein erträgliches Maß reduziert haben, und deshalb gar nicht mehr groß darüber nachdenken. Wir gehen zum Beispiel einfach nicht bei Rot über eine Ampel!

Genauso läuft das auch im Internet!

Die wichtigsten Gefahren und wie man sich davor schützen muss will ich dir in diesem Abschnitt kurz darstellen:

8.4.1 Viren

Können Computer auch Grippe bekommen? Natürlich nicht! Computerviren heißen so, weil sie ganz ähnliche Eigenschaften haben wie die Viren, denen wir die Grippe und andere Krankheiten verdanken.

Viren sind kleine Programme, die irgendeinen Schaden an deinem Rechner anrichten können: Manche sind harmlos aber lästig, indem sie irgendwelche Erscheinungen auf dem Bildschirm erzeugen, andere lassen den Computer → abstürzen und ganz üble Vertreter zerstören Daten auf deiner Festplatte.

Genau wie ihre biologischen Verwandten brauchen sie immer einen *Wirt*, mit dessen Hilfe sie von Computer zu Computer gelangen. Das kann entweder eine Diskette sein, auf der sich Viren auf den → Bootsektor schreiben, oder irgendeine Datei, in die Viren ihren Code gewissermaßen einbetten. Diese Datei wandert dann zum Beispiel ans Anhang einer e-Mail von einem Rechner zum nächsten. Meist sorgt der Virus sogar selbst dafür, dass er verschickt wird!

Ganz raffinierte Viren verändern sogar im Laufe der Zeit ihren Code selbst, um möglichst lange unentdeckt zu bleiben.

Die wichtigsten Wege, auf denen Viren deinen Rechner erreichen können, habe ich schon genannt: Disketten und das Internet. Einen Computer, den du mit dem Internet verbindest, musst du auf jeden Fall gegen Viren schützen. Du kannst ihn sozusagen gegen Grippe impfen.

Dazu gibt es Anti-Virenprogramme (auch *Virens Scanner* genannt), die deinen Rechner laufend auf Virenbefall überwachen. Damit sie das können, müssen diese Programme immer aktuell gehalten werden, um auch neueste Viren finden zu können.

8.4.2 Trojaner

Kennst du die Sage vom Trojanischen Pferd? Als die Griechen Troja belagerten und es ihnen lange nicht gelang, die Stadt zu erobern, ersann Odysseus, einer Ihrer Heerführer, folgende List: Sie bauten heimlich ein riesengroßes hölzernes Pferd, in dessen Innern Platz war für einige Soldaten. Dieses Pferd stellten sie nachts vor dem Stadttor Trojas ab und zogen sich zurück. Die Trojaner glaubten an ein Geschenk der Götter und holten das Pferd in die Stadt, um den Göttern für die Befreiung zu danken. In der Nacht schlichen die griechischen Soldaten aus dem Innern des Pferdes und öffneten den heimlich zurückgekehrten Griechen von innen das Stadttor. So wurde Troja dann schließlich doch eingenommen.

Ein Trojanisches Pferd auf einem Computer nennt man in Anlehnung an diese Sage eine Datei, die etwas anderes tut als du glaubst! Zum Beispiel könnte man dich glauben machen, ein Programm, das du im Internet zum Laden auf deinen Rechner angeboten bekommst, sei ein tolles Spiel. Wenn du es dann auf deinem Rechner ausführst, löscht es statt dessen den Inhalt deiner Festplatte!

Gegen Trojanische Pferde wehrt man sich am besten auch mit Hilfe eines Virens scanners und mit einem gesunden Misstrauen gegen alles, wovon man noch nichts gehört oder gelesen hat!

8.4.3 Dialer

Ein paar Stunden im Internet surfen kosten normalerweise wenige Euro. Diese Gebühren werden über die Telefonrechnung mit abgerechnet, weil die Telefongesellschaft weiß, von wann bis wann du mit welcher Telefonnummer verbunden warst, und wie viel Geld sie dafür deinem Internetprovider abgeben muss.

Diesen Service, Geld über die Telefonrechnung einzutreiben, bieten die Telefongesellschaften nicht nur Internet Providern an. Jede Firma kann für eine Dienstleistung (zum Beispiel eine Beratung) das Honorar über die Telefonrechnung bekommen. Dazu lässt sich die Firma von der Telefongesellschaft eine

spezielle Rufnummer geben. Preiswerte Dienste beginnen dabei mit 0180, teure Dienste mit 0900. (In jeder Werbung für eine solche Dienstleistung muss übrigens auch immer stehen, wie teuer die Verbindung pro Minute ist.)

Nun gibt es Programme, die gelangen wie ein Virus oder Trojanisches Pferd auf deinen Computer und installieren sich, ohne dass du es merkst. Danach wählen sie das Internet nicht mehr über deine gewohnte Rufnummer an sondern über eine teure 0900-Nummer – wieder ohne dass du etwas davon merkst! Am Monatsende, wenn die Telefonrechnung kommt, gibt es dann die böse Überraschung!

Solche Programme heißen *Dialer* (sprich: deiler, *dial* bedeutet auf Englisch „am Telefon wählen“).

Am wirkungsvollsten schützt man sich gegen Dialer, indem man bei der Telefongesellschaft den eigenen Telefonanschluss für 0900-Dienste sperren lässt.

Eine weitere Möglichkeit sind *Firewall* Programme (sprich: feierwuol): Das sind Programme, die jede Verbindung deines Rechners überwachen und mit denen man einstellen kann, welche Programme welche Rufnummern wählen dürfen.

Firewall ist Englisch und bedeutet so viel wie Brandschutzmauer. Dieser Name kommt daher, dass diese Programme wie eine Brandschutzmauer zwischen deinem Rechner und dem Internet stehen und jede unzulässige Verbindung abschirmen.

8.4.4 Würmer

Würmer sind so etwas wie elektronische Kettenbriefe; e-Mails, die dich dazu veranlassen sollen, sie möglichst schnell an möglichst viele andere Leute weiterzuleiten. Der Witz für die Urheber solcher Würmer liegt darin, dass sich die Zahl der verschickten e-Mails rasend schnell erhöht und damit das Internet unnötig verstopft wird.

Dir selbst wird unmittelbar kein Schaden zugefügt, außer vielleicht einem mulmigen Gefühl, ob du wohl das Richtige tust, wenn du eine solche Mail nicht weiterleitest. Meistens werden in einer Wurm-e-Mail nämlich ganz herzerreißende Geschichten erzählt, zum Beispiel von Leuten, die sterben müssen, wenn sie nicht ganz schnell etwas ganz Seltenes finden, was nur durch diese weltweite Umfrage möglich sei.

Warum vervielfältigt sich die Zahl der e-Mails bei einem Wurm so schnell? Nehmen wir der Einfachheit einmal an, dass jeder Empfänger den Wurm an nur zwei Leute weitersendet, die er kennt (in Wirklichkeit würdest du eine solche Geschichte - wenn du sie glaubst - an jeden senden, den du kennst, oder?).

Nun stelle dir ein Schachbrett vor. Für jede weitergeleitete Mail tust du ein Reiskorn auf eines der 64 Felder des Schachbretts. Auf dem ersten Feld liegen

also zwei Körner, auf dem zweiten liegen vier, weil ja jedes der beiden Reiskörner vom ersten Feld die Mail an zwei Leute weitergeschickt hat. Auf dem dritten liegen dann acht. Am Ende der ersten Reihe des Schachbretts liegen schon 256 Reiskörner; schon am Ende der zweiten Reihe hast du Schwierigkeiten, die notwendigen 65.536 Reiskörner unterzubringen! Am Ende der dritten Reihe brauchst du 16.777.216. Auf dem letzten Feld liegen schließlich 18.446.744.073.709.551.616 (18 Trillionen) Reiskörner! Dafür musste sich die Zahl der e-Mails nur 64 Mal verdoppeln.

8.4.5 Datensammler

Es gibt im Internet Seiten, auf denen du aufgefordert wirst, persönliche Daten wie deinen Namen, deine Adresse und deine e-Mail anzugeben. Manchmal verspricht man dir dafür ein Geschenk oder eine kostenlose Dienstleistung.

Solche Daten werden häufig an Firmen, die damit Werbebriefe oder Massen-e-Mail-Werbung (→ Spam) versenden, verkauft.

Du solltest solche Angaben nur dann machen, wenn du selbst einen bestimmten Dienst im Internet in Anspruch nehmen willst (zum Beispiel ein e-Mail-Konto einrichten, Mitglied bei einem Auktionshaus werden, usw.), und wenn dir der Anbieter bekannt ist.

Solltest du einmal im Internet einkaufen und Bankverbindungen oder Kreditkartennummern angeben, achte darauf, dass sie auf einer verschlüsselten Webseite (kleines gelbes Schloss unten rechts im Internet-Explorer, Protokoll „https“) eingegeben werden!

Die Regel gegen Datensammler lautet: Habe ein gesundes Misstrauen gegen alles, was du nicht verlangt hast und nicht kennst!

9 Programmieren

Nun bist du im Gebrauch deines Rechners schon ziemlich fortgeschritten! Das einzige was du noch nicht kannst, ist deinem Rechner beizubringen, was er tun soll, also: *Programmieren*.

Auch das ist wieder ein Begriff, der für die meisten mit etwas Geheimnisvollem und Schwierigem verbunden ist, aber das ist alles halb so wild!

Erinnern wir uns erst mal an unsere Definition eines Programms, denn programmieren heißt ja nichts anderes als ein Programm zu erstellen:

Ein Programm ist ein im Voraus festgelegter Ablauf. Es legt fest, was wann gemacht wird.

Das hört sich doch eigentlich ganz einfach an – und das ist es auch! Das einzige was ein Programm schwierig aussehen lässt, ist die Sprache, in der man es dem Computer nachher mitteilt – die Programmiersprache. Aber dazu kommen wir später! Wer nämlich ordentlich programmieren will, der kümmert sich zunächst mal *überhaupt nicht* um die Programmiersprache. Das gilt auch für professionelle Programmierer.

Stelle dir vor, du bist im Urlaub in einem fremden Land und suchst den Weg zu einer Sehenswürdigkeit. Wenn dir dann jemand den Weg beschreibt (also den Programmablauf deines Spaziergangs), ist es doch viel wichtiger, dass er dir die richtigen „Befehle“ gibt, also zum Beispiel „erste Straße links, dritte Straße rechts und dann 500m geradeaus“, als die Frage, in welcher Sprache er das tut. Die einzige Bedingung ist, dass du die Sprache verstehst.

Man fängt beim Programmieren also immer damit an, sich den Weg ans Ziel zu überlegen. Dazu muss man wissen, von wo man losgeht und wo man hin möchte. Übertragen auf ein Computerproblem bedeutet die Frage, wo man losgeht zum Beispiel, welche Eingabedaten zur Verfügung stehen, und das Ziel entspricht dem, was der Computer später als Ergebnis ausgeben soll.

Erinnerst du dich an das Beispiel mit der Waschmaschine aus Kapitel 3.6? Dabei sind die Eingabegrößen: Schmutzige Wäsche, maximale Wassertemperatur, maximale Schleuderdrehzahl, Wäscheart (zum Beispiel „pflegeleicht“), Wasser und Waschmittel. Das gewünschte Ziel ist saubere, nicht eingelaufene, möglichst trockene Wäsche. Der Ingenieur, der eine Waschmaschine bauen will, muss sich das Programm überlegen, wie er von der Eingabe zur Ausgabe kommt. Dabei entsteht dann zum einen die Waschmaschine selbst, also die Hardware, und das Programm, das die Maschine beim Waschen abarbeitet.

Über die Hardware denken wir beim Programmieren meistens nicht mehr nach; sie ist nur der Grund, warum wir deinen Computer *nicht für jede* Aufgabe programmieren können – Zimmer aufräumen kann er eben wie gesagt nicht!

9.1 Unsere erste Programmieraufgabe

Betrachten wir also mal ein einfaches Problem, für das du deinen Computer programmieren könntest:

Du möchtest eine Zahl eingeben können und wissen, durch welche Zahlen man sie ohne Rest teilen kann. Du möchtest also die Teiler einer Zahl berechnen.

Versuche zunächst einmal selbst, dir zu überlegen, was der Computer tun muss, damit du vom Ausgangspunkt „Zahl eingeben“ zum Ziel „Teiler ausgeben“ kommst!

Hier ein Lösungsvorschlag:

Wir probieren einfach alle in Frage kommenden Zahlen aus! Erst teilen wir unsere Zahl durch 2 und schauen ob ein Rest bleibt, dann durch 3, durch 4 usw. so lange bis unser Teiler genauso groß ist wie die eingegebene Zahl. Jedes Mal wenn es uns gelingt, unsere Zahl ohne Rest zu teilen, geben wir die Zahl, durch die wir gerade geteilt haben, als Teiler unserer Eingabezahl aus.

1. Dividend (so heißt in der Sprache der Mathematik eine Zahl, die geteilt werden soll) eingeben und speichern
2. Divisor (so heißt in der Sprache der Mathematik die Zahl, durch die der Dividend geteilt werden soll) auf 2 setzen
3. Rest der Division „Dividend durch Divisor“ berechnen
4. Falls der Rest Null ist, Divisor als Teiler ausgeben
5. Divisor um 1 vergrößern (*→ inkrementieren*)
6. Prüfen ob der Divisor gleich dem Dividenten ist,
falls ja, Programm beenden
falls nein zu Schritt 3 zurückgehen.

Schon wenn man versucht, das Programm als Text zu formulieren, fällt einem auf, dass man für die Zahlen, mit denen gerechnet werden soll, Bezeichnungen braucht – irgendwelche Namen oder Platzhalter. Deshalb habe ich diese komischen Worte, Dividend und Divisor eingeführt. Man weiß ja noch gar nicht, welche Zahl nachher wirklich der Dividend, der Divisor oder der Rest sein wird.

Beim richtigen Programmieren ist das genauso: Egal welche Sprache man verwendet, es gibt immer sogenannte *Variablen*, die als Platzhalter für die Daten verwendet werden, mit denen später gerechnet wird. Du kannst dir Variablen

auch als Namen der Speicherplätze vorstellen, in denen die Rechengrößen gespeichert werden.

Der Begriff kommt von „variabel“, was *veränderlich* bedeutet. Variablen verwendet man, um die veränderlichen Werte zu beschreiben.

Unser Programm beginnt also im Schritt 1 mit der Eingabe der Zahl, die auf Teiler untersucht werden soll, dem Dividend. Und natürlich muss die Zahl gespeichert werden, weil der Computer sie in den Berechnungen ja immer wieder brauchen wird.

Im nächsten Schritt überlegen wir, welches die erste Zahl ist, die wir als Teiler ausprobieren wollen. Wir könnten auch 1 als ersten Wert wählen, aber das ist unnötig, weil ja *jede* ganze Zahl durch 1 teilbar ist; also brauchen wir das auch nicht ausprobieren.

In Schritt 3 findet dann der Test statt, ob die Division einen Rest hat. Je nachdem, ob unsere Programmiersprache einen Befehl zur Berechnung des Rests einer Division hat oder nicht, müssen wir diesen Schritt später vielleicht mit mehreren Programmbefehlen realisieren. Aber das ist das Schöne an einer solchen Programmbeschreibung: Wenn das Gerüst erst mal steht, kann man die einzelnen Schritte später immer noch genauer beschreiben.

Wenn wir die Division durchgeführt haben, müssen wir das Ergebnis in Schritt 4 noch bewerten: War unser Divisor nun ein Teiler oder nicht. Wenn er ein Teiler war, wenn also der Rest der Division Null war, dann geben wir den Divisor aus, und sagen damit dem Bediener des Programms, dass dieser Wert des Divisors ein Teiler des Dividenden ist.

Danach wollen wir mit der nächsten Zahl weitermachen und erhöhen deshalb in Schritt 5 den Divisor um Eins.

Natürlich soll das Programm auch irgendwann fertig werden. Deshalb müssen wir prüfen, ob der Divisor denn schon seinen Endwert erreicht hat, bevor wir erneut versuchen, ob es einen Rest gibt. Der letzte Wert, den der Divisor annehmen soll, ist der des Dividenden. Dafür brauchen wir die Division aber nicht mehr ausprobieren, weil wir genau wie bei der Division durch 1 genau wissen, dass es nie einen Rest gibt, wenn wir zwei gleiche Zahlen durcheinander teilen: $3:3=1$; $12:12=1$ und ohne groß nachzudenken kann ich behaupten

$$323.297.632.974.563.875 : 323.297.632.974.563.875 = 1$$

Wenn der Divisor also gleich dem Dividenden ist, endet das Programm!

*Eine solche Beschreibung, wie man in endlicher Zeit von einer definierten Ausgangssituation zu einem bestimmten gewünschten Ergebnis kommt, nennt man auch einen **Algorithmus**. Das Programm ist dann die Übersetzung eines Algorithmus in eine bestimmte Computersprache.*

9.1.1 Testen des Algorithmus

Nachdem man sich einen Algorithmus für die Lösung seines Problems überlegt hat, probiert man ihn erst mal aus! Überlegen wir also, was unser Computer machen würde, wenn wir uns die Teiler der Zahl 6 berechnen lassen wollen:

1. Eingabe der Zahl 6, Dividend = 6
2. Divisor = 2
3. Dividend durch Divisor teilen:
 $6/2 = 3$, Rest = 0
4. Rest = 0, also Divisor 2 als Teiler ausgeben
5. Divisor inkrementieren:
 $2 + 1 = 3$
6. Ist Divisor ungleich Dividend?
3 ist ungleich 6, also zu Schritt 3 gehen
3. Dividend durch Divisor teilen:
 $6/3 = 2$, Rest = 0
4. Rest = 0, also wird der Divisor 3 als Teiler ausgegeben
5. Divisor inkrementieren:
 $3 + 1 = 4$
6. Ist Divisor ungleich Dividend?
4 ist ungleich 6, also zu Schritt 3 gehen
7. Dividend durch Divisor teilen:
 $6/4 = 1$, Rest = 2
8. Rest ungleich Null, keine Ausgabe eines Teilers
9. Divisor inkrementieren:
 $4 + 1 = 5$
10. Ist Divisor ungleich Dividend?
5 ist ungleich 6, also zu Schritt 3 gehen

11. Dividend durch Divisor teilen

$$6 : 5 = 1, \text{ Rest} = 1$$

12. Rest ungleich Null, keine Ausgabe eines Teilers

13. Divisor inkrementieren:

$$5 + 1 = 6$$

14. Ist Divisor ungleich Dividend?

Divisor = Dividend, Programm beendet.

Unser Programm hat für den Dividenten 6 die Teiler 2 und 3 ausgegeben! Für dieses Beispiel funktioniert es also! Müssen wir den Algorithmus jetzt für alle Zahlen ausprobieren? Nein! Aber es macht durchaus Sinn, sich neben „normalen“ Eingabewerten Extremfälle zu überlegen, bei denen es zu Schwierigkeiten kommen kann.

Überlege dir, was passiert, wenn wir als Dividenten 999.999.999 eingeben! Kann im Programmablauf irgendein Problem auftreten? Außer dass unser Rechner viel länger rechnet als bei 6 eigentlich nicht.

Was passiert, wenn wir 1 als Dividenten eingeben? Überlege einmal selbst bevor du weiterliest! Gehe den Algorithmus auf die selbe Weise schrittweise durch, wie wir das oben gemacht haben!

Wir würden erwarten, dass der Rechner bei einer so kleinen Eingabe sehr schnell fertig ist! Aber das Gegenteil ist der Fall: Er rechnet, und rechnet und rechnet und rechnet und rechnet und rechnet und rechnet und rechnet und rechnet und rechnet

Warum?

Unser Programm wird nur dann beendet, wenn der Divisor nach dem Erhöhen (INKrementieren) gleich dem Dividenten ist. Der kleinste Divisor, den wir verwenden, ist 2! Damit ist er schon größer als der eingegebene Divident. Nachdem diese Festlegung Divisor = 2 in Schritt Nummer 2 getroffen worden ist, wird der Divisor in unserem Programm immer nur noch größer (Schritt 5); er kann also niemals mehr den gleichen Wert annehmen wie der Divident und deshalb endet das Programm nie!

Dürfen wir denn 2 als Divident eingeben? Probiere den Algorithmus aus, um zu verstehen, warum auch das zu einem niemals endenden Programm führt: Wir erhöhen den Divisor, *bevor* wir ihn mit dem Dividenten vergleichen! Beim ersten Vergleich ist der Divisor in diesem Fall also auch schon größer als der Divident!

3 ist die kleinste Zahl, die wir als Divident eingeben dürfen!

Lass uns noch mehr komische Fälle untersuchen!

Was passiert mit unserem Algorithmus, wenn wir „2,8“ als Dividenden eingeben? Als erstes ist natürlich klar, dass der Anwender, der eine Kommazahl auf ganzzahlige Teiler untersuchen lassen will, nicht viel Ahnung von Mathematik hat, aber man muss beim Programmieren wirklich jede Eventualität einkalkulieren; es könnte sich ja auch um einen Tippfehler handeln, und wenn unser Programm damit nicht umgehen kann, wäre das ja ärgerlich.

Was passiert ist wieder folgendes: Der Divisor ist in unserem Programm immer eine ganze Zahl weil wir bei 2 beginnen und nie etwas anderes tun als 1 dazuzuzählen. Infolgedessen kann der Divisor nie gleich dem Dividenden sein, der ja keine ganze Zahl ist und im Programm auch nie verändert wird. Unser Programm wird wieder endlos laufen!

Was können wir dagegen tun? Ganz einfach: Wir fügen nach der Eingabe des Dividenden eine Prüfabfrage ein:

1. Dividend eingeben und speichern
 - 1a. Wenn der Dividend kleiner ist als 3 oder wenn es sich nicht um eine ganze Zahl handelt, Fehlermeldung ausgeben und zurück zu Schritt 1.
2. Divisor auf 2 setzen
3. ...

Mit dieser Ergänzung funktioniert unser Programm erst mal zufriedenstellend!

Bist du enttäuscht, dass wir den Computer unser Problem durch *Ausprobieren* lösen lassen? Denke an Kapitel 3! Die Aufgaben, die ein Computer lösen können sollte, mussten ganz einfach sein. Dafür kann er davon dann ganz viele in ganz kurzer Zeit lösen. Das wird dir beim Programmieren immer wieder begegnen, dass du die eigentliche Aufgabe in ganz viele ganz kleine und einfache Teilaufgaben zerlegen musst.

9.1.2 Verbessern des Algorithmus

Trotzdem kann man die Rechnung noch etwas intelligenter gestalten! Es gibt nämlich meistens mehrere Algorithmen, mit denen man ein Problem lösen kann, und die haben unterschiedliche Vor- und Nachteile. Ein Qualitätskriterium ist dabei immer, wie schnell sich ein Problem mit einem bestimmten Algorithmus lösen lässt.

Versuchen wir also unseren Algorithmus etwas schneller zu machen! Wir könnten uns zum Beispiel überlegen, ob wir wirklich jede Zahl ausprobieren müssen, ob sie ein Teiler des Dividenden ist:

Beispielsweise sind die Teiler von 36: 2, 3, 4, 6, 9, 12, 18

$$2 \times 18 = 36$$

$$3 \times 12 = 36$$

$$4 \times 9 = 36$$

$$6 \times 6 = 36$$

Wenn wir den kleinsten Teiler einer Zahl gefunden haben (hier: 2), dann haben wir gleichzeitig auch schon den größten gefunden! Das ist nämlich das Ergebnis der Division durch diesen kleinsten Teiler, im Beispiel 18. Alle Zahlen, die größer sind als dieser größte Teiler, brauchen wir also gar nicht mehr ausprobieren, ob sie auch noch Teiler sind! Haben wir den zweitkleinsten Teiler gefunden, haben wir gleichzeitig auch den zweitgrößten. Es ist wieder das Ergebnis der Division durch den gefundenen Teiler, in unserem Beispiel also 12. Zwischen dem größten und dem zweitgrößten Teiler kann es ebenfalls keinen weiteren Teiler geben; also brauchen wir auch diese Zahlen nicht auszuprobieren.

Um unseren Algorithmus zu verbessern, brauchen wir uns also nur den jeweiligen „Partner“ eines gefundenen Teilers zu merken; weiter als bis zum jeweils letzten gefundenen Partner brauchen wir keine weiteren Teiler zu suchen!

Genau genommen können wir noch ein wenig sparen:

Die Quadratwurzel einer Zahl ist diejenige Zahl, die mit sich selbst mal genommen die ursprüngliche Zahl ergibt. Beispiel: Die Quadratwurzel von 36 ist 6, weil $6 \times 6 = 36$. Man schreibt das mit einem Zeichen so:

$$\sqrt{36} = 6$$

Genau bis zu dieser Zahl müssen wir Teiler suchen, weil das die größte Paarung von Teilern ist, die auftreten kann.

Eine Quadratwurzel auszurechnen, ist nicht so einfach! Bei 36 ging das ja noch, aber versuche es einmal für 10! $\sqrt{10} = 3,1622776601683793319988935444327\dots$

Die meisten Computer können das heute. Um das Programmieren zu üben, wollen wir aber annehmen, unser Computer hätte dafür keinen Befehl und uns wäre das auch zu kompliziert, einen zu programmieren. Wir programmieren deshalb erst mal die zuerst beschriebene Variante, bei der wir immer den Partner des letzten gefundenen Teilers als Abbruchkriterium benutzen:

1. Dividend eingeben und speichern
 - 1a. Wenn der Dividend kleiner ist als 3 oder wenn es sich nicht um eine ganze Zahl handelt, Fehlermeldung ausgeben und zurück zu Schritt 1.
2. Divisor auf 2 setzen
3. Rest der Division „Dividend durch Divisor“ berechnen
4. Falls der Rest Null ist,
Divisor als Teiler ausgeben
Partner neu berechnen als Dividend : Divisor
Partner als Teiler ausgeben

5. Divisor um 1 vergrößern
6. Prüfen, ob Divisor gleich Partner ist,
falls ja, Programm beenden
falls nein, zu Schritt 3 zurückgehen.

Der Vorteil dieses Algorithmus ist, dass er erheblich schneller ist. Der Nachteil ist, dass die Teiler nicht mehr in aufsteigender Reihenfolge ausgegeben werden. Außerdem wird bei Quadratzahlen die Wurzel zweimal als Teiler ausgegeben. Dieses Manko lässt sich aber leicht beheben.

Leider gibt es noch einen weiteren Nachteil: Dieser Algorithmus rechnet bei Primzahlen (Zahlen, die genau zwei Teiler haben: 1 und sich selbst) immer noch bis zum Dividenden. Wie könnten wir diesen Mangel noch beheben?

Können wir eine andere Grenze finden, die einfacher zu berechnen ist als die Quadratwurzel? Irgendwie hat die Anordnung der Teiler auf dem Zahlenstrahl doch etwas symmetrisches: Die eine Hälfte ist kleiner als die Quadratwurzel des Dividenden, die andere größer und unter Umständen ist noch ein Teiler gleich der Quadratwurzel (daraus folgt übrigens: Nur Quadratzahlen haben eine ungerade Anzahl von Teilern!).

Wie wäre es denn, wenn wir die Suche bei der Hälfte des Dividenden abbrechen? Diese Festlegung ist rein willkürlich intuitiv gewählt, weil Symmetrie immer etwas mit Halbierung zu tun hat. Tatsächlich können wir zeigen, dass wir bei der Hälfte einer Zahl auf jeden Fall mit der Suche aufhören können, weil die Hälfte nämlich ab der Zahl 4 immer größer bzw. bei 4 gleich der Quadratwurzel ist. 2 und 3 haben keine ganzzahligen Teiler und bei der 4 finden wir die 2 als Teiler, bevor wir die Suche abbrechen. Wir können also noch eine Zeile 2a einfügen, in der wir unser Abbruchkriterium „Partner“ auf die Hälfte des Dividenden setzen:

1. Dividend eingeben und speichern
 - 1a. Wenn der Dividend kleiner ist als 3 oder wenn es sich nicht um eine ganze Zahl handelt, Fehlermeldung ausgeben und zurück zu Schritt 1.
2. Divisor auf 2 setzen
 - 2a. Partner = Dividend : 2
3. Rest der Division „Dividend durch Divisor“ berechnen
4. Falls der Rest Null ist,
Divisor als Teiler ausgeben
Partner neu berechnen als Dividend : Divisor
Partner als Teiler ausgeben
5. Divisor um 1 vergrößern

6. Prüfen, ob Divisor gleich Partner ist,
falls ja, Programm beenden
falls nein, zu Schritt 3 zurückgehen.

Führe mit diesem neuen Algorithmus den gleichen Test durch, wie wir es für den alten getan haben, um seine Funktionsweise zu verstehen!

9.2 Programmelemente

Vielleicht hast du dich darüber gewundert, dass ich die Programme bisher nur als einfache Textbeschreibung hingeschrieben, also nur die Algorithmen beschrieben habe. Es scheint noch ein wenig weit weg, diesen Text in Befehle zu gießen, die ein Computer verstehen kann. Warum das gar nicht schwierig ist, will ich dir in diesem Kapitel erklären und im nächsten siehst du dann, wie ein richtiges Programm aussieht.

Um von der Vielfalt der Sprache zu einfachen Befehlen zu kommen, schauen wir doch mal, ob in der Beschreibung der Algorithmen nicht Vorgänge auftauchen, die sich ähneln, die man also mit Befehlen (und dazugehörigen → Parametern) vereinfacht darstellen kann.

Eingabe

Das erste was wir in unserem Programm beschreiben, ist die Eingabe und Speicherung einer Zahl. Es lässt sich leicht vorstellen, einen Befehl „Eingabe“ zu realisieren, dem wir als Parameter mitgeben, in welchen Variablen die einzugebenden Werte zu speichern sind. Ein weiterer Parameter könnte vielleicht noch ein Text sein, der dem Benutzer angezeigt wird, damit er weiß was er eingeben soll.

Bedingung

In Schritt 1a unseres Algorithmus wird eine Bedingung abgefragt „*Wenn ... dann ...*“! Allein von der Sprache her wäre denkbar, dass auch noch ein „*sonst*“ hinterherkommt. *Nur wenn* die Bedingung erfüllt ist sollen bestimmte Programmteile abgearbeitet werden, sonst andere.

Solche Bedingungen sind in der Programmierung sehr wichtig! Man nennt sie auch *Verzweigung*, weil sich abhängig von der Bedingung der „Weg durch das Programm“ verändert.

Ein Befehl „Wenn“ bräuchte als Parameter die Bedingung und den Programmteil, der ausgeführt werden soll, wenn die Bedingung erfüllt ist, und den der ausgeführt werden soll, wenn sie nicht erfüllt ist. Natürlich kann man sich auch vorstellen, dass eine Bedingung mehr als zwei Auswahlmöglichkeiten zulassen soll. Du kannst dir aber leicht vorstellen, dass man das mit verschachtelten

Wenn-Befehlen lösen kann; dafür braucht man also keinen eigenen Befehl. Ein Beispiel:

Es soll das Gewicht eines Briefes eingegeben und das Porto berechnet werden. Ein Brief bis 20g kostet 0,50€, ein Brief zwischen 20g und 50g kostet 1,00€ und ein Brief über 50g kostet 2,00€:

1. Eingabe und Speicherung von Gewicht
2. Wenn Gewicht kleiner oder gleich 20g, dann
Porto 0,50€ ausgeben.
Programm beenden.
sonst weiter mit 3.
3. Wenn Gewicht kleiner oder gleich 50g, dann
Porto 1,00€ ausgeben.
Programm beenden.
sonst weiter mit 4.
4. Porto 2,00€ ausgeben.

Blöcke

Eine weitere Funktion beim Programmeschreiben ist nicht so offensichtlich, aber es leuchtet irgendwie ein, dass wir etwas brauchen, womit wir Programmteile zu *Blöcken* zusammenfassen können, damit wir zum Beispiel in einer Bedingung wie oben sagen können, was alles zu dem Programmteil gehört, der beispielsweise dann ausgeführt werden soll, wenn die Bedingung erfüllt ist.

Wir sind immer noch bei Schritt Nummer 1a der Algorithmusbeschreibung und haben schon drei Sachen gefunden, die wir in Befehlen formulieren können!

Vergleiche

Ebenso wie die Blockbildung ist auch die nächste Gruppe von Befehlen eng mit der Möglichkeit Bedingungen zu formulieren verknüpft: Wir müssen vergleichen können! Außerdem wollen wir logisch Verknüpfen können. Damit sind die Formulierungen „oder“, „und“, „nicht“ in den Bedingungen gemeint. Wenn wir uns das genau überlegen, sind solche *Operationen* auch nichts anderes als die mathematischen Berechnungen plus und minus: Zwei Größen werden miteinander verknüpft und heraus kommt ein Ergebnis, das von der Definition der Operation abhängt. Der einzige Unterschied ist, dass das Ergebnis keine richtige Zahl sondern sie Aussage *wahr* oder *falsch* ist. Beispiele:

5 ist größer als 3 hat das Ergebnis *wahr*.

7 ist kleiner als 4 hat das Ergebnis *falsch*.

„Äpfel sind rund“ und „Birnen sind viereckig“ hat das Ergebnis *falsch*.

„Äpfel sind rund“ oder „Birnen sind viereckig“ hat das Ergebnis *wahr*.

„Birnen sind nicht viereckig“ hat das Ergebnis *wahr*.

Für alle diese Operationen brauchen wir also Befehle, die als Parameter die beiden (oder manchmal auch mehr) verknüpften Größen, man nennt sie die *Operanden*, erhalten und die ein Ergebnis liefern.

Wertzuweisung

Und wohin dann mit dem Ergebnis? Auch dafür brauchen wir einen Befehl! Das sieht man sehr schön im Schritt 2 oder noch besser im Schritt 2a: Dort wird einer Variablen ein *Wert zugewiesen*. Bei diesem Befehl muss man ein wenig aufpassen, wie man ihn nennt! Man könnte ja, wie ich das in der Textbeschreibung auch getan habe, das Gleichheitszeichen benutzen und sagen „Variable = Wert!“.

Wie schreibe ich dann aber meine Vergleichsoperation, mit der ich prüfe, ob zwei Größen gleich sind, wenn ich also Frage: „Ist Variable1 = Variable2?“

Es gibt Programmiersprachen die dafür das selbe Zeichen benutzen, aber man muss sich immer klar machen, dass das unterschiedliche Dinge sind!

Ausgabe

Wir haben im Schritt Nummer 1a noch zwei Arten von Befehlen ausgelassen! Das eine ist die *Ausgabe*. Das kann man sich relativ leicht vorstellen, daraus einen Befehl zu formulieren, der als Parameter mitbekommt, was er ausgeben soll, und vielleicht noch wohin (Bildschirm, Drucker, Datei, ...) und in welchem Format.

Sprünge

Wichtiger ist da schon der *Sprungbefehl*: Ganz am Ende von Schritt Nummer 1a steht doch, wo das Programm weitermachen soll, nachdem der Befehlsblock für die erfüllte Bedingung abgearbeitet ist. Ein solcher Befehl lässt sich sehr einfach realisieren, als Parameter hat er nur das Sprungziel. Wenn du dich an Kapitel 3.6 erinnerst und was dort so beiläufig über das Befehlszähler-Register erzählt wurde (siehe Seite 55), dann kannst du dir vorstellen, dass ein Sprungbefehl nichts anderes tut, als eine neue Adresse ins Befehlszähler-Register zu schreiben.

Schleifen

Die letzte wichtige Sorte von Befehlen, die wir noch brauchen, ist auf den ersten Blick in unserem Beispielalgorithmus nicht so ohne weiteres zu erkennen, aber trotzdem drin enthalten: Betrachte noch einmal die Schritte Nummer 3 bis 6! Dort wird ein Block von Befehlen mehrmals durchlaufen, und zwar so lange bis eine bestimmte Bedingung erfüllt ist.

Eine solche Konstruktion nennt man eine *Schleife*. Dabei unterscheidet man drei Typen: Bei der ersten weiß man vor Beginn der Schleife, wie oft sie durch-

laufen werden soll. Bei den anderen beiden ergibt sich die Anzahl der Durchläufe aus den Berechnungen, die in der Schleife gemacht werden. Man unterscheidet dann nur noch, ob die Bedingung *vor Eintritt in den Schleifenkörper* überprüft wird, oder erst danach. Im ersten Fall kann es sein, dass die Befehle, die zur Schleife gehören, gar nicht durchlaufen werden, im zweiten Fall werden sie mindestens einmal durchlaufen.

Erkennst du, welchen Schleifentyp unser Algorithmus verwendet?

Es hängt von den gefundenen Teilern und ihren Partnern ab, wie viele Divisoren wir ausprobieren, also wie oft wir die Schleife durchlaufen. Der erste Schleifentyp kann es also nicht sein.

Die Prüfung, ob wir die Schleife verlassen können, findet am Ende statt! Die Schritte 3 bis 6 werden also mindestens einmal durchlaufen.

Das war jetzt ziemlich viel Text, der dir eigentlich nur zeigen sollte, welche Arten von Befehlen wir brauchen, um unseren Computer programmieren zu können. Deshalb fasse ich das noch einmal kurz zusammen: Wir brauchen:

- Eingabebefehle
- Ausgabebefehle
- Bedingungen (*wenn ... dann ... sonst*)
- Blockbildung
- Operationen (*Rechenarten*)
 - Vergleiche (*kleiner, größer, gleich*)
 - Logische Verknüpfungen (*und, oder, nicht*)
 - Mathematische (*plus, minus, mal, geteilt, Rest, Wurzel, ...*)
- Sprünge
- Schleife (*drei Typen*)

Das sind die Werkzeuge, die eine Programmiersprache uns zur Verfügung stellen muss, damit wir ein Programm schreiben können – eigentlich nicht viel, oder?

Leider sind das noch nicht die Befehle, die der Computer versteht! Wenn du dich an das Beispiel eines Assemblerbefehls aus Kapitel 3.6 auf Seite 54 erinnerst, passt das noch nicht ganz zusammen.

Der Rest ist aber ganz einfach: Es gibt Programme, die Befehle der oben beschriebenen Kategorien verstehen und für den Computer übersetzen. – Wieder kommt das altbewährte Prinzip zum Einsatz, Aufgaben, die für den Computer zu schwierig sind, in kleinere Teilaufgaben zu zerlegen. In diesem Fall tut das ein Programm für uns.

9.3 Programmierumgebungen

Programmierbefehle auf der Ebene der oben beschriebenen Kategorien nennt man auch *höhere Programmiersprachen*. Sie können von Menschen ziemlich leicht verstanden werden, auch wenn man sich an strikte Regeln halten muss.

Es gibt eine Reihe von Programmen, die eine solche Sprache (es gibt verschiedene davon) für den Computer übersetzen können. In jedem Fall schreibst du dein Programm erst mal in der höheren Programmiersprache als Textdatei. Diesen Text nennt man auch den *Quellcode*.

Dann gibt es zwei grundsätzlich verschiedene Möglichkeiten:

Entweder wird dieser Quellcode jedes Mal, wenn du das Programm ausführst, in Computerbefehle übersetzt. Dann nennt man das Programm, das das macht einen *Interpreter*.

Oder du lässt das Programm *einmal* in eine ausführbare Datei (→ Dateiendung „.exe“) übersetzen, die die Computerbefehle in binärer Form, also als Nullen und Einsen enthält. Zum Ausführen des Programms rufst du dann nur noch diese Datei auf. Ein Programm, das so etwas kann, nennt man einen *Compiler* (sprich: kompeiler).

Beide Varianten haben Vor- und Nachteile: Programme, die von einem Interpreter übersetzt werden müssen, können nie alleine laufen, brauchen also immer zusätzlich das Interpreterprogramm, was sie zum einen langsamer macht, denn neben den eigentlichen Programmbefehlen muss ja gleichzeitig auch die Übersetzung abgearbeitet werden, und zum anderen umständlich ist, wenn man sie weitergeben will. Andererseits kann man mit einer Interpretersprache das Programm sehr einfach testen, weil man sich um die Übersetzung nicht kümmern muss; die läuft immer automatisch.

Zu einer *Programmierungsumgebung* gehört neben dem Interpreter oder Compiler noch ein Editor, also ein Programm, in dem du dein Programm schreiben kannst. Es hilft dir beim Schreiben zum Beispiel dadurch, dass es dir sagt, wenn du dich versehentlich nicht an die Regeln der Programmiersprache hältst, oder dadurch dass es dir Hilfen zu den verschiedenen Programmierbefehlen anzeigt.

Einige Beispiele für Programmiersprachen, die es so gibt, sind: Basic, Pascal, Ada, Cobol, Java, Javascript, C++, Fortran, Algol,

Welche man verwendet ist im Wesentlichen eine Frage der persönlichen Vorliebe!

Die Entwicklung der Programmiersprachen und Programmierungsumgebungen hat noch etwas hervorgebracht, das es dem Programmierer erleichtern soll „sauber“ zu programmieren, also Fehler zu vermeiden und den Programmcode leicht nachvollziehbar zu machen: Die *Objektorientierte Programmierung!* Was

ist das nun schon wieder? Ein Objekt ist der vornehme Ausdruck für „Gegenstand“ oder „Ding“.

Traditionell sind die Dinge, die ein Programm bearbeitet, → Variablen. Diese Variablen kann man mit verschiedenen → Operatoren, man könnte auch sagen mit verschiedenen *Methoden* bearbeiten und verändern, und man kann ihren Wert auslesen. Der Wert, den eine Variable gerade hat, ist sozusagen eine *Eigenschaft* der Variablen; genauso ihr Typ, also ob es sich um Text, ganze Zahlen, Kommazahlen usw. handelt.

Bei der Objektorientierten Programmierung hat der Programmierer die Möglichkeit, außer Variablen noch andere Objekte zu definieren, die er mit Methoden, die er ebenfalls festlegt, bearbeiten kann, oder die ihm in seinem Programm bestimmte Eigenschaften mitteilen können. Solche Objekte können dann auch noch verschachtelt werden – damit es übersichtlich bleibt!

Zum Beispiel könnte ein solches Objekt ein Fenster auf dem Bildschirm sein. Für dieses Fenster wären beispielsweise die Methoden „maximieren“, „minimieren“, „wiederherstellen“ und „schließen“ (entsprechend des → Schaltflächen-*trios*) sinnvoll. Eigenschaften des Fensters können zum Beispiel Breite, Höhe, Position auf dem Bildschirm oder das Vorhandensein von Rollbalken sein.

Es hat sich in der Objektorientierten Programmierung eingebürgert, Objekte, Unterobjekte und Methoden bzw. Eigenschaften jeweils durch einen Punkt getrennt zu schreiben.

```
x = programmfenster.breite / 2
programmfenster.maximieren()
```

sind typische Befehle in einem objektorientiert geschriebenen Programm. Der erste würde zum Beispiel die Eigenschaft der Fensterbreite halbieren und einer Variablen x zuweisen. Der zweite würde die Methode „maximieren“ auf das Programmfenster anwenden. Methoden erkennt man immer an dem Klammerpaar am Ende, weil man dazu manchmal Parameter angeben muss, die in diesen Klammern übergeben werden.

Natürlich müssen die Methoden nach wie vor programmiert werden! Der Vorteil ist aber, dass gleichartige Objekte (wie zum Beispiel Fenster) immer den selben Code benutzen.

Solche Objekte wie Fenster, die jedes Programm braucht, sind in der Programmierumgebung in der Regel mit ihren Eigenschaften und Methoden schon festgelegt, so dass man sich um solche Programmteile schon gar nicht mehr kümmern muss.

9.4 Implementieren des Programms

Der letzte Abschnitt war immer noch ziemlich abstrakt. Wir haben immer noch kein richtiges Programm geschrieben, das dein Computer hätte verstehen können. Das wollen wir jetzt tun!

Das Umsetzen eines Algorithmus in Quellcode nennt man auch *implementieren*, was eigentlich „einbauen“ heißt; man baut den Algorithmus quasi in ein Programm ein.

Wenn du in Kapitel 9.3 aufgepasst hast, weißt du, dass wir, um ein richtiges Programm zu schreiben, einen Compiler oder Interpreter brauchen. Hast du nicht? Doch hast du! Dein Internet-Browser ist gleichzeitig ein Interpreter für die Programmiersprache *Javascript*!

Mit Javascript kann man kleine Programme in eine Internetseite einbauen, die dann entweder automatisch ablaufen, wenn die Seite geladen wird, oder die der Betrachter der Seite über einen → Link aufrufen kann.

Damit Internetseiten bei ihren Besuchern nicht irgendwelchen Schaden anrichten können, ist die Programmiersprache in bestimmten Bereichen sehr eingeschränkt. Zum Beispiel kann man damit nicht auf irgendwelche Dateien zugreifen. (Sonst könnte man ja Internetseiten programmieren, die deine Festplatte löschen!)

Wir wollen in diesem Kapitel den Algorithmus zum Finden der Teiler einer Zahl aus Kapitel 9.1 in Javascript implementieren. Eine Besonderheit dabei ist erstens, dass das Programm in einer html-Seite steht, und zweitens dass Javascript eine *objektorientierte* Sprache ist. Du wirst deshalb aus unten stehendem Quellcode noch etwas mehr über html lernen als in Kapitel 8.1 und gleichzeitig die Besonderheiten objektorientierter Programmierung. Keine Angst, es ist alles nicht so schlimm wie es sich anhört und ich erkläre dir jede einzelne Zeile der Datei!

Wenn du alles was in Tabelle 21 in der Spalte Quellcode steht, zum Beispiel mit Notepad in eine Datei schreibst und sie zum Beispiel als „teiler.html“ speicherst, kannst du das Programm an deinem Rechner ausprobieren! Achte dabei auch auf Kleinigkeiten wie die Semikolons am Ende der Zeilen! Sonst funktioniert das Programm nicht!

Die Zeilennummern habe ich nur hinzugefügt, um im Text die einzelnen Zeilen ansprechen zu können. In deine html-Datei dürfen sie nicht mit rein! In der rechten Spalte steht eine kurze Erläuterung jeder Zeile; unten im Text erkläre ich die Zusammenhänge.

Das eigentliche Programm steht zwischen den → Tags `<script>` und `</script>` (in der Tabelle gelb hinterlegt).

Zeile	Quellcode	Erklärung
1	<html>	Mit diesem Befehl beginnt jede html-Datei.
2	<head>	<i>Head</i> ist Englisch und bedeutet "Kopf". Das ist ein Teil der Datei, der Informationen enthält, die nicht angezeigt werden sollen – zum Beispiel Javascript-Programme.
3	<title>Teiler finden mit JavaScript</title>	Dieser Text wird in der Titelzeile des Browserfensters angezeigt (<i>title</i> = englisch Titel).
4	<script type="text/javascript">	Hiermit wird der Beginn eines Javascripts gekennzeichnet.
5	function teiler()	Das ist eine Überschrift für Programmteile, die separat zum Beispiel aus einer Internetseite aufgerufen werden können.
6	{	Geschweifte Klammern kennzeichnen Programmblöcke. Alles was jetzt kommt, gehört zu der Funktion „teiler“.
7	var dividend = 0, divisor = 2, partner;	Man muss in Javascript <i>vorher</i> sagen, welche → Variablen man verwenden möchte. Zusammen mit der Namensfestlegung kann man auch schon Werte vorgeben.
8	do	Leitet eine Schleife ein, deren Abbruchbedingung am Ende geprüft wird (englisch <i>do</i> = tue, sprich: du)
9	{	Hier wird der Block eingeleitet der zur do-Schleife gehört.
10	dividend = window.prompt("Die Teiler welcher Zahl sollen berechnet werden?", "");	Eingabe der Zahl, deren Teiler gesucht werden soll. „window“ ist ein vordefiniertes Objekt und „prompt()“ die Methode, mit der man aus dem Fenster eine Eingabe abholt. Diese Methode erhält als Parameter einen Text der dem Benutzer angezeigt wird und einen Standardwert, der genommen werden soll, wenn der Nutzer nichts eingibt.
11	if (dividend <= 3 dividend % 1 != 0) alert("Der Dividend muss ganzzahlig und größer als 3 sein!\nVersuche es noch einmal!");	Hier wird geprüft, ob der eingegebene Wert den Bedingungen entspricht: <i>if</i> ist Englisch und bedeutet „wenn“; in den Klammern wird die Bedingung formuliert und dahinter kommt der Programmteil, der nur dann ausgeführt wird, wenn die Bedingung wahr ist. <= bedeutet „kleiner oder gleich“ bedeutet „oder“ % berechnet den Rest einer Division != bedeutet „ungleich“, (! bedeutet „nicht“) alert() gibt eine Meldung aus (englisch: „Alarm“) \n beginnt eine neue Zeile
12	}	Ende des Blocks, der zur do-Schleife gehört
13	while (dividend <= 3 dividend % 1 != 0);	Abbruchbedingung der do-Schleife: Das Eingabefenster und die Fehlermeldung kommen so oft, bis dividend ganzzahlig und größer 3 ist. <i>while</i> (sprich: weil) ist Englisch und bedeutet „so lange wie“. So lange wie die Bedingung erfüllt ist, wird die Schleife ausgeführt.
14	partner = dividend / 2;	Die Variable Partner wird auf die Hälfte des Dividenden gesetzt.
15	document.write("Die Teiler von " + dividend + " sind: ");	Auf das vordefinierte Objekt „document“ (was im aktiven Fenster angezeigt wird), wird die Methode „write()“ (englisch: schreiben, sprich <i>wreit</i>) angewandt. Sie schreibt den in der Klammer angegebenen Text in das aktive Fenster. Mit den Pluszeichen werden die auszugebenden Teile verknüpft.
16	do	Beginn der Schleife zur Teiler-Suche
17	{	Beginn des Schleifenblocks
18	if (dividend % divisor == 0)	Ist der Rest der Division (%) Null? Beachte: Der Rest der Division wird mit Null <i>verglichen</i> ! Deshalb „==“ anstatt „=“ wie bei der <i>Wertzuweisung</i> in Zeile 14!
19	{	Beginn des Blocks, der nur ausgeführt wird, wenn die Bedingung in Zeile 18 wahr ist.

Zeile	Quellcode	Erklärung
20	<code>document.write(divisor + ", ");</code>	Es wird wieder etwas in das aktive Fenster geschrieben: Der aktuelle Divisor, der ja wegen Zeile 18 ein Teiler von dividend sein muss, gefolgt von einem Komma, weil wir ja noch mehr Teiler erwarten.
21	<code>partner = dividend / divisor;</code>	Der Partnerteiler des Divisors wird berechnet. / ist das Zeichen für die normale Division
22	<code>document.write(partner + ", ");</code>	Auch der Partnerteiler wird als Partner ausgegeben.
23	<code>}</code>	Ende der Anweisungen, die nur ausgeführt werden, wenn Zeile 18 wahr ist.
24	<code>divisor++;</code>	Der Divisor wird → inkrementiert (um 1 erhöht).
25	<code>}</code>	Ende des Schleifenblocks Teilersuche
26	<code>while(divisor < partner);</code>	Abbruchbedingung der Teilersuche: So lange wie der Divisor kleiner ist als der Partnerteiler wird die nächste Zahl getestet, ob sie auch ein Teiler ist.
27	<code>}</code>	Ende des Blocks, der zur Funktion teiler() gehört.
28	<code></script></code>	Ende des Javascripts
29	<code></head></code>	Ende des Kopfs der Internetseite
30	<code><body></code>	<i>Body</i> ist Englisch und bedeutet „Körper“. Alles was zwischen <code><body></code> und <code></body></code> steht wird auf der Internetseite angezeigt (außer es ist speziell als Kommentar gekennzeichnet.)
31	<code><p></code>	Beginn eines Absatzes (paragraph)
32	Von dieser Seite aus kannst du ein Programm starten, das alle Teiler einer Zahl berechnet. 	Dieser text wird im Browser angezeigt. erzeugt eine neue Zeile.
33	Klicke dazu auf den Knopf:	In der neuen Zeile steht dieser Text.
34	<code></p></code>	Ende des Absatzes
35	<code><form></code>	Dieses Tag leitet ein Formular ein, mit dem man auf einer Internetseite Benutzereingaben machen lassen kann.
36	<code><p></code>	Beginn eines Absatzes
37	<code><input type="button" value="Teiler berechnen" onClick="teiler()"></code>	Eingabemöglichkeit in einem Formular: <i>input</i> = englisch: „Eingabe“ <i>button</i> = englisch: Knopf (es gibt auch andere Eingabemöglichkeiten wie Auswahlfelder etc.) <i>value</i> = englisch: Wert (sprich: wälju) enthält den Text, der auf dem Knopf draufstehen soll <i>onClick</i> = englisch: bei Klick; hier steht, was passieren soll, wenn der Knopf angeklickt wird; es soll die Funktion teiler() aufgerufen werden, die wir im Kopf der Internetseite definiert haben.
38	<code></p></code>	Ende des Absatzes
39	<code></form></code>	Ende des Formulars
40	<code></body></code>	Ende des Dateikörpers
41	<code></html></code>	Ende von html

Tabelle 21: JavaScript-Programm zur Teilerberechnung

Wenn du alles richtig getippt hast, müsste beim Aufruf der Datei eine solche Seite in deinem Browser erscheinen:



Abbildung 91: Internetseite zum Aufrufen des Programms Teilerberechnung

Wenn du dann auf den Knopf geklickt hast, erscheint ein Fenster wie in Abbildung 92, in das ich schon eine – falsche – Eingabe gemacht habe:

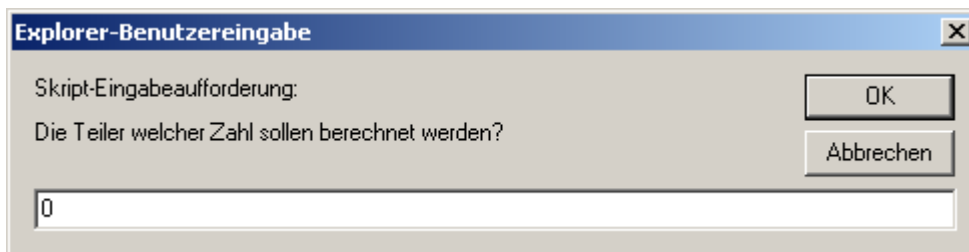


Abbildung 92: Eingabefenster des Programms zur Teilerberechnung

Die Überprüfung der Eingabe erzeugt in Zeile 11 des Programms (siehe Tabelle 21) folgende Fehlermeldung:



Abbildung 93: Fehlermeldung des Programms zur Teilerberechnung

Wenn man einen zugelassenen Wert für den Dividenden eingibt (im Beispiel 144) erscheint folgende Ausgabe:

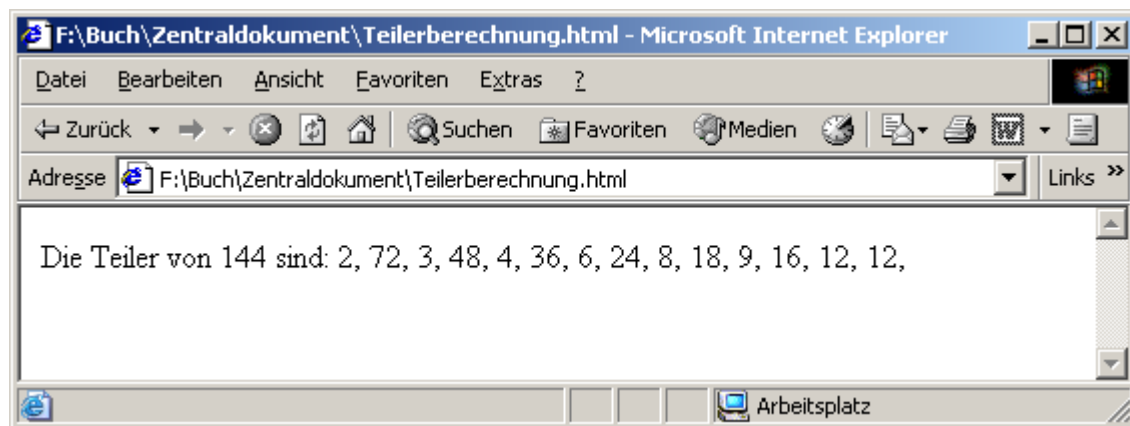


Abbildung 94: Ausgabefenster des Programms zur Teilerberechnung

Für ein relativ kleines Programm sieht das doch schon recht professionell aus, oder? Das liegt natürlich daran, dass Javascript die ganzen Möglichkeiten mit Fenstern schon zur Verfügung stellt.

Gehen wir das Programm noch einmal durch:

Als erstes ist der Kleinkram wichtig: Jede Zeile muss mit einem Semikolon enden, falls kein Anweisungsblock mit geschweiften Klammern folgt. Interpreter (und Compiler auch!) sind so pedantisch, weil sie anders das Programm nicht übersetzen können.

Eine Besonderheit bzw. Abweichung von der ursprünglichen Formulierung unseres Algorithmus ist die Überprüfung der Gültigkeit des eingegebenen Dividenden in den Zeilen 8 bis 13: Ein einfacher Befehl in der if-Anweisung in Zeile 11, zu Zeile 10 zurückzugehen, würde ja auch genügen. Wir müssen aber eine Schleife programmieren, weil Javascript keine reinen Sprungbefehle zur Verfügung stellt. Wie du siehst, braucht man sie auch nicht unbedingt, so lange man Schleifen und Unterprogramme (Funktionen) hat.

In Zeile 13 prüfen wir den Dividenden darauf, ob es sich um eine ganze Zahl handelt. Weil Javascript den Befehl zur Berechnung des Rests einer Division mit dem Operator „%“¹² hat, können wir das ganz einfach bewerkstelligen, indem wir probieren, ob die Division durch 1 einen Rest ergibt. Falls der Rest Null ist, ist der Dividend eine ganze Zahl.

Gäbe es diesen Befehl nicht müssten wir uns etwas anderes einfallen lassen: In der Programmiersprache Basic gibt es zum Beispiel den Befehl „INT“, was für integer steht und „ganzzahlig“ bedeutet. Dieser Befehl berechnet den ganzzahligen Anteil einer Zahl. Zum Beispiel ist $\text{INT}(5,3) = 5$

¹² Das ist eine etwas unglückliche Zeichenwahl! Sie hat nichts mit Prozentrechnung zu tun!
In Javascript ist $5 \% 3 = 2$, weil 5 geteilt durch 3 das Ergebnis 1 Rest 2 hat.

Wir müssten die Prüfung dann so durchführen: $\text{dividend} = \text{INT}(\text{dividend})$?

Das Beispiel zeigt sehr schön, dass man sich bei der Implementierung eines Algorithmus immer die Werkzeuge (Befehle) zu Nutze machen muss, die die jeweilige Programmiersprache bereitstellt.

Zeile 18 zeigt noch einmal sehr schön den Unterschied zwischen dem *Vergleichsoperator*, der prüft, ob zwei Zahlen gleich sind (in Javascript „==“) und dem Zuweisungsoperator (in Javascript „=“), der einer Variablen einen Wert zuweist (wie zum Beispiel in Zeile 10 oder 14).

Eigentlich sieht das Programm ja schon ganz gut aus. Wenn man ein wenig damit herumspielt, findet man aber ein paar Dinge, die man noch verbessern könnte:

Die Aufzählungsliste endet mit einem Komma. Das ist unschön, weil es so aussieht, als sei die Liste unvollständig. Das könnte man beheben, indem man das Programm mitzählen lässt, wie viele Teiler es gefunden hat. Bei allen Teilern außer beim ersten, schreibt man dann zuerst das Komma und dann den Teiler.

Mitzählen der Teiler hilft auch gegen folgendes kleines Problem: Wenn man Primzahlen testen lässt, steht erst verheißungsvoll da: „Die Teiler von sind:“ und dann kommt gar nichts mehr! Dem könnte man abhelfen, indem man zuerst die Teiler ausgibt, und dann nur wenn es Teiler gibt schreibt: „sind alle Teiler von ...“

Findet man keine Teiler schreibt man: „... ist eine Primzahl!“ (Statt der Punkte kommt natürlich immer der Dividend hin!)

Außerdem sollte man verhindern, dass zweimal der selbe Teiler ausgegeben wird. Das lässt sich einfach bewerkstelligen, indem man vor der Ausgabe von `partner` in Zeile 22 prüft, ob `partner = divisor` ist. Falls ja gibt man `Partner` nicht aus; stattdessen könnte man den Hinweis ausgeben, dass der Dividend eine Quadratzahl des letzten Teilers ist. Beachte: Falls du die Anzahl der gefundenen Teiler mitzählst, darfst du `divisor` und `partner` in diesem Fall nur einmal zählen.

Schön wäre doch auch, wenn das Ausgabefenster nach dem Durchlaufen des Programms einen Knopf hätte, mit dessen Hilfe man das Programm erneut starten könnte. Das lässt sich ganz einfach wie folgt realisieren: Die Methode `document.write()` kann ja alles Mögliche in ein Fenster schreiben – auch `html`-Formulare! So brauchen wir zwischen Zeile 26 und 27 nur eine Zeile einfügen, die das gleiche in das Ausgabefenster schreibt wie in Zeile 35 bis 39 steht.

Am aufwändigsten zu lösen ist das Problem, dass die Teiler unsortiert ausgegeben werden. Um das zu ändern gibt es grundsätzlich zwei verschiedene Möglichkeiten:

Entweder wir machen es so wie in unserem ersten Algorithmus, dass wir einfach *alle* Zahlen zwischen 1 und dem Dividenden als Teiler ausprobieren. Das ist von der Programmierung her am einfachsten, aber irgendwie unelegant, weil man ja mehr rechnen lässt als man muss – was bei heutigen Rechenleistungen aber nicht mehr wirklich eine Rolle spielt.

Oder wir speichern erst mal alle gefundenen Teiler und geben sie dann sortiert aus. Das Problem daran ist, dass wir nicht im Vorhinein wissen, wie viele Teiler wir finden werden! Wir müssen aber zu Beginn des Programms die Variable (= den Speicherplatz) reservieren, den wir dafür brauchen.

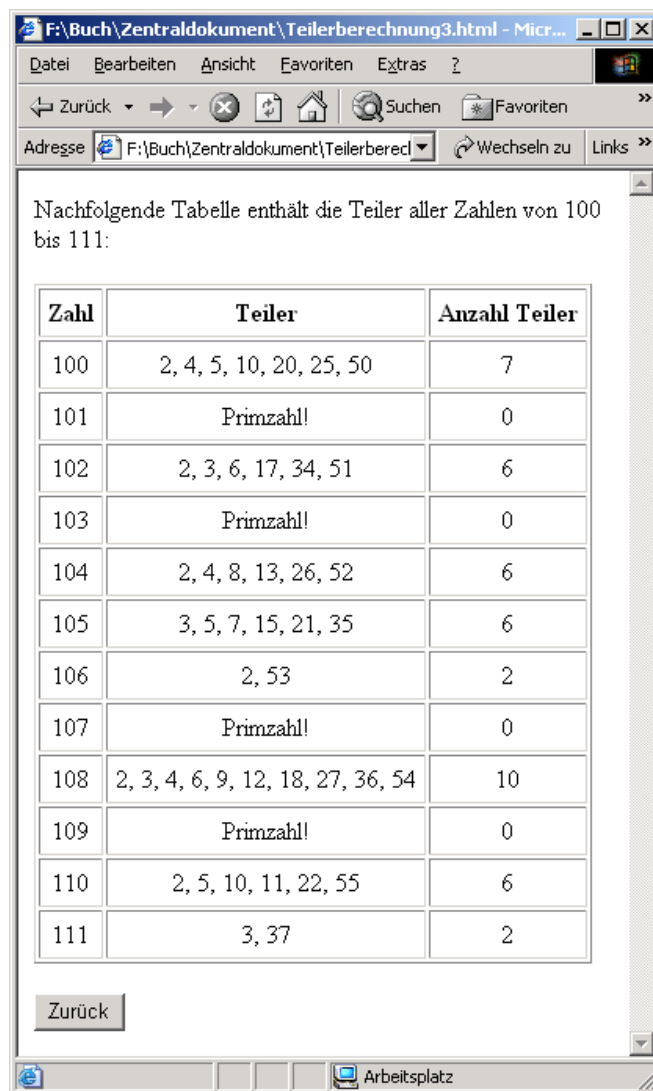
Eine letzte schöne Modifikation des Programms wäre, wenn wir nicht nur eine Zahl eingeben könnten, deren Teiler berechnet werden, sondern einen ganzen Bereich. Das ist wieder ziemlich einfach! Wir müssen nur statt einer Zahl „dividend“ zwei Zahlen eingeben lassen (den kleinsten und den größten Dividenden) und um den Programmbereich Zeile 14 bis 27 herum eine Schleife programmieren, die der Variablen dividend der Reihe nach die entsprechenden Werte zuweist.

Eine Reihe dieser Verbesserungen sind in dem Programm in Tabelle 22 realisiert. Es soll dir nur zeigen, was man machen kann, und dir Geschmack machen, dich weiter mit Javascript oder einer anderen Programmiersprache zu beschäftigen. Dieses Thema eingehend zu behandeln, würde den Rahmen dieses Buches sprengen! Deshalb stehen auch nur an den wesentlichen neuen Stellen Erläuterungen (außerdem fehlt der html-Code drumherum).

<code><script type="text/javascript"></code>	
<code>function teiler()</code>	
<code>{</code>	
<code>var dividend, divisor, teilerzahl = 0, min, max;</code>	
do	Neue Eingabeschleife
<code>{</code>	
<code>do</code>	
<code>{</code>	
<code>min = window.prompt("Gib den kleinsten Dividenden ein:", "4");</code>	Eingabe des kleinsten Dividenden
<code>min = parseInt(min);</code>	Aus Texteingabe eine Zahl machen
<code>if (min <= 3 min % 1 != 0) alert("Dividenden müssen ganzzahlig und größer als 3 sein!\nVersuche es noch einmal!");</code>	
<code>}</code>	
<code>while (min <= 3 min % 1 != 0);</code>	
<code>do</code>	
<code>{</code>	
<code>max = window.prompt("Gib den größten Dividenden ein:", "50");</code>	Eingabe des größten Dividenden
<code>max=parseInt(max);</code>	Aus Texteingabe eine Zahl machen

<pre>if (max <= 3 max % 1 != 0) alert("Dividenden müssen ganzzahlig und größer als 3 sein!\nVersuche es noch einmal!"); }</pre>	
<pre>while (max <= 3 max % 1 != 0); if (min >= max) alert("Der kleinste Dividend muss kleiner sein als der größte!\nVersuche es noch einmal!"); }</pre>	Hier wird getestet, ob die Eingabe plausibel ist und der Anfangswert kleiner ist als der Endwert.
<pre>while (min >= max); document.write("<p>Nachfolgende Tabelle enthält die Teiler aller Zahlen von " + min + " bis " + max + ":\</p>"); document.write("<table border=\<1\< cellpad- ding=\<5\< <tr><td a- lign=\<center\< Zahl<\<\</td><td a- lign=\<center\< Teiler<\<\</td><td a- lign=\<center\< Anzahl Tei- ler<\<\</td><\</tr><tr>");</pre>	Tabelle und ihre Überschriftenzeile erzeugen
<pre>dividend = min; while (dividend <= max)</pre>	Schleife für alle Dividenden
<pre>{ document.write("<td align=\<center\< >" + divi- dend + "<\</td><td align=\<center\< >"); divisor = 2; do</pre>	Dividend in erste Spalte schreiben
<pre>{ if (dividend % divisor == 0)</pre>	
<pre>{ teilerzahl++; if (teilerzahl > 1) document.write(", ");</pre>	Teiler zählen Komma erst ab dem zweiten gefundenen Teiler
<pre>document.write(divisor); } divisor++; }</pre>	
<pre>while (divisor < dividend); if (teilerzahl == 0) docu- ment.write("Primzahl!\<\</td>");</pre>	Kennzeichnung einer Primzahl
<pre>if (teilerzahl >= 1) document.write("<\</td>");</pre>	Spalte der Teiler schließen
<pre>document.write("<td align=\<center\< >" + teiler- zahl + "<\</td><\</tr>"); teilerzahl = 0; dividend++; }</pre>	Teilerzahl in dritte Spalte schreiben
<pre>document.write("<\</table>"); document.write("<form><p><input type=\<button\< value=\<Zurück\< onC- lick=\<history.back()\< ><\</p><\</form>"); }</pre>	Tabelle beenden Knopf um zur Startseite zurückzukehren.
<pre></script></pre>	

Tabelle 22: *Verbessertes Programm zur Teilerberechnung*



Dieses Programm erzeugt eine Ausgabe wie in links stehender Abbildung 95 zu sehen.

Du siehst wie viele Gedanken man sich schon um so ein kleines Programm machen kann! Deiner Kreativität sind keine Grenzen gesetzt!

Programmieren hat etwas von Spielen mit Bauklötzen: Aus den vorgegebenen Befehlen, die die Programmiersprache bietet, einen Programmablauf zusammenbauen, der das tut, was man gerne möchte, beziehungsweise was man sich zuvor als Algorithmus überlegt hat! Wenn man sich einmal ein wenig eingefuchst hat, macht es einen Riesenspaß, wenn der Computer nachher wirklich „gehört“!

Abbildung 95: Komfortablere Teilerberechnung

10 Quiz

In diesem Abschnitt stehen ein paar Fragen zum Inhalt des Buches.

Du kannst sie wie in einem Quiz entweder für dich alleine oder im Wettbewerb mit jemandem, der dieses Buch auch gelesen hat (oder sich sowieso mit Computern auskennt) beantworten und so ganz nebenbei festzustellen, ob du alles verstanden hast, was du in diesem Buch gelesen hast.

Es ist immer genau eine von vier Antwortmöglichkeiten richtig.

Die richtigen Lösungen findest du auf Seite 247! Hinter jeder Lösung steht auch ein Hinweis, an welcher Stelle des Buches du nachlesen kannst, falls dir die richtige Antwort nicht klar ist.

10.1 Fragen

- 1) **Woher „weiß“ ein Computer, was er tun soll und warum eignen sich Computer für so viele verschiedene Aufgaben?**
- a) Von den Befehlen in der CPU. Es gibt davon so viele, dass es für jede Aufgabe einen passenden gibt.
 - b) Aus dem Programm das er ausführt. Programme lassen sich individuell für jede Aufgabe schreiben.
 - c) Durch die Eingaben des Bedieners. Die Tastatur erlaubt in Verbindung mit der Maus jede beliebige Eingabe, die eine Aufgabe erfordern könnte.
 - d) Aus dem BIOS. Darin stehen für eine Vielzahl von Anwendungen alle notwendigen Befehle für den Computer.

Du kannst diese Frage beantworten, wenn du Seite 10 gelesen hast!

- 2) **Was muss man tun, um schwierige Aufgaben von einer dummen aber schnellen Maschine wie dem Computer lösen zu lassen?**
- a) Den Arbeitsspeicher des Computers möglichst groß machen.
 - b) Die Taktfrequenz des Computers möglichst groß machen.
 - c) Die Aufgabe in viele kleine einfache Teilaufgaben zerlegen, die sich mit Standardbefehlen lösen lassen.
 - d) Einen möglichst intelligenten Bediener an den Rechner setzen.
- Du kannst diese Frage beantworten, wenn du Seite 10 gelesen hast!*

3) Was ist eine Ziffer?

- a) Eine Ziffer ist ein Zeichen mit dem Zahlen dargestellt werden.
- b) Eine Ziffer ist das selbe wie eine Zahl.
- c) Ziffern sind die Zahlen auf dem Zifferblatt einer Uhr.
- d) Eine Ziffer ist der Code zum Speichern von Text.

Du kannst diese Frage beantworten, wenn du Seite 12 gelesen hast!

4) Alle Ziffern des Binär-, Dezimal- und Hexadezimalsystems sind...?

- a) 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- b) 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
- c) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- d) 0000, 0001, 0010, 0011, 0100, ..., 1110, 1111

Du kannst diese Frage beantworten, wenn du Kapitel 3.3 gelesen hast!

5) Was ist ein Bit?

- a) Eine Altbiersorte.
- b) Die Zusammenfassung von acht Byte.
- c) Ein Taktzyklus der Taktfrequenz der CPU.
- d) Die Einheit für die kleinstmögliche Informationsmenge, nämlich die Unterscheidung zwischen zwei Zuständen.

Du kannst diese Frage beantworten, wenn du Seite 13 gelesen hast!

6) Was ist ein Byte?

- a) Die Maßeinheit für die Taktfrequenz der CPU.
- b) Die Zusammenfassung von acht Bit.
- c) Der englische Begriff für die Laufwerksbuchstaben A:\, B:\, C:\ usw.
- d) Der Arbeitsspeicher des Computers.

Du kannst diese Frage beantworten, wenn du Seite 23 gelesen hast!

7) Wie viele verschiedene Zahlen kann man mit 4 Bit darstellen?

- a) 4
- b) 8
- c) 15
- d) 16
- e) 9999

Du kannst diese Frage beantworten, wenn du Kapitel 3.2 gelesen hast!

8) Was ist ein *Code*?

- a) Eine Zuordnung von Zeichen eines Zeichenvorrats zu den Zeichen eines anderen Zeichenvorrats.
- b) Eine Geheimtinte.
- c) Ein Geheimwort, mit dem die Zugangsberechtigung beim Starten des Computers geprüft wird.
- d) Eine Folge von Nullen und Einsen in einer Datei.

Du kannst diese Frage beantworten, wenn du Seite 26 gelesen hast!

9) Wie wird Text in einer Computerdatei dargestellt?

- a) Ganz normal mit den Zeichen A-Z, a-z und den Sonderzeichen.
- b) Durch den dritten Zustand eines Bits, den man Textebene nennt.
- c) Durch Programme.
- d) Durch die Zuordnung bestimmter Bitfolgen zu den Textzeichen; diese Zuordnung heißt ASCII-Code.

Du kannst diese Frage beantworten, wenn du Kapitel 3.4 gelesen hast!

10) Wie kann ein Bild in einer Computerdatei dargestellt werden?

- a) Durch Unterteilung in Bildpunkte, für die jeweils eine Farbinformation in Form einer Bitfolge gespeichert wird (sogenannte Bitmap).
- b) Durch analoge Speicherung auf der Festplatte.
- c) Durch Speicherung der Form jedes Pixels.
- d) Durch Speicherung im Verzeichnis „Eigene Bilder“.

Du kannst diese Frage beantworten, wenn du Kapitel 3.5 gelesen hast!

11) Was ist eine *Vektorgrafik*?

- a) Ein Grafikformat, bei dem geometrische Formen statt Bildpunkte binär codiert werden, um verlustfreie Vergrößerungen zu ermöglichen.
- b) Eine Grafik, die von der Firma Vektor erstellt wurde.
- c) Das Bild eines Pfeils.
- d) Eine von Hugo Vektor gezeichnete Grafik.

Du kannst diese Frage beantworten, wenn du Seite 36 gelesen hast!

12) Was ist ein *Programm*?

- a) Das deutsche Wort für den englischen Begriff *Hardware*.
- b) Ein kleines Bildchen, das als Symbol für bestimmte Funktionen am oberen Rand eines Fensters erscheint.
- c) Ein im Voraus festgelegter Ablauf. Bei Computern insbesondere die festgeschriebene Abfolge von Befehlen, die der Computer ausführen soll.
- d) Ein Eintrag im Menü Start.

Du kannst diese Frage beantworten, wenn du Seite 41 gelesen hast!

13) Was versteht man unter dem Begriff *digital*?

- a) Uhren mit Quarzlaufwerk nennt man digital.
- b) Digital bedeutet, dass ein Wert nur in bestimmten Stufen dargestellt werden kann.
- c) Digital bedeutet, dass eine Zahl nur die Werte Null oder Eins annehmen kann.
- d) Digital bedeutet, dass keine Benutzereingaben erforderlich sind.

Du kannst diese Frage beantworten, wenn du Seite 45 gelesen hast!

14) Wie heißen die elektronischen Schalter, die in Computern die Darstellung von Nullen und Einsen im Arbeitsspeicher und in der CPU übernehmen?

- a) Transformator
- b) Taster
- c) Transistor
- d) Traktor

Du kannst diese Frage beantworten, wenn du Seite 47 gelesen hast!

15) Was versteht man unter der *Taktfrequenz* eines Computers?

- a) Die Taktfrequenz bestimmt, wie schnell Daten von der Festplatte gelesen werden können.
- b) Die Taktfrequenz bestimmt die Geschwindigkeit, mit der der Computer die Befehle eines Programms abarbeitet.
- c) Die Taktfrequenz ist die Häufigkeit, mit der ein Computer höchstens abstürzen darf.
- d) Die Taktfrequenz ist die höchste Frequenz, die mit den Computerausgängen erzeugt werden kann.

Du kannst diese Frage beantworten, wenn du Seite 48 gelesen hast!

-
- 16) Was bezeichnet man mit der Abkürzung *RAM*?
- a) Das **R**aster **A**nalyse **M**odul, mit dem die Anzahl Bildpunkte des Bildschirms an den Computer übertragen werden.
 - b) Das **R**isk **A**ssessment **M**apping, das vor dem Testen von Programmen das Risiko eines Systemabsturzes beurteilt.
 - c) Die drei Farben **R**ot, **A**zur, **M**alve, aus denen alle Farben der Bildschirmdarstellung gemischt werden.
 - d) Den Arbeitsspeicher, englisch: **R**andom **A**ccess **M**emory
- Du kannst diese Frage beantworten, wenn du Seite 50 gelesen hast!*
- 17) Was kann sich alles im Arbeitsspeicher eines Computers befinden?
- a) Wenn der Computer ausgeschaltet wird, werden dort die unerledigten Arbeiten gespeichert.
 - b) Bei eingeschaltetem Computer das Betriebssystem, Programme und Daten.
 - c) Nur die Daten, an denen der Computer gerade arbeitet.
 - d) Alle installierten Programme.
- Du kannst diese Frage beantworten, wenn du Kapitel 4.3 gelesen hast!*
- 18) Wie heißt das wichtigste Bauteil des Computers, in dem die eigentliche Rechenarbeit erledigt wird?
- a) Prozessor oder CPU
 - b) Computerhirn
 - c) Arbeitsspeicher
 - d) Festplatte
- Du kannst diese Frage beantworten, wenn du Kapitel 4.4 gelesen hast!*
- 19) Wie schreibt man die hexadezimale Zahl 2Ch binär und dezimal?
- a) binär: 00101100, dezimal: 44
 - b) binär: 00101100, dezimal: 212
 - c) binär: 00010110, dezimal: 44
 - d) binär: 00010110, dezimal: 32
- Du kannst diese Frage beantworten, wenn du Seite 54 gelesen hast!*
-

20) Wofür braucht ein Computer das *BIOS*?

- a) Das BIOS regelt die Spannungsversorgung des Netzteils, so dass die empfindlichen Transistoren der CPU und des Arbeitsspeichers immer mit exakt 5 Volt Gleichspannung versorgt werden.
- b) Das BIOS gleicht Farbdifferenzen zwischen Bildschirm, Drucker und Scanner aus.
- c) Im BIOS stehen alle Informationen, die der Computer beim Starten als erstes braucht, noch bevor er ein Betriebssystem laden kann.
- d) Das BIOS speichert alle Fehlermeldungen, die während des Betriebs auftreten, damit man sie später nachvollziehen kann.

Du kannst diese Frage beantworten, wenn du Kapitel 6.1 gelesen hast!

21) Wie viel Daten kann man ungefähr auf einer CD speichern?

- a) 200 Textseiten.
- b) 512 Kilobyte
- c) 700 Megabyte
- d) 50 Gigabyte

Du kannst diese Frage beantworten, wenn du Seite 63 gelesen hast!

22) Welche Bedingung müssen Namen für Ordner und Dateien erfüllen?

- a) Es dürfen keine Ziffern darin enthalten sein.
- b) Eindeutigkeit: Ein Name darf innerhalb eines Verzeichnisses nur einmal vorkommen.
- c) Dateinamen dürfen nur klein geschrieben werden, Ordnernamen nur groß!
- d) Länge: Ein Name darf höchstens acht Zeichen lang sein plus weitere drei Zeichen durch einen Punkt getrennt.

Du kannst diese Frage beantworten, wenn du Seite 67 gelesen hast!

23) Was ist eine *Schnittstelle*?

- a) Die Unterteilung des Explorerfensters, links Verzeichnisstruktur, rechts Ordnerinhalt nennt man Schnittstelle.
- b) An einer Schnittstelle lassen sich verschiedene Komponenten eines Systems (z.B. eines Computers) voneinander trennen. Also ist z.B. eine Steckerverbindung eine Schnittstelle.
- c) Die Programmiersprache, in der ein Programm geschrieben ist, nennt man auch Schnittstelle.
- d) Die Stelle, wo die Gehäuseteile eines Rechners aufeinander treffen und wieder getrennt werden können, nennt man Schnittstelle.

Du kannst diese Frage beantworten, wenn du Seite 73 gelesen hast!

24) Wofür braucht man die *Umschalttaste*?

- a) Man schaltet damit zwischen den verschiedenen Stromversorgungen um, damit man einen Computer auch im Ausland benutzen kann.
- b) Man schaltet damit das Betriebssystem zwischen englischer und deutscher Benutzeroberfläche um.
- c) Die Umschalttaste braucht man, um zwischen der Eingabe per Maus oder per Tastatur umzuschalten.
- d) Damit werden die verschiedenen Bedeutungen einer Taste der Tastatur umgeschaltet, also zum Beispiel große und kleine Buchstaben.

Du kannst diese Frage beantworten, wenn du Seite 86 gelesen hast!

25) Was versteht man unter *WYSIWYG*?

- a) Das ist eine Druckertechnologie, die besonders schnell drucken kann.
- b) So werden Bildschirme genannt, deren Längen- zu Breitenverhältnis 16:9 statt 4:3 beträgt.
- c) Das ist die Abkürzung von „What you see is what you get“ und bedeutet, dass die Darstellung auf dem Bildschirm genau so aussieht, wie auf dem Drucker.
- d) Das ist die Abkürzung für „What you send is what you get“ und bezeichnet ein Verfahren zum Testen von Datenleitungen, bei dem eine Nachricht einmal hin und wieder zurück gesendet wird.

Du kannst diese Frage beantworten, wenn du Seite 109 gelesen hast!

26) Was versteht man unter dem Begriff *booten*?

- a) Den Startvorgang des Computers.
- b) Den Einsatz von Computern auf Wasserfahrzeugen.
- c) Den Computer vor unerlaubtem Zugriff schützen, zum Beispiel durch die Einrichtung von Passwörtern.
- d) Das Reinigen der beweglichen Teile der Maus (Kugel, Andruckrolle, Aufnehmerachsen und Gleitflächen).

Du kannst diese Frage beantworten, wenn du Seite 115 gelesen hast!

27) Welche Aussage trifft auf ein *Betriebssystem* zu?

- a) Es gibt nur Windows-Betriebssysteme (Windows 95, Windows 98, Windows Me, Windows 2000, Windows XP).
- b) Das Betriebssystem steht im BIOS.
- c) Das Betriebssystem ist ein Baustein auf dem Mainbord.
- d) Ein Betriebssystem ist ein Programm, das den Computer erst nutzbar macht, weil es seine Ressourcen verwaltet.

Du kannst diese Frage beantworten, wenn du Seite 118 gelesen hast!

28) Was ist eine *Verknüpfung*?

- a) Der Eintrag in der FAT (File Allocation Table), der dem Computer sagt, wo auf der Festplatte eine bestimmte Datei steht.
- b) Eine Datei, in der Informationen darüber stehen, wo sich eine andere Datei (z.B. ein Programm) befindet. Verknüpfungen kommen vor allem im Startmenü und auf dem Desktop vor.
- c) Eine Verbindung zwischen 2 Computern über ein Netzkabel.
- d) Jede Kabelverbindung zwischen dem Computer und einem seiner Peripheriegeräte (Drucker, Scanner usw.).

Du kannst diese Frage beantworten, wenn du Seite 120 gelesen hast!

29) Was versteht man unter einem *Multitasking-System*?

- a) Ein Computer, an dem mehrere Benutzer gleichzeitig (an verschiedenen Bildschirmen und Tastaturen) arbeiten können.
- b) Ein Kombi-Gerät, das sowohl Drucker als auch Scanner als auch Faxgerät in einem Gehäuse vereint.
- c) Ein Betriebssystem, das mehrere Programme im Arbeitsspeicher verwalten und quasi-gleichzeitig ausführen kann.
- d) Die Methode mit zehn Fingern gleichzeitig zu tippen.

Du kannst diese Frage beantworten, wenn du Seite 131 gelesen hast!

30) Was bedeutet *virtueller Arbeitsspeicher*?

- a) Arbeitsspeicher-Bausteine der Firma „Virtu“
- b) Arbeitsspeicher, den man gerne hätte, aber nicht hat.
- c) Speicherbereiche auf der Festplatte, in die gerade nicht benutzte Inhalte des Arbeitsspeichers geschrieben werden, bis sie wieder gebraucht werden.
- d) Arbeitsspeicher mit Schreib- und Lesezeiten unter 10ns.

Du kannst diese Frage beantworten, wenn du Seite 131 gelesen hast!

31) Was ist das *Kontextmenü*?

- a) Das Kontextmenü öffnet sich, wenn man auf den Start-Knopf drückt oder die Windows-Taste betätigt. Darüber kann man dann alle installierten Programme und die Systemsteuerung aufrufen.
- b) Das Kontextmenü gibt es für alle externen Geräte, die Text ausgeben können, wie z.B. Drucker, Bildschirme, usw.
- c) Das Kontextmenü erscheint, wenn man etwas mit der rechten Maustaste anklickt und enthält die häufigsten Befehle, die im Zusammenhang (Kontext) mit dem Angeklickten gebraucht werden.
- d) Das Kontextmenü erscheint automatisch, wenn eine Eingabe oder ein Befehl nicht eindeutig war, und der Computer weitere Informationen benötigt, welche Aktion er ausführen soll.

Du kannst diese Frage beantworten, wenn du Seite 131 gelesen hast!

32) Was ist die *Zwischenablage*?

- a) Ein Verzeichnis im Windows-Ordner, wo temporäre Dateien zwischengespeichert werden.
- b) Die deutsche Bezeichnung für den Desktop, auf dem man Verknüpfungen ablegen kann, die Angaben zum Speicherort von Programmen enthalten.
- c) Ein Bereich der Festplatte, wo man Textbausteine mit Hilfe des Befehls *Bearbeiten/Kopieren* zwischenspeichern kann, um sie in dieser Datei an einer anderen Stelle mit Hilfe des Befehls *Bearbeiten/Einfügen* wieder einzufügen.
- d) Ein Bereich des Arbeitsspeichers, in dem man beliebige Daten mit Hilfe der Befehle *Bearbeiten/Kopieren* und *Bearbeiten/Ausschneiden* zwischenspeichern kann, um sie später an anderer Stelle (auch in anderen Programmen!) mit Hilfe des Befehls *Bearbeiten/Einfügen* wieder einzufügen.

Du kannst diese Frage beantworten, wenn du Kapitel 6.2.3 gelesen hast!

33) Was kannst du mit dem Programm *Explorer* nicht machen?

- a) Dateien löschen, verschieben, kopieren und umbenennen.
- b) Neue Verzeichnisse anlegen.
- c) Den Inhalt von Dateien bearbeiten.
- d) Nach Dateien suchen, deren Speicherort du vergessen hast.

Du kannst diese Frage beantworten, wenn du Kapitel 6.2.5 gelesen hast!

34) Wozu braucht man den *Papierkorb*?

- a) Um versehentlich gelöschte Dateien und Ordner wiederherstellen zu können.
- b) Um dort dauerhaft Dokumente abzulegen.
- c) Um die Speicherkapazität der Festplatte zu erhöhen.
- d) Der Papierkorb ist ein Computerspiel, das mit Windows mitgeliefert wird.

Du kannst diese Frage beantworten, wenn du Seite 153 gelesen hast!

35) Welche Möglichkeit gibt es, Hilfetexte zu Windows und anderen Programmen zu erhalten?

- a) Menü *Datei / Hilfe ...*
- b) Hersteller anrufen.
- c) Taste [F1] drücken.
- d) Menü *Ansicht / Hilfe...*

Du kannst diese Frage beantworten, wenn du Seite 156 gelesen hast!

36) Welche Voraussetzung(en) muss/müssen gegeben sein, damit du ein Programm auf deinem Computer ausführen kannst?

- a) Es muss auf eine CD gebrannt sein.
- b) Es muss installiert sein und in den Arbeitsspeicher geladen worden sein.
- c) Es muss mit der Bildschirmauflösung kompatibel sein.
- d) Es muss in die CPU geladen worden sein.

Du kannst diese Frage beantworten, wenn du Seite 159 gelesen hast!

-
- 37) Was sollte man beim Arbeiten mit Textverarbeitungsprogrammen beachten:
- a) Bei jedem Zeilenende die Eingabetaste betätigen.
 - b) Einzüge nicht mit Leerzeichen sondern mit Hilfe des Absatzformats einzustellen.
 - c) Formatierungen immer sofort bei Eingabe des Textes einzugeben.
 - d) Keine Dokumente von mehr als 10 Seiten Länge zu erstellen, weil sonst der Arbeitsspeicher voll werden kann.

Du kannst diese Frage beantworten, wenn du Seite 163 gelesen hast!

- 38) Wie setzt man sinnvollerweise die Satzzeichen?
- a) Gar nicht, das macht der Computer automatisch!
 - b) Mit jeweils einem Leerzeichen davor und einem dahinter:
...Wort , Wort...
 - c) Mit einem Leerzeichen davor:
...Wort ,Wort...
 - d) Mit einem Leerzeichen dahinter:
...Wort, Wort...

Du kannst diese Frage beantworten, wenn du Seite 159 gelesen hast!

- 39) Wofür gibt es in Microsoft Word *keine* separaten Formatierungsbefehle?
- a) Zeichen
 - b) Zeile
 - c) Absatz
 - d) Abschnitt

Du kannst diese Frage beantworten, wenn du Kapitel 7.3.1.2 gelesen hast!

- 40) Wofür braucht man eine Tabellenkalkulation?
- a) Um Daten übersichtlich darzustellen und Berechnungen damit durchzuführen.
 - b) Um schwierige mathematische Sachverhalte einfach darzustellen.
 - c) Um Datenbanken zu verwalten.
 - d) Um in einer Textverarbeitung auch Tabellen darstellen zu können.

Du kannst diese Frage beantworten, wenn du Seite 181 gelesen hast!

- 41) Woran erkennt Excel, dass es sich beim Inhalt einer Tabellenzelle nicht um Daten sondern um eine Formel handelt, die berechnet werden soll?
- a) Das wird mit der Zellenformatierung eingestellt (Menü *Format / Zellen...*)
 - b) Wenn der Inhalt *nicht* in Anführungszeichen steht.
 - c) Wenn der Inhalt mit einem Gleichheitszeichen beginnt.
 - d) Excel kann innerhalb einer Tabelle keine Berechnungen durchführen!

Du kannst diese Frage beantworten, wenn du Seite 184 gelesen hast!

- 42) Was ist das Besondere an einer *relationalen* Datenbank?
- a) Sie enthält nur Daten über Beziehungen, also Adressen, Telefonnummern usw.
 - b) Alle Daten stehen in einer Tabelle, so dass sie besonders leicht zu handhaben ist.
 - c) Jede Information steht nur *ein Mal* in der Datenbank. Das wird durch mehrer Tabellen erreicht, die untereinander in Beziehung gesetzt werden.
 - d) Alle Felder dieser Datenbank werden in separaten Dateien gespeichert, die untereinander in Beziehung gesetzt werden.

Du kannst diese Frage beantworten, wenn du Seite 192 gelesen hast!

- 43) Was ist das Internet?
- a) Die Verbindung aller Rechner einer Firma über Ethernet-Schnittstellen.
 - b) Weltweit über Datenleitungen miteinander verbundene Computer. Über Telefonleitungen kann man den eigenen Computer mit diesem Netzwerk verbinden.
 - c) Jede Verbindung zwischen zwei oder mehreren Rechnern.
 - d) Ein Datenprotokoll zwischen der CPU und dem Arbeitsspeicher.

Du kannst diese Frage beantworten, wenn du Kapitel 8 gelesen hast!

- 44) Was ist html und das Besondere daran?
- a) Html ist ein Dateiformat, in dem Internetseiten geschrieben sind. Es enthält den Text und alle Formatierungsbefehle als ASCII-Zeichen. Der wichtigste Befehl erlaubt das Erzeugen von Verknüpfungen (Links) mit beliebigen anderen Internetseiten.
 - b) Html wurde als Datenformat speziell für e-Mails entwickelt. Es erlaubt eine besonders schnelle Datenübertragung.

-
- c) Html ist das Datenprotokoll im Internet, mit dem die Internetseiten von Rechner zu Rechner übertragen werden. Die Besonderheit ist, dass die Daten in Pakete aufgeteilt werden und unterschiedliche Wege zum Empfänger nehmen können.
 - d) Html ist eine Programmiersprache, mit der Internetseiten kleine Programme auf den Rechnern der Besucher ausführen können. Die Programme werden als ASCII-Text übertragen und vom Browser des Besuchers interpretiert.

Du kannst diese Frage beantworten, wenn du Kapitel 8.1 gelesen hast!

45) **Mit welcher Adresseingabe wird dir eine Internetseite angezeigt?**

- a) `html://www.MiJan.de`
- b) `http://www-MiJan-de`
- c) `http://www@MiJan.de`
- d) `http://www.MiJan.de`

Du kannst diese Frage beantworten, wenn du Seite 200 gelesen hast!

46) **Was ist das Besondere an e-Mails?**

- a) Sie sind besonders zuverlässig; man kann immer sicher sein, dass sie den Empfänger auch erreichen.
- b) Sie sind besonders vertraulich: Man kann sicher sein, dass sie unterwegs nicht gelesen werden und dass der Empfänger sie nicht unbefugt weitergeben kann.
- c) Man weiß immer genau, von wem sie kommen! Im Gegensatz zu Unterschriften lassen sich e-Mails nicht fälschen.
- d) Sie sind billig, schnell und bequem.

Du kannst diese Frage beantworten, wenn du Kapitel 8.3 gelesen hast!

47) **Wie kannst du sicher sein, dass keine Viren, Würmer oder Trojaner auf deinen Rechner gelangen?**

- a) Wenn ich nur e-Mails von Absendern öffne, die ich kenne, kann nichts passieren!
- b) Ich prüfe alle Disketten sofort nachdem ich sie eingelegt habe auf verdächtige Dateien.
- c) Ich arbeite nicht am Computer, wenn ich krank bin, so dass er sich nicht anstecken kann.
- d) Ich installiere ein Anti-Viren- und möglichst auch ein Firewall-Programm auf meinem Rechner und halte sie immer aktuell.

Du kannst diese Frage beantworten, wenn du 8.4 gelesen hast!

48) Was ist der wichtigste Schritt beim Erstellen eines Programms?

- a) Das Testen des fertig implementierten Programms.
- b) Das Übersetzen (Compilieren).
- c) Das Erarbeiten und Beschreiben eines Algorithmus zur Lösung der Problemstellung.
- d) Die Auswahl der Programmiersprache.

Du kannst diese Frage beantworten, wenn du Seite 210 gelesen hast!

49) Welche Elemente braucht eine Programmiersprache *nicht*?

- a) Schleifen
- b) Hardware-Überwachung
- c) Bedingte Verzweigungen
- d) Ein-/Ausgabeoperationen

Du kannst diese Frage beantworten, wenn du Kapitel 9.2 gelesen hast!

50) Welche der folgenden Aussagen ist falsch?

- a) Der Algorithmus wird von einem Compiler oder Interpreter in Befehle übersetzt, die der Computer verstehen kann.
- b) Ein Compiler übersetzt Programmcode und erzeugt eine ausführbare Datei, so dass zur Laufzeit des Programms keine Übersetzung mehr erforderlich ist.
- c) Ein Interpreter übersetzt Programmcode zur Laufzeit des Programms, also jedes Mal aufs Neue, wenn das Programm ausgeführt wird.
- d) Der Browser Microsoft Internet Explorer beinhaltet einen Interpreter für die Programmiersprache Javascript.

Du kannst diese Frage beantworten, wenn du Seite 213 gelesen hast!

10.2 Lösungen

Aufgabe	Richtige Lösung	Nachzulesen
1)	b)	Kapitel 2
2)	c)	Kapitel 2
3)	a)	Seite 12
4)	c)	Seite 24
5)	d)	Seite 13
6)	b)	Seite 23
7)	d)	Seite 21
8)	a)	Seite 26
9)	d)	Seite 27
10)	a)	Kapitel 3.5 ab Seite 31
11)	a)	Seite 36
12)	c)	Seite 41
13)	b)	Seite 45
14)	c)	Kapitel 4.1
15)	b)	Kapitel 4.2
16)	d)	Seite 50
17)	b)	Kapitel 4.3 und 5.1
18)	a)	Kapitel 4.4 und 5.1
19)	a)	Seite 23
20)	c)	Seite 58
21)	c)	Kapitel 5.1.3.2
22)	b)	Seite 67
23)	b)	Kapitel 5.2

24)	d)	Seite 86
25)	c)	Seite 109
26)	a)	Kapitel 6.1
27)	d)	Kapitel 6.2
28)	b)	Kapitel 6.2.1.3
29)	c)	Seite 131
30)	c)	Seite 131
31)	c)	Kapitel 6.2.1.5
32)	d)	Kapitel 6.2.3
33)	c)	Kapitel 6.2.5
34)	a)	Kapitel 6.2.5.5
35)	c)	Kapitel 6.2.6
36)	b)	Kapitel 7.2
37)	b)	Kapitel 7.3
38)	d)	Kapitel 7.3.1.1
39)	b)	Kapitel 7.3.1.2
40)	a)	Kapitel 7.3.2
41)	c)	Kapitel 7.3.2.1
42)	c)	Seite 192
43)	b)	Kapitel 8
44)	a)	Kapitel 8.1
45)	d)	Kapitel 8.2.1
46)	d)	Kapitel 8.3
47)	d)	Kapitel 8.4
48)	c)	Kapitel 9.1
49)	b)	Kapitel 9.2
50)	a)	Kapitel 9.3

11 Glossar

In diesem Glossar werden Begriffe kurz erklärt, die im Text mit einem → Hinweisfeil versehen sind. Das soll dir ermöglichen, während des Lesens einen Begriff, der dir nicht klar ist, kurz nachschlagen zu können.

Willst du ausführlicher über einen Begriff nachlesen, verwende den Index ab Seite 255, der zu diesen und weiteren Schlagwörtern Seitenzahlen nennt, wo im Buch darüber geschrieben wird.

A

Adresse	Angabe eines Fundortes. In der Computertechnik kann dies eine Speicherzelle im Arbeitsspeicher sein, die Nummer eines Gerätes, das an einem → Bus arbeitet, oder die Bezeichnung (Name oder Nummer) eines Computers in einem Netzwerk.
Analog	bedeutet eigentlich: „gleich“ oder „entsprechend“; in der Technik ist damit gemeint, dass alle Werte (zwischen zwei Grenzen) angenommen werden können. Gegenteil: → Digital, diskret. Zeigerinstrumente arbeiten z.B. analog.
Arbeitsspeicher	Elektronischer Speicher in einem Computer, auf den der → Prozessor direkt zugreifen kann. Befehle eines → Programms, die der Computer ausführen soll, müssen im Arbeitsspeicher stehen. Ebenso die → Daten, die mit einem Programm bearbeitet oder von ihm benutzt werden sollen.
ASCII	Bezeichnung für die gängige Zuordnung von Buchstaben zu Bitmustern, die für das Speichern von Texten benutzt wird.

B

Betriebssystem	→ Programm, das den Betrieb eines Computers steuert, also zum Beispiel das Starten von Programmen erlaubt oder den Zugriff auf Laufwerke ermöglicht.
Binärsystem	Zahlensystem, in dem nur zwei Ziffern verwendet werden. Auch: Dualsystem.
BIOS	Programm, das beim Starten des Computers ausgeführt wird und in einem → ROM fest eingespeichert ist.
Bit	Kleinstmögliche Menge an Information. Die Unterscheidung zwischen zwei Zuständen.

Bitmap	Format zur → digitalen Darstellung von Bildern: Das Bild wird in viele kleine Punkte unterteilt. Jedem Punkt wird ein seiner Farbe entsprechendes Bitmuster zugeordnet. Diese Bitmuster werden gespeichert.
Browser	Programm zum Betrachten von Internetseiten. Ein Browser interpretiert html-Code und stellt die Dateien dementsprechend dar.
Bus	Datenleitung. Unterscheidung zwischen parallelem und seriellen Bus: Bei einem seriellen Bus (z.B. USB, RS-232, Firewire) werden alle Daten nacheinander auf einer Leitung übertragen. Bei einem parallelen Bus werden auf mehreren (z.B. acht) Datenleitungen Daten gleichzeitig und synchron übertragen.
Byte	Zusammenfassung von 8 → Bit. → Arbeitsspeicher wird byteweise adressiert, d.h. es werden immer acht Bit gleichzeitig gelesen oder geschrieben.

C

Code	Zuordnungsvorschrift. Z.B. beim → ASCII-Code werden den Buchstaben Bitmuster zugeordnet, um Texte mit Computern bearbeiten zu können.
CPU	→ Prozessor

D

Datei	Zusammenfassung von zusammengehörenden Daten. Dateien werden mit einem eindeutigen Dateinamen identifiziert.
Dateityp	→ Dateien können unterschiedliche Informationen in unterschiedlicher Anordnung (Dateiformat) enthalten. Der Dateityp legt fest, welche Art von Daten in welchem Format in einer Datei enthalten sind. Meist ist mit einem Dateityp eine Anwendung verknüpft. Microsoft Word speichert beispielsweise Text in einer ganz bestimmten Art und Weise. Im Dateinamen wird der Dateityp durch die letzten Buchstaben nach einem Punkt angegeben.
Dezimalsystem	Zahlensystem, das zehn Zeichen verwendet.
Digital	Gegenteil von → analog: Ein Wert kann nur ganz bestimmte Größen annehmen, die durch die Verwendung von → Ziffern (engl.: digit) und einer bestimmten Anzahl Stellen definiert

	sind.
Digitalkamera	Fotoapparat oder Videokamera, die Bilder/Filme → digital erfasst und speichert.
Directory	→ Verzeichnis
Dualsystem	→ Binärsystem

G

Grafikkarte	Elektronische Platine, die als Erweiterungskarte in einem Computer das vom → Prozessor errechnete Bild in elektrische Signale Umwandelt, die ein Monitor darstellen kann.
-------------	---

H

Hardware	Alle gegenständlichen Teile eines Computers; alles was man im Gegensatz zur → Software anfassen kann.
Hauptplatine	Kernstück eines Computers, worauf sich → Prozessor, → Arbeitsspeicher und Erweiterungsplätze befinden.
Hertz	Einheit der Häufigkeit. Abkürzung: Hz 1 Hz bedeutet ein Mal pro Sekunde.
Hexadezimalsystem	Zahlensystem, das 16 Ziffern verwendet: 0,1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

I

Installieren	Vorbereiten der Benutzung einer → Software auf einem Computer. Dazu zählt z.B. Speichern der verschiedenen → Dateien auf einem Rechner und Erzeugen von Startmenüeinträgen.
Internetbrowser	→ Browser

K

Kilo	Abkürzung: „k“. Steht bei Einheiten für „Tausend“: 1 kHz = 1.000 Hz, 1kByte = 1.000 Byte
Kontextmenü	Menü, dessen Einträge davon abhängen, in welchem Zusammenhang es aufgerufen wurde (was angeklickt wurde). Aufrufen mit rechter Maustaste oder Kontextmenütaste der

Tastatur.

L

Laden	Kopieren von Daten und / oder Programmen von einem Datenträger (Festplatte, CD, ...) in den → Arbeitsspeicher.
Laufwerk	Gerät, das einen Datenträger enthält oder aufnehmen kann, z.B. Festplatte, CD, DVD, Magnetband usw.

M

Mainboard	→ Hauptplatine
Mega	Abkürzung: „M“. Steht bei Einheiten für „Million“: 1 MHz = 1.000.000 Hz, 1MByte = 1.000.000 Byte

N

Netzteil	Gerät mit dessen Hilfe man den Computer ans Stromnetz anschließt. Es wandelt den Strom aus der Steckdose (230V, 50Hz Wechselspannung) in die Spannungen um, die im Computer benötigt werden (z.B. 5V Gleichspannung).
Netzwerk	Verbindung zwischen Computern zum Datenaustausch.
Nibble	4 Bit, ein halbes Byte

O

Ordner	→ Verzeichnis
--------	---------------

P

Palette	Gesamtheit der verwendeten Farben und ihre Zuordnung zu einem Bitmuster.
Pfad	Vollständige Angabe, wo eine Datei zu finden ist: Enthält Laufwerk, ggf. Verzeichnis, ggf. Unterverzeichnis und Dateinamen; in Netzwerken vor dem Laufwerk auch noch den Computernamen.
Programm	Abfolge von Befehlen, die der Computer ausführt. Sammel-

begriff „Software“. Programme werden in ausführbaren Dateien gespeichert und müssen in den Arbeitsspeicher geladen werden, um ausgeführt werden zu können.

Prozessor Herzstück eines Computers. Hier werden die Befehle eines Programms ausgeführt und die Daten verarbeitet.

R

RAM → Arbeitsspeicher

Register Speicherzellen, die direkt in der → CPU sitzen. Die Befehle, die die CPU ausführen kann, greifen für die Ein- und Ausgabe auf die Register zu.

Ressourcen Sammelbegriff für alle Geräte und Hilfsmittel, die dem Computer im Betrieb zur Verfügung stehen: Arbeitsspeicher, Festplatte, Drucker, usw.

ROM Speicher der nur gelesen und nicht beschrieben werden kann. Zum Beispiel zum Speichern des → BIOS.

S

Software Sammelbegriff für → Programme. Im Gegensatz zur → Hardware lässt sich Software leicht verändern und während des Betriebs des Computers wechseln. Man kann z.B. zuerst Textverarbeitung und dann Tabellenkalkulation auf der selben Hardware machen.

Soundkarte Erweiterungskarte, die aus berechneten Tondaten Stromsignale erzeugt, die in einem Lautsprecher hörbar gemacht werden können. In der Regel lassen sich auch Mikrofone anschließen und damit Töne digitalisieren und auf dem Computer speichern und bearbeiten.

Startmenü Zentrale Windows-Funktion: Menü das beim Klicken auf den Knopf „Start“ in der Taskleiste erscheint, von dem aus man Programme starten oder Einstellungen vornehmen kann.

Steuerzeichen Zeichen, die in einer Textdatei gespeichert aber nicht gedruckt werden. Sie dienen dazu, den Drucker zu steuern, also z.B. dafür zu sorgen, dass eine neue Zeile begonnen oder ein neues Blatt benutzt wird.

Symbolleiste Bereich in Windows-Fenstern, in dem Schaltflächen abgebil-

det sind, auf die man klicken kann, um bestimmte Programmfunktionen (z.B. Drucken) aufzurufen.

T

Taktfrequenz Häufigkeit mit der der → Prozessor die Befehlsschalter (vgl. Modellcomputer in Kapitel 3.1) umschaltet. Maß für die Rechengeschwindigkeit eines Computers.

U

USB-Schnittstelle Serielle Schnittstelle

V

Variable Platzhalter, Speicherplatz für sich ändernde Werte in einer Programmiersprache

Vektorgrafik Grafikformat, bei dem ganze Formen beschrieben werden statt einzelne Bildpunkte.

Verknüpfung Kleine Datei, die Informationen darüber enthält, wo sich eine andere Datei befindet, und mit welchen Parametern sie ggf. ausgeführt werden soll.

Verschieben Kopieren und anschließendes Löschen der Quelldatei.

Verzeichnis Möglichkeit, Dateien unter einem Namen zusammenzufassen. Bildlich vorstellbar als ein „Behälter für Dateien“.

W

Wechseldatenträger → Laufwerk, bei dem man das Medium (den Datenträger) wechseln kann, z.B. CD und Diskette

Wechselschaltung Verdrahtung, bei der man einen Verbraucher an zwei verschiedenen Schaltern beliebig und unabhängig voneinander ein- und ausschalten kann.

Z

Zahl Eine oder mehrere → Ziffern, mit denen eine Menge bestimmt

	wird. Die Position, die eine Ziffer innerhalb einer Zahl einnimmt, bestimmt ihren Wert, also die dargestellte Menge.
Ziffern	Zeichen, mit deren Hilfe → Zahlen gebildet werden können.

12 Index

1		Bus	74
		Byte	23

16er-System	23	C	
-------------	----	----------	--

A		CapsLock	87
Absatz	161, 165	CD	62
Abschnitt	166	CD-Laufwerk	117
Addieren	14, 43	Clipboard	144
Adresse	66, 115	Code	26
Adressen	200	COM	77
aktives Fenster	137	Compact Disc	Siehe CD
Algorithmus	212, 231	Compiler	221
Alt	89	CPU	50, 55
AltGr	91	Cursortasten	102
analog	44		
Anwendung	158	D	
Arbeitsplatz	145	Datei	66, 146
Arbeitsspeicher	48, 55, 58, 118	Dateiformat	66
ASCII	26, 27, 85, 199	Dateiname	66, 120, 151
Assembler	53	Dateityp	67, 159
Auswählen	156	Daten	50, 159

B		Datenbank	190
Baud	77	relational	191
Beamer	108	Datenformat	50
Bedingung	218	dekrementieren	54
Befehl	52	Desktop	121
Befehlszähler	54	Dezimalsystem	24
Benutzerschnittstelle	81	Dezimalzahl	23
Betriebssystem	57, 118	Diagramme	187
Bild	31	Dialer	207
Bild ab	102	Digital	44
Bild auf	102	Digitalkamera	113
Bildlaufleiste	140	Directory	Siehe Verzeichnis
Bildpunkt	Siehe Pixel	Diskette	61
Bildschirm	79, 107	Dokumente	124
binär	43	DOS	119
Binärsystem	25	dpi	112
BIOS	57, 115	Drop-Down-Feld	39
Bit	13	Drucken	100, 168
Bitmap	34	Drucker	77, 108
Bluetooth	80	Druckertreiber	110
booten	115	Dualsystem	25
Browser	199	DVD	64
Buchstaben	26	DVD-Laufwerk	117

Index

E

EEPROM	57
Eigenschaft	222
Einf	100
Eingabetaste	94
E-Mail	202
Ende	101
Entf	101
EPROM	57
Erweiterung	74
Erweiterungen	56
Esc	96
Ethernet	80
Explorer	146

F

FAT	67
Fenster	135
Festplatte	61
Feststelltaste	86
File	Siehe Datei
Firewire	76
Floppy-Disk	61
Font	109
Format	165
Formel	184
Frequenz	47
Funktion	186
Funktionstasten	97

G

Generator	59
Giga	47
Gliederung	161
Grafikkarte	57, 75, 79
GUI	81

H

Hardcopy	100
Hardware	55, 74
Hauptplatine	56
Hertz	47
Hexadezimalsystem	24
Hilfe	157
Homepage	196
HTML	197

Hz	47
----	----

I

IC	47
Icon	121, 142
Infrarot	80
inkrementieren	54
installieren	158
Integrierten Schaltkreis	47
Internet	195
Interpreter	221
invertieren	29
IrDA	80
ISA	74
ISDN	80, 111

J

Joystick	75
----------	----

K

Kilo	47
Kontextmenü	99, 129, 132, 147
Kopieren	154

L

laden	118
LAN	80, 196
Laufwerk	56, 61, 116
Laufwerksbuchstabe	117
Leertaste	85
Lineal	177
löschen	152
LPT1	77

M

Magnetband	60
Mainboard	56
Markieren	156
Maus	78, 103
Mauszeiger	103
Maximieren	138
Medium	64
Mega	47

Menü	121, 141, 143
Methoden	222
Minimieren	138
Modem	75, 80, 111
Monitor	79
Motherboard	56
Multiplizieren	43
Multitasking	131

N

Netzteil	56
Netzwerk	79, 195
Netzwerkkarte	75
Nibble	25
NumLock	94

O

Objektorientierte Programmierung	222
OCR	113
Oeffnen	134, 167
Ordner	71, 146, 147

P

Palette	33
Papierkorb	153
paralleler Bus	74
Parallelport	77
Pause	100
PCI	74
PCMCIA	76, 78
Pfad	71, 120
Pfeiltasten	102
Pixel	32, 112
Platine	49, 56
Plug & Play	76
Pos 1	101
Postfach	203
Präsentation	194
Primzahl	228
Programm	10, 40, 158, 210
Programmiersprachen	221
Protokolle	205
Provider	203
Prozessor	50, 56
PS/2	78

Q

Quadratwurzel	215
Quanten	44
Quick Info	142

R

Rahmen	140
RAM	49
Rechtschreibprüfung	179
Register	48, 51
Ressourcen	118
Rollen	100
ROM	57
Root	72
RS-232	77

S

Satzzeichen	163
Scanner	112
Schalter	46, 48
Schaltfläche	138, 142
Schleife	220
Schließen	138
Schnittstelle	72
scroll bar	140
Seite	166
Seitenansicht	176
serielle Schnittstelle	77
serieller Bus	74, 76
Shortcut	143
Sortieren	147
Soundkarte	57, 75
Speicher	48, 58
Speichern	168
Sprungbefehl	219
Startdiskette	116
Startmenü	98, 121
Statuszeile	140
Stecker	72
Steckkarte	56
Steckplatz	74
Steuerzeichen	28
Strg	88
Suchen	148
Symbole	121
Symbolleisten	142, 170
Systemmenü	137

T

Tabellenkalkulation	180
Tabulator	92, 177
Takt	47
Taskleiste	129
Tastatur	78, 81, 143
Tastenkombination	128, 143
Texterkennungssprogramme	113
Textfeld	172
Textverarbeitung	160
Thesaurus	174
Titelleiste	136
Transformator	56
Transistor	46
Treiber	74
Trojaner	207

U

Uhr	57
Umbenennen	151
Umschalttaste	85
Unterverzeichnisse	71
Urheberrecht	154
USB	76, 117
USB-Stick	64

V

Variablen	222
Vektorgrafik	36
Verknüpfungen	120, 155
Verschieben	134, 155
Verzeichnis	71, 153
Verzeichnisstruktur	147
VGA	79
Viren	206
Virtueller Arbeitsspeicher	131
Vorlagen	178

W

wahlfreier Zugriff	50
WAN	80, 196
Wechselschaltung	18
Wechselstrom	73
Wiederherstellen	138
Windows	120
Würmer	208

WYSIWYG 109, 160

Z

Zahlensysteme	25
Zeichen	165
Zeilenumbruch	161
Zentraleinheit	55
Ziffer	12, 93
Zwischenablage	144