Czes Kosniowski

Commodore Sachbuchreihe Band 8

MATHEMATIK MIT DEM COMMODORE 64

Wichtige Programmier-Routinen

Czes Kosniowski

MATHEMATIK MIT DEM COMMODORE 64

Commodore Sachbuchreihe Band 8

Czes Kosniowski

MATHEMATIK MIT DEM COMMODORE 64

Commodore

Titel der Originalausgabe: Mathematics on the Commodore 64; Essential routines for programming.
Copyright © Czes Kosniowski, 1983
First published 1983 by:
Sunshine Books (an imprint of Scot Press Ltd.)
12–13 Little Newport Street, London WC2R 3LD

Aus dem Englischen übertragen von Dr. Wolfgang Wefelmeier und Brigitte Pohl.

Artikel-Nr. 556430/12.84 Änderungen vorbehalten ISBN-Nr. 3-89133-008-1

INHALT

VORWORT

Dieses Buch ist für alle geschrieben, die einen Commodore 64 besitzen und gern etwas mehr über einige mathematische Verfahren wissen möchten. Wahrscheinlich wissen Sie, was für ein Programm Sie schreiben wollen, sind aber vielleicht nicht ganz sicher, was Sie an Mathematik dazu brauchen. Können Sie etwas mit COS, ABS oder SGN anfangen?
Hier sind alle mathematischen Funktionen des Commodore 64 beschrieben. Ihre Anwendung ist in kurzen Programmen erläutert, die Sie wörtlich übernehmen und in Ihre eigenen Programme einbauen können.
Aber das Buch ist nicht nur eine Einführung in diese elementaren mathematischen Funktionen. Es enthält auch Einführungen und Programme zu so unterschiedlichen Gebieten wie Codes und Kryptographie, Zufallszahlen, Trigonometrie, Primzahlen und Auswertung statistischer Daten. Diese Informationen können Sie sowohl für ernsthaftes Programmieren wie für Spiele verwenden.
Mein Dank gilt Ann, Kora und Inga für ihre Geduld beim Schreiben dieses Buches.

Czes Kosniowski
Newcastle upon Tyne, September 1983

BEMERKUNGEN ZU DEN PROGRAMMEN

Der Commodore 64 verwendet 'Steuerzeichen', um Funktionen wie Cursorbewegung und Zeichenfarbe zu kontrollieren. Diese Steuerzeichen werden gewöhnlich revers dargestellt. Ein revers dargestelltes Herz in Anführungszeichen löscht zum Beispiel den Bildschirm und bringt den Cursor in die Home-Position. Um Schwierigkeiten bei den Programmlistings zu vermeiden, sind keine Steuerzeichen benutzt worden. Statt dessen sind die zugehörigen CHR$-Codes verwendet worden. Die gebräuchlicheren sind in der folgenden Liste zusammengestellt. Weitere sind im Anfang F des Commodore 64 Handbuchs enthalten.

CHR$(5)	Weiß
CHR$(17)	Cursor nach unten
CHR$(28)	Rot
CHR$(30)	Grün
CHR$(31)	Blau
CHR$(145)	Cursor nach oben
CHR$(147)	Bildschirm löschen und Cursor Home
CHR$(154)	Hellblau
CHR$(157)	Cursor nach links
CHR$(158)	Gelb
CHR$(159)	Türkis

Beim Eingeben der Programme können Sie die CHR$-Codes durch Steuerzeichen ersetzen.

KAPITEL 1

EINFACHE FUNKTIONEN

ÜBERSICHTLICHE DARSTELLUNG VON ZAHLEN

Ganze Zahlen sind Zahlen, die keine Stellen nach dem Dezimalpunkt haben. Wenn Sie die folgenden Zeilen in einem Programm verwenden, erscheinen ganze Zahlen übersichtlich angeordnet auf Ihrem Bildschirm.

```
L = LEN(STR$(X))
PRINT TAB(25-L) X
```

Die Funktion STR$(X) wandelt die Zahl X in einen String um, LEN berechnet seine Länge, und TAB führt den Cursor an die richtige Stelle des Bildschirms.

```
   9
 123
- 10
  89
```

Bei nicht-ganzen Zahlen mißlingt diese Darstellung.

```
    89
   1.2
-13.89
  .126
```

Die Zahlen sind rechtsbündig; es wäre aber schöner, wenn die Dezimalpunkte untereinander ständen. Das erreicht man mit den Funktionen INT(X) und ABS(X). Die Funktion INT(X) bestimmt den ganzzahligen Anteil von X, also die größte ganze Zahl, die kleiner oder gleich X ist. Zum Beispiel

```
INT(1.21) = 1
INT(2) = 2
INT(2.1) = 2
```

```
INT(−2) = −2
INT(−2.1) = −3
INT(9.1) = 9
INT(−9.2) = −10
```

Die Funktion ABS(X) bestimmt den absoluten Wert von X, d. h. die Zahl ohne das Vorzeichen + oder −. Zum Beispiel

```
ABS(9.1) = 9.1
ABS(−9.1) = 9.1
```

Das folgende Programm stellt beliebige Zahlen übersichtlich auf Ihrem Bildschirm dar:

```
Y = INT(ABS(X)) : L = LEN(STR$(Y))
IF Y = 0 AND X <> 0 THEN L = L−1
PRINT TAB (25−L) X
```

Eine typische Darstellung sieht so aus:

```
      3
      .23
  −89.14
6712399.1
      2.23871
    −1.22
     −.13
```

In der ersten Zeile des Programms kümmert sich ABS um die negativen Zahlen und INT um die nicht-ganzen Zahlen. Beachten Sie, daß die Funktion INT allein nicht dasselbe leistet. (Nehmen Sie etwa die Zahl −9.1) Die zweite Zeile behandelt Zahlen, die größer als −1 und kleiner als 1 sind.
Das obige Programm zeigt eine einfache Anwendung der Funktionen INT und ABS. Es funktioniert nicht für Zahlen nahe 0 (absoluter Wert kleiner oder gleich 0.01) und sehr große Zahlen (absoluter Wert größer oder gleich 1000000000). Ansonsten mißglückt die Darstellung, wenn die wissenschaftliche Notation E vorkommt.

```
      3
  −89.14
      1E−04
9.9E+16
```

Wenn Sie wollen, können Sie unserem Programm zwei Zeilen hinzufügen, um Zahlen zu behandeln, die in der wissenschaftlichen Notation E angezeigt sind.

RUNDEN VON ZAHLEN

Die Funktion INT läßt sich zum 'Runden' von Zahlen gebrauchen. Wenn Sie beispielsweise DM 565,58 auf einem Bankkonto haben und 9% Zinsen im Jahr bekommen, beträgt Ihr Guthaben nach einem Jahr

```
565.58 + 565.58*9/100
```

Mit Ihrem Commodore 64 können Sie ausrechnen, daß das 616,4822 ergibt. Die Bank würde diesen Wert natürlich auf DM 616,48 *ab*runden. Entsprechend würde etwa ein Betrag von 76,6752 auf DM 76,68 *auf*gerundet. Dieses Runden bewerkstelligt Ihr Commodore 64 mit der Zeile

```
X = INT(X*100 + 0.5)/100
```

Zur Umwandlung in Pfennig wird X erst mit 100 multipliziert. Das Hinzuaddieren von 0,5 bewirkt dann ein Aufrunden, wenn die Pfennig-Bruchteile mindestens Einhalb ausmachen. Die Funktion INT ignoriert alle Bruchteile, und die Division durch 100 wandelt die Zahl wieder in DM um.
Allgemein ergibt die Programmzeile

```
B = INT(A*10↑D+0.5)/10↑D
```

den Wert von A, gerundet auf D Dezimalstellen.

BANKKONTEN

Das oben ausgelistete Programm zur übersichtlichen Darstellung von Zahlen auf dem Bildschirm läßt sich zum Beispiel für ein Saldierprogramm verwenden. Auf diese Weise erhält man etwa folgende Darstellung:

ART	AUSGABEN	EINNAHMEN	SALDO
ÜBERTRAG			596.61
869162	46.22		550.39
869164	169		381.39
869165	15.01		366.38
SCHECK		75.7	442.08

Wie Sie sehen, fehlt die Null, wenn der Betrag keine Pfennige oder ein Vielfaches von zehn Pfennig enthält. Um das zu ändern, müssen wir unsere Zahlen in STRings umwandeln und die nötige Anzahl von Nullen hinzufügen. Nachfolgende Programmzeilen zeigen, wie man das macht.

```
X$ = STR$(X) : L = LEN(STR$(X*100)) : M = LEN(X$)
IF M = L THEN X$ = X$ + "0"
IF M = L-2 THEN X$ = X$ + ".00"
PRINT TAB(25-L) X$
```

Beachten Sie, daß wir die Funktionen INT und ABS nicht brauchen. Der eben dargestellte Kontoauszug würde nun wie folgt aussehen:

ART	AUSGABEN	EINNAHMEN	SALDO
ÜBERTRAG			596.61
869162	46.22		550.39
869164	169.00		381.39
869165	15.01		366.38
SCHECK		75.70	442.08

ÜBERZOGENE KONTEN

Bankkonten werden gelegentlich überzogen (oder geraten in die roten Zahlen). Das geschieht, wenn der Saldo negativ (kleiner als Null) wird. So bedeutet ein Saldo von −DM 64,00, daß Sie DM 64,00 überzogen haben. Die folgenden Programmzeilen zeigen, wie man mit Hilfe der Funktion ABS Kontoauszüge ausdrucken und überzogene Beträge kennzeichnen kann (mit DB für Debit).

```
X$ = STR$(ABS(X)) : L = LEN(STR$(X*100)) : M = LEN(X$)
IF M = L THEN X$ = X$ + "0"
IF M = L-2 THEN X$ = X$ + ".00"
PRINT TAB(25-L) X$;
```

```
IF X < 0 THEN PRINT „DB";
PRINT
```

Zum Beispiel:

ART	AUSGABEN	EINNAHMEN	SALDO
ÜBERTRAG			442.08
869166	52.80		389.29
869167	422.00		32.72 DB

FARBIGE KONTOAUSZÜGE

Die Funktion SGN(X) ist die Vorzeichen-Funktion, die das Vorzeichen (positiv, negativ oder Null) der Zahl X ergibt. Das Ergebnis ist +1 für eine positive Zahl, −1 für eine negative Zahl und 0 für die Zahl 0. Zum Beispiel:

```
SGN(9.21) = 1
SGN(−9.1) = −1
SGN(0) = 0
```

Die Funktion SGN wird insbesondere verwendet, wenn das Programm verschiedene Unterprogramme abarbeiten soll, je nachdem, ob das Vorzeichen einer Zahl positiv, negativ oder Null ist. So würde das Programm aufgrund der Programmzeile

```
ON SGN(X) + 2 GOSUB 1000, 1100, 1200
```

das Unterprogramm 1000 ausführen, falls X negativ ist, das Unterprogramm 1100, falls X gleich Null ist, und das Unterprogramm 1200, falls X positiv ist.
Eine interessante Anwendung der Funktion SGN besteht darin, die Zeichenfarbe auf einfache Weise zu ändern. Auf dem Commodore 64 bedeutet CHR$(28) rot und CHR$(30) grün. Also ist CHR$(29+SGN(X)) rot oder grün, je nachdem, ob X negativ oder positiv ist. Das nächste Programm ergänzt unser Saldierprogramm um diesen Farbwechsel. Das erste POKE wählt eine Hintergrundfarbe, vor der rot und grün deutlich sichtbar sind.

```
POKE 53281,7
X$ STR$(ABS(X)) : L = LEN(STR$(X*100)) : M = LEN(X$)
IF M = L THEN X$ = X$ + "0"
IF M = L-2 THEN X$ = X$ + ".00"
PRINT CHR$(29+SGN(X)) TAB(25-L) X$;
IF X < 0 THEN PRINT "DB";
PRINT CHR$(30)
```

Dieses kurze Programm druckt Zahlen mit zwei Dezimalstellen (zum Beispiel DM und Pfennig) senkrecht untereinander aus. Dabei wird der ABSolute Wert einer negativen Zahl rot und mit DB dahinter ausgedruckt.

KAPITEL 2

TRIGONOMETRIE

MASSSTABTREUE ZEICHNUNGEN

Nur selten kann man die Höhe großer Gebäude, Berge, Bäume usw. direkt messen. Man kann aber die Höhe eines Gebäudes oder Baumes bestimmen, indem man aus einiger Entfernung den Winkel zwischen der Horizontalen und dem höchsten Punkt des Objekts mißt (mit einem Klinometer, das im Grunde nur ein veredelter Winkelmesser ist) und dann den Abstand bestimmt, den man zum Objekt hat. Siehe **Abbildung 1.**

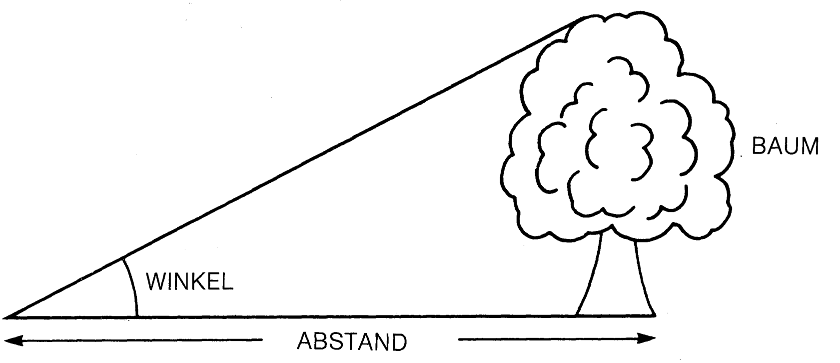


Abbildung 1.

Zur Bestimmung der Höhe einer kilometerweit entfernten und in Wolken gehüllten Bergspitze könnten Sie dieses Verfahren nicht anwenden. Die Wolken würden stören, und der horizontale Abstand könnte nicht gemessen werden. In solchen Fällen würde ein Instrument wie das Tellurometer weiterhelfen. Es lokalisiert die Bergspitze mit Hilfe von Radar. Außerdem mißt es Winkel und Abstand zwischen Ihnen und der Spitze. Mit einer maßstabtreuen Zeichnung können Sie dann die Höhe des Berges bestimmen. Siehe **Abbildung 2.**

Ein weiteres Beispiel ist die Bestimmung der Breite eines großen Teiches oder Sees; siehe **Abbildung 3.**

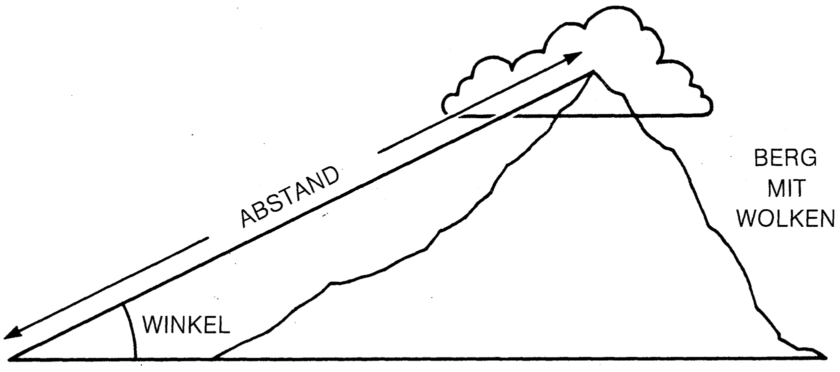


Abbildung 2.

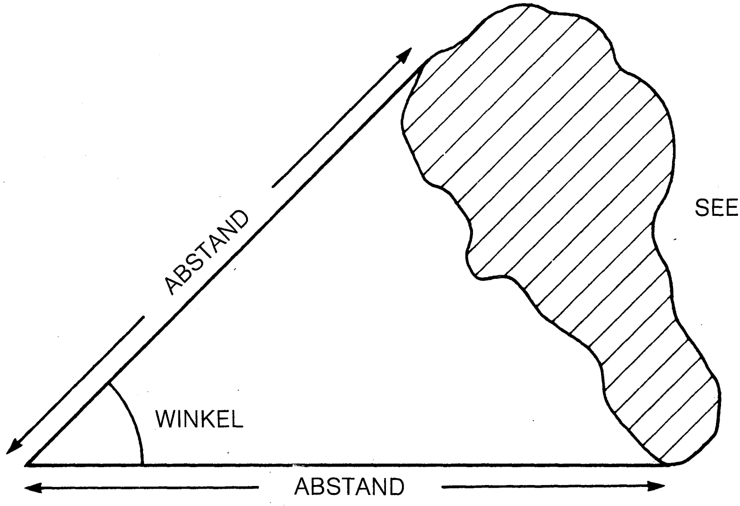


Abbildung 3.

Mit den gemessenen Werten könnte man eine maßstabtreue Zeichnung anfertigen und die fragliche Länge schätzen.
Hier ist ein ähnliches Beispiel: Ein Navigator befindet sich in einer bestimmten Position A. Er ist 150 km westlich von der Stadt B und 188 km von der Stadt C entfernt. Von seiner Position aus ist der Winkel zwischen beiden Städten 23 Grad. Wie weit sind die beiden Städte voneinander entfernt? Eine maßstabtreue Zeichnung könnte auch diese Aufgabe lösen.

Maßstabtreue Zeichnungen beantworten zwar die oben erwähnten Fragen, aber sie liefern nur eine große Näherung. Sie sind weder genau noch immer praktikabel. Einen anderen Zugang bietet die Trigonometrie, und dabei hilft Ihnen Ihr Commodore 64.

DIE TRIGONOMETRISCHEN FUNKTIONEN

Die drei wichtigsten trigonometrischen Funktionen sind SIN (die Sinus-Funktion), COS (die Cosinus-Funktion) und TAN (die Tangens-Funktion). Alle stellen Verhältnisse verschiedener Seiten eines rechtwinkligen Dreiecks dar. Das unten gezeigte Dreieck ist ein solches rechtwinkliges Dreieck. Der Winkel an der linken Ecke ist mit dem Symbol X bezeichnet. Die drei Seiten des Dreiecks heißen Ankathete von X, Gegenkathete von X und Hypootenuse (längste Seite).

$$TAN(X) = \frac{\text{Gegenkathete}}{\text{Ankathete}}$$

$$SIN(X) = \frac{\text{Gegenkathete}}{\text{Hypotenuse}}$$

$$COS(X) = \frac{\text{Ankathete}}{\text{Hypotenuse}}$$

Es ist sinnvoll, sich die folgenden Werte zu merken:

SIN(0°) = 0	COS(0°) = 1	TAN(0°) = 0
SIN(30°) = 0.5	COS(30°) = SQR(3)/2	TAN(30°) = 1/SQR(3)
SIN(45°) = 1/SQR(2)	COS(45°) = 1/SQR(2)	TAN(45°) = 1
SIN(60°) = SQR(3)/2	COS(60°) = 1/2	TAN(60°) = SQR(3)
SIN(90°) = 1	COS(90°) = 0	

Kennt man den Winkel X und eine der drei Seiten eines rechtwinkligen Dreiecks, so kann man die anderen beiden Seiten bestimmen. Sind beispielsweise der Winkel X und die Länge der Ankathete bekannt, so ergeben sich die beiden anderen Seiten aus folgenden Formeln:

Gegenkathete = TAN(X) * Ankathete
Hypotenuse = Ankathete / COS(X)

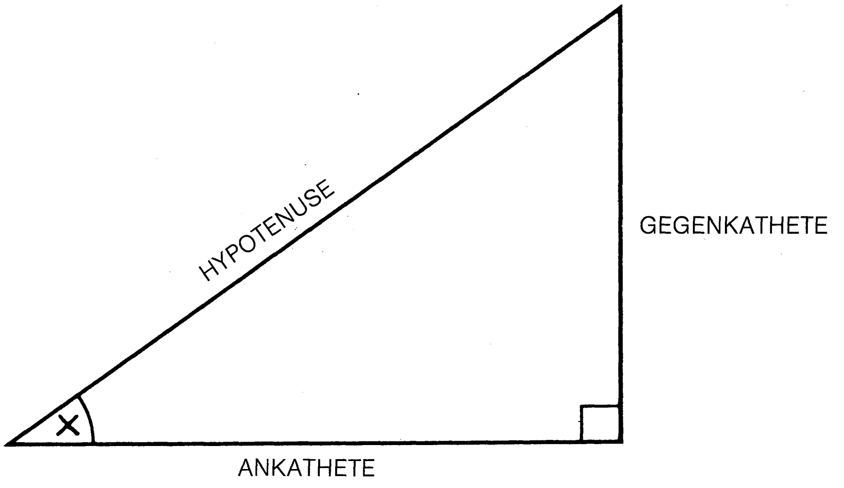


Abbildung 4.

Man kann die trigonometrischen Funktionen auch anhand eines Kreises mit dem Radius 1 beschreiben. Dazu zeichnet man den gegebenen Winkel wie in **Abbildung 5** ein. Die Werte der verschiedenen trigonometrischen Funktionen lassen sich aus der Abbildung ablesen.

Mathematisch werden Längen horizontal von links nach rechts und vertikal von unten nach oben gemessen. Das erklärt, warum z. B. COS(X) in **Abbildung 6** einen negativen Wert hat.

Sie erhalten SIN, COS und TAN eines Winkels X durch Eingeben von

PRINT SIN(X) etc.,

wobei für X der gewünschte Wert einzusetzen ist. Das einzige Problem dabei ist, daß der Commodore 64 wie die meisten Mikrocomputer die Winkel im Bogenmaß, nicht in Grad erwartet. Glücklicherweise kann man Grad leicht in Radiant (engl.: radian) umrechnen und umgekehrt.
Zunächst: Was ist ein Radiant? Zeichnen Sie einen Kreis vom Radius 1. Tragen Sie auf dem Umfang eine Strecke von der Länge des Radius ab. Der von diesem Bogen aufgespannte Winkel ist 1 Radiant oder ungefähr 57°. Siehe **Abbildung 7.**

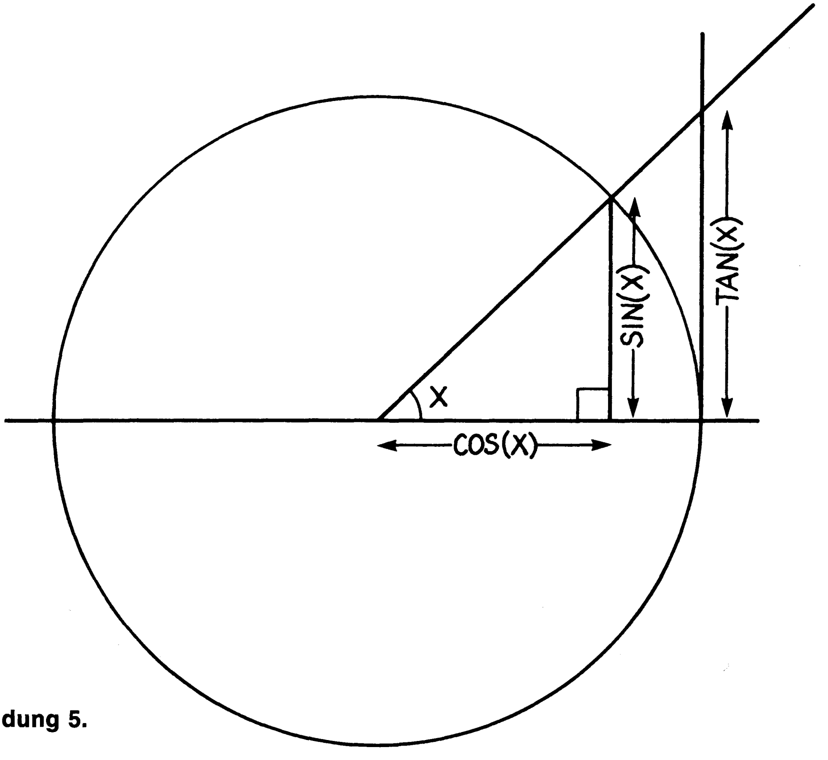


Abbildung 5.

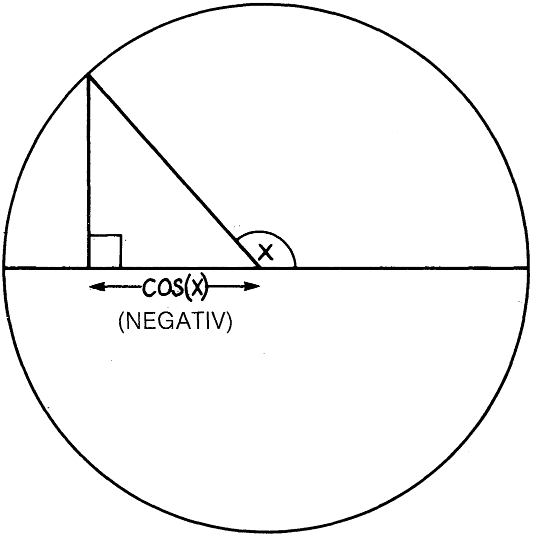


Abbildung 6.

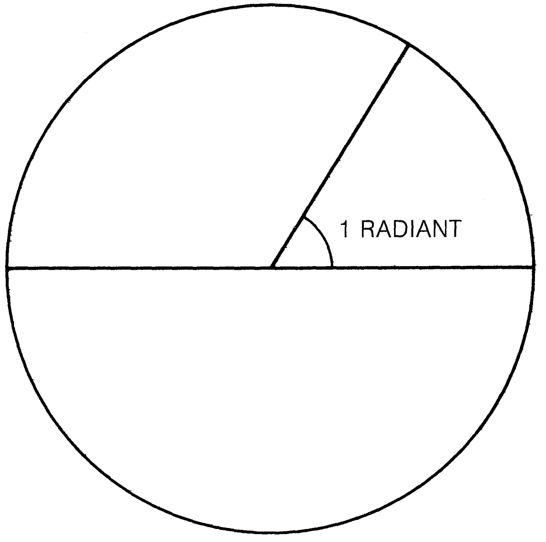


Abbildung 7.

Die Zahl π (oder pi) ist bemerkenswert und berühmt. Sie ist definiert als das Verhältnis des Umfangs eines Kreises zu seinem Durchmesser. Der (ungefähre) Wert von π ist in Ihrem Commodore 64 gespeichert. Tippen Sie einfach

PRINT π

ein, und Sie erhalten den gespeicherten Wert. In einem Kreis vom Radius 1 ist der Durchmesser gleich 2. Also ist der Umfang des Kreises gleich $2*\pi$, und deshalb hat ein ganzer Kreis $2*\pi$ Radiant. Da ein ganzer Kreis 360 Grad hat, ergibt sich

360 Grad = $2*\pi$ Radiant,
180 Grad = π Radiant.

Mit folgenden einfachen Formeln läßt sich Grad in Radiant und Radiant in Grad umrechnen:

X Grad = $X*\pi/180$ Radiant,
Y Radiant = $Y*180/\pi$ Grad.

Mit dem nächsten Programm können Sie die Seiten eines rechtwinkligen Dreiecks bestimmen. Sie müssen dazu einen Winkel und die Länge einer Seite eingeben. Das Programm berechnet dann die beiden anderen Seiten.

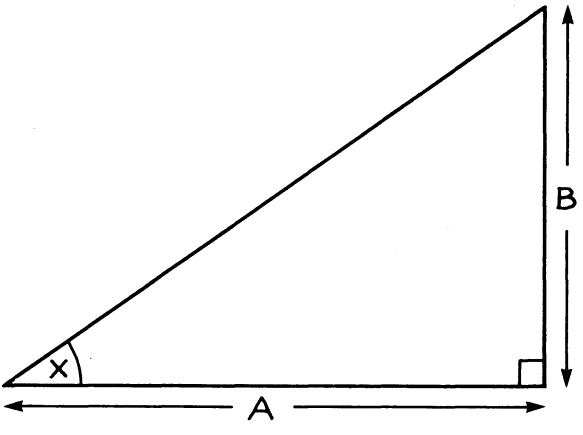
```
10 REM 1RECHTWINKLIGE DREIECKE
20 POKE53280,6:PRINT CHR$(147):PRINT"
   ***RECHTWINKLIGE DREIECKE***"CHR$(17)
30 PRINT"DIESES PROGRAMM ERMOEGLICHT ES,
 SEITEN"
40 PRINT"EINES RECHTWINKLIGEN DREIECKS Z
U"
50 PRINT"FINDEN, ANGENOMMEN SIE KENNEN E
INE       SEITE UND EINEN WINKEL"CHR$(17)
54 PRINT"                *"
55 PRINT"               **"
60 PRINT"              * *"
65 PRINT"             *  *"
70 PRINT"            *   *"
75 PRINT" HYPOTENUSE*    *GEGENKATHETE"
80 PRINT"          *     *"
85 PRINT"         *      *"
90 PRINT"        *WINKEL *"
95 PRINT"       **********"
96 PRINT"       ANKATHETE" CHR$(17)
100 REM EINGABE DETAILS
110 INPUT "WINKEL IN GRAD: ";X
120 IF X<=0 OR X>=90THEN PRINT "FEHLER-K
EIN DREIECK":GOTO110
125 PRINTCHR$(147)
130 PRINT CHR$(17)"WELCHE SEITE IST BEKA
NNT?"
135 PRINT"          1 (GEGENKATHETE)
140 PRINT"          2 (ANKATHETE)
150 PRINT"          3 (HYPOTENUSE)"CHR$(
17)
155 INPUT " 1,2 ODER 3 ";T
160 IF T<1 OR T>3 OR T<>INT(T) THEN 155
170 PRINTCHR$(17)"GEBEN SIE DIE LAENGE D
ER SEITE EIN."
180 INPUT "LAENGE: ";L
190 IF L<=0 THEN180
200 REM UMWANDLUNG IN BOGENMASS"
210 X=X*π/180
220 REM AUFTEILEN
230 ON T GOSUB 300,350,400
```

```
240 PRINTCHR$(17) "DAS WAR'S - NOCH EIN
VERSUCH? (J/N):"
250 GET G$:IF G$="" THEN250
260 IF G$="J"THEN RUN
270 IF G$="N"THEN PRINT CHR$(147) " AUF
WIEDERSEHEN":END
300 REM GEGENKATHETE BEKANNT
310 PRINT CHR$(17) "ANKATHETE   :" L/TAN(
X)
320 PRINT"HYPOTENUSE :" L/SIN(X)
330 RETURN
350 REM ANKATHETE BEKANNT
360 PRINTCHR$(17)"GEGENKATHETE :" TAN(X)
*L
370 PRINT"HYPOTENUSE :"L/COS(X)
380 RETURN
400 REM HYPOTENUSE BEKANNT
410 PRINT CHR$(17) "GEGENKATHETE :" SIN(
X)*L
420 PRINT "ANKATHETE     :" L*COS(X)
430 RETURN
```

INVERSE FUNKTIONEN

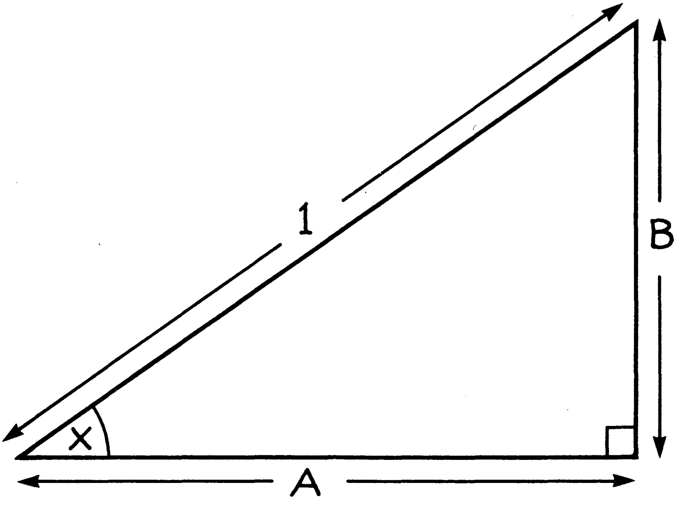
Angenommen, wir kennen die Seiten eines rechtwinkligen Dreiecks – können wir dann die verschiedenen Winkel bestimmen? Die Antwort ist ja, doch dazu benötigen wir die inversen trigonometrischen Funktionen. Ist ein Winkel X gegeben, so ergibt TAN(X) eine Zahl, den Tangens des Winkels X. Wenn umgekehrt eine Zahl N gegeben ist, gibt es einen Winkel, dessen Tangens gleich dieser Zahl ist. Einen solchen Winkel könnte man den inversen Tangens von N nennen. Gewöhnlich wird er mit ATN(N), das heißt Arcustangens von N, bezeichnet.
Sehen Sie sich das Dreieck in **Abbildung 8** an.
Wenn wir die Werte von A und B kennen, können wir den Wert des Winkels X bestimmen. Wir wissen, daß TAN(X) = B/A ist, also X = ATN(B/A). Sie können die betreffenden Werte in diesen Ausdruck einsetzen und die Antwort von Ihrem Commodore 64 ausdrucken lassen. Die Antwort würde natürlich in Radiant angegeben. Für eine Antwort in Grad müssen Sie das Ergebnis mit $180/\pi$ multiplizieren.
Die trigonometrischen Funktionen SIN und COS haben ebenfalls inverse Funktionen; sie werden mit ASN (Arcussinus) und ACS (Arcuscosinus) bezeichnet.

Abbildung 8.



ASN(N) ist der Winkel, dessen Sinus gleich N ist; entsprechend ist ACS(N) der Winkel, dessen Cosinus gleich N ist. Leider sind in Commodore-64-Basic wie bei vielen anderen Mikrocomputern diese Funktionen nicht vordefiniert. Sie können aber leicht aus der Funktion ATN abgeleitet werden.
Um zu verstehen, wie man ASN aus ATN ableitet, betrachten Sie das rechtwinklige Dreieck mit Hypotenuse 1 in **Abbildung 9.**

Abbildung 9.



Angenommen, wir kennen den Wert von B und wollen den Winkel X finden. Wir wissen, daß SIN(X) = B ist, also X = ASN(B), aber wie erwähnt ist ASN nicht im Commodore 64 vorhanden. Wenn uns aber der Wert von A bekannt wäre, könnten wir ATN verwenden, weil auch X = ATN(B/A) gilt. Um A zu bestimmen, benutzen wir den Satz des Pythagoras.
Der Satz des Pythagoras ist Ihnen sicher bekannt. Wörtlich besagt er, daß das Quadrat der Hypotenuse eines rechtwinkligen Dreiecks gleich der Summe der Quadrate der beiden anderen Seiten ist. In Symbolen ausgedrückt:

$$C^2 = A^2 + B^2$$

Dabei bezeichnet C die Länge der Hypotenuse. Da unsere Hypotenuse die Länge 1 hat, erhalten wir

```
1 = A*A + B*B
```

oder

```
A*A = 1 - B*B
```

also

```
A = SQR(1 - B*B)
```

Wegen X = ATN(B/A) ergibt sich

```
X = ATN(B/SQR(1 - B*B))
```

Andererseits gilt auch ASN(B) = X und deshalb

```
ASN(B) = ATN(B/SQR(1 - B*B))
```

Auf ähnliche Weise können wir folgende Formel für ACS(A) herleiten:

```
ACS(A) = π/2 - ATN(A/SQR(1 - A*A))
```

Beachten Sie, daß ACS(A) = π/2 − ASN(A) ist. Anhang H des Commodore-64-Handbuchs enthält andere Beispiele mathematischer Funktionen, die nützlich sein können, aber in Commodore-64-Basic nicht vordefiniert sind.

NICHT-RECHTWINKLIGE DREIECKE

Die ersten beiden Beispielaufgaben aus dem Abschnitt über maßstabtreue Zeichnungen können mit dem Programm 'Rechtwinklige Dreiecke' gelöst werden. Im dritten Beispiel allerdings ist das Dreieck (im allgemeinen) nicht rechtwinklig.
Ein Dreieck hat drei Winkel und drei Seiten. Wenn wir drei beliebige Werte davon kennen (mit Ausnahme von drei Winkeln), können wir die anderen Werte bestimmen. Nehmen wir an, wir kennen zwei Seiten und einen Winkel. Die dritte Seite und die beiden anderen Winkel können wir dann mit Hilfe einer Formel berechnen.
Bezeichnen wir die drei Winkel unseres Dreiecks mit X, Y und Z, und die drei Seiten mit SX, SY und SZ, wobei die Seite SX dem Winkel X gegenüberliegt, etc. Siehe **Abbildung 10.**

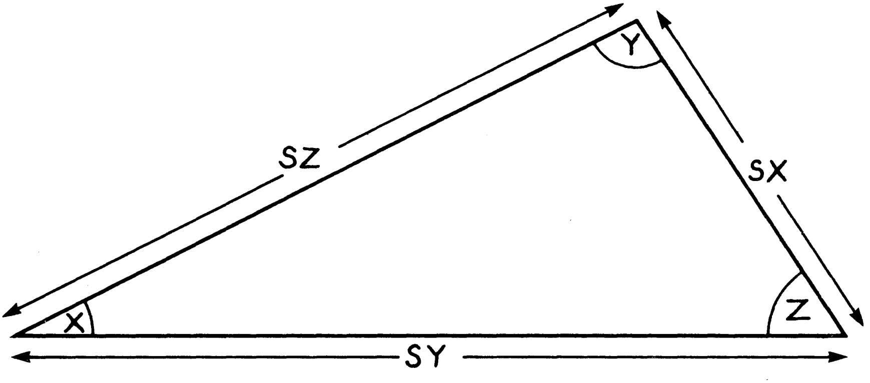


Abbildung 10.

Die verschiedenen Seiten und Winkel stehen über folgende Gleichungen miteinander in Beziehung:
Der Cosinus-Satz:

SZ*SZ = SX*SX + SY*SY − 2*SX*SY*COS(Z)
SY*SY = SX*SX + SZ*SZ − 2*SX*SZ*COS(Y)
SX*SX = SY*SY + SZ*SZ − 2*SY*SZ*COS(X)

Der Sinus-Satz: SIN(X)/SX = SIN(Y)/SY = SIN(Z)/SZ

Beachten Sie, daß COS(Z) = 0 ist, wenn Z ein rechter Winkel ist (also 90 Grad hat).
Die erste Gleichung hat dann folgende Form:

SZ*SZ = SX*SX + SY*SY

Das ist gerade der Satz des Pythagoras.
Das nächste Programm bestimmt die übrigen Winkel und Seiten, wenn eine der folgenden Angaben bekannt ist:

Seite Seite Seite: Sie kennen alle drei Seiten und wollen die drei Winkel bestimmen.
Seite Seite Winkel: Sie kennen zwei Seiten und einen Winkel, der nicht zwischen ihnen liegt (einen nicht eingeschlossenen Winkel), und wollen die dritte Seite und die übrigen Winkel bestimmen.
Seite Winkel Seite: Sie kennen zwei Seiten und den Winkel dazwischen (den eingeschlossenen Winkel) und wollen die dritte Seite und die übrigen Winkel bestimmen.
Seite Winkel Winkel: Sie kennen zwei Winkel und eine nicht dazwischen liegende Seite (eine nicht eingeschlossene Seite) und wollen die übrigen Seiten und den dritten Winkel bestimmen.
Winkel Seite Winkel: Sie kennen zwei Winkel und die dazwischen liegende Seite (die eingeschlossene Seite) und wollen die übrigen Seiten und den dritten Winkel bestimmen.

Beachten Sie, daß im zweiten Fall (Seite Seite Winkel) gewöhnlich zwei verschiedene Dreiecke möglich sind, je nachdem, ob der der dritten Seite gegenüberliegende Winkel größer oder kleiner als 90 Grad ist. **Abbildung 11** verdeutlicht das.

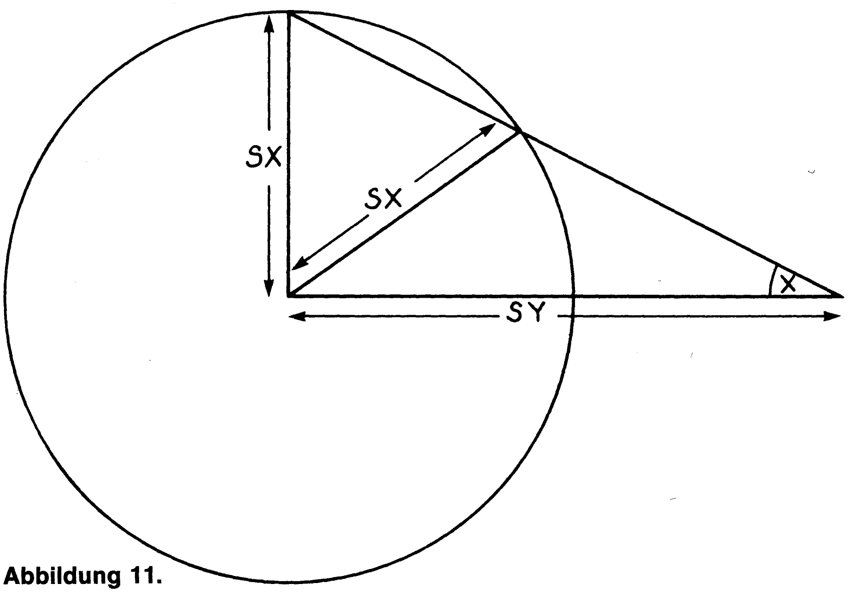


Abbildung 11.

```
10 REM 2DREIECKE
15 POKE53280,6
20 PRINTCHR$(147)"           *** DREIECKE
 ***"CHR$(17)
30 PRINT"DIESES PROGRAMM FINDET DIE FEHL
ENDEN "
40 PRINT"SEITEN UND WINKEL EINES DREIECK
ES"
45 PRINT
50 PRINT"WELCHE INFORMATIONEN HABEN SIE?
"
55 PRINT
60 PRINT"1) SSS : ALLE 3 SEITEN"
70 PRINT"2) SSW : 2 SEITEN UND NICHT EIN
GESCHL.            WINKEL"
80 PRINT"3) SWS : 2 SEITEN UND EINGESCHL
.WINKEL"
90 PRINT"4) SWW : 2 WINKEL UND NICHT EIN
GESCHL.            SEITE
100 PRINT"5) WSW : 2 WINKEL UND EINGESCH
L.SEITE"
110 REM WAHL
115 PRINT
120 INPUT "BITTE NUMMER EINGEBEN ";N
125 IFN=1ORN=2ORN=4ORN=3ORN=5GOTO140
130 PRINTCHR$(17)"BITTE 1,2,3,4 ODER 5 V
ERSUCHEN":GOTO120
140 REM ARCUS - SINUS FUNKTION DEFENIERE
N(AUF ZWEI STELLEN HINTERM KOMMA)
150 DEF FNAS(X)=INT(18000*ATN(X/SQR(1-X*
X))/π+.5)/100
160 REM AUFTEILEN
165 PRINTCHR$(147)
170 PRINT:ON N GOSUB 310,510,710,910,101
0
180 PRINT:PRINTCHR$(17) "EIN NEUER START
 (J/N)"
190 GET G$:IF G$=""THEN190
200 IF G$="J"THENRUN
210 PRINTCHR$(147):PRINT"AUF WIEDERSEHEN
":END
```

```
300 REM 3 SEITEN
310 PRINT"      ***     3 SEITEN BEKANNT
   ***"
320 M=1:GOSUB 1110:SX=S
330 M=2:GOSUB 1110:SY=S
340 M=3:GOSUB 1110:SZ=S
350 A=(SY*SY+SZ*SZ-SX*SX)/(2*SY*SZ)
360 IF ABS(A)>=1 THENPRINT"KEIN DREIECK"
365 PRINT:PRINT
370 PRINT"WINKEL GEGENUEBER SEITE 1 = "9
0-FNAS(A):PRINT
380 A=(SX*SX+SZ*SZ-SY*SY)/(2*SX*SZ)
390 PRINT"WINKEL GEGENUEBER SEITE 2 = "9
0-FNAS(A):PRINT
400 A=(SX*SX+SY*SY-SZ*SZ)/(2*SX*SY)
410 PRINT"WINKEL GEGENUEBER SEITE 3 = "9
0-FNAS(A):PRINT
420 RETURN
500 REM 2 SEITEN UND NICHT EINGSCHL. WIN
KEL
510 PRINT"** 2 SEITEN UND NICHT EINGSCHL
.WINKEL **"
520 PRINT"GEBEN SIE DIE SEITE EIN, VON D
ER          DER GEGENUEBERLIEGENDE WINKEL
BE-"
525 PRINT"KANNT IST"
530 M=1:GOSUB 1110:SX=S:GOSUB 1210:AX=A
540 M=2:GOSUB 1110:SY=S
550 A=SIN(AX)*SY/SX:IF ABS(A)>1ORA=0THEN
PRINT:PRINT"KEIN DREIECK":RETURN
559 PRINT
560 PRINT"IST DER WINKEL GEGENUEBER SEIT
E 2 GROE- SSER (>) ODER KLEINER (<) ";
565 PRINT"ALS 90 GRAD ?"
570 INPUT"BITTE > ODER < EINGEBEN"; A$
580 IF A$<>"<" AND A$<>">"THEN570
590 AY=FNAS(A):IFA$=">"AND AY<90 THENAY=
90+AY
600 PRINT:PRINT:PRINT     "WINKEL GEGENU
EBER SEITE 2 = "AY
610 AZ=π-AX-AY*π/180
```

```
620 PRINT:PRINT"LAENGE VON SEITE 3
   = " SX*SIN(AZ)/SIN(AX)
630 PRINT:PRINT"WINKEL GEGENUEBER SEITE
3 = " INT(18000*AZ/π+.5)/100 :PRINT
640 RETURN
700 REM 2 SEITEN UND EINGESCHL. WINKEL
710 PRINT"*** 2 SEITEN UND EINGESCHL. WI
NKEL ***"
720 M=1:GOSUB 1110:SX=S
730 M=2:GOSUB 1110:SY=S
740 M=3:GOSUB 1210:AZ=A
750 SZ=SQR(SX*SX+SY*SY-2*SX*SY*COS(AZ))
760 IF SZ=0 THENPRINT:PRINT"KEIN DREIECK
":RETURN
770 PRINT:PRINT:PRINT"LAENGE SEITE 2
         = "SZ
780 A=(SY*SY+SZ*SZ-SX*SX)/(2*SY*SZ)
790 PRINT"WINKEL GEGENUEBER SEITE 1 = "9
0-FNAS(A)
800 A=(SX*SX+SZ*SZ-SY*SY)/(2*S*SZ)
810 PRINTCHR$(17)"WINKEL GEGENUEBER SEIT
E 2 = " 90-FNAS(A)
820 RETURN
900 REM 2 WINKEL UND NICHT EINGESCHL. SE
ITE
910 PRINT"* 2 WINKEL UND NICHT EINGESCHL
. SEITE *"
915 PRINT
920 PRINT"GEBEN SIE DEN WINKEL, DER DER
BEKANNTEN SEITE GEGENUEBER LIEGT ";
925 PRINT"ZUERST EIN."
930 M=1:GOSUB 1210:AX=A:GOSUB 1110:SX=S
940 M=2:GOSUB 1210:AY=A
950 A=π-AX-AY:IFA<=0 THEN PRINT"KEIN DRE
IECK":RETURN
960 PRINT:PRINT: PRINT"LAENGE SEITE 2
          = "SX*SIN(AY)/SIN(AX)
970 PRINT: PRINT"WINKEL GEGENUEBER SEITE
 3 = "INT(18000*A/π+.5)/100
980 PRINT:PRINT"LAENGE SEITE 3
   = "SX*SIN(A)/SIN(AX)
```

```
990 RETURN
1000 REM 2 WINKEL UND EINGESCHL. SEITE
1010 PRINT"**** 2 WINKEL UND EINGESCHL.
SEITE ****"
1020 M=1:GOSUB 1210:AX=A
1030 M=2:GOSUB 1210:AY=A
1040 M=3:GOSUB 1110:SZ=S
1050 A=π-AX-AY:IF A<0 THEN PRINT "KEIN D
REIECK":RETURN
1060 PRINT:PRINT:PRINT"WINKEL GEGENUEBER
 SEITE 3 = "INT(18000*A/π+.5)/100CHR$(17
)
1070 PRINT"LAENGE SEITE 1             = "
SZ*SIN(AX)/SIN(A) CHR$(17)
1080 PRINT"LAENGE SEITE 2             = "
SZ*SIN(AY)/SIN(A)
1090 RETURN
1100 REM EINE SEITE BERECHNEN
1110 PRINT:S=0:PRINT"LAENGE DER SEITE" M
 "EINGEBEN : ";:INPUT S
1120 IF S<=0 THEN PRINT"KEIN DREIECK":GO
TO1110
1130 RETURN
1200 REM EINEN WINKEL BERECHNEN
1210 PRINT:A=0:PRINT "WINKEL GEGENUEBER
SEITE " M " : ";:INPUT A
1220 IF A<=0.001 OR A>180 THEN PRINT"KEI
N DREIECK":GOTO 1210
1230 A=A*π/180: RETURN
```

BRECHUNG

Betrachtet man Gegenstände durch Glas oder Plastik hindurch, sehen sie oft verzerrt aus. Gewässer scheinen seichter als sie wirklich sind. Das liegt an der Brechung des Lichts. Tritt ein Lichtstrahl aus einem Medium (Luft) in ein anderes (Glas, Wasser . . .) ein, so wird er gebeugt oder gebrochen. Der Winkel, unter dem der Strahl auf das Glas eintrifft, heißt Einfallswinkel; der Winkel nach der Brechung heißt Brechungswinkel – siehe **Abbildung 12.**

Für ein gegebenes Material besteht eine feste Beziehung zwischen Einfallswinkel und Brechungswinkel. Sie wird durch das Snelliussche Gesetz ausgedrückt, das besagt, daß das Verhältnis der Sinuswerte für jedes Material (im Vergleich zur Luft) konstant ist. Dieses Verhältnis heißt Brechungsindex (Brechzahl).

$$\text{Brechungsindex} = \frac{\text{SIN (Einfallswinkel)}}{\text{SIN (Brechungswinkel)}}$$

Der Brechungsindex für Glas ist ungefähr 1.5, für Wasser beträgt er 1.333, für Diamant ist er 2.417.

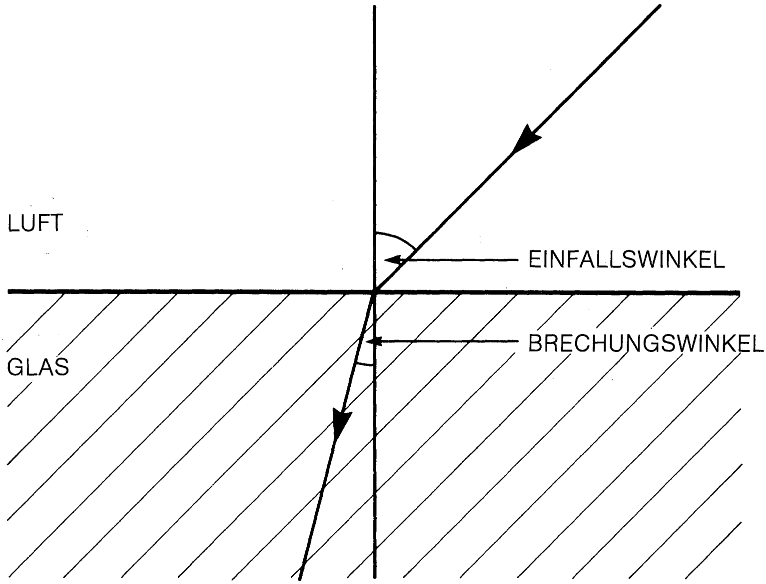


Abbildung 12.

Mit dem folgenden Programm können Sie den Brechungswinkel berechnen, wenn Sie den Einfallswinkel und den Brechungsindex kennen:

```
10 REM 3BRECHUNG
20 POKE53280,6:PRINTCHR$(147)"          ***
**  BRECHUNG  *****"
30 PRINT:PRINT"DIESES PROGRAMM ERMOEGLIC
HT ES, DEN AUS-FALLSWINKEL EINES LICHT";
40 PRINT"STRAHLS ZU BE-   RECHNEN, WENN
ER
```

```
AUF EIN ANDERES MEDIUM TRIFFT"
100 REM EINGABE DETAILS
110 PRINT:PRINT"BITTE EINFALLSWINKEL EIN
GEBEN ( IN GRAD)"
120 INPUT "WINKEL        : ";X
130 IF X<=0ORX>=90THENPRINT"!!  IRRTUM
!!":GOTO120
140 PRINT:PRINT"BITTE BRECHUNGSINDEX EIN
GEBEN"
145 INPUT"BRECHUNGSINDEX: ";R
150 IF R<=0THENPRINT"!!  IRRTUM  !!":GOT
O145
160 REM UMRECHNUNG IN BOGENMASS
170 X=X*π/180
180 REM BERECHNUNG
190 Y=SIN(X)/R:Y=Y/SQR(1-Y*Y)
200 PRINT:PRINT:PRINT"AUSFALLSWINKEL: "A
TN(Y)*180/π"GRAD"
210 PRINT:PRINT"PROZENTSATZ DES EINFALLS
WINKELS :"INT(ATN(Y)*100/X+.5)
240 PRINT:PRINT:PRINT"NEUER START (J/N)?
"
250 GET A$:IF A$="" THEN250
260 IF A$="J"THENRUN
270 IF A$="N"THENPRINTCHR$(147)"AUF WIED
ERSEHEN":END
280 GOTO250
```

TOTALREFLEXION

Ein Stück Glas oder eine Wasseroberfläche verhalten sich gelegentlich wie ein gewöhnlicher Spiegel: Sie reflektieren alles. Das geschieht, wenn der Einfallswinkel zu groß ist und der Lichtstrahl voll zurückgeworfen wird. Der kleinste Winkel, bei dem dies auftritt, heißt Grenzwinkel der Totalreflexion. Er ergibt sich aus dieser einfachen Formel:

$$\text{SIN (Grenzwinkel)} = \frac{1}{\text{Brechungsindex}}$$

Also läßt sich der Grenzwinkel mit Hilfe der oben beschriebenen Funktion ASN aus dem Brechungsindex bestimmen.

KAPITEL 3

TRIGONOMETRIE DER ERDE

DIE ERDE

Die Erde ist fast eine Kugel. Nur am Nord- und Südpol ist sie leicht abgeflacht. Der Äquator hat einen Radius von 6378 Kilometern. Der Pol-Radius ist 6357 Kilometer. Da der Unterschied nur 0,3% beträgt, ist die Erde kaum von einer Kugel zu unterscheiden. Der mittlere Radius beträgt 6371 Kilometer.
Wir sind daran gewöhnt, uns unter der kürzesten Verbindung zwischen zwei Punkten eine 'gerade Linie' vorzustellen. Das gilt aber nur für eine Ebene, bzw. für jede 'flache' Oberfläche. Auf einer Kugel wie der Erde ist die kürzeste Verbindung zwischen zwei Punkten ein Teil eines sogenannten Großkreises. Ein Großkreis ist ein Kreis, dessen Mittelpunkt im Mittelpunkt der Erde liegt.
Großkreise durch den Nord- und Südpol heißen *Längenkreise.* Jedem Längenkreis ist ein Winkel zugeordnet. Der durch Greenwich in England verlaufende Längenkreis wird mit 0° (0 Grad) bezeichnet. Die anderen ergeben sich jeweils aus dem Winkel zwischen Längenkreis, Erdmittelpunkt und Längenkreis durch Greenwich, gemessen am Äquator – siehe **Abbildung 13.**

Gewöhnlich zählt man die geographische Länge von 0° bis 180°, sowohl nach Osten als auch nach Westen.
An den Längengraden erkennt man, wie weit westlich oder östlich von Greenwich ein Punkt auf der Erde liegt. Wie weit nördlich oder südlich vom Äquator ein Punkt liegt, sieht man an den *Breitenkreisen.* Der Äquator hat den Breitengrad 0°. Alle Kreise auf der Erde, die parallel zum Äquator verlaufen, heißen Breitenkreise. Jedem Breitenkreis ist der Winkel zugeordnet, den er am Erdmittelpunkt mit dem Äquator bildet – siehe **Abbildung 14.**

Die geographische Breite zählt von 0° bis 90° nach Norden und Süden. Der Breitengrad des Nordpols ist 90° Nord, der des Südpols 90° Süd.
Jeder Punkt auf der Erde ist durch seinen Breiten- und Längengrad festgelegt. Beispielsweise liegt Newcastle upon Tyne (England) ungefähr 55° nördlicher Breite und 1,5° westlicher Länge. Genauere Werte sind 54° 58' nördliche Breite und 1° 36' westliche Länge, wobei ' als Minuten gelesen wird und eine Minute 1/60 Grad beträgt.
Breiten- und Längengrade bilden einen Satz von Koordinaten auf der Erdoberfläche.

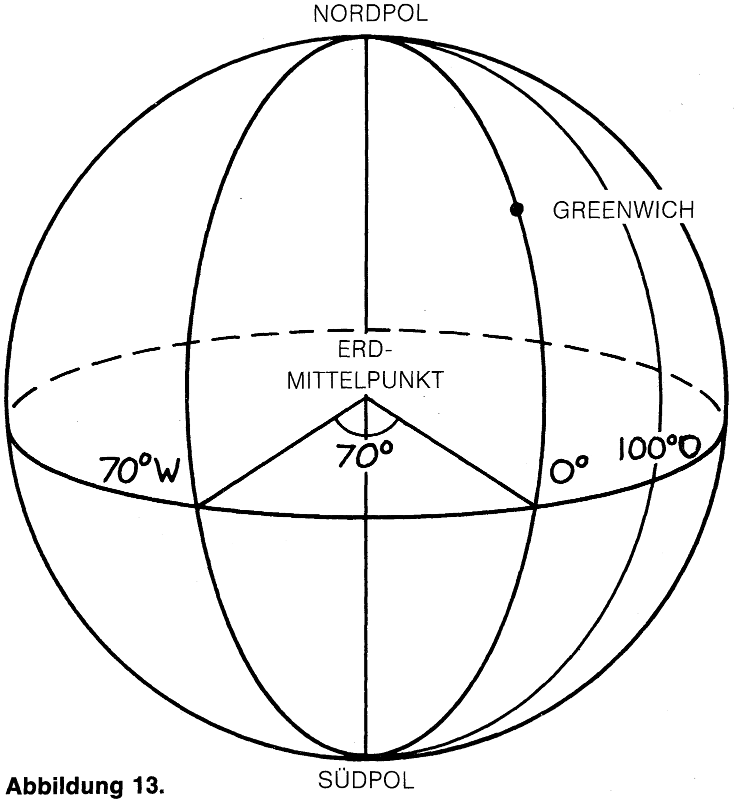


Abbildung 13.

Es ist nicht leicht, die (kürzeste) Entfernung (entlang eines Großkreises) zwischen zwei Punkten auf der Erde zu berechenn. Wie weit ist z. B. Newcastle upon Tyne von Paris entfernt? (Paris liegt ungefähr bei 49° nördlicher Breite, 2° östlicher Länge.) Mit einem geeigneten Programm stellt das auf dem Commodore 64 kein Problem dar.

Das folgende Programm berechnet die Entfernung zwischen zwei Punkten auf der Erdoberfläche. Die Mathematik hinter diesem Programm beruht auf den Cosinus- und Sinus-Gesetzen, die im vorigen Kapitel behandelt wurden.

```
10 REM 4ERDTRIGONOMETRIE
20 POKE53280,6:PRINTCHR$(147)"         ***
ERDTRIGONOMETRIE ***"
25 PRINT"          (GROSSKREISNAVIGATION) "
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 DIE KUERZESTE ENTFERNUNG ZWISCHEN";
```

```
40 PRINT"ZWEI PUNKTEN AUF DER ERDE"
50 REM EINGABE DATEN
60 FOR I=1TO2
70 PRINT:PRINTCHR$(144)"POSITION";I:PRIN
T
80 PRINTCHR$(154):INPUT"LAENGE    ";A(I)
90 IFA(I)<0 OR A(I)>90 THEN PRINT: PRINT
"!! ZWISCHEN 0 UND 90 GRAD !!":GOTO80
100 INPUT "N ODER S ";A$
110 IF A$="N"ORA$="S"THEN120
115 PRINT:PRINT"!! NORD ODER SUED !!":PR
INT:GOTO100
120 IFA$="S"THENA(I)=-A(I)
130 PRINT:INPUT"BREITE    ";B(I)
140 IFB(I)<0 OR B(I)>180THENPRINT:PRINT"
!! ZWISCHEN 0 UND 180 GRAD !!":GOTO130
150 INPUT "O ODER W ";A$
160 IF A$="O"ORA$="W"THEN170
165 PRINT:PRINT"!! OST ODER WEST !!":PRI
NT:GOTO150
170 IF A$="O" THENB(I)=-B(I)
180 NEXT
190 PRINT CHR$(147):PRINT:PRINT"ENTFERNU
NG IN KILOMETERN ODER MEILEN ?"
200 INPUT "K ODER M ";A$
210 IF A$="K"ORA$="M"THEN220
215 GOTO200
220 R=6371:B$="KILOMETER":IFA$="M"THENB$
="MEILEN":R=3960
230 REM BERECHNUNG
240 A1=A(1)*π/180:A2=A(2)*π/180:B1=B(1)*
π/180:B2=B(2)*π/180
250 B=ABS(B(1)-B(2)):IFB>180THENB=180-B
260 A=ABS(A(1)-A(2))*π/360:B=B*π/360
270 X=COS(A1)*SIN(B)*COS(A2)*SIN(B)+SIN(
A)*SIN(A)
280 D=2*R*ATN(SQR(X/(1-X)))
290 PRINTCHR$(17)CHR$(17)CHR$(144)"ENTFE
RNUNG =";INT(D*100+.5)/100;B$
300 PRINT:PRINT:PRINTCHR$(154)"NEUER STA
```

```
RT (J/N)?"
310 GET A$:IFA$="" THEN310
320 IFA$="J"THENRUN
330 IFA$="N"THENPRINTCHR$(147)CHR$(154)"
AUF WIEDERSEHEN"CHR$(154):END
340 GOTO310
```

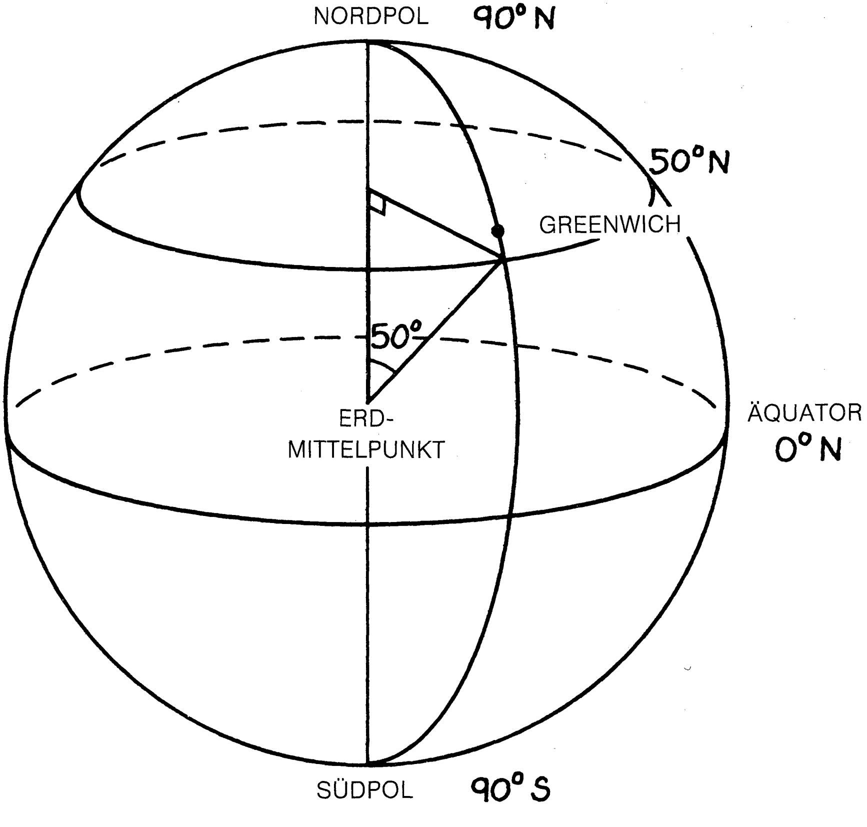


Abbildung 14.

KAPITEL 4

POTENZEN

Das Quadrat einer Zahl X ist das Produkt der Zahl mit sich selbst, also X*X. Es wird mit X^2 oder X↑2 bezeichnet. Man nennt es auch die zweite Potenz von X.
Die Potenzen von X sind entsprechende mehrfache Produkte der Zahl X mit sich selbst. Die erste, zweite, dritte, vierte und fünfte Potenz von 2 sind die Zahlen

2, 4, 8, 16 und 32.

Man bezeichnet diese Potenzen entsprechend mit

2^1, 2^2, 2^3, 2^4 und 2^5.

Mit dem folgenden einfachen Programm können Sie die Potenzen von 2 (oder jeder anderen Zahl) berechnen:

```
10 REM 5POTENZEN
20 POKE53280,6:PRINTCHR$( 147)CHR$( 5)"
      *** POTENZEN *** "
30 PRINT:PRINT"DIESES PROGRAMM ERRECHNET
 DIE POTENZEN  EINER EINGEGEBENEN ZAHL"
40 PRINT:INPUT "ZAHL EINGEBEN    ";X
45 IFX=0THEN40
50 PRINT:INPUT "WIEVIEL POTENZEN ";N
60 IF N< 1ORN< >INT( N)THENPRINT:PRINT"NUR
NATUERLICHE ZAHLEN BITTE":GOTO50
70 REM BERECHNUNG
80 PRINTCHR$( 147):PRINT:PRINT" I  "X"↑I"
86 PRINT
90 Y=1
100 FOR I=1TON:Y=Y*X:PRINTI;Y:NEXT
110 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
120 GET A$: IFA$=""THEN120
130 IFA$="J"THENRUN
140 IFA$="N"THENPRINTCHR$( 154)CHR$( 147)"
AUF WIEDERSEHEN":END
150 GOTO120
```

Falls die Zahl N zu groß ist, wird OVERFLOW ERROR gemeldet.

Mit einigen zusätzlichen Zeilen können Sie ein mögliches Überlaufen testen und vermeiden:

```
10 REM 5POTENZEN 1
20 POKE53280,6:PRINTCHR$(147)CHR$(5)"
      *** POTENZEN *** "
30 PRINT:PRINT"DIESES PROGRAMM ERRECHNET
 DIE POTENZEN  EINER EINGEGEBENEN ZAHL"
40 PRINT:INPUT "ZAHL EINGEBEN    ";X
45 IFX=0THEN40
50 PRINT:INPUT "WIEVIEL POTENZEN ";N
60 IF N<1ORN<>INT(N)THENPRINT:PRINT"NUR
NATUERLICHE ZAHLEN BITTE":GOTO50
61 IF(N+1)*LOG(ABS(X))<126*LOG(2)THEN70
62 PRINT:PRINT"RECHNERKAPAZITAET WIRD UE
BERSCHRITTEN !"
63 PRINT:PRINT"NEU START (J/N) ?"
64 GETA$:IFA$=""THEN64
65 IFA$="N"THEN70
66 IFA$="J"THENRUN
67 GOTO64
70 REM BERECHNUNG
80 PRINTCHR$(147):PRINT:PRINT" I "X"↑I"
86 PRINT
90 Y=1
100 FOR I=1TON:Y=Y*X:PRINTI;Y:NEXT
110 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
120 GET A$:IFA$=""THEN120
130 IFA$="J"THENRUN
140 IFA$="N"THENPRINTCHR$(154)CHR$(147)"
AUF WIEDERSEHEN":END
150 GOTO120
```

Der Commodore 64 kann die N-te Potenz einer Zahl X auch mit PRINT X↑N berechnen.

```
10 REM 6POTENZEN 2
20 POKE53280,6:PRINTCHR$(147)CHR$(5)"
      *** POTENZEN *** "
```

```
30 PRINT:PRINT"DIESES PROGRAMM ERRECHNET
 DIE POTENZEN  EINER EINGEGEBENEN ZAHL"
40 PRINT:INPUT "ZAHL EINGEBEN    ";X
45 IFX=0THEN40
50 PRINT:INPUT "WIEVIEL POTENZEN ";N
60 IF N<1ORN<>INT(N)THENPRINT:PRINT"NUR
NATUERLICHE ZAHLEN BITTE":GOTO50
61 IF(N+1)*LOG(ABS(X))<126*LOG(2)THEN70
62 PRINT:PRINT"RECHNERKAPAZITAET WIRD UE
BERSCHRITTEN !"
63 PRINT:PRINT"NEUSTART  (J/N)  ?"
64 GETA$:IFA$=""THEN64
65 IFA$="N"THEN70
66 IFA$="J"THENRUN
67 GOTO64
70 REM BERECHNUNG
80 PRINT:PRINT" I  "X"↑I"
86 PRINT
100 FOR I=1TON:PRINTI;X↑I:NEXT
110 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
120 GET A$:IFA$=""THEN120
130 IFA$="J"THENRUN
140 IFA$="N"THENPRINTCHR$(154)CHR$(147)"
AUF WIEDERSEHEN":END
150 GOTO120
```

Wie Sie vielleicht wissen oder leicht nachprüfen können, braucht N keine ganze Zahl zu sein.
Aber was bedeutet 2↑1.7? Es sollte natürlich eine Zahl zwischen 2↑1 und 2↑2 sein, also zwischen 2 und 4. Tatsächlich hat 2↑1.7 ungefähr den Wert 3.24900959. Näherungsweise kann man den Wert von 2↑1.7 (ohne Computer) mit Hilfe von Millimeterpapier bestimmen. Tragen Sie die erste bis fünfte Potenz von 2 ein – siehe **Abbildung 15.** Zeichnen Sie eine glatte Kurve durch diese Punkte – siehe **Abbildung 16.** Der ungefähre Wert von 2↑1.7 läßt sich aus dieser Darstellung ablesen.

Ähnlich könnte man die Potenzen anderer Zahlen graphisch darstellen. Das ist natürlich unnötig, da Ihr Commodore 64 Ihnen die Antwort sofort gibt.
Negative Potenzen von Zahlen sind ebenfalls sinnvoll; sie sind einfach durch folgende Regel definiert:

$X^{-N} = 1/X^N$.

Also ist $2^{-2} = 1/2^2$, das heißt 1/4 oder 0.25. Die nullte Potenz einer Zahl ist konventionsgemäß gleich 1.
Potenzen von Zahlen gehorchen der folgenden einfachen Multiplikationsregel:

$X^M * X^N = X^{M+N}$.

So ist zum Beispiel

$3^4 * 3^2 = 3^6$

und

$10^{-2} * 10^3 = 10^1$.

QUADRATWURZELN

Für eine Zahl X hat die Zahl $X^{1/2}$ einen besonderen Namen: sie heißt *Quadratwurzel* von X. Die Quadratwurzel einer Zahl ist diejenige Zahl, deren Quadrat die ursprüngliche Zahl ergibt. So ist die Quadratwurzel von 9 gleich 3 und die Quadratwurzel von 2 ungefähr 1.41421356, wie Sie leicht nachprüfen können, indem Sie diese Zahl mit sich selbst multiplizieren.

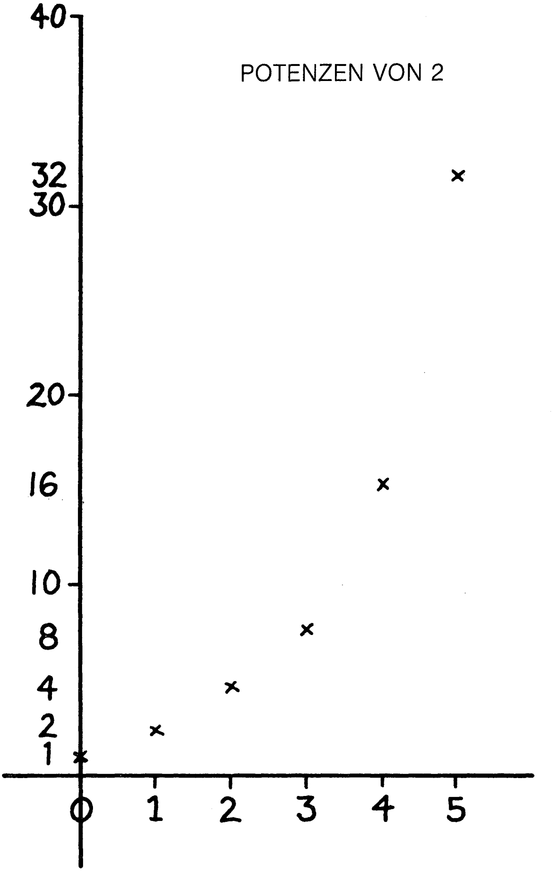


Abbildung 15.

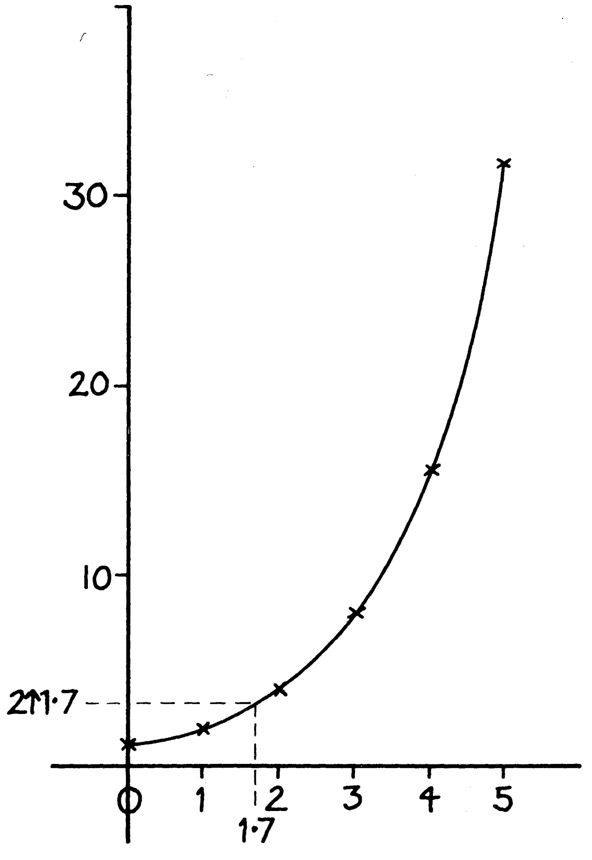


Abbildung 16. Ablesen von 2↑1.7 aus der Graphik

Die Schreibweise $X^{1/2}$ für die Quadratwurzel von X paßt zur Multiplikationsregel für Potenzen:

$$X^{1/2} * X^{1/2} = X^1$$

Das Ergebnis ist natürlich X.
Die Quadratwurzel einer Zahl X kann man auch mit dem Commodore 64 bestimmen, indem man PRINT SQR(X) eingibt.

IMAGINÄRE ZAHLEN

Das Quadrat einer Zahl ist immer positiv, und deshalb sollte man nicht erwarten, die Quadratwurzel einer negativen Zahl finden zu können. Tatsächlich wird Ihnen der Commodore 64 auf die Frage PRINT SQR(−1) mit ILLEGAL QUANTITY ERROR antworten.
Vielleicht haben Sie aber schon von imaginären und komplexen Zahlen gehört, die den Quadratwurzeln negativer Zahlen entsprechen. Mathematiker lassen sich niemals von scheinbar Unmöglichem wie der Quadratwurzel von −1 abschrecken. Man schafft einfach ein neues Symbol für diese Zahl. So schreibt man I für die Quadratwurzel von −1. Daran ist nichts Seltsames; I existiert ebenso wie negative Zahlen.
Man kann I zu sich selbst und anderen Zahlen addieren und mit anderen Zahlen multiplizieren. Auf diese Weise entstehen Zahlen wie

$$2 + 3*I,\ 1.41412*I,\ 9 - I,\ 10 - 8*I.$$

Zahlen, die I enthalten, heißen komplexe Zahlen, im Unterschied zu gewöhnlichen oder reellen Zahlen. Jede komplexe Zahl läßt sich in der Form

$$X + Y*I$$

schreiben, wobei X und Y reelle Zahlen sind. X heißt Realteil und Y Imaginärteil der komplexen Zahl.
Sobald wir einmal das Symbol I für SQR(−1) gewählt haben, können wir die Quadratwurzeln anderer negativer Zahlen bestimmen.

```
SQR(X)  = SQR(ABS(X)*SGN(X))
        = SQR(ABS(X)) * SQR(SGN(X))
```

Zum Beispiel gilt

SQR(−9) = SQR(9) * SQR(−1)
= 3*I.

Natürlich arbeitet der Commodore 64 nicht in dieser Weise, denn er kennt keine komplexen Zahlen.

Das folgende Programm zeigt, wie man komplexe Zahlen auf dem Commodore 64 darstellen kann. So lassen sich einfache Rechnungen durchführen. Beachten Sie, daß komplexe Zahlen nach dieser Regel multipliziert werden:

(A + B*I) + (C + D*I) = A*C + A*D*I + B*I*C + B*I*D*I
= A*C + A*D*I + B*C*I + B*D*I*I
= A*C + A*D*I + B*C*I + B*D*−1
= A*C − B*D + (A*D + B*C)*I

```
10 REM 7KOMPLEXE ZAHLEN
20 POKE53280,6:PRINTCHR$(147)"          ***
 KOMPLEXE ZAHLEN ***"
30 PRINT:PRINT"DIESES PROGRAMM MULTIPLIZ
IERT UND DIVI- DIERT KOMPLEXE ZAHLEN.
40 PRINT:PRINT"KOMPLEXE ZAHLEN WIE GEFRA
GT EINGEBEN"
60 PRINT:PRINT"ERSTE  ZAHL"
70 PRINT:INPUT"REALTEIL        ";A
80 INPUT"IMAGINAERTEIL ";B
90 PRINT:PRINT"ZWEITE ZAHL"
100 PRINT:INPUT"REALTEIL        ";C
110 INPUT"IMAGINAERTEIL ";D
120 PRINT:PRINT"WAS SOLL GEMACHT WERDEN
?"
130 PRINT:PRINT"1) SOLLEN DIE ZAHLEN MUL
TIPLIZIRT          WERDEN ?"
140 PRINT"2) SOLL DIE ERSTE VON DER ZWEI
TEN          ZAHL DIVIDIERT WERDEN ?"
150 PRINT:INPUT"BITTE 1 ODER 2 EINGEBEN"
;N
160 IFN=1ORN=2THEN170
165 GOTO150
170 ON N GOSUB 300,350
180 PRINT:PRINT:PRINT"EIN NEUER START (J
/N) ?"
```

```
190 GETA$:IFA$=""THEN190
200 IF A$="J"THENRUN
210 IF A$="N"THENPRINTCHR$(147)"AUF WIED
ERSEHEN":END
220 GOTO190
300 REM MULTIPLIKATION
310 PRINTCHR$(147):PRINT:PRINT:PRINT"PRO
DUKT DER ZAHLEN :"
320 PRINT:PRINT"REALTEIL        :";A*C-B*D

330 PRINT"IMAGINAERTEIL :";A*D+B*C
340 RETURN
350 REM DIVISION
360 X=C*C+D*D:IFX=0THENPRINTCHR$(147)"DI
VISION DURCH NULL NICHT MOEGLICH"RETURN
370 PRINTCHR$(147):PRINT:PRINT:PRINT"QUO
TIENT DER ZAHLEN :"
380 PRINT:PRINT"REALTEIL        :";(A*C+B*
D)/X
390 PRINT"IMAGINAERTEIL :";(-A*D+B*C)/X
400 RETURN
```

QUADRATISCHE GLEICHUNGEN

Quadratische Gleichungen ergeben sich bei der Lösung ganz unterschiedlicher Probleme. Die allgemeine Form einer quadratischen Gleichung ist

$$A*X^2 + B*X + C = 0$$

wobei A, B und C bekannte Zahlen sind und A von Null verschieden ist. Das Problem besteht darin, diejenigen Zahlen X zu finden, die die Gleichung erfüllen. Sie heißen die *Wurzeln* der quadratischen Gleichung. Zum Beispiel erfüllen die Werte X=1 und X=2 die folgende quadratische Gleichung:

$$X^2 - 3*X + 2 = 0$$

wie Sie leicht überprüfen können.
Gewöhnlich hat eine quadratische Gleichung zwei Wurzeln, und häufig sind die Wurzeln komplexe Zahlen.

Es gibt eine ganz einfache Formel für die Wurzeln einer quadratischen Gleichung, und zwar sind die Wurzeln gegeben durch

$$\frac{(-B+SQR(B^2-4*A*C))}{2*A}$$

und

$$\frac{(-B-SQR(B^2-4*A*C))}{2*A}$$

Die Art der Wurzeln ergibt sich aus dem Teil der Formel, der die Quadratwurzel enthält,

$SQR(B^2 - 4*A*C)$

die sogenannte *Diskriminante.*

Wenn $B^2 - 4*A*C > 0$ ist, gibt es zwei reelle Wurzeln.
Wenn $B^2 - 4*A*C = 0$ ist, sind die beiden Wurzeln reell und stimmen überein.
Wenn $B^2 - 4*A*C < 0$ ist, gibt es zwei komplexe Wurzeln.

Das nächste Programm berechnet die Wurzeln einer quadratischen Gleichung.

```
10 REM 8QUADRATISCHE GLEICHUNGEN
20 POKE53280,6:PRINTCHR$(147)"    ***   QU
ADRATISCHE GLEICHUNGEN  ***"
30 PRINT:PRINT"DIESES PROGRAMM LOEST QUA
DRATISCHE      GLEICHUNGEN
40 PRINT:PRINT"GLEICHUNGSSCHEMA:   A*X*X
+ B*X + C = 0 "
50 PRINT:INPUT"WERT VON A";A
60 IFA=0THENPRINT"A KANN NICHT NULL SEIN
":GOTO50
70 PRINT:INPUT"WERT VON B";B
80 PRINT:INPUT"WERT VON C";C
90 REM BEGINN DER RECHNUNG
100 D=B*B-4*A*C
```

```
110 ON SGN(D)+2 GOSUB 200,300,400
120 PRINT:PRINT:PRINT"EIN NEUER START (J
/N) ?"
130 GETA$:IFA$=""THEN130
140 IFA$="J"THENRUN
150 IFA$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
200 REM ZWEI KOMPLEXE WURZELN
210 PRINT:PRINT:PRINT:PRINT"ES GIBT ZWEI
 KOMPLEXE WURZELN"
220 X=-B/A/2:Y=ABS(SQR(-D)/A/2)
230 PRINT:PRINTX;"+";Y;"*I"
240 PRINT:PRINTX;"-";Y;"*I"
250 RETURN
300 REM ZWEI GLEICHE WURZELN
310 PRINT:PRINT:PRINT:PRINT"ES GIBT ZWEI
 GLEICHE WURZELN"
320 PRINT:PRINT-B/A/2
330 RETURN
400 REM ZWEI REALE WURZELN
410 PRINT:PRINT:PRINT:PRINT"ES GIBT ZWEI
 REALE WURZELN"
420 D=SQR(D):X=(-B-D)/2:IFB<0THENX=(-B+D
)/2
430 PRINT:PRINTX/A:PRINTC/X
440 RETURN
```

Falls es zwei reelle Wurzeln gibt, berechnet das Programm mit Hilfe der Formel für die Wurzeln zunächst eine Wurzel, und zwar die mit dem größten Betrag. Die andere Wurzel ergibt sich dann daraus, daß das Produkt der beiden Wurzeln gleich C/A ist.

LÖSEN ANDERER GLEICHUNGEN

Quadratische Gleichungen sind recht einfach zu lösen. Das gilt nicht für andere Gleichungen wie

$$9*X^5 - 3*X^4 + X^3 - X^2 + 5*X - 4 = 0$$

Wenn die Gleichung nur nichtnegative ganze Potenzen enthält wie in diesem Beispiel, wird sie *polynomische* Gleichung genannt. Die größte von Null verschiedene Potenz darin heißt der *Grad* des Polynoms. Die obige Gleichung hat den Grad 5. Eine quadratische Gleichung ist eine polynomische Gleichung vom Grad 2. Quadratische Gleichungen haben gewöhnlich 2 Wurzeln; polynomische Gleichungen vom Grad N haben gewöhnlich N Wurzeln.

Mit Ausnahme einiger Sonderfälle gibt es keine allgemeinen Formeln zur Lösung polynomischer Gleichungen. Das hat für die Mathematik sehr interessante Konsequenzen gehabt, aber das ist ein anderes Thema, auf das wir hier nicht eingehen werden.

Wir können jedoch unseren Commodore 64 dazu benutzen, durch wiederholtes Raten die Wurzeln einer polynomischen Gleichung zu bestimmen. Im wesentlichen prüft der Computer viele verschiedene Zahlen daraufhin, ob sie die Gleichung erfüllen.

Das nächste Programm liefert ein Verfahren zur Bestimmung einer reellen Wurzel einer polynomischen Gleichung. Es ist mathematisch grob und findet gelegentlich auch dann keine Wurzel, wenn es eine gibt. Die zugrundeliegende Methode läßt sich jedoch an diesem Programm erkennen:

```
10 REM 9POLYNOME
20 POKE53280,6:PRINTCHR$(147)"    *** POL
YNOMISCHE GLEICHUNGEN ***"
30 PRINT:PRINT"DIESES PROGRAMM VERSUCHT
LOESUNGEN ZU   FINDEN FUER GLEICHUN";
35 PRINT"GEN WIE :"
40 PRINT:PRINT"A*X↑N + B*X↑(N-1) + ... +
 C*X + D = 0"
50 PRINT
60 INPUT "GRAD DES POLYNOMS ";N
70 IF N<2 OR N<>INT(N) THENPRINT"NATUERL
ICHE ZAHLEN EINGEBEN":GOTO60
80 DIMA(N):PRINT:PRINT"KOEFFIZENTEN TERM
 NACH TERM EINGEBEN"
85 PRINT
90 FOR I=0 TO N
100 IF I<N THENPRINT" KOEFFIZENT FUER X↑
";MID$(STR$(N-I),2);
110 IF I=N THENPRINT" KONSTANTE (D)";
120 INPUTA(I)
130 IFA(0)=0THENPRINT"KEINE NULL BITTE":
GOTO100
```

```
140 NEXT
150 PRINT:PRINT"GEBEN SIE DIE GRENZEN EI
N, IN DENEN DIE WURZELN GESUCHT WERDEN";

160 PRINT" SOLLEN"
170 PRINT:INPUT"UNTERER WERT";A
180 INPUT"OBERER  WERT";B
190 IFA>=BTHENPRINT:PRINT"ERSTER WERT MU
SS KLEINER SEIN":GOTO170
200 REM SUCHE
210 S=B-A:T=0:TEST=-1:D=1E-9
220 PRINTCHR$(147):PRINT:PRINT"TEST LAUF
";T+1;"**";10↑T;"DIVISIONEN **":GOSUB300

230 IF TEST THEN S=S/10:T=T+1:IF T<4 THE
N 220
240 IFT=4THENPRINT:PRINT"KANN LEIDER KEI
NE WURZEL FINDEN"
250 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
260 GETG$:IFG$=""THEN260
270 IFG$="J"THENRUN
280 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
290 GOTO260
300 REM SCHRITT BEI SCHRITT SUCHE
310 X=A:GOSUB400:X1=A:Y1=Y
320 FORX=(A+S)TOBSTEPS
330 GOSUB400:X2=X:Y2=Y
340 IFY1*Y2<=DTHENGOSUB500
350 Y1=Y2:X1=X
360 NEXT
370 RETURN
400 REM BERECHNUNG DES POLYNOMS
410 Y=A(0):FORI=1TON:Y=Y*X+A(I):NEXT
420 RETURN
500 REM SCHLUSS BERECHNUNG
510 PRINT:PRINT"SCHLUSS BERECHNUNG":B$="
"
520 IFABS(Y1)<DTHENX=X1:GOTO590
```

```
530 IFABS(Y2)<DTHENX=X1:GOTO590
540 Z=X:X=(X1+X2)/2:GOSUB400
550 IF ABS(Y)<DTHENB$="WAHRSCHEINLICH":G
OTO590
570 IFY*Y2>0THENX2=X:GOTO540
580 X1=X:GOTO540
590 PRINT:PRINTB$" WURZEL BEI";X:TEST=0:
X=B
600 RETURN
```

Manchmal sagt das Programm von einer Zahl, daß sie *vielleicht* eine Wurzel ist. Um zu prüfen, ob sie tatsächlich eine ist, oder wie nahe sie einer Wurzel ist, beantworten Sie zunächst die Frage nach einem weiteren Programmdurchlauf mit N, und geben Sie dann folgendes ein:

X=Z:GOSUB 400:PRINT Y

Der Wert der daraufhin ausgegebenen Zahl sagt Ihnen, wie nahe Sie an einer Wurzel sind.

DAS NEWTONSCHE VERFAHREN

Das im vorigen Abschnitt angegebene Verfahren zur Bestimmung von Wurzeln einer Gleichung läßt sich mit dem sogenannten Newtonschen Verfahren verbessern.
Angenommen, wir wollen Wurzeln der polynomischen Gleichung

$$9*X^5 - 3*X^4 + X^3 - X^2 + 5*X - 4 = 0$$

finden. Wir bezeichnen das Polynom mit P(X) und schreiben P'(X) für das folgende Polynom:

$$5*9*X^4 - 4*3*X^3 + 3*X^2 - 2*X + 5.$$

P'(X) ist die *Ableitung* von P(X), aber das braucht uns nicht zu kümmern. Wenn Y näherungsweise eine Wurzel der Gleichung P(X) = 0 ist, so ist

Y – P(Y)/P'(Y)

gewöhnlich eine bessere Näherung, falls P'(Y) verschieden von Null ist.
Das nächste Programm verwendet diese Technik zur Bestimmung von Wurzeln polynomischer Gleichungen.

```
10 REM 10POLYNOME 2
20 POKE53280,6:PRINTCHR$(147)"    *** POL
YNOMISCHE GLEICHUNGEN ***"
25 PRINT"         *** NEWTON'S METHODE ***
"
30 PRINT:PRINT"DIESES PROGRAMM FINDET LO
ESUNGEN NACH   NEWTON'S METHODE FUER ";
35 PRINT"GLEICHUNGEN WIE   "
40 PRINT:PRINT"A*X↑N + B*X↑(N-1) + ... +
 C*X + D = 0"
50 PRINT
60 INPUT "GRAD DES POLYNOMS ";N
70 IF N<2 OR N<>INT(N) THENPRINT"NATUERL
ICHE ZAHLEN EINGEBEN":GOTO60
80 DIMA(N):PRINT:PRINT"KOEFFIZENTEN TERM
 NACH TERM EINGEBEN"
85 PRINT
90 FOR I=0 TO N
100 IF I<N THENPRINT" KOEFFIZENT FUER X↑
";MID$(STR$(N-I),2);
110 IF I=N THENPRINT" KONSTANTE (D)";
120 INPUTA(I)
130 IFA(0)=0THENPRINT"KEINE NULL BITTE":
GOTO100
140 NEXT
150 PRINT:PRINT"GEBEN SIE EINEN ANGENOMM
EN WERT FUER DIEWURZELN EIN";
160 PRINT:INPUT"WERT";X
170 REM BERECHNUNG P'(X)
180 DIM B(N):FORI=0TON:B(I)=(N-1)*A(I):N
EXT
190 GOSUB500:REM WURZELN FINDEN
200 REM ENDE
210 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
220 GETG$:IFG$=""THEN220
```

```
230  IFG$="J"THENRUN
240  IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
250  GOTO210
400  REM LOESUNG POLYNOM
410  Y=A(0):FORI=1TON:Y=Y*X+A(I):NEXT
420  Y1=B(0):FORI=1TO(N-1):Y1=Y1*X+B(I):N
EXT
430  RETURN
500  REM LOESUNGEN FINDEN
505  PRINTCHR$(147)
510  J=1:D=1E-9:GOSUB400
520  IFABS(Y)<DTHEN590
530  IFY1=0THENPRINT"DIVISION DURCH NULL
- NEUER VERSUCH":RETURN
540  Z=X:X=X-Y/Y1:J=J+1:GOSUB400
550  IFABS(Y)<DTHEN590
560  IFABS(Z-X)<DTHENB$="WAHRSCHEINLICH "
:GOTO590
570  IFJ>1000THENPRINT"LEIDER KEINE WURZE
L GEFUNDEN"
580  GOTO530
590  PRINT:PRINTB$"WURZEL BEI";X
600  RETURN
```

DIE EXPONENTIALFUNKTION

Funktionen wie 2^x oder 10^x heißen *Potenzfunktionen,* weil die Variable X als Potenz eingeht. Potenzfunktionen erhält man auf dem Commodore 64 durch PRINT 2↑X etc. Eine wichtige Potenzfunktion ist im Commodore 64 eingebaut: die Exponentialfunktion EXP(X).

Die Exponentialfunktion beruht auf Potenzen der Zahl E, die ungefähr den Wert 2.71828183 hat. Verwechseln Sie dieses E nicht mit dem E, das in der Gleitkommadarstellung von Zahlen vorkommt. Die Zahl E ist definiert durch

$$E = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \frac{1}{5!} + \frac{1}{6!} + \dots$$

Dabei bedeutet . . ., daß die Summe endlos fortgesetzt ist, und das Symbol ! steht für Fakultät, definiert durch

N! = N * (N −1) * (N−2) * . . . * 2 * 1

also das Produkt der ganzen Zahlen von 1 bis N. Z.B. ist 4! gleich 4*3*2*1, also 24. Die Exponentialfunktion ist definiert durch

$$\begin{aligned} \text{EXP(X)} &= \text{E}^{\text{X}} \\ &= \text{E}\uparrow\text{X}. \end{aligned}$$

Insbesondere gilt also E = EXP(1). Geben Sie Ihrem 64 versuchsweise folgendes ein:

```
E = EXP(1)
PRINT E↑5, EXP(5)
```

Aus den Eigenschaften von Potenzfunktionen ergeben sich die Eigenschaften der Exponentialfunktion:

EXP(X)*EXP(Y) = EXP(X+Y)
EXP(X)/EXP(Y) = EXP(X−Y)

Mit Hilfe einer ganz einfachen Formel läßt sich EXP(X) für jedes X berechnen. Diese Formel sieht so aus:

$$\text{EXP(X)} = 1 + \frac{X}{1!} + \frac{X^2}{2!} + \frac{X^3}{3!} + \frac{X^4}{4!} + \frac{X^5}{5!} + \frac{X^6}{6!} + \ldots$$

DIE LOGARITHMUSFUNKTION

Welche Zahl X erfüllt EXP(X) = 3? Da EXP(1) = 2.71828183 gilt, muß X etwas größer als 1 sein. Tatsächlich lautet die Antwort: ungefähr 1.09861229.
Die Zahl X, die EXP(X) = N erfüllt, heißt (natürlicher) Logarithmus von N. Sie wird gewöhnlich mit LN(N) bezeichnet. Wie die meisten Mikrocomputer schreibt der Commodore sie jedoch als LOG(N).
Die Logarithmusfunktion hat folgende Eigenschaften:

LOG(X*Y) = LOG(X) + LOG(Y)
LOG(X/Y) = LOG(X) − LOG(Y)
$\text{LOG}(\text{X}^{\text{N}}) = \text{N*LOG(X)}$

Wegen dieser Eigenschaften sind Logarithmen wichtig für Multiplikationen, Divisionen etc. ohne Computer.
Zwischen der Exponentialfunktion und der Logarithmusfunktion bestehen die Beziehungen

EXP(LOG(N)) = N
LOG(EXP(N)) = N

Eine einfache Anwendung der Logarithmusfunktion ist das Testen großer Zahlen. Z.B. erfüllt die Zahl X die Beziehung

X< 10↑N

dann und nur dann, wenn die Beziehung gilt:

LOG(X)<N*LOG(10).

Eine solche Umformulierung ist nützlich, weil die Zahl 10↑N selbst einen OVERFLOW ERROR hervorrufen kann.

WURZELN ANDERER FUNKTIONEN

Ist eine Funktion wie zum Beispiel X*EXP(X)+1 gegeben, so versteht man unter einer Wurzel der Funktion eine Zahl, für die die Funktion den Wert 0 ergibt.
Zwei Programme zur Bestimmung der Wurzeln einer quadratischen Gleichung wurden weiter oben vorgestellt. Das erste davon läßt sich so verändern, daß es Wurzeln anderer Funktionen bestimmt. Wenn Sie das Programm starten, werden Sie zur Eingabe von zwei Zeilen aufgefordert:

100 DEF FNA(X) = Funktion von X eingeben
GOTO 100

Geben Sie in der ersten Zeile die Funktion ein, von der Sie die Wurzel bestimmen wollen.

```
10 REM 11WURZELN
20 POKE53280,6:PRINTCHR$(147)"   *** WURZ
ELN ANDERER FUNKTIONEN ***"
30 PRINT:PRINT"DIESE PROGRAMM VERSUCHT W
URZELN VON      FUNKTIONEN ZU FINDEN. ";
```

```
40 PRINT"BITTE FUNKTION IN DER FOLGENDEN
 FORM EINGEBEN:"
50 PRINT:PRINT"■100 DEF FNA(X) = X*EXP(X
)
60 PRINT:PRINT"      ◘DANACH"
70 PRINT:PRINT"■     GOTO 100◘":END
100 DEF FNA(X)=X*EXP(X)*LOG(ABS(X0)+1)
150 PRINT CHR$(147):PRINT:PRINT"GEBEN SI
E DIE GRENZEN EIN, IN DENEN EINE";
160 PRINT"WURZEL GESUCHT WERDEN SOLL !"
170 PRINT:INPUT"UNTERER WERT ";A
180 INPUT"OBERER  WERT ";B
190 IF A>B THEN PRINT:PRINT"ERSTER WERT
MUSS KLEINER SEIN"
200 REM SUCHE
210 S=B-A:T=0:TEST=-1:D=1E-9
220 PRINTCHR$(147):PRINT:PRINT"TEST LAUF
 ";T+1;" ** ";10↑T;" DIVISIONEN **":GOSU
B300
230 IF TEST THEN S=S/10:T=T+1:IF T<4 THE
N GOTO220
240 IF T=4 THEN PRINT:PRINT"LEIDER KEINE
 WURZEL GEFUNDEN"
250 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
260 GET G$:IF G$="" THEN 260
270 IF G$="J" THEN RUN
280 IF G$="N" THEN PRINT CHR$(147)"AUF W
IEDERSEHEN":END
300 REM SCHRITT FUER SCHRITT SUCHE
310 X=A:Y=FNA(X):X1=A:Y1=Y
320 FOR X=A+S TO B STEP S
330 Y=FNA(X):X2=X:Y2=Y
340 IF Y1*Y2<=D THEN GOSUB500
350 Y1=Y2:X1=X
360 NEXT
370 RETURN
500 REM SCHLUSSBERECHNUNG
510 PRINT:PRINT"SCHLUSSBERECHNUNG"
520 IF ABS(Y1)<D THEN X=X1:GOTO590
```

```
530 IF ABS(Y2)<D THEN X=X2:GOTO590
540 Z=X:X=(X1+X2)/2:Y=FNA(X)
550 IF ABS(Y)<D THEN590
560 IF ABS(Z-X)<D THEN B$="WAHRSCHEINLIC
H ":GOTO590
570 IF Y*Y2>0 THEN X2=X:GOTO540
580 X1=X:GOTO540
590 PRINT:PRINTB$"WURZEL BEI ";X:TEST=0:
X=B
600 RETURN
```

Wenn Sie ein Ergebnis mit dem Zusatz PROBABLY (vielleicht) erhalten, stoppen Sie das Programm und geben

PRINT FNA(Z)

ein, um zu ermitteln, wie nahe Sie an einer Wurzel sind.

KAPITEL 5

FOLGEN

Folgen (und Reihen) sind wichtige Begriffe, die überall vorkommen. Eine *Folge* ist einfach eine Liste von Zahlen, zum Beispiele

1, 2, 3, 4, 5, 19, 7, 8, 12
20, 18, 16, 14, 12, 10, 8
1, 0.1, 0.01, 0.001, 0.0001. 0.00001. 0.000001

Die einzelnen Zahlen oder Elemente der Folge heißen *Glieder* der Folge. Gewöhnlich ist eine Folge durch eine bestimmte Regel erzeugt, wie etwa die letzten beiden oben. Die zweite ist aus der Formel 22 − 2*N für N = 1 bis 7 entstanden, und die letzte aus $10/10^N$ für N = 1 bis 7. Ihr Commodore 64 kann leicht Folgen erzeugen, wie das nächste Programm zeigt. Setzen Sie Ihre eigene Formel (von N abhängig) in der zweiten Zeile ein.

```
10 REM SEQUENCE GENERATOR
20 DEF FNA(N)= (von N abhängige Formel einsetzen)
30 FOR N = 1 TO 10
40 PRINT FNA(N);:IF N < 10 THEN PRINT CHR$(157);",";
50 NEXT
60 PRINT
```

Hier sind einige mit diesem Programm erzeugte Folgen. Können Sie die jeweils zugrundeliegende Formel erkennen? Überprüfen Sie Ihre Vermutung, indem Sie die Formel in das obige Programm einsetzen. (Die Antworten werden weiter unten in diesem Abschnitt gegeben; die nachfolgenden Abschnitte enthalten weitere Beispiele.)

(a) 1, 6, 11, 16, 21, 26, 31, 36, 41, 46
(b) 1, 2, 4, 8, 16, 32, 64, 128, 256, 512
(c) 1, 4, 9, 16, 25, 36, 49, 64, 81, 100
(d) 1, 2, 4, 7, 11, 16, 22, 29, 37, 46
(e) 4, 4, 8, 12, 20, 32, 52, 84, 136, 220

Für die ersten Folgen sind die Formeln nicht schwer zu bestimmen. Für (a) ist es 5*N−4, für (b) ist es 2↑N/2, und für (c) ist es N*N. Die vierte (d) ist nicht so leicht zu erraten, es ist (N*N − N + 2)/2. Auf die Formel für die fünfte Folge (e) schließlich kann man unmöglich kommen, wenn man sie noch nie gesehen hat; sie lautet:

1+(−1)↑N
N*(−1)↑N
INT(SIN(N)*10)

ARITHMETISCHE FOLGEN

Eine *arithmetische* Folge oder arithmetische Progression ist eine Folge, in der jedes Glied die Summe des vorhergehenden Gliedes und einer Konstanten ist. Die allgemeine Formel für eine arithmetische Folge ist

A + (N − 1)*D

wobei A das erste Glied der Folge und D die gemeinsame Differenz ist. Hier sind noch mehr Beispiele für arithmetische Folgen:

5, 10, 15, 20, 25, 30, 35, 40, 45, 50	(A = 5, D = 5)
1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5	(A = 1, D = 0.5)
0, 2, 4, 6, 8, 10, 12, 14, 16, 18	(A = 0, D = 2)

Das unten abgedruckte Programm kann Ihnen helfen, arithmetische Folgen zu untersuchen. Sie geben das erste Glied der Folge, die gemeinsame Differenz und die gewünschte Anzahl der Glieder ein. Beachten Sie, daß keine Formel erforderlich ist, weil der Commodore 64 die Folge iterativ berechnet. Zusätzlich berechnet das Programm die Summe der Folgeglieder und druckt das Ergebnis aus.

```
10 REM 12FOLGEN ARITHMETRISCHE
20 POKE53280,6:PRINTCHR$(147)"       *** A
RITHMETISCHE FOLGEN ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 ARITHMETISCHE FOLGEN"
40 PRINT:INPUT"ERSTE GLIED DER FOLGE";A
50 PRINT:INPUT"GEMEINSAME DIFFERENZ ";D
60 PRINT:INPUT"ANZAHL DER GLIEDER   ";N
70 IF N<1 OR N<>INT(N) THEN 60
80 REM DIE FOLGE
90 PRINTCHR$(147):PRINT:PRINT"DIE FOLGE
IN DEN GRENZEN 1 BIS";N
100 TERM=A:SUM=0
105 PRINT
110 FOR I=1 TO N
```

```
120 IF38-POS(0)<LEN(STR$(TERM)) THENPRIN
T
130 PRINTCHR$(26)TERM
140 SUM=SUM+TERM:TERM=TERM+D
150 NEXT I
160 PRINT:PRINT:PRINT"DIE SUMME = "SUM
170 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N)"
180 GETG$:IFG$=""THEN180
190 IFG$="J"THENRUN
200 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
210 GOTO180
```

WAS MÖCHTEN SIE LIEBER?

Angenommen, der Verleger dieses Buches bietet Ihnen eine Stelle als freier Mitarbeiter an und fragt Sie dann, wie Sie am liebsten bezahlt werden möchten: 'Was möchten Sie lieber? Mit DM 3000 für sechs Monate anfangen und dann alle sechs Monate DM 120 mehr bekommen, oder mit DM 6120 für ein Jahr anfangen und dann jedes Jahr DM 240 mehr bekommen?' Die in einem Jahr erhaltenen Geldbeträge bilden in beiden Fällen eine arithmetische Folge. Aber ein Angebot ist viel besser als das andere: das erste. Erkennen Sie warum? Sehen Sie sich die folgenden Berechnungen an:

	Erstes Angebot	Zweites Angebot
Erstes Jahr	(Die ersten 6 Monate DM 3000)	
	(Die zweiten 6 Monate DM 3120)	
	DM 6120	DM 6120
Zweites Jahr	(Die ersten 6 Monate DM 3240)	
	(Die zweiten 6 Monate DM 3360)	
	DM 6600	DM 6360
Drittes Jahr	(Die ersten 6 Monate DM 3480)	
	(Die zweiten 6 Monate DM 3600)	
	DM 7080	DM 6600

Sie sehen, daß das erste Angebot besser ist, wenn Sie die Stelle länger als ein Jahr behalten. Beachten Sie, daß beim ersten Angebot das Gehalt für die ersten 6 Monate jeden Jahres um DM 240 wächst, demnach die jährliche Erhöhung DM 480 beträgt. Das jährliche Gehalt beim ersten Angebot ist eine arithmetische Folge nach der Formel:

6120 + (N−1)*480,

beim zweiten dagegen nach der Formel:

6120 + (N−1)*240

Angenommen, Sie bekämen ein drittes Angebot: 'Sie fangen mit DM 1440 im Quartal an und bekommen alle drei Monate DM 60 mehr.' Welches Angebot würden Sie jetzt vorziehen? Die Antwort ist Ihnen hoffentlich klar. Die Rechnung sähe so aus:

	Drittes Angebot	
Erstes Jahr	(Die ersten 3 Monate DM 1440)	
	(Die zweiten 3 Monate DM 1500)	
	(Die dritten 3 Monate DM 1560)	
	(Die vierten 3 Monate DM 1620)	DM 6120
Zweites Jahr	(Die ersten 3 Monate DM 1680)	
	(Die zweiten 3 Monate DM 1740)	
	(Die dritten 3 Monate DM 1800)	
	(Die vierten 3 Monate DM 1860)	DM 7080

Die Erhöhung für ein Quartal beträgt von Jahr zu Jahr DM 240, aber das bekommen Sie alle drei Monate. Also ist die jährliche Erhöhung DM 960.

GEOMETRISCHE FOLGEN

Eine andere verbreitete Art von Folgen ist die *geometrische* Folge oder geometrische Progression. In einer geometrischen Folge ist das Verhältnis zweier aufeinander folgender Glieder konstant, zum Beispiel

2, 6, 18, 54, 162

wo jedes Glied (außer dem ersten) das Dreifache des vorhergehenden ist. Die allgemeine Formel für eine geometrische Folge ist

A * R↑(N−1)

wobei A das erste Glied und R das gemeinsame Verhältnis ist.
Hier sind noch mehr Beispiele für geometrische Folgen:

4, 2, 1, 0.5, 0.25, 0.125, 0.0625	(A = 4, R = 0.5)
2, −4, 8, −16, 32, −64, 128	(A = 2, R = −2)

Das nächste Programm kann Ihnen helfen, geometrische Folgen zu untersuchen. Sie geben das erste Glied der Folge, das gemeinsame Verhältnis und die gewünschte Anzahl der Glieder ein. Beachten Sie, daß keine Formel erforderlich ist, weil der Commodore 64 die Folge iterativ berechnet. Zusätzlich addiert das Programm alle Glieder der Folge und druckt das Ergebnis aus.

```
10 REM 13FOLGEN GEOMETRISCHE
20 POKE53280,6:PRINTCHR$(147)"       *** G
EOMETRISCHE FOLGEN ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 GEOMETRISCHE  FOLGEN"
40 PRINT:INPUT"ERSTE GLIED DER FOLGE    "
;A
50 PRINT:INPUT"GEMEINSAMES VERHAELTNIS  "
;R
60 PRINT:INPUT"ANZAHL DER GLIEDER       "
;N
70 IF N<1 OR N<>INT(N) THEN 60
80 REM DIE FOLGE
90 PRINTCHR$(147):PRINT:PRINT"DIE FOLGE
IN DEN GRENZEN 1 BIS";N
100 TERM=A:SUM=0
105 PRINT
110 FOR I=1 TO N
120 IF38-POS(0)<LEN(STR$(TERM)) THENPRIN
T
130 PRINTCHR$(26)TERM
140 SUM=SUM+TERM:TERM=TERM*R
150 NEXT I
160 PRINT:PRINT:PRINT"DIE SUMME = "SUM
170 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N)"
180 GETG$:IFG$=""THEN180
```

```
190  IFG$="J"THENRUN
200  IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
210 GOTO180
```

ZINSEN

Am 1. Januar zahlt eine Frau DM 100 in eine Bank ein, die jedes Jahr (am Jahresende) 6% Zinsen gewährt. Auf welchen Betrag sind die DM 100 nach 10 Jahren auf der Bank angewachsen?
Nach einem Jahr hat die

```
100 + 100 * 6/100
    = 100 + 100 * 0.06
    = 100 * 1.06
```

also DM 106. Nach zwei Jahren hat sie

```
100 * 1.06 + (100 * 1.06) * 0.06
           = 100 * 1.06 * 1.06
```

Wie Sie jetzt herausfinden können, hat sie nach 10 Jahren

```
100 * 1.06↑10.
```

Der Gesamtbetrag nach jedem Jahr bildet eine geometrische Reihe wie folgt:

```
100 * 1.06, 100 * 1.06↑2, 100 * 1.06↑3, 100 * 1.06↑4,
     100 * 1.06↑5, 100 * 1.06↑6, 100 * 1.06↑7, 100 * 1.06↑8,
          100 * 1.06↑9, 100 * 1.06↑10
```

Beginnt man allgemeiner mit einem Betrag A und erhält I% Zinsen im Jahr, dann ist der ursprüngliche Betrag nach N Jahren auf folgende Summe angewachsen:

```
A * (1 + I/100)↑N
```

TÄGLICHE ZINSEN

Werden DM 1000 bei einer Sparkasse angelegt, die am Ende jeden Jahres 6% Zinsen zahlt, so beträgt das Guthaben nach einem Jahr

1000 * 1.06

vorausgesetzt, es finden keine weiteren Einzahlungen oder Auszahlungen statt. Wenn jedoch die Bank alle 6 Monate Zinsen zahlt (und auch die schon gezahlten Zinsen verzinst, also Zinseszins zahlt), ist das Guthaben am Jahresende

1000 * (1.03)↑2.

Allgemeiner ausgedrückt: Zahlt die Sparkasse 6% Zinseszins N-mal im Jahr, so sind die DM 1000 am Ende eines Jahres auf den folgenden Betrag angewachsen:

1000 * (1 + 0.06/N)↑N

Nachstehende Tabelle zeigt die Beträge, die sich bei unterschiedlich häufiger Zinseszinszahlung ergeben:

N	6% Zinseszins	Guthaben am Ende des Jahres (auf Pfennig gerundet)
1	jährlich	DM 1060.60
2	halbjährlich	DM 1060.90
4	vierteljährlich	DM 1061.36
6	zweimonatlich	DM 1061.52
12	monatlich	DM 1061.68
52	wöchentlich	DM 1061.80
365	täglich	DM 1061.83
8760	stündlich	DM 1061.84

Diese Tabelle hat der Commodore 64 mit dem folgenden einfachen Programm erstellt:

```
10 REM 14ZINSESZINS
20 POKE53280,6:PRINTCHR$(147)"        ***
ZINSESZINS FORMEL ***"
25 PRINT:PRINT"WIE VERAENDERT SICH DER B
ETRAG VON        1000 DM, WENN IM JAHR ";
26 PRINT"MEHRFACH ZINSEN   GEZAHLT WERDE
N":PRINT
```

```
30 PRINT:PRINT"BITTE ANZAHL DER ZINSPERI
ODEN ANGEBEN ."
40 PRINT:INPUT"ZINSPERIODEN ";N:IFN<=0OR
N<>INT(N)THEN40
50 T=1000*(1+0.06/N)↑N:PRINT:PRINTINT(T*
100)/100
60 PRINT:PRINT"NEUER START (J/N) ?"
70 GETG$:IFG$=""THEN70
80 IFG$="J"THENRUN
90 IFG$="N"THENPRINTCHR$(147)"AUF WIEDER
SEHEN":END
100 GOTO70
```

Um andere Zinssätze einsetzen zu können, sind einige Zusätze und Änderungen erforderlich. Veränderungen und Zusätze im nächsten Listing sind durch ein Sternchen am Anfang einer Zeile gekennzeichnet.

```
10 REM 15ZINSESZINS VARIABELER ZINSSATZ
20 POKE53280,6:PRINTCHR$(147)" *** VARIA
BLE ZINSESZINS FORMEL ***"
30 PRINT:PRINT"WIE VERAENDERT SICH EIN B
ETRAG VON        1000 DM, WENN IM JAHR ";
40 PRINT"MEHRFACH ZINSEN   GEZAHLT WERDE
N":PRINT
50 PRINT:PRINT"BITTE ZINSSATZ EINGEBEN"
60 PRINT:INPUT"ZINSSATZ       ";I
70 PRINT:PRINT"BITTE ANZAHL DER ZINSPERI
ODEN ANGEBEN ."
80 PRINT:INPUT"ZINSPERIODEN ";N:IFN<=0OR
N<>INT(N)THEN80
90 T=1000*(1+I/100/N)↑N:PRINT:PRINTINT(T
*100)/100
100 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
110 GETG$:IFG$=""THEN110
120 IFG$="J"THENRUN
130 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
140 GOTO110
```

VERDOPPELN ODER AUFGEBEN

Manche halten es für möglich, bei Glücksspielen niemals zu verlieren.
Nehmen Sie zur Veranschaulichung dieses Glücksspiel: 'Eine unverfälschte Münze wird geworfen, gleichzeitig machen Sie Ihren Einsatz. Zeigt die Münze Kopf, erhalten Sie das Doppelte Ihres Einsatzes zurück.'
Das folgende Argument soll begründen, daß Sie nie zu verlieren brauchen: Setzen Sie erst DM 1. Wenn Sie gewinnen, hören Sie auf. Wenn Sie verlieren, spielen Sie mit einem Einsatz von DM 2 weiter. Verdoppeln Sie Ihren Einsatz nach jedem verlorenen Spiel und spielen Sie weiter. Sobald Sie gewinnen, hören Sie auf und haben auch dann mehr als Ihren Einsatz zurück.

Nehmen Sie an, Sie verlieren die ersten vier Spiele und gewinnen das fünfte. Es ergibt sich die Tabelle:

	Einsatz	Verlust	Gewinn
Erster Wurf	DM 1	DM 1	
Zweiter Wurf	DM 2	DM 2	
Dritter Wurf	DM 4	DM 4	
Vierter Wurf	DM 8	DM 8	
Fünfter Wurf	DM 16		DM 16
INSGESAMT		DM 15	DM 16
NETTOGEWINN = DM 1			

Die untenstehende Folge ist eine geometrische Folge. Überzeugt Sie dieses Argument davon, daß man niemals zu verlieren braucht?

FIBONACCI-FOLGEN

Am Anfang dieses Kapitels hatten wir die Folge

(e) 4, 4, 8, 12, 20, 32, 52, 84, 136, 220

die durch die Formel erzeugt wird:

4*INT(((0.5 + SQR(5)/2)↑N – (0.5 – SQR(5)/2)↑N) / SQR(5)).

Es gibt jedoch ein einleuchtenderes Verfahren, diese Folge zu erzeugen. Jedes Glied außer den ersten beiden ist die Summe der zwei vorhergehenden Glieder.

4 + 4 = 8
4 + 8 = 12
8 + 12 = 20

und so weiter.
So erzeugte Folgen heißen Fibonacci-Folgen. Im Jahr 1202 beobachtete Leonardo von Pisa, genannt Fibonacci, eine solche Folge im Zusammenhang mit der Kaninchenzucht.
Hier sind zwei weitere Fibonacci-Folgen:

2, 5, 7, 12, 19, 31, 50
3, 5, 8, 13, 21, 34, 55

Das nächste Programm erzeugt Fibonacci-Folgen bis zum Überdruß:

```
10 REM 16FIBONACCI FOLGEN
20 POKE53280,6:PRINTCHR$(147)"          ***
 FIBONACCI FOLGEN ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 FIBONACCI-     FOLGEN"
40 PRINT:PRINT"GEBEN SIE ZWEI GANZE ZAHL
EN EIN. TRENNENSIE DIESE DURCH ";
45 PRINT"EIN KOMMA."
50 PRINT:INPUT"ZAHLEN";U,V
60 IF U<>INT(U)ORV<>INT(V)THENPRINT:PRIN
T"GANZE ZAHLEN BITTE"
70 PRINT:INPUT"WIEVIELE FOLGEGLIEDER";N
80 IFN<1ORN<>INT(N)THEN70
90 REM DIE FOLGE
100 PRINTCHR$(147):PRINT:PRINT"FIBONACCI
-FOLGE MIT GRUNDGLIEDERN";U;",";V
105 PRINT
110 FOR I=1TON
120 IF 38-POS(0)<LEN(STR$(U))THENPRINT
130 PRINTCHR$(29);U
140 W=U+V:U=V:V=W
150 NEXTI
160 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N) ?"
170 GETG$:IFG$=""THEN170
180 IFG$="J"THENRUN
```

```
190 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
200 GOTO170
```

Es folgt eine Übungsaufgabe, die Sie vielleicht lösen möchten: Schreiben Sie ein kurzes Programm für Ihren Commodore 64.
'Schreibe zwei beliebige ganze Zahlen auf. Erzeuge eine Fibonacci-Folge durch paarweises Addieren von Gliedern zur Bildung eines weiteren Gliedes. Bestimme das Verhältnis jedes Gliedes zum vorhergehenden. Wohin tendiert das Verhältnis, wenn die Anzahl der Glieder groß wird? Berechne den Wert von 0.5 + SQR(5)/2.'

KAPITEL 6

ZAHLENBASEN

Gewöhnlich schreiben wir Zahlen im Dezimalsystem auf. Zum Beispiel steht 1432, in Worten: Eintausendvierhundertzweiunddreißig, für den umständlicheren Ausdruck

$$1*10^3 + 4*10^2 + 3*10 + 2.$$

Eine noch etwas umständlichere Schreibweise wäre

$$1*10^3 + 4*10^2 + 3*10^1 + 2*10^0$$

denn $10^1 = 10$ und $10^0 = 1$. Anders gesagt ist die Zahl 1432 als Summe von Vielfachen von Zehnerpotenzen zu interpretieren. Die ganzen Zahlen 1, 4, 3 und 2 heißen die *Stellen* der Zahl; 1 ist dabei die Tausenderstelle, 4 die Hunderterstelle, 3 die Zehnerstelle und 2 die Einerstelle. Diese Darstellung der Zahl nennt man ihre *Dezimaldarstellung;* man sagt, daß die Zahl zur *Basis* 10 dargestellt ist. Das Wort dezimal ist abgeleitet vom lateinischen *decem,* zehn.
Das Dezimalsystem hat 10 zur Basis. Man kann aber auch andere Basen als 10 nehmen. Es ist nicht nur interessant, sondern auch nützlich, Zahlen mit verschiedenen Basen auszudrücken. Beispielsweise hat sich die Darstellung von Zahlen zur Basis 2 als außerordentlich wichtig für Computer und alles damit Zusammenhängende erwiesen.
Jede ganze Zahl größer als 1 kann als Basis dienen, und jede Zahl kann zu jeder beliebigen Basis dargestellt werden. Ihrem Commodore fällt es im übrigen leicht, Zahlendarstellungen von einer Basis auf eine andere umzurechnen.
Sei N irgendeine positive Zahl und B eine ganze Zahl größer als 1. Um N zur Basis B auszudrücken, müssen wir N wie folgt schreiben:

$$N = X_m*B^m + X_{m-1}*B^{m-1} + \ldots + X_1*B + X_0$$

wobei jede der Zahlen $X_0, X_1, \ldots, X_m$ eine ganze Zahl zwischen 0 und $B-1$ ist. (Sehen Sie sich an, was herauskommt, wenn Sie 10 für B einsetzen). Die Stellen X_0, X_1, etc. heißen *Koeffizienten* der Zahl N zur Basis B.
Kleine Werte der Basis B führen zu Zahlendarstellungen von großer Länge. Aber sie haben den Vorteil, daß sie mit weniger Koeffizienten auskommen. Der Extremfall ist $B = 2$. Das zugehörige System heißt *binäres* Zahlensystem (vom lateinischen *binarius,* zwei). Schreibt man eine Zahl im Binärsystem, können nur 0 und 1 als Koeffizienten vorkommen. Zum Beispiel

$$86 = 64 + 16 + 4 + 2$$
$$= 1*2^6 + 0*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2 + 0$$

Die Zahl 86 hat also die Binärstellung 1010110. Binärzahlen werden von Computern verwendet, weil sie als Folgen von Nullen und Einsen ausgedrückt werden können. Der Grund ist, daß 0 und 1 in einem Computer einfach den Stellungen 'Aus' und 'Ein' eines Schalters entsprechen.
Bei Basen größer als 10 braucht man zusätzliche Symbole. Es liegt nahe, die Buchstaben A, B, C etc. des Alphabets zu verwenden. Eine häufig benutzte Basis ist 16. Eine zur Basis 16 dargestellte Zahl heißt *hexadezimal.* Der Vorteil dieser Basis ist es, daß sie wenige Koeffizienten braucht und sich Hexadezimalzahlen trotzdem leicht in Binärzahlen umwandeln lassen.
Die Umwandlung einer Zahl von der Basis 10 zur Basis B ist ganz einfach. Angenommen, wir wollen die Zahl N von der Basis 10 zur Basis B umwandeln. Erst subtrahieren wir von N alle Vielfachen von B.

$$M = INT(N/B) : R = N - B * M$$

Den Rest notieren wir und nennen ihn R_0. Dann wiederholen wir das Verfahren mit M, indem wir N = M setzen. Den neuen Rest nennen wir R_1. So fahren wir fort, bis der Wert von M schließlich 0 erreicht. Falls R_S der letzte so erhaltene Rest ist, ist die ursprüngliche Zahl zur Basis B gleich

$$R_S \ldots R_2R_1R_0$$

Wir wollen nun ein konkretes Beispiel durchrechnen. Angenommen, wir möchten die Zahl 29 zu Basis 3 schreiben. Dann rechnen wir wie folgt:

Schritt 1.

$$\begin{aligned} N &= 29 \\ M &= INT(29/3) \\ &= 9 \\ R_0 &= 29 - 3*9 \\ &= 2 \end{aligned}$$

Schritt 2.

$$\begin{aligned} N &= M \\ &= 9 \\ M &= INT(9/3) \\ &= 3 \\ R_1 &= 9 - 3*3 \\ &= 0 \end{aligned}$$

Schritt 3.

$$\begin{aligned} N &= M \\ &= 3 \\ M &= INT(3/3) \\ &= 1 \\ R_2 &= 3 - 3*1 \\ &= 0 \end{aligned}$$

Schritt 4

$$\begin{aligned} N &= M \\ &= 1 \\ M &= INT(1/3) \\ &= 0 \\ R_3 &= 1 - 3*0 \\ &= 1 \end{aligned}$$

Das Verfahren bricht nach vier Schritten ab, wenn M Null erreicht. Der Wert von 29 zur Basis 3 ist also 1002.

Das unten abgedruckte Programm wandelt ganze Zahlen von einer Basis in eine andere um. Beispielsweise kann man damit Zahlen von der Basis 10 auf die Basis 16 umschreiben. Für Basen größer als 10 stellen die Buchstaben A, B, C etc. die Zahlen 10, 11, 12 etc. dar.

```
10 REM 17BASIS UMWANDLER
20 POKE53280,6:PRINTCHR$(147)"          **
* BASIS-UMWANDLER ***"
30 PRINT:PRINT"DIESES PROGRAMM WANDELT Z
AHLEN VON EINERBASIS ZUR ANDEREN."
40 PRINT:INPUT"UMZUWANDELNDE BASIS ";A
45 AA=54+A:IFA<11THENAA=47+A
50 IFA<2ORA>35ORINT(A)<>ATHENPRINT"NEUE
EINGABE":GOTO40
60 PRINT:INPUT"UMZUWANDELNDE ZAHL  ";M$
65 N$=M$
70 REM KONTROLLE N$
80 IFN$=""THENPRINT"KEINE ZAHL":GOTO60
90 L=LEN(N$):I=0
100 I=I+1:N=ASC(MID$(N$,I,1))
110 IFN<48OR(N>57 AND N<65) OR N>AA THEN
 PRINT"KEINE ZAHL":GOTO60
120 IFI<L THEN 100
124 DIM A(L)
```

```
125 FORI=1TOL:N=ASC(MID$(N$,I,1)):IF N<5
8 THEN A(I)=N-48
126 IF N>64 THEN A(I)=N-55
127 NEXT
150 REM UMWANDLUNG VON BASIS A ZU BASIS
10
160 N=VAL(N$)
170 IFA<>10THENN=0:FORI=1TOL:N=A(I)+N*A:
NEXT
180 PRINTCHR$(147):PRINT:PRINT"DEZIMALFO
RM DER ZAHL: "N
190 PRINT:INPUT"ZIELBASIS ";B:PRINT
200 IFB<>INT(B)ORB<2ORB>35THENPRINT"UNMO
EGLICH-NEUER VERSUCH":GOTO190
210 REM UMWANDLUNG N NACH B
220 N$=""
230 M=INT(N/B):R=N-B*M:N=M
240 IFR<10THENN$=CHR$(48+R)+N$
250 IFR>9THENN$=CHR$(55+R)+N$
260 IFN<>0THEN230
270 PRINT:PRINTM$" MIT DER BASIS"A" ERGI
BT":PRINTN$" MIT DER BASIS"B
280 PRINT:PRINT:PRINT"NEUER START (J/N)"

290 GETA$:IFA$<>"N"ANDA$<>"J"THEN290
300 IFA$="J"THENRUN
310 PRINTCHR$(147)"AUF WIEDERSEHEN":POKE
53280,14:END
```

64ER ZAHLEN

Will man eine Zahl zwischen 0 und 255 als Binärzahl darstellen, braucht man höchstens 8 Koeffizienten. Zum Beispiel

$$255 = 1*2^7 + 1*2^6 + 1*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 1*2 + 1$$
$$128 = 1*2^7 + 0*2^6 + 0*2^5 + 0*2^4 + 0*2^3 + 1*2^2 + 0*2 + 0$$

Diese 8 Koeffizienten, oder 8 Bits auf einem Computer, nennt man ein Byte. Der Commodore 64 speichert ganze Zahlen in zwei Bytes, die High Byte und Low Byte genannt werden. Das High Byte stellt Vielfache von 256 dar. Beispielsweise würde

die Zahl 999 mit High Byte 3 und Low Byte 231 gespeichert, denn 999 = 3*256 + 231. Zahlen in Bytes zu speichern bedeutet nichts anderes als Zahlen zur Basis 256 zu speichern.
Durch PEEKen können Sie sehen, wie der 64 High Byte und Low Byte einer ganzen Zahl speichert. Erinnern Sie sich, daß Integervariablen auf dem Commodore 64 durch das Prozentzeichen (%) hinter dem Variablennamen gekennzeichnet sind. Geben Sie folgendes ein, und drücken Sie die Return-Taste am Ende jeder Zeile:

```
NEW : CLR
X% = 999 (beliebige ganze Zahl eingeben)
PRINT "HIGH BYTE=" PEEK(2053), "LOW BYTE=" PEEK (2054)
```

Die größte ganze Zahl, die der Commodore 64 speichern kann, ist 32767, also 127*256 + 255. Zahlen mit einem High Byte von 128 oder größer sind negative Zahlen. Negative Zahlen werden nämlich gespeichert, indem man erst den ABSolutbetrag der Zahl betrachtet, High und Low Byte berechnet und dann den Wert des High Byte von 255 subtrahiert und den des Low Byte von 256 subtrahiert. Auf diese Weise erhält z. B. −1 ein High Byte von 255 und ein Low Byte von 255.
Eine Zahl wird wie folgt aus High und Low Byte berechnet. Bezeichne H das High Byte und L das Low Byte.

```
NUMBER = H*256 + L
IF H > = 128 THEN NUMER = −((255−H)*256 + (256−L))
```

Die letzte Zeile kann auch so geschrieben werden:

```
IF H > = 128 THEN NUMBER = H*256 + L − 256*256
```

Versuchen Sie, Zahlen in die Adressen 2053 und 2054 zu POKEn, geben Sie PRINT X% ein und vergleichen Sie die Antworten. Tippen Sie z. B. folgendes ein, und drücken Sie die Return-Taste am Ende jeder Zeile:

```
NEW : CLR : X% = 0
POKE 2053,98 : POKE 2054,99 (Variableninhalt ersetzen)
N = PEEK(2053)*256 + PEEK(2054)
IF PEEK(2053) > = 128 THEN N = N − 256*256
PRINT X%, N
```

KLEINE ZAHLEN

Das Programm im vorletzten Abschnitt verarbeitet positive ganze Zahlen. Jede Zahl, ob ganz oder nicht ganz, hat jedoch eine Darstellung in jeder Basis. Beispielsweise wird die Dezimalzahl 0.25 binär als 0.01 dargestellt, und die Dezimalzahl 0.125 binär als 0.001. Um das nachzuvollziehen, wollen wir uns zunächst klarmachen, was die Dezimalzahl 0.25 bedeutet. Diese Zahl steht für 2 Zehntel und 5 Hundertstel, also

$$\begin{aligned} 0.25 &= 2/10 + 5/100 \\ &= 2*10^{-1} + 5*10^{-2}. \end{aligned}$$

Um sie zur Basis B darzustellen, müssen wir sie in dieser Form schreiben:

$$Y_1*B^{-1} + Y_2*B^{-2} + \ldots$$

wobei wie üblich $B^{-1} = 1/B$, $B^{-2} = 1/B^2$, etc.
Die Dezimalzahlen 0.25 und 0.125 können wie folgt geschrieben werden:

$$\begin{aligned} 0.25 &= 1/4 \\ &= 0*2^{-1} + 1*2^{-2} \\ 0.125 &= 1/8 \\ &= 0*2^{-1} + 0*2^{-2} + 1*2^{-3} \end{aligned}$$

Daraus erklärt sich die Binärdarstellung dieser Zahlen.
Sehen Sie als weiteres Beispiel, wie die Zahl 0.6 in (negativen) Potenzen von 2 dargestellt wird:

$$\begin{aligned} 0.6 = {} & 1*2^{-1} + 0*2^{-2} + 0*2^{-3} + 1*2^{-4} \\ & + 1*2^{-5} + 0*2^{-6} + 0*2^{-7} + 1*2^{-8} \\ & + 1*2^{-9} + 0*2^{-10} + 0*2^{-11} + 1*2^{-12} \\ & + 1*2^{-13} + 0*2^{-14} + 0*2^{-15} + 1*2^{-16} \\ & + \ldots \end{aligned}$$

Tatsächlich braucht man unendlich viele Stellen, um 0.6 exakt binär aufzuschreiben. Es ergibt sich die Darstellung

0.1001100110011001100110011001 . . .

Der Commodore 64 speichert nur 32 dieser Stellen, angefangen bei der ersten von Null verschiedenen; zusätzlich rundet er auf, wenn die 33. signifikante Stelle ungleich Null ist. Also speichert der 64 die Zahl 0.6 binär als

0.10011001100110011001100110011010

Das nächste Programm druckt die Binärdarstellung einer Dezimalzahl zwischen 0 und 1 so aus, wie sie vom Commodore 64 gespeichert wird.

```
10 REM 18BINAER UMWANDLER
20 POKE53280,6:PRINTCHR$(147)" *** UMWAN
DLUNG DEZIMAL ZU BINAER ***"
30 PRINT:PRINT"DIESES PROGRAMM ZEIGT DIE
 BINAERFORM    EINER ZAHL ZWISCHEN ";
40 PRINT"NULL UND EINS ."
50 N=0:PRINT:INPUT"ZAHL EINGEBEN";N
60 IFN<=0ORN>=1THENPRINT:PRINT"!!ZWISCHE
N NULL UND EINS!!":GOTO50
70 N$="0"
80 FORI=1TO32
90 N=N*2:N$=N$+MID$(STR$(INT(N)),2,1):N=
N-INT(N)
100 NEXT I
110 PRINTCHR$(147):PRINT:PRINT"DIE BINAE
RFORM DER ZAHL IST:"
115 PRINT:PRINTN$
120 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N) ?"
130 GET G$:IF G$ = "" THEN GOTO130
140 IF G$="J" THEN RUN
150 IF G$="N" THENPRINTCHR$(147)"AUF WIE
DERSEHEN":END
160 GOTO130
```

GLEITKOMMAZAHLEN

Ganze Zahlen werden im Commodore 64 in zwei Bytes gespeichert. Aber reelle Zahlen werden in 5 Bytes gespeichert, selbst wenn ihr Wert eine ganze Zahl ist. Wenn Sie eine Zahl nicht mit dem Prozentzeichen zur ganzen Zahl erklären, wird sie als reelle Zahl in 5 Bytes gespeichert. Eine Zahl kann binär wie folgt dargestellt werden:

$$1.X_1X_2X_3 \ldots X_m * 2^N$$

wobei X_1, X_2 . . ., X_m entweder 0 oder 1 sind und N eine ganze Zahl ist (positiv, negativ oder Null). Die ganze Zahl N heißt *binärer Exponent* der Zahl, der Rest heißt *binäre Mantisse.* Beispielsweise ist die Dezimalzahl 10 binär 1010 und läßt sich schreiben als

$1.01 * 2^3$

Also hat 10 den binären Exponenten 3 und die binäre Mantisse 1.01. Ein anderes Beispiel ist die Dezimalzahl 0.375, die binär 0.011 ist und sich schreiben läßt als

$1.1 * 2^{-2}$

Also hat die Dezimalzahl 0.375 den binären Exponenten −2 und die binäre Mantisse 1.1.
Wir haben erwähnt, daß der 64 zum Speichern von Zahlen 5 Bytes benutzt. Das erste Byte ist der binäre Exponent plus 129. Die übrigen vier Bytes bestimmen die binäre Mantisse und das Vorzeichen einer Zahl. Da die erste Stelle einer binären Mantisse immer 1 ist, braucht sie nicht gespeichert zu werden – es reicht, die Stellen rechts von Dezimalpunkt der binären Mantisse zu speichern. Das erste Bit von Byte 2 bestimmt das Vorzeichen der Zahl, die übrigen 31 Bit der letzten vier Bytes enthalten die binäre Mantisse (ohne die führende 1).
Beispielsweise würde die Dezimalzahl 10 wie folgt gespeichert: Das erste Byte ist 129 plus den binären Exponenten 3, also 132. Das erste Bit des zweiten Byte ist 0, da die Zahl positiv ist. Die übrigen 31 Bit wären

01000000000000000000000000000

weil 10 die binäre Mantisse 1.01 hat und die führende 1 ignoriert wird. Die 32 Bit in den letzten vier Bytes wären also

00100000000000000000000000000

Aufgeteilt in vier Blöcke zu 8 Bit ergibt das

00100000 00000000 00000000 00000000

was wiederum 32, 0, 0, 0 ist. Also würde die Dezimalzahl 10 in den 5 Bytes 132, 32, 0, 0, 0 gespeichert.
Das Verfahren läßt sich umkehren, um die Zahl zu ermitteln, die der 64 in 5 Bytes abgelegt hat. Nehmen wir an, die Zahl N ist in den 5 Bytes P, Q, R, S, T gespeichert. Folgende Programmzeilen berechnen N aus P, Q, R, S und T:

```
X = 1 : IF Q > = 128 THEN Q = Q−128 : X = − 1
```

$N = X * 2^{P-129} * (1 + Q*2^{-7} + R*2^{-15} + S*2^{-23} + T*2^{-31})$

Um den 64 in Aktion zu erleben, geben Sie diese Zeilen ein, und drücken Sie Return am Ende jeder Zeile:

```
NEW : CLR
X = 10 (beliebige Zahl eingeben)
FOR I=0 TO 4 : PRINT "BYTE" I+1 "=" PEEK(2053+I) : NEXT I
```

Sie werden allerdings feststellen, daß der Commodore 64 beim Multiplizieren gelegentlich kleine Fehler macht. Die Zahl $1 + 2^{-24}$ wird (korrekt) in den folgenden 5 Bytes gespeichert:

129, 0, 0, 0, 128

Wird dieselbe Zahl jedoch als $1*(1 + 2^{-24})$ geschrieben, wird sie in diesen 5 Bytes gespeichert:

129, 0, 0, 0, 64

Anders gesagt erhält der Commodore 64 das Ergebnis

```
1 * (1 + 2↑−24) = 1 + 2↑−25
```

Dasselbe Problem läßt sich auch so sichtbar machen:

```
X = 1 + 2↑−24
PRINT X − X, X − 1*X
```

Versuchen Sie es auch mit

```
X = 1 + 2↑−24
X1 = 1*X
X2 = 1*X1
X3 = 1*X2
X4 = 1*X3
PRINT X, X1, X2, X3, X4
```

Probleme dieser Art scheinen nicht aufzutauchen, wenn die mittleren 3 Bytes der zum Speichern der Zahl verwendeten 5 Bytes nicht alle gleich 0 sind. Versuchen Sie, Beispiele zu finden.

KAPITEL 7

TAGE UND WOCHEN

TAGE

Die Zellersche Kongruenz ist eine kompliziert aussehende Formel, die den Wochentag (Sonntag, Montag, etc.) zu jedem gegebenen Datum bestimmt. Mit dieser Formel können Sie z. B. herausfinden, an welchem Wochentag jemand geboren ist. Und falls Sie ein schlechtes Gedächtnis haben, könnten Sie berechnen, auf welchen Wochentag ein bestimmtes Jubiläum gefallen ist.
Die Zellersche Formel sieht so aus:

A = INT(2.6*M − 0.1) + D + Y + INT(Y/4) + INT(C/4) −2*C
X = A − 7*INT(A/7)

Die Zahl X liegt zwischen 0 und 6, weil alle Vielfachen von 7, die kleiner als A sind, von A subtrahiert worden sind. Diese Zahlen entsprechen den 7 Wochentagen wie folgt:

0 : Sonntag
1 : Montag
2 : Dienstag
3 : Mittwoch
4 : Donnerstag
5 : Freitag
6 : Samstag

Die Zahlen D, M, Y und C sind definiert als:

D : der Monatstag.
M : die Monatszahl – aber nicht die übliche. Januar und Februar tragen die Zahlen 11 und 12 des vorhergehenden Jahren (das beeinflußt das unten beschriebene Y und gelegentlich auch C). März trägt die Zahl 1, April die Zahl 2, Mai die Zahl 3, . . . und Dezember die Zahl 10.
Y : das Jahr im Jahrhundert.
C : die Zahl der Jahrhunderte, d. h. die ersten zwei Stellen der Jahreszahl.

Für den 26. August 1983 ist beispielsweise die übliche Schreibweise 26/08/1983. Für die Zellersche Formel verwenden wir D = 26, M = 6, Y = 83 und C = 19.

Durch Einsetzen dieser Zahlen in die Zellersche Formel ergibt sich

$$
\begin{aligned}
A &= \mathrm{INT}(2.6*6 - 0.1) + 26 + 83 + \mathrm{INT}(83/4) + \mathrm{INT}(19/4) - 2*19 \\
&= \mathrm{INT}(15.6 - 0.1) + 26 + 83 + \mathrm{INT}(20.75) + \mathrm{INT}(4.75) - 38 \\
&= 15 + 26 + 83 + 20 + 4 - 38 \\
&= 110
\end{aligned}
$$

und

$$
\begin{aligned}
Y &= 110 - 7*\mathrm{INT}(110/7) \\
&= 110 - 7*\mathrm{INT}(15.714285) \\
&= 110 - 105 \\
&= 5.
\end{aligned}
$$

Wir schließen daraus, daß der Wochentag des 26. August 1983 ein Freitag ist.
Hier sind einige Beispiele für die übliche Schreibweise des Datums und die Werte, die in die Zellersche Formel eingehen.

ÜBLICHE SCHREIBWEISE	SCHREIBWEISE FÜR DIE ZELLERSCHE FORMEL			
	D	M	C	Y
03/03/1947	3	1	19	47
01/01/2000	1	11	19	99
26/02/1983	26	12	19	82
29/11/1984	29	9	19	84

Das nachstehende Programm benutzt die Zellersche Formel, um den Wochentag zu jedem vorgegebenen Datum auszurechnen. Ihr Commodore 64 bestimmt automatisch die richtigen Werte von D, M, C und Y zu einem beliebigen eingegebenen Datum. Beachten Sie, daß das Programm mit einigen Kontrollen sicherstellt, daß das eingegebene Datum sinnvoll ist. Beispielsweise wird der 30. Februar 1983 nicht akzeptiert. Außerdem muß das eingegebene Jahr eine ganze Zahl im Bereich von 1752 bis 4902 sein. Die Zellersche Formel bezieht sich auf den Bereich von 1582 bis 4902, aber der Gregorianische Kalender ist in Großbritannien, den Britischen Kolonien und den USA erst seit 1752 in Gebrauch.
Schaltjahre werden automatisch vom Programm berücksichtigt. Beachten Sie, daß ein Jahr ein Schaltjahr ist, wenn die Jahreszahl ohne Rest durch 4 teilbar ist, es sei denn, sie ist durch 100, aber nicht durch 400 teilbar. Also war 1900 kein Schaltjahr, aber 2000 wird eines sein.

```
10 REM 19WOCHENTAGE
20 DIM A(12),A$(12):FORI=1TO12:READ A(I)
,A$(I):NEXTI
30 FORI=0TO6:READB$(I):NEXTI
40 POKE53280,6:PRINTCHR$(147)"
*** WOCHENTAGE ***"
50 PRINT:PRINT"DIESES PROGRAMM ZEIGT DEN
 WOCHENTAG IR- GENDEINES ";
60 PRINT"SPEZIELLEN TAGES"
70 PRINT:PRINT"BITTE DATUM EINGEBEN"
80 A(2)=29:D=0:PRINT:INPUT"TAG (ZW.1 UND
 31)    ";D
90 IF D<1 OR D>31 OR D<>INT(D) THENPRINT
"NOCH MAL VERSUCHEN":GOTO80
100 Y=0:PRINT:INPUT"MONAT (ZW.1 UND 12)
";M
110 IFM<1ORM>12ORM<>INT(M)THENPRINT"NOCH
 MAL VERSUCHEN":GOTO100
120 IF D>A(M)THENPRINT:PRINT"ZUVIEL TAGE
 IM MONAT":FORI=1TO2000:NEXT:GOTO40
130 Y=0:PRINT:INPUT"JAHR (VIERSTELLIG)
";Y
140 IF Y<1582 OR Y>4902 OR Y<>INT(Y) THE
NPRINT"NOCH MAL VERSUCHEN":GOTO130
150 REM SCHALTJAHR
160 L=0:IF INT(Y/4)*4=Y THENL=-1
170 IFLANDINT(Y/100)*100=YTHENL=0:IFINT(
Y/400)*400=YTHENL=-1
175 PRINTCHR$(147)
180 A(2)=28-L:IFLTHENPRINT:PRINT:PRINTY"
IST/WAR EIN SCHALTJAHR"
190 IF M=2ANDD>A(2)THENPRINT:PRINT:PRINT
" !! 28 TAGE IM FEBRUAR !!":RT=1
195 IFRT=1THEND=D-1
200 PRINT:PRINT:PRINT" DER"D"█."M"█."Y"I
ST/WAR EIN ";
210 M=M-2:IFM<1THENM=M+12:Y=Y-1
220 C=INT(Y/100):Y=Y-C*100
230 REM ZELLER'SCHE KONGRUENZ
240 DAY=INT(2.6*M-0.1)+D+Y+INT(C/4)+INT(
Y/4)-2*C
```

```
250 DAY=DAY-7*INT(DAY/7):PRINTB$(DAY)
260 PRINT:PRINT:PRINT" NEUER START (J/N)
 ?"
270 GET G$:IF G$="" THEN270
280 IFG$="J"THENRUN
290 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
300 GOTO270
400 REM DATA
410 DATA 31,JANUAR,29,FEBRUAR,31,MAERZ,3
0,APRIL
420 DATA 31,MAI,30,JUNI,31,JULY,31,AUGUS
T
430 DATA 30,SPTEMBER,31,OKTOBER,30,NOVEM
BER,31,DEZEMBER
440 DATA SONNTAG,MONTAG,DIENSTAG,MITTWOC
H,DONNERSTAG,FREITAG,SAMSTAG
```

Anmerkung: Sie haben vielleicht die Zellersche Formel vorher schon einmal gesehen und möglicherweise bemerkt, daß die hier verwendete Formel etwas davon abweicht. Häufig ist der erste Ausdruck INT(2.6*M – 0.1) durch den Ausdruck INT (2.6*M – 0.2) ersetzt. Daß der zweite Ausdruck hier nicht benutzt wird, hat seinen Grund in dem Verfahren, mit dem der Commodore den ganzen Teil INT einer Zahl berechnet. Für M = 7 ergibt sich zum Beispiel:

$$\begin{aligned} \text{INT}(2.6*7 - 0.2) &= \text{INT}(18.2 - 0.2) \\ &= \text{INT}(18) \\ &= 18 \end{aligned}$$

Der Commodore 64 weist jedoch INT(2.6*7 – 0.2) den Wert 17 zu, obwohl er korrekt 18 ausdruckt, wenn Sie ihn nach 2.6*7 – 0.2 fragen.

KALENDER

Sobald wir den Wochentag für jedes Datum kennen, können wir einen Kalender herstellen. Das nächste Programm druckt einen Kalender für jeden Monat eines beliebigen Jahres. (Auf dem Bildschirm kann man nur einen Monat übersichtlich darstellen.) Das Programm berechnet mit der Zellerschen Formel den Wochentag, auf den der erste Tag des Monats fällt. Wie im obigen Programm für den Wochentag werden Schaltjahre automatisch berücksichtigt.

```
10 REM 20KALENDER
20 DIMA(12),A$(12):FORI=1TO12:READA(I),A
$(I):NEXTI
30 REM START
40 POKE53280,6:POKE53281,6:PRINTCHR$(147
)"            *** KALENDER ***"
50 PRINT:PRINT"DIESES PROGRAMM ZEIGT EIN
EN KALENDER VONJEDEM MONAT ";
60 PRINT"EINES GEWUENSCHTEN JAHRES."
70 PRINT:PRINT"BITTE MONAT UND JAHR EING
EBEN"
80 M=0:PRINT:INPUT"MONAT (ZW.  1 UND 12)"
;M
90 IFM<1ORM>12ORM<>INT(M)THENPRINT"NOCH
MAL VERSUCHEN":GOTO80
100 Y=0:PRINT:INPUT"JAHR (VIERSTELLIG) "
;Y
110 IFY<1752ORY>4902ORY<>INT(Y)THENPRINT
"NOCH MAL VERSUCHEN":GOTO100
130 REM SCHALTJAHRE
140 L=0:IFINT(Y/4)*4=YTHENL=-1
150 IFLANDINT(Y/100)*100=YTHENL=0:IFINT(
Y/400)*400=YTHENL=-1
155 PRINTCHR$(147)CHR$(31):POKE53281,7:P
OKE53280,7
157 PRINT:PRINTA$(M);Y
160 A(2)=28-L:IFLTHENPRINT:PRINT"DIES IS
T EIN SCHALTJAHR"
170 A$=A$(M)+STR$(Y):MM=A(M)
210 M=M-2:IFM<1THENM=M+12:Y=Y-1
220 C=INT(Y/100):Y=Y-C*100
230 REM ZELLER'SCHE KONGRUENZ
240 DAY=INT(2.6*M-0.1)+1+Y+INT(C/4)+INT(
Y/4)-2*C
250 DAY=DAY-7*INT(DAY/7)
260 REM KALENDER ZEIGEN
280 PRINTTAB(20-LEN(A$)/2)
290 PRINT:PRINT:PRINTCHR$(28)"    SO    MO
   DI    MI    DO    FR    SA    "CHR$(31)
295 PRINT
```

```
300 FORI=1TOMM
310 DAY=DAY+1
320 PRINTTAB(DAY*5-2+(I>9)) I;
330 IFDAY=7THENDAY=0:PRINT:PRINT
340 NEXTI
350 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N) ?"
360 GETG$:IFG$=""THEN360
370 IFG$="J"THENPRINTCHR$(154):RUN
380 IFG$="N"THENPRINTCHR$(154):POKE53281
,6:POKE53280,14:PRINTCHR$(147)
385 IFG$="N"THENPRINT"AUF WIEDERSEHEN":E
ND
390 GOTO360
400 REM DATA
410 DATA 31,JANUAR,29,FEBRUAR,31,MAERZ,3
0,APRIL
420 DATA 31,MAI,30,JUNI,31,JULY,31,AUGUS
T
430 DATA 30,SEPTEMBER,31,OKTOBER,30,NOVE
MBER,31,DEZEMBER
```

TERMINKALENDER

Man braucht gelegentlich eine Liste von Terminen, die eine feste Anzahl von Tagen auseinanderliegen, z. B. Behandlungstermine im Krankenhaus oder Zahltage.
Um eine solche Liste anzufertigen, verwenden wir das 'pseudo-Julianische' Datum. Das ist einfach die Zahl der Tage seit einem beliebigen festen Datum. (Zum Beispiel hat der erste Januar des Jahres 1 das 'pseudo-Julianische' Datum 1.) Eine verhältnismäßig einfache Formel wandelt das tatsächliche Datum in ein pseudo-Julianisches Datum um und umgekehrt.
Sei D/M/Y das Datum, wobei D den Tag, M den Monat und Y das Jahr (einschließlich des Jahrhunderts) bezeichnet. Dann berechnet sich das pseudo-Julianische Datum wie folgt:

$X = INT(30.57*M) + INT(365.25*Y - 395.25) + D$
Ist $M > 2$ und Y ein Schaltjahr, subtrahiere 1 von X.
Ist $M > 2$ und Y kein Schaltjahr, subtrahiere 2 von X.

Als Beispiel wollen wir das pseudo-Julianische Datum des 26. August 1983 berechnen. Die Werte von D, M und Y sind durch D = 26, M = 8, Y = 1983 gegeben. In die Formel eingesetzt liefern diese Werte:

$$\begin{aligned} X &= \mathrm{INT}(30.57*8) + \mathrm{INT}(365.25*1983 - 395.25) + 26 \\ &= \mathrm{INT}(244.56) + \mathrm{INT}(723895.5) + 26 \\ &= 244 + 723895 + 26 \\ &= 724165 \end{aligned}$$

Da aber die Monatszahl M größer als 2 ist und 1983 kein Schaltjahr ist, ziehen wir 2 von X ab und erhalten das pseudo-Julianische Datum 724163.
Um umgekehrt das Datum aus dem pseudo-Julianischen Datum zu bestimmen, gehen wir wie folgt vor; sei X das pseudo-Julianische Datum:

Die erste Approximation für das Jahr ist:
Y = INT(X/365.26) + 1

Der Jahrestag ergibt sich als:
D = X − INT(365.25*Y − 395.25)

Eine Schaltjahr-Korrektur wird vorgenommen:
D1 = 2; ist Y ein Schaltjahr, setze D1 = 1
Ist D> 91−D1, addiere D1 zu D.

Monat und Tage werden berechnet:
M = INT(D/30.57)
D = D − INT(30.57*M)

Monat und Jahr werden nötigenfalls korrigiert:
Ist M> 12, setze M = 1 und addiere 1 zu Y.

Wir möchten beispielsweise das Datum bestimmen, das dem pseudo-Julianischen Datum 724164 entspricht (das ist 1 höher als das vorhin berechnete pseudo-Julianische Datum). Folgende Berechnungen sind nötig:

$$\begin{aligned} Y &= \mathrm{INT}(724164/365.26) + 1 \\ &= \mathrm{INT}(1982.5987) + 1 \\ &= 1983 \\ D &= 724164 - \mathrm{INT}(365.25*1983 - 395.25) \\ &= 724164 - \mathrm{INT}(724290.75 - 395.25) \\ &= 724164 - \mathrm{INT}(723895.5) \\ &= 269 \end{aligned}$$

Das Jahr 1983 ist kein Schaltjahr, also ist D1 gleich 2. Weil der Wert von D größer als 91 – D1 ist, addieren wir D1 zu D. Es ergibt sich daraus

$$\begin{aligned} D &= 271 \\ M &= INT(271/30.57) \\ &= INT(8.86490023) \\ &= 8 \\ D &= 271 - INT(30.57*8) \\ &= 271 - INT(244.56) \\ &= 271 - 244 \\ &= 27 \end{aligned}$$

Da der Wert von M nicht größer als 12 ist, sind wir fertig: D = 27, M = 8 und Y = 1983. Also entspricht dem pseudo-Julianischen Datum 724164 der 27. August 1983.

Das folgende Programm erledigt in kurzer Zeit alle diese Berechnungen und fertigt eine Liste von Terminen an, die eine vorgegebene Zahl von Tagen auseinanderliegen. Der Einfachheit halber funktioniert das Programm nur für Daten im 20. Jahrhundert. Die nötigen Änderungen für ein anderes Jahrhundert sollten Ihnen nicht schwerfallen.

Achtung: Daten werden in der Form DD/MM/YY eingegeben; z. B. würde 12. März 1984 als 12/03/84 oder 12/ 3/84 eingegeben, aber nicht als 12/3/84. Das Datum wird nach der Eingabe nicht gerade vielen Kontrollen unterzogen, daher könnten Sie beispielsweise 30/02/83 eingeben, was vom Programm als 2. März 1983 interpretiert würde. (Erkennen Sie warum?)

```
10 REM 21DATEN MANAGEMENT
20 POKE53280,6:PRINTCHR$(147)"          **
* TERMIN MANAGEMENT ***"
30 PRINT:PRINT"DIESES PROGRAMM LIEFERT T
ERMINE EINER   BESTIMMTEN ZEITSPANNE"
40 REM EINGABE ANFANGSDATUM
60 PRINT:PRINT:PRINT"EINGABE DES ANFANGS
DATUM (TT/MM/JJ)"
80 S$="":PRINT:INPUT"ANFANGSDATUM :  ";S$

90 IFLEN(S$)<>8THENPRINT:PRINT"IN FORM T
T/MM/JJ":GOTO80
100 IF MID$(S$,3,1)<>"/"OR MID$(S$,6,1)<
>"/"THENPRINT:PRINT"IN FORM TT/MM/JJ":GO
```

```
110 T=VAL(MID$(S$,1,2)):M=VAL(MID$(S$,4,
2)):J=VAL(MID$(S$,7,2))+1900
120 IFT<=0ORT>31ORM<=0ORM>12THENPRINT"TA
G, MONAT FALSCH":GOTO80
130 REM DATUM IST IN DER KORREKTEN FORM
140 REM BERECHNUNG DES PSEUDO-JULIAN DAY

150 X=INT(30.57*M)+INT(365.25*J-395.25)+
T
160 REM SCHALTJAHR
170 IFM>2THENX=X-2:IF INT(J/4)*4=J THEN
X=X+1
180 REM EINGABE ZEITSPANNE
190 PRINT:PRINT:PRINT"EINGABE DER ZEITSP
ANNE IN TAGEN"
200 P=0:PRINT:INPUT"ZEITSPANNE  : ";P
210 IF P<=0 OR INT(P)<>P THENPRINT:PRINT
"GANZE ZAHLEN BITTE":GOTO200
220 PRINT:PRINT:PRINT"ANZAHL DER ZEITSPA
NNEN"
230 PRINT:INPUT"ZAHL          : ";N
240 IF N<=0 OR INT(N)<>N THENPRINT:PRINT
"GANZE ZAHLEN BITTE":GOTO230
250 IF N<=0 OR N>100 THEN PRINT"VERNUENF
TIG BLEIBEN":GOTO230
260 PRINTCHR$(147)CHR$(5)
270 FOR I=0TO N
280 :J=INT(X/365.26)+1:REM JAHR
290 :T=X-INT(365.25*J-395.25)
300 :REM SCHALTJAHR
310 :T1=2:IF INT(J/4)*4=J THENT1=1
320 :IFT>91-T1 THEN T=T+T1
330 :M=INT(T/30.57) :REM MONAT
340 :T=T-INT(30.57*M):REM TAG
350 :IFM>12 THENM=1:J=J+1
360 :J=J-1900
370 :REM AUSGABE
380 :Z=T:GOSUB 500:T$=Z$+"/"
390 :Z=M:GOSUB 500:T$=T$+Z$+"/"
400 :Z=J:GOSUB 500:T$=T$+Z$
```

```
410 :PRINTT$
420 :X=X+P
430 NEXT
440 PRINT:PRINT:PRINT:PRINTCHR$(154)"NEU
ER START (J/N) ?"
450 GETG$:IFG$=""THEN450
460 IFG$="J"THENRUN
470 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
480 GOTO450
500 REM FORMATIERUNG
510 Z$=MID$(STR$(Z),2):IF LEN(Z$)<2THENZ
$="0"+Z$
520 RETURN
```

KAPITEL 8

DER GRÖSSTE GEMEINSAME TEILER

Wenn A und B ganze Zahlen sind, dann ist ein *gemeinsamer Teiler* oder *gemeinsamer Faktor* von A und B eine ganze Zahl, die beide Zahlen teilt. Der *größte Faktor* von A und B ist die größte solcher ganzen Zahlen.
Zum Beispiel ist 3 ein gemeinsamer Teiler von 12 und 18, aber der größte gemeinsame Teiler von 12 und 18 ist 6.
Das Berechnen des größten gemeinsamen Teilers zweier Zahlen ist nicht sonderlich kompliziert, vor allem nicht für einen Computer. Die dabei verwendete Methode läßt sich gut zur Veranschaulichung eines Rechenalgorithmus benutzen.
Der *Euklidische Algorithmus* ist die bekannteste und älteste (Drittes Jahrhundert v. Chr.) Methode zur Berechnung des größten gemeinsamen Teilers. Wenn man den größten gemeinsamen Teiler von A und B finden will, verfährt man wie folgt:

1. **Benenne A und B (wenn nötig) um, so daß A größer ist als B.**
2. **Dividiere A und B und bestimme den Rest R_1.**
   $R_1 = A - B * INT(A/B)$

Man beachte, daß jede Zahl, die A und B teilt, auch R_1 teilt. Und umgekehrt ist jeder gemeinsame Teiler von B und R_1 auch ein Teiler von A. Daraus ergibt sich, daß die gemeinsamen Teiler von A und B dieselben sind wie die von B und R_1. Also ist der größte gemeinsame Teiler von A und B gleich dem größten gemeinsamen Teiler von B und R_1.

3. **Teile nun B durch R_1 und bestimme den Rest R_2.**
   $R_2 = B - R_1 * INT(B/R_1)$

Was oben zu den Zahlen B, R_1 bemerkt wurde, gilt auch für R_1, R_2. Also ist der größte gemeinsame Teiler von A und B gleich dem größten gemeinsamen Teiler von R_1 und R_2.

4. **Als nächstes teile R_2 durch R_1 und bestimme den Rest R_3.**
   $R_3 = R_1 - R_2 * INT(R_1/R_2)$

Das Verfahren wird in dieser Weise fortgesetzt, bis der Rest Null ist. Beachten Sie, daß die Reste jedesmal kleiner werden und deshalb nach einer Anzahl von Schritten Null erreichen.

$$R_1 = A - B * INT(A/B) \qquad (0 <= R_1<B)$$
$$R_2 = B - R_1 * INT(B/R_1) \qquad (0 <= R_2<R_1)$$
$$R_3 = R_1 - R_2 * INT(R_1/R_2) \qquad (0 <= R_3<R_2)$$

● ●

● ●

● ●

$$R_{N-1} = R_{N-3} - R_{N-2} * INT(R_{N-3}/R_{N-2}) \qquad (0 <= R_{N-1}\ R_{N-2})$$
$$R_N = R_{N-2} - R_{N-1} * INT(R_{N-2}/R_{N-1}) \qquad (R_N = 0)$$

Hat der Rest 0 erreicht, so sieht man, daß der vorhergehende Rest R_{N-1} der größte gemeinsame Teiler von R_{N-1} und R_{N-2} ist. Führt man dieses Argument aus, so erkennt man R_{N-1} als größten gemeinsamen Teiler von A und B.

Der oben skizzierte Prozeß läßt sich leicht auf den Computer übertragen. Das leistet das folgende Programm:

```
10 REM 22GROESSTER GEMEINSAMER TEILER
20 POKE53280,6:PRINTCHR$(147)" *** GROES
STER GEMEINSAMER TEILER ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 DEN GROESSTEN GEMEINSAMEN TEILER ZWEIER
";
40 PRINT" GANZER ZAHLEN UNTER ZUHILFENAH
ME DES EUCLIDSCHEN         ALOGORITHMUS"
50 PRINT:PRINT"EINGABE ZWEIER GANZER ZAH
LEN"
60 PRINT:INPUT"ERSTE  ZAHL ";A
70 IFA<1ORA<>INT(A)THENPRINT:PRINT"NOCH
MAL VERSUCHEN"GOTO80
80 PRINT:INPUT"ZWEITE ZAHL ";B
90 IFA<1ORA<>INT(A)THENPRINT:PRINT"NOCH
MAL VERSUCHEN"GOTO80
100 IFA<BTHENC=A:A=B:B=C
110 REM EUCLIDSCHER        ALOGORITHMUS
120 R=A:S=B
130 T=R-S*INT(R/S)
140 IFT<>0THENR=S:S=T:GOTO130
150 PRINTCHR$(147):PRINT:PRINT:PRINT"GRO
ESSTER GEMEINSAMER TEILER :";S
160 PRINT:PRINT"ERSTES GEMEINSAMES VIELF
ACHES: "A*B/S
170 PRINT:PRINT:PRINT:PRINT"NEUER START
```

```
(J/N) ?"
180 GETG$:IFG$=""THEN180
190 IFG$="J"THENRUN
200 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
210 GOTO180
```

Das Programm berechnet auch das kleinste gemeinsame Vielfache von A und B. Das *kleinste gemeinsame Vielfache* zweier Zahlen A und B ist die kleinste Zahl, die sich sowohl durch A als auch durch B teilen läßt. Das kleinste gemeinsame Vielfache von A und B hat den Wert

A*B/(größter gemeinsamer Teiler).

Bezeichnet D den größten gemeinsamen Teiler von A und B, so läßt sich D als Kombination von A und B schreiben:

$$D = S*A + T*B.$$

Dabei sind S und T ganze Zahlen. Die Werte von S und T kann man bestimmen, indem man den Euklidischen Algorithmus rückwärts anwendet. Ändern Sie das Programm zur Bestimmung des größten gemeinsamen Teilers so ab, daß es auch S und T berechnet.

KAPITEL 9

PRIMZAHLEN

Eine *Primzahl* ist eine ganze Zahl (größer als 1), die durch keine andere ganze Zahl außer 1 und sich selbst teilbar ist (unter Teilbarkeit verstehen wir natürlich Teilbarkeit ohne Rest). Die Zahlen 2, 3, 5, 7 und 11 sind prim, aber 4, 6, 8, 9 und 10 nicht. Nicht-Primzahlen werden *zusammengesetzt* genannt.
Es gibt keine allgemeine Formel für Primzahlen, aber wie Euklid (etwa 300 v. Chr.) gezeigt hat, gibt es unendlich viele Primzahlen. Man weiß auch, daß Primzahlen unter großen Zahlen weniger häufig vorkommen.
Primzahltests sind von großer Bedeutung. In jüngster Zeit haben sie in der Kryptographie Beachtung gefunden.
Eine einfache und direkte Methode, zu entscheiden, ob eine Zahl N prim ist, ist das sogenannte Sieb des Eratosthenes. (Eratosthenes von Kyrene war ein griechischer Mathematiker, ca. 276–196 v. Chr., der auch den Umfang der Erde berechnet hat.)
Die Idee besteht darin, alle ganzen Zahlen von 1 bis N aufzuschreiben. Lasse dann 2 stehen und streiche alle geraden Zahlen nach 2. Die nächste nicht durchgestrichene Zahl ist prim; das ist 3. Streiche nun jede dritte Zahl nach 3. Die nächste verbliebene Zahl nach 3 ist 5. Diese Zahl muß prim sein. Streiche nun jede fünfte Zahl nach 5. Setze dieses Verfahren fort. Was übrigbleibt, sind die Primzahlen zwischen 1 und N.
Die folgende Tabelle zeigt das Ergebnis eines Siebs für die Zahlen bis 100. Die Vielfachen von 2 sind mit / durchgestrichen, die Vielfachen von 3 mit –, die von 5 mit \ und die von 7 mit |.

	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	~~35~~	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	~~49~~	~~50~~
~~51~~	~~52~~	53	~~54~~	~~55~~	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	~~65~~	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	~~77~~	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	~~85~~	~~86~~	~~87~~	~~88~~	89	~~90~~
91	~~92~~	~~93~~	~~94~~	~~95~~	~~96~~	97	~~98~~	~~99~~	~~100~~

Das nächste Programm verwendet das Sieb des Eratosthenes, um eine Liste von Primzahlen unterhalb einer gegebenen Zahl N zu erstellen. Diese Zahl wird zu Beginn mit INPUT eingegeben.

```
10 REM SIEB DES ERASTOSTHENES
20 POKE53280,6:PRINTCHR$(147)"     *** SI
EB DES ERASTOSTHENES ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 PRIMZAHLEN BISZU EINER EINGEGEBENEN";
40 PRINT" ZAHL"
50 PRINT:INPUT"BIS ZU WELCHER ZAHL ";N
60 IF N<4 OR N>15000 OR N<>INT(N) THEN P
RINT:PRINT"VERNUENFTIG BLEIBEN":GOTO50
70 DIM A%(N):PRINTCHR$(147):PRINT:PRINT"
PRIMZAHLEN VON 2 BIS";N
75 PRINT:PRINTCHR$(5)
80 FORI=2TON
90 IFA%(I)=1 THEN130
100 IF 39-POS(0)<LEN(STR$(I))THENPRINT
110 PRINTI,
120 FORJ=ITON STEPI:A%(J)=1:NEXT
130 NEXT:PRINT
135 PRINT:PRINT:PRINT:PRINT:PRINTCHR$(15
4)"NEUER START (J/N) ?"
150 GET G$:IFG$=""THEN150
160 IFG$="J"THENRUN
170 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
180 GOTO150
```

Beachten Sie, daß das Programm für große Zahlen langsam anläuft; nach kurzer Zeit beginnt es jedoch, sehr schnell Primzahlen auszudrucken. Es dauert etwa 7 Minuten und 21 Sekunden, die Primzahlen zwischen 2 und 15 000 auszudrucken. Das Sieb des Eratosthenes ist begrifflich einfach und nützlich, wenn man an einer Primzahl-Liste interessiert ist. Aber es ist keine sehr praktische Methode, wenn man prüfen will, ob eine gegebene Zahl prim ist.

Man kann auf einfache Weise feststellen, ob eine Zahl N prim ist, indem man schrittweise prüft, ob sie durch die Zahlen kleiner als N teilbar ist. Das folgende Programm führt diese Methode vor. Es enthält eine Uhr, die anzeigt, wie lange der Primzahltest dauert.

```
10 REM 24PRIMZAHLEN 1
20 POKE53280,6:PRINTCHR$(147)" *** LANGS
AMER PRIMZAHLEN TESTER I ***"
30 PRINT:PRINT:PRINT:PRINT:INPUT"ZU TEST
```

```
 ENDE ZAHL ";N
 40 IFN<4 OR N<>INT(N) THEN PRINT"VERNUEN
 FTIG BLEIBEN":GOTO30
 45 PRINTCHR$(147):PRINT:PRINT
 50 A$=" EINE ":TI$="000000":X=N-1
 60 FOR I=2 TO X
 70 T=N/I:IF INT(T)=T THENA$=" KEINE ":I=
 X
 80 NEXT
90 PRINTN "IST"A$"PRIMZAHL."
100 PRINT:PRINT:PRINT" ZEIT UM ZAHL ZU T
ESTEN"INT(TI/60+0.5*100)/100"SEKUNDEN"
110 PRINT:PRINT:PRINT:PRINT" NEUER START
 (J/N) ?"
120 GETG$:IFG$=""THEN120
130 IFG$="J"THENRUN
140 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
150 GOTO120
```

Das Programm funktioniert zwar, ist aber ziemlich ineffizient und langsam. Beispielsweise dauert der Test, ob 9001 prim ist, etwa 113 Sekunden. Mit etwas Nachdenken läßt sich das enorm beschleunigen.
Zunächst einmal brauchen wir nicht alle Zahlen zwischen 2 und N zu berücksichtigen, sondern nur die Zahlen zwischen 2 und INT(SQR(N)). Denn wenn M eine ganze Zahl ist, die N teilt und größer als INT(SQR(N)) ist, dann ist N/M eine ganze Zahl, die N teilt und kleiner als INT(SQR(N)) ist. Die nachstehende Version 2 enthält diese einfache Ergänzung.

```
10 REM 25PRIMZAHLEN 2
20 POKE53280,6:PRINTCHR$(147)" *** LANGS
AMER PRIMZAHLEN TESTER II ***"
30 PRINT:PRINT:PRINT:PRINT:INPUT"ZU TEST
ENDE ZAHL ";N
40 IFN<4 OR N<>INT(N) THEN PRINT"VERNUEN
FTIG BLEIBEN":GOTO30
45 PRINTCHR$(147):PRINT:PRINT
50 A$=" EINE ":TI$="000000":X=SQR(N)
60 FOR I=2 TO X
70 T=N/I:IF INT(T)=T THENA$=" KEINE ":I=
X
```

```
80 NEXT
90 PRINTN "IST"A$"PRIMZAHL."
100 PRINT:PRINT:PRINT" ZEIT UM ZAHL ZU T
ESTEN :"INT(TI/60+0.5*100)/100"SEKUNDEN"

110 PRINT:PRINT:PRINT:PRINT" NEUER START
 (J/N) ?"
120 GETG$:IFG$=""THEN120
130 IFG$="J"THENRUN
140 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
150 GOTO120
```

Diese Version arbeitet wesentlich schneller. Der Primzahltest für 9001 dauert jetzt kaum länger als eine Sekunde. Eine größere Zahl, etwa 987654323, wird in ungefähr 387 Sekunden getestet. Versuchen Sie nicht, eine so große Zahl mit Version 1 zu testen – wenn es Ihnen nicht gerade Spaß macht, Löcher in den Bildschirm zu starrren.
Anmerkung: Die in Version 2 des Primzahltests enthaltene Ergänzung könnte auch in das Programm 'Eratosthenes' eingebaut werden. Mit dem so erzeugten 'Eratosthenes 2' dauert es 6 Minuten und 21 Sekunden, die Primzahlen von 2 bis 15 000 auszudrucken.

```
10 REM 26ERASTOSTHENES2
20 POKE53280,6:PRINTCHR$(147)"     *** SI
EB DES ERASTOSTHENES ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 PRIMZAHLEN BISZU EINER EINGEGEBENEN";
40 PRINT" ZAHL"
50 PRINT:INPUT"BIS ZU WELCHER ZAHL ";N
60 IF N<4 OR N>15000 OR N<>INT(N) THEN P
RINT:PRINT"VERNUENFTIG BLEIBEN":GOTO50
70 DIM A%(N):PRINTCHR$(147):PRINT:PRINT"
PRIMZAHLEN VON 2 BIS";N:S=INT(SQR(N)+1)
75 PRINT:PRINTCHR$(5)
80 FORI=2TON
90 IFA%(I)=1 THEN130
100 IF 39-POS(0)<LEN(STR$(I))THENPRINT
110 PRINTI,
120 IF I<=S THEN FORJ=ITON STEPI:A%(J)=1
:NEXT
```

```
130 NEXT:PRINT
135 PRINT:PRINT:PRINT:PRINT:PRINTCHR$(15
4)"NEUER START (J/N) ?"
150 GET G$: IFG$=""THEN150
160 IFG$="J"THENRUN
170 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
180 GOTO150
```

Version 2 des Primzahltests läßt sich mit einer Idee aus dem Sieb des Eratosthenes weiter verbessern. Wir können alle geraden Zahlen größer als 2 und jede dritte Zahl nach 3 weglassen. Diese Änderungen sind in die dritte Version des Programms eingebunden worden.

```
10 REM 27PRIMZAHLEN 3
20 POKE53280,6:PRINTCHR$(147)"    *** P
RIMZAHLEN TESTER III ***"
30 PRINT:PRINT:PRINT:PRINT:INPUT"ZU TEST
ENDE ZAHL ";N
40 IFN<4 OR N<>INT(N) THEN PRINT"VERNUEN
FTIG BLEIBEN":GOTO30
45 PRINTCHR$(147):PRINT:PRINT
50 A$=" EINE ":TI$="000000":X=SQR(N)
52 IF INT(N/2)=N/2 THENA$=" KEINE ":GOTO
90
53 IF INT(N/3)=N/3 THENA$=" KEINE ":GOTO
90
60 FOR I=5 TO X STEP 6
70 T=N/I:IF INT(T)=T THENA$=" KEINE ":I=
X
75 T=N/(I+2):IF INT(T)=T THENA$=" KEINE
":I=X
80 NEXT
90 PRINTN "IST"A$"PRIMZAHL."
100 PRINT:PRINT:PRINT" ZEIT UM ZAHL ZU T
ESTEN :"INT(TI/60+0.5*100)/100"SEKUNDEN"

110 PRINT:PRINT:PRINT:PRINT" NEUER START
 (J/N) ?"
120 GETG$:IFG$=""THEN120
130 IFG$="J"THENRUN
```

```
140 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
150 GOTO120
```

Damit läuft das Programm etwas schneller ab. Der Test der Primzahl 987654323 dauert jetzt etwa 133 Sekunden, was einigermaßen akzeptabel ist.
Eine zusammengesetzte Zahl läßt sich als Produkt von Primzahlen schreiben. Zum Beispiel

6 = 3 * 2
24 = 3 * 2 * 2 * 2
81018001 = 9001 * 9001

und so weiter. Die vorkommenden Primzahlen heißen *Faktoren* der jeweiligen Zahl. Mit einigen zusätzlichen Zeilen zu 'Version 3' kann man alle Faktoren einer Zahl ausgeben lassen.
Das nächste Programm druckt die Faktoren einer Zahl aus.

```
10 REM 28PRIMFAKTOREN
20 P=0:GOTO70:REM SUBROUTINE
30 Y=N/T:IF38-POS(0)<LEN(STR$(Y)) THEN P
RINT
40 N=T:T=N/Y:IFT=INT(T)THENE=E+1:GOTO30
50 P=P+1:PRINTY"↑"E,:S=SQR(N)+1:IFP=4THE
NPRINT:P=0
55 RETURN
60 REM HAUPTTEIL
70 POKE53280,6:PRINTCHR$(147)"         ***
 PRIMFAKTOREN ***"
75 PRINT:PRINT:PRINT"DIESES PROGRAMM ZER
LEGT BELIEBIGE ZAHLENIN PRIMFAKTOREN."
80 PRINT:INPUT"EINGABE ZAHL  ";N
90 IF N<4 OR N<>INT(N)THEN PRINT:PRINT"V
ERNUEFTIG BLEIBEN":GOTO80
100 PRINTCHR$(147):PRINT:PRINT:PRINT"DIE
 PRIMFAKTOREN VON"N"SIND :"CHR$(5):PRINT

110 TI$="000000":X=SQR(N):S=X+1
120 E=1:T=N/2:IF INT(T) = T THEN GOSUB30

130 E=1:T=N/3:IF INT(T) = T THEN GOSUB30
```

```
140 FORI=5TOXSTEP6
150 E=1:T=N/I:IF INT(T) = T THEN GOSUB30

160 E=1:T=N/(I+2):IF INT(T) = T THEN GOS
UB30
170 IFI>STHENI=X
180 NEXT
190 IF 38-POS(0)<LEN(STR$(N)) THEN PRINT

200 IF N>1 THENPRINTN"↑ 1"
210 PRINT:PRINT:PRINT:PRINTCHR$(154)"ZEI
T UM FAKTOREN ZU BERECHNEN : "
220 PRINT:PRINTCHR$(5)INT(TI/60*100+.5)/
100 "SEKUNDEN"CHR$(154)
230 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N) ?"
240 GETG$:IFG$=""THEN240
250 IFG$="J"THENRUN
260 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
270 GOTO240
```

Wenn Sie eine sehr große Zahl in das Programm eingeben, rundet der Commodore 64 allerdings die Zahl und berechnet die Faktoren der gerundeten Zahl.

GROSSE PRIMZAHLEN

Es gibt Leute, die zum Zeitvertreib nach großen Primzahlen suchen. Die größten bekannten Primzahlen sind gewöhnlich Mersennesche Primzahlen. Zahlen der Form

$$2^P - 1$$

heißen *Mersennesche Zahlen* nach dem französischen Mönch Marin Mersenne, der sich mit dem Problem beschäftigt hat, welche dieser Zahlen prim sind.
Früher glaubten viele einschlägige Autoren, daß Mersennesche Zahlen prim sind, wenn der Exponent P prim ist. Für die ersten paar Exponenten ist das tatsächlich der Fall.

$2^2 - 1 = 3$
$2^3 - 1 = 7$
$2^5 - 1 = 31$
$2^7 - 1 = 127$

Aber für P = 11 ist die Zahl nicht prim.

$2^{11} - 1 = 2047 = 23 * 89$

Zumindest in den letzten 100 Jahren war die größte bekannte Primzahl der Welt immer eine Mersennesche Primzahl (mit Ausnahme einer kurzen Phase im Jahre 1951). Bis Januar 1983 war die größte bekannte Primzahl

$2^{44497} - 1$

eine Mersennesche Primzahl, die im April 1979 von David Slowinski entdeckt wurde. Es war die 27. bekannte Mersennesche Primzahl. Ungefähr im Januar 1983 fand Slowinski die viel größere Primzahl

$2^{86243} - 1$

die eine Mersennesche Primzahl mit 25 962 Dezimalstellen ist. (Vollständig ausgeschrieben würde diese Zahl mehrere Seiten dieses Buches füllen.) Der Computer, den Slowinski dabei benutzte, war ein Cray-1. Obwohl das ein ungeheuer schnelles Gerät ist, brauchte der Test 1 Stunde, 36 Minuten und 22 Sekunden Computerzeit. Es wäre unmöglich, eine beliebige Zahl in der Größenordnung von

$2^{286243} - 1$

mit einem der Programme aus dem vorigen Abschnitt einem Primzahltest zu unterziehen. Es gibt aber besondere Techniken für Mersennesche Zahlen, die in den letzten 100 Jahren entwickelt worden sind. Um zu prüfen, ob eine Zahl der Form $N = 2^P - 1$ prim ist, definiert man eine Folge:

$U_1 = 4$
$U_2 = U_1*U_1 - N*INT((U_1*U_1-2)/N)$

$\bullet \quad \bullet$
$\bullet \quad \bullet$
$\bullet \quad \bullet$

$U_{p-1} = U_{p-2}*U_{p-2}-2- N*INT((U_{p-2}*U_{p-2}-2)/N)$

Dann ist N prim genau dann, wenn $U_{p-1} = 0$ ist. Das heißt, wir benötigen ungefähr P einfache Rechenschritte, um zu testen, ob N prim ist. Für den Commodore 64 würde es ziemlich einfach, 86 243 Operationen auszuführen, wenn die betreffenden Zahlen klein wären. Hier hat man es aber mit Zahlen zu tun, die 86 243 Binärstellen haben, während der Commodore 64 nur 32 Binärstellen speichert. Man kann dieses Problem mit geeigneten Methoden umgehen, aber wir wollen hier nicht ins Einzelne gehen. Lesen Sie dazu das nächste Kapitel.

STOCHASTISCHE PRIMZAHLTESTS

Bei unbegrenzter Zeit könnten wir jede ganze Zahl daraufhin prüfen, ob sie prim ist, indem wir sie zu teilen versuchen. Aber die Zeit ist beschränkt, selbst für einen Computer.

In den letzten Jahren hat man jedoch einen neuen Test auf der Grundlage eines alten Satzes von Pierre Fermat entworfen. (Fermat lebte im 17. Jahrhundert.) Fermat hat gezeigt, daß $B^{P-1} - 1$ durch P teilbar ist, wenn P prim ist und B irgend eine andere Zahl zwischen 1 und $P - 1$ ist. Sei z. B. $P = 11$ und $B = 2$. Dann ist

$$2^{11-1} - 1$$

gleich 1023, also tatsächlich durch 11 teilbar.

Fermat hat seinen Satz für alle Werte von B bewiesen; jedoch kannten chinesische Mathematiker schon im 5. Jahrhundert v. Chr. den Satz für $B = 2$. Außerdem nahmen sie irrtümlich an, daß die Umkehrung richtig sei. Mit anderen Worten, sie waren der Ansicht, daß P prim ist, wenn 2^{P-1} durch P teilbar ist. Aber 341 teilt

$$2^{340} - 1$$

ohne daß 341 prim ist. So eine Zahl nennt man *pseudoprim zur Basis 2.* Allgemein heißt eine zusammengesetzte Zahl P, die $B^{P-1} - 1$ teilt, *pseudoprim zur Basis B.* Die meisten Zahlen, die pseudoprim scheinen, sind in Wahrheit prim. Auf dieser Tatsache beruht der Test.

```
10 REM 29PRIMZAHLEN 4
20 POKE53280,6:PRINTCHR$(147)"    *** STOC
HASTISCHER PRIMZAHLTEST ***"
30 PRINT:PRINT:PRINT:PRINT:INPUT"EINGABE
 ZAHL";N
40 IFN<4ORN<>INT(N)THENPRINT:PRINT"VERNU
ENFTIG BLEIBEN":GOTO30
```

```
50 REM FAKTOR N-1 = (2↑T)*X
60 T=0:X=N-1
70 D=X/2:IF INT(D)=D THEN T=T+1:X=D:GOTO
70
80 REM BASIS B
90 B=RND(-TI):B=INT(RND(1)*(N-3.0001)+2)

100 PRINTCHR$(147):PRINT:PRINT:PRINT"TES
T BASIS";BCHR$(5)
120 P=1
130 IF X=0 THEN 170
140 D=X/2:IFD<>INT(D) THENP=P*B:P=P-N*IN
T(P/N)
150 B=B*B:B=B-N*INT(B/N):X=INT(D):GOTO13
0
160 REM KONTROLLE B↑X
170 IFP=1ORP=N-1THEN200
180 IFT<2THENPRINT:PRINT:PRINTN"IST KEIN
E PRIMZAHL":GOTO250
190 P=P*P:P=P-N*INT(P/N):T=T-1:GOTO170
200 PRINT:PRINT:PRINTN"IST WAHRSCHLICH P
RIMZAHL"
210 PRINT:PRINT:PRINTCHR$(154)"SOLL NOCH
 MAL MIT EINER ANDEREN BASIS"
220 PRINT"GETESTET WERDEN (J/N) ?"
230 GETG$:IFG$<>"J"ANDG$<>"N"THEN230
240 IFG$="J"THEN60
250 PRINT:PRINT:PRINT:PRINTCHR$(154)"NEU
ER START (J/N) ?"
260 GETG$:IFG$=""THEN260
270 IFG$="J"THENRUN
280 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
290 GOTO260
```

Versuchen Sie es mit Zahlen wie 341 und 561, die beide nicht prim sind. Falls sich herausstellt, daß sie pseudoprim zu einer bestimmten Basis sind, nehmen Sie eine andere Basis.

Anmerkung: Das Programm ist schnell, aber auf Grund von Rundungsfehlern funktioniert es bei großen Zahlen nicht sehr gut.

KAPITEL 10

VERMISCHTES

PYTHAGOREISCHE TRIPEL

Sie erinnern sich bestimmt an den schon erwähnten Satz des Pythagoras. Er besagt, daß in einem rechtwinkligen Dreieck mit den Seiten X, Y und Z, wobei Z die Hypotenuse (die längste Seite) ist, die folgende Beziehung gilt:

$$X^2 + Y^2 = Z^2$$

Das klassische Beispiel, das einem dabei sofort einfällt, ist das 3, 4, 5-Dreieck:

$$3^2 + 4^2 = 5^2$$

Ein anderes Beispiel ist das 5, 12, 13-Dreieck:

$$5^2 + 12^2 = 13^2$$

Es gibt natürlich unendlich viele verschiedene Beispiele für rechtwinklige Dreiecke. Aber wie viele Fälle gibt es, in denen X, Y und Z ganze Zahlen sind? Die Antwort lautet: unendlich viele. Solche Zahlen X, Y und Z heißen *Pythagoreische Tripel.*
Wie erzeugt man eine Liste Pythagoreischer Tripel? Natürlich sollte eine solche Liste keine Tripel enthalten, die Vielfache eines anderen sind. Weil z. B. 3, 4, 5 ein Pythagoreisches Tripel ist, ist auch 6, 8, 10 eines. Pythagoreische Tripel, die keinen gemeinsamen Faktor haben, heißen *primitiv.* Also ist 3, 4, 5 ein primitives Pythagoreisches Tripel, während 6, 8, 10 nicht primitiv ist.
Es wäre ziemlich zeitraubend, Pythagoreische Tripel zu finden, wenn es nicht einigen Mathematikern gelungen wäre, eine elegante Methode zur Erzeugung primitiver Pythagoreischer Tripel zu entwickeln. Die Technik läßt sich so beschreiben:

1. Wähle zwei positive ganze Zahlen A und B, so daß gilt:

   (a) A ist größer als B.
   (b) A + B ist ungerade (also ist eine der Zahlen ungerade und die andere gerade).
   (c) A und B haben keinen gemeinsamen Teiler außer 1.

2. Berechne

$X = A^2 - B^2$
$Y = 2*A*B$
$Z = A^2 + B^2$

Die Zahlen X, Y, Z bilden ein Pythagoreisches Tripel. Umgekehrt erhält man auf diese Weise jedes primitive Pythagoreische Tripel. Mit dieser Technik erzeugt das nächste Programm eine Liste Pythagoreischer Tripel. Das Verfahren beginnt bei A = 2. Das Programm bestimmt dann alle möglichen Werte von B, die die geforderten Bedingungen erfüllen. Der Wert von A wird dann erhöht und das Verfahren wiederholt. Das kann man so lange fortsetzen, wie man will. Man kann auch mit einem anderen Anfangswert A beginnen.

```
10 REM 30TRIPEL
20 POKE53280,6:PRINTCHR$(147)"     *** PY
THAGOREISCHE TRIPEL ***"
30 PRINT:PRINT"DIESES PROGRAMM ZEIGT EIN
IGE PRIMITIVE  PYTHAGOREISCHE TRIPEL"
40 PRINT:PRINT:PRINT:PRINT:PRINT"
LEER-TASTE FUER START":K=0:A=2:B=3
50 GETG$:IFG$<>" "THEN50
70 PRINTCHR$(147)"     *** PYTHAGOREISCHE
 TRIPEL ***"
80 PRINT:PRINT"ZAHL     ** X **     **Y**
   **Y**"CHR$(5):PRINT
90 B=B-2:IFB<1THENA=A+1:B=A-1
100 A1=A:B1=B
110 N=INT(A1/B1):A2=A1-N*B1
120 IFA2<>0THENA1=B1:B1=A2:GOTO110
130 IFB1<>1THEN90
140 K=K+1:PRINTK,A*A-B*B,2*A*B,A*A+B*B
150 IFINT(K/15)<>K/15THEN90
160 PRINT:PRINT:PRINTCHR$(154)"WEITERE T
RIPEL (J/N) ?"
170 GETG$:IFG$=""THEN170
180 IFG$="J"THEN70
190 IFG$="N"THENPOKE53280,14:PRINTCHR$(1
47)"AUF WIEDERSEHEN":END
200 GOTO170
```

MEHRFACHGENAUE POTENZEN

Der Commodore 64 verwaltet Zahlen mit 9 signifikanten Stellen genau. Also werden ganze Zahlen, die kleiner als 999 999 999 sind, korrekt gespeichert. Größere Zahlen werden gerundet. Wie kann man dann große Potenzen von Zahlen genau berechnen? Wieviel ist z. B. 2^{130}? Das Problem läßt sich mit mehrfachgenauer Arithmetik lösen.

Mehrfachgenaue Arithmetik ist u. a. möglich, wenn man die Stellen einer Zahl in einem Feld M(I) ablegt. Dabei speichert M(0) die letzten 4 Stellen, M(1) die vorhergehenden 4, und so weiter. Auf diese Weise würde 987654 gespeichert als

M(1) = 98, M(0) = 7654.

Die ursprüngliche Zahl erhält man zurück, indem man die Elemente des Feldes in Strings umwandelt und nacheinander ausdruckt. Natürlich kann man sie auch durch diese Formel gewinnen:

$$M = M(1)*10^4 + M(0)$$

Angenommen, wir wollen M = 987654 mit N = 23456 multiplizieren. Als erstes schreiben wir diese Zahlen in die Felder M(I) und N(I).

M(1) = 98, M(0) = 7654
N(1) = 2, N(0) =3456

Dann bilden wir das folgende Feld:

C(0) = M(0)*N(0)
C(1) = M(1)*N(0) + M(0)*N(1)
C(2) = M(1)*N(1)

Wir erhalten:

C(0) = 26452224
C(1) = 353996
C(2) = 196

Daraus berechnen wir das Produkt M*N als

$$196*10^8 + 353996*10^4 + 26452224$$
$$= 19600000000 + 35399960000 + 26452224$$
$$= 23166412224$$

Natürlich geschieht die Berechnung nicht wie eben gezeigt – der Commodore 64 würde die Zahlen einfach runden. Wir streichen vielmehr von links Stellen von C(0), bis nur noch 4 übrig sind. Dann addieren wir die gestrichenen Stellen zu C(1).

C(0) → 2224
C(1) → 353996 + 2645 = 356641

Als nächstes streichen wir von links Stellen von C(1), bis noch 4 übrig sind. Die gestrichenen Stellen addieren wir zu C(2).

C(1) → 6641
C(2) → 196 + 35 = 231

Damit erhalten wir:

C(2) = 231, C(1) = 6641, C(0) =2224

Daraus lesen wir sofort das Ergebnis ab:

987654*23456 = 23166412224

Dieses Verfahren gilt ganz allgemein. Bei anderen, größeren Zahlen existieren vielleicht Werte für die Feldelemente M(2), M(3) etc. In dem Fall würden wir das folgende Feld bilden:

C(0) = M(0)*N(0)
C(1) = M(1)*N(0) + M(0)*N(1)
C(2) = M(2)*N(0) + M(1)*N(1) + M(0)*N(2)
etc.

Danach wird wie oben nach der Streich-Methode verfahren, bis schließlich das Resultat ausgedruckt werden kann.
Das Programm 'Mehrfachgenaue Potenzen' zeigt, wie man Potenzen einer gegebenen Zahl mit Hilfe der eben beschriebenen Technik genau berechnet. Das Programm läuft, bis eine 40stellige Zahl erreicht ist. Um die Genauigkeit zu vergrößern, brauchen Sie nur den Wert von X in Zeile 80 zu ändern. Der Wert von X + 1 mal 4 ist der Genauigkeitsgrad.

```
10 REM 31AUF 40 STELLEN
20 POKE53280,6:PRINTCHR$(147)"      *** E
RHOEHTE STELLENZAHL ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 POTENZEN VON  GANZEN ZAHLEN AUF ";
40 PRINT"VIERZIG STELLEN GENAU"
50 PRINT:PRINT"BITTE DIE ZU POTENZIEREND
E ZAHL EINGEBEN"
60 PRINT:INPUT"ZAHL";N
70 IFN<2ORN>99999999ORN<>INT(N)THENPRIN
TPRINT:"!! FEHLER !!":GOTO60
80 T=10000:X=9:Y=X+2:DIMM(X),N(X),L(Y):K
=1:N(0)=N:M(0)=N
90 FORI=0TO2
100 IFM(I)>=TTHENQ=INT(M(I)/T):M(I)=M(I)
-Q*T:M(I+1)=M(I+1)+Q
110 N(I)=M(I)
120 NEXT
130 PRINTCHR$(147)"ERHOEHTE STELLENZAHL
VON";N:PRINTCHR$(5)
140 K=K+1
150 FORI=0 TO Y:L(I)=0:NEXT
160 FORJ=0 TO 2:FORI=0 TO X
170 L(I+J)=L(I+J)+N(I)*M(J)
180 NEXT:NEXT
190 FORI=0 TO X:N(I)=L(I):NEXT
200 FORI=0 TO X
210 IFN(I)>=T THENQ=INT(N(I)/T):N(I)=N(I
)-Q*T:N(I+1)=N(I+1)+Q
220 NEXT
230 FORI=0 TO X
240 IFN(I)>0 THEN L=I
250 NEXT
260 PRINTN"↑"K
270 FORI=L TO 0 STEP -1
280 A$=MID$(STR$(N(I)),2)
290 IFI<LANDLEN(A$)<>4 THEN A$="0"+A$:GO
TO290
300 PRINTA$;:NEXT:PRINT
310 IF (N(X)*T+N(X-1))*N>=T*TTHEN360
```

```
320  IFINT(K/10)<>K/10THEN140
330  PRINT:PRINTCHR$(154)"WEITERMACHEN (J
/N) ?"
340 GETG$:IFG$<>"N"ANDG$<>"J"THEN340
350  IFG$="J"THEN130
355 PRINTCHR$(147)
360 PRINTCHR$(154):PRINT:PRINT"NEUER STA
RT (J/N) ?"
370 GETG$:IFG$<>"J"ANDG$<>"N"THEN370
380  IFG$="J"THENRUN
390 PRINT:PRINTCHR$(147)"AUF WIEDERSEHEN
 ":END
```

KAPITEL 11

MATRIZEN

Matrizen sind rechteckige Zahlenfelder, bei denen Position und Wert jeder Zahl von Bedeutung sind. Eine Matrix wird gewöhnlich, aber nicht immer, in Klammern dargestellt:

$$\begin{bmatrix} 2 & -3 & 6 \\ -4 & 4 & 5 \end{bmatrix}$$

Die Größe einer Matrix ist durch die Anzahl der Zeilen und Spalten gegeben. Eine Matrix heißt M-mal-N-Matrix, wenn sie M Zeilen und N Spalten hat. Das obige Beispiel ist also eine 2-mal-3-Matrix. Hier sind weitere Beispiele für Matrizen:

$$\begin{bmatrix} 8 & 12 & 0 \\ 1 & 1 & 8 \\ 8 & 12 & 1 \end{bmatrix}$$ Eine 3-mal-3-Matrix

$$\begin{bmatrix} 9 & 12 \\ 0 & 1 \\ 8 & -2 \end{bmatrix}$$ Eine 3-mal-2-Matrix

$$\begin{bmatrix} 9 & 12 \\ 3 & 2 \end{bmatrix}$$ Eine 2-mal-2-Matrix

Ist die Zeilenzahl einer Matrix gleich der Zahl ihrer Spalten, nennt man sie eine quadratische Matrix.
Matrizen sind nützlich, um Informationen auf übersichtliche Weise zu speichern. Nehmen Sie z. B. an, Sie haben den Betrag errechnet, den Ihr Haushalt in jedem Quartal eines Jahres für drei Arten von Brennstoff ausgibt. Diese Information könnten Sie wie folgt in einer Tabelle darstellen:

Quartal	Gas	Elektrizität	Fester Brennstoff
1	30	28	5
2	27	19	4
3	25	15	0
4	32	27	3

Diese Daten können in eine 4-mal-3-Matrix geschrieben werden.

$$\begin{bmatrix} 30 & 28 & 5 \\ 27 & 19 & 4 \\ 25 & 15 & 0 \\ 32 & 27 & 3 \end{bmatrix}$$

In Ihrem Commodore 64 können Sie Matrizen mit Hilfe zweidimensionaler Felder wie etwa A(I,J) speichern. Zum Beispiel könnte die Matrix mit den Brennstoffausgaben durch folgende Zuweisungen gespeichert werden:

A(1,1) = 30 A(1,2) = 28 A(1,3) = 5
A(2,1) = 27 A(2,2) = 19 A(2,3) = 4
A(3,1) = 25 A(3,2) = 15 A(3,3) = 0
A(4,1) = 32 A(4,2) = 27 A(4,3) = 3

Die Informationen würden natürlich mit einem READ-Befehl eingegeben:

```
FOR I=1 TO 4:FOR J=1 TO 3:READ A(I,J):NEXT:NEXT
DATA 30,28,5,27,19,4,25,15,0,32,27,3
```

Das nächste kurze Programm verdeutlicht, wie Daten eingelesen und dann ausgedruckt werden.

```
5 REM 32DEMO MATRIZEN
10 PRINTCHR$(147)
20 FORI=1TO4:FORJ=1TO3:READA(I,J):NEXT:N
EXT
30 FORI=1TO4
40 FORJ=1TO3:PRINTA(I,J);:NEXT
50 PRINT
60 NEXT
100 DATA 30,28,5,27,19,4,25,15,0,32,27,3
```

Beachten Sie, daß wir die Zähler I und J besser bei 0 statt bei 1 hätten beginnen lassen, um den Speicherplatz des Commodore 64 besser auszunutzen, denn Felder fangen bei 0 an. Die Zählung von 1 an ist allerdings begrifflich einfacher. Ebenso ist zu beachten, daß der Commodore 64 eine DIM-Anweisung erwartet, wenn die Anzahl der Zeilen oder Spalten einer Matrix 10 überschreitet.

ADDIEREN VON MATRIZEN

Zwei Matrizen können addiert werden, wenn sie gleich groß sind. Anders gesagt, zwei Matrizen können addiert werden, wenn sie die gleiche Anzahl von Zeilen und die gleiche Anzahl von Spalten haben. Diese Addition wird nacheinander für je 2 Elemente in den gleichen Positionen ausgeführt. Zum Beispiel

$$\begin{bmatrix} 1 & 5 & 4 \\ -2 & 3 & 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 & 5 \\ 3 & 2 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+2 & 5+0 & 4+5 \\ -2+3 & 3+2 & 1+1 \end{bmatrix}$$

$$= \begin{bmatrix} 3 & 5 & 9 \\ 1 & 5 & 2 \end{bmatrix}$$

WOZU MATRIZEN ADDIEREN?

Weiter oben hatten wir ein Beispiel mit Brennstoffkosten für vier Quartale eines Jahres. Nehmen wir an, die entsprechende Matrix sieht im nächsten Jahr so aus:

$$\begin{bmatrix} 34 & 27 & 4 \\ 30 & 20 & 3 \\ 20 & 16 & 1 \\ 35 & 29 & 5 \end{bmatrix}$$

Wenn wir die beiden Matrizen addieren, erhalten wir die Ausgaben, die wir für die drei Brennstoffe in einem gegebenen Quartal über zwei Jahre gehabt haben. Das Ergebnis ist diese Matrix:

$$\begin{bmatrix} 64 & 55 & 9 \\ 57 & 39 & 7 \\ 45 & 31 & 1 \\ 67 & 56 & 8 \end{bmatrix}$$

Ihr Commodore 64 kann Matrizen oder zweidimensionale Felder sehr leicht addieren. Seien zwei Felder A(I,J) und B(I,J) gegeben, wobei jedesmal I von 1 bis 4 und J von 1 bis 3 läuft. Durch Addition der beiden Felder ergibt sich ein drittes Feld wie folgt:

```
FOR I=1 TO 4:FOR J=1 TO 3
C(I,J) = A(I,J) + B(I,J)
NEXT:NEXT
```

Das Subtrahieren von Matrizen geschieht nach den gleichen Regeln wie das Addieren.

MULTIPLIZIEREN VON MATRIZEN

Das Multiplizieren von Matrizen ist begrifflich nicht ganz einfach. Um zwei Matrizen multiplizieren zu können, muß die Anzahl der Spalten in der ersten Matrix mit der Anzahl der Zeilen in der zweiten Matrix übereinstimmen. Man kann also eine 4-mal-3-Matrix mit einer 3-mal-2-Matrix multiplizieren, aber nicht mit einer 2-mal-3-Matrix. Das Produkt einer M-mal-N-Matrix mit einer N-mal-P-Matrix ist eine M mal P-Matrix. Wie das Verfahren tatsächlich abläuft, läßt sich am besten an einem Beispiel zeigen, das anschließend erläutert wird. Angenommen, wir wollen die folgenden Matrizen multiplizieren:

$$\begin{bmatrix} 2 & -3 & 6 \\ -4 & 4 & 5 \end{bmatrix}$$

Die erste Matrix, eine 2-mal-3-Matrix

$$\begin{bmatrix} 9 & 12 \\ 0 & 1 \\ 8 & -2 \end{bmatrix}$$

Die zweite Matrix, eine 3-mal-2-Matrix

Da die Anzahl der Spalten der ersten Matrix mit der Anzahl der Zeilen der zweiten Matrix übereinstimmt, kann die erste mit der zweiten multipliziert werden.

$$\begin{bmatrix} 2 & -3 & 6 \\ -4 & 4 & 5 \end{bmatrix} * \begin{bmatrix} 9 & 12 \\ 0 & 1 \\ 8 & -2 \end{bmatrix}$$

$$= \begin{bmatrix} 2*9 + -3*0 + 6*8 & 2*12 + -3*1 + 6*-2 \\ -4*9 + 4*0 + 5*8 & -4*12 + 4*1 + 4*-2 \end{bmatrix}$$

$$= \begin{bmatrix} 66 & 9 \\ 4 & -52 \end{bmatrix}$$

Zum Multiplizieren sehen Sie sich zunächst jede Zeile der ersten Matrix und jede Spalte der zweiten Matrix an. Alle haben die gleiche Anzahl von Elementen. Bei jeder dieser Zeilen und Spalten multipliziert man zunächst jeweils die ersten Elemente, dann die zweiten und so fort, und zum Schluß addiert man alle so gewonnenen Produkte zusammen. Auf diese Weise erhält man ein Element der Produktmatrix.

Auf Ihrem Commodore 64 können Sie das Produkt der M-mal-N-Matrix A(I,J) mit der N-mal-P-Matrix B(I,J) wie folgt bilden. Das Ergebnis ist eine M mal P-Matrix C(I,J).

```
FOR I=1 TO M
FOR J= 1 TO P
C(I,J) = 0
FOR K=1 TO N : C(I,J) = C(I,J) + A(I,K)*B(K,J) : NEXT
NEXT
NEXT
```

Wenn man das Produkt A*B zweier Matrizen A und B bilden kann, braucht man nicht auch das Produkt B*A bilden zu können. Selbst wenn das Produkt B*A existiert, muß es nicht gleich A*B sein. Suchen Sie nach einfachen Beispielen!

WOZU MATRIZEN MULTIPLIZIEREN?

Wir wollen uns noch einmal das Beispiel mit den Brennstoffkosten ansehen. Vergegenwärtigen Sie sich die einzelnen Daten:

Quartal	Gas	Elektrizität	Fester Brennstoff
1	30	28	5
2	27	19	4
3	25	15	0
4	32	27	3

Angenommen, es liegen zwei Schätzungen für den mutmaßlichen Kostenanstieg für diese Brennstoffe vor:

	Schätzung 1	Schätzung 2
Gas	10%	5%
Elektrizität	5%	10%
Fester Brennstoff	10%	10%

Die nächste Tabelle zeigt diesen Anstieg in Dezimaldarstellung.

	Schätzung 1	Schätzung 2
Gas	0.1	0.05
Elektrizität	0.05	0.1
Fester Brennstoff	0.1	0.1

Wieviel mehr würden Sie in jedem Quartal für die drei Brennstoffe zahlen müssen? Das hängt davon ab, welche Schätzung Sie zugrundelegen. Das Ergebnis läßt sich wie folgt tabellieren:

Quartal	Schätzung 1	Schätzung 2
1	30*.1 + 28*.05 + 5*.1	30*.05 + 28*.1 + 5*.1
2	27*.1 + 19*.05 + 4*.1	27*.05 + 19*.1 + 4*.1
3	25*.1 + 15*.05 + 0*.1	25*.05 + 15*.1 + 0*.1
4	32*.1 + 27*.05 + 3*.1	32*.05 + 27*.1 + 3*.1

Nach dem Ausrechnen ergibt sich diese Tabelle:

Quartal	Schätzung 1	Schätzung 2
1	4.9	4.8
2	4.05	3.65
3	3.24	2.75
4	4.85	4.6

Wie Sie vielleicht bemerkt haben, ist die so erhaltene Matrix das Produkt der Matrix der Ausgaben mit der Matrix der Schätzungen. Das heißt, es ist das nachstehende Matrizenprodukt:

$$\begin{bmatrix} 30 & 28 & 5 \\ 27 & 19 & 4 \\ 25 & 15 & 0 \\ 32 & 27 & 3 \end{bmatrix} * \begin{bmatrix} 0.1 & 0.05 \\ 0.05 & 0.1 \\ 0.1 & 0.1 \end{bmatrix}$$

NULLEN UND EINSEN

Matrizen, die nur aus Nullen bestehen, heißen Nullmatrizen. Multipliziert man eine Matrix mit einer Nullmatrix, so erhält man eine Nullmatrix. Zum Beispiel:

$$\begin{bmatrix} 30 & 28 \\ 27 & 19 \\ 25 & 15 \end{bmatrix} * \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 30*0 + 28*0 & 30*0 + 28*0 \\ 27*0 + 19*0 & 27*0 + 19*0 \\ 25*0 + 15*0 & 25*0 + 15*0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Das Produkt zweier Nicht-Nullmatrizen kann eine Nullmatrix ergeben, wie das nächste Beispiel zeigt.

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1*1 + 1*-1 & 1*-1 + 1*1 \\ 1*1 + 1*-1 & 1*-1 + 1*1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Eine quadratische Matrix mit Einsen auf der Diagonalen (von links oben nach rechts unten) und sonst nur Nullen heißt *Einheitsmatrix.* Diese beiden Matrizen z. B. sind Einheitsmatrizen:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Multiplikation einer Matrix mit einer Einheitsmatrix läßt die Matrix unverändert. In anderen Worten verhält sich die Einheitsmatrix ganz ähnlich wie die 1 bei der üblichen Multiplikation von Zahlen. Zum Beispiel:

$$\begin{bmatrix} 30 & 28 \\ 27 & 19 \\ 25 & 15 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 30*1 + 28*0 & 30*0 + 28*1 \\ 27*1 + 19*0 & 27*0 + 19*1 \\ 25*1 + 15*0 & 25*0 + 15*1 \end{bmatrix}$$

$$= \begin{bmatrix} 30 & 28 \\ 27 & 19 \\ 25 & 15 \end{bmatrix}$$

INVERTIEREN VON MATRIZEN

Die Inverse oder Umgekehrte einer Matrix, sofern sie existiert, hat dieselbe Eigenschaft wie das Inverse einer gewöhnlichen Zahl. Das Inverse der Zahl 4 ist 0.25, und es gilt

$$4 * 0.25 = 1, \qquad 0.25*4 = 1$$

Für eine Matrix A wird die Inverse (wenn sie existiert) mit A^{-1} bezeichnet, und es gilt

$$A * A^{-1} = I, \qquad A^{-1} * A = 1$$

wobei I für eine Einheitsmatrix steht.

Nur quadratische Matrizen können Inverse besitzen. Tatsächlich haben nur bestimmte quadratische Matrizen eine Inverse.

Es gibt verschiedene Methoden, die Inverse einer quadratischen Matrix zu finden. Für Ihren Commodore 64 ist das unten beschriebene schrittweise Verfahren gut geeignet. Es besteht darin, sogenannte Zeilenoperationen auszuführen. Vereinfachend könnte man sagen, wir setzen zunächst eine Einheitsmatrix neben die Matrix, die wir invertieren wollen. Die Zeilenoperationen werden dann auf beide Matrizen gleichzeitig angewandt, bis die ursprüngliche in eine Einheitsmatrix umgewandelt ist. Die Matrix, die zu Beginn eine Einheitsmatrix war, ist jetzt die Inverse.

Ein Programm zur Berechnung der Inversen von Matrizen geben wir später an. Vorher beschreiben wir kurz die Methode.

Angenommen, wir wollen die Inverse der folgenden Matrix finden:

$$\begin{bmatrix} 2 & 1 \\ -2 & 4 \end{bmatrix}$$

Wir setzen eine 2-mal-2-Einheitsmatrix neben diese Matrix:

$$\begin{bmatrix} 2 & 1 \\ -2 & 4 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Da die linke obere Eintragung eine 1 werden soll, teilen wir die ganze erste Zeile durch 2:

$$\begin{bmatrix} 1 & 0.5 \\ -2 & 4 \end{bmatrix} \qquad \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}$$

Als nächstes soll die linke untere Eintragung 0 werden; also addieren wir das Zweifache der ersten Zeile zur zweiten Zeile:

$$\begin{bmatrix} 1 & 0.5 \\ 0 & 5 \end{bmatrix} \qquad \begin{bmatrix} 0.5 & 0 \\ 1 & 1 \end{bmatrix}$$

Nun soll aus der rechten unteren Eintragung (der linken Matrix) eine 1 werden. Dazu teilen wir die zweite Zeile durch 5:

$$\begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0.5 & 0 \\ 0.2 & 0.2 \end{bmatrix}$$

Um aus der linken Matrix eine Einheitsmatrix zu machen, subtrahieren wir das Einhalbfache der zweiten Zeile von der ersten:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0.4 & -0.1 \\ 0.2 & 0.2 \end{bmatrix}$$

Die rechte Matrix ist jetzt die Inverse der Ausgangsmatrix. Folgende Rechnung bestätigt das:

$$\begin{bmatrix} 2 & 1 \\ -2 & 4 \end{bmatrix} * \begin{bmatrix} 0.4 & -0.1 \\ 0.2 & 0.2 \end{bmatrix}$$

$$= \begin{bmatrix} 2^*.4 + 1^*.2 & 2^*-.1 + 1^*.2 \\ -2^*.4 + 4^*.2 & -2^*-.1 + 4^*.2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Das unten angeführte Programm berechnet die Inverse quadratischer Matrizen, sofern überhaupt eine Inverse existiert.

```
10 REM 33INVERSE MATRIZEN
20 POKE53280,6:PRINTCHR$(147)"        ***
INVERSE MATRIZEN ***"
30 PRINT:PRINT"DIESES PROGRAMM BERECHNET
 DIE INVERSE   EINER N MAL N MATRIX"
40 PRINT:PRINT"BITTE DIE GROESSE DER MAT
RIX EINGEBEN"
50 PRINT:INPUT"GROESSE VON N  ";N
60 IFN<1ORN<>INT(N)THENPRINT"!! UNMOEGLI
CH !!":GOTO50
70 REM EINGABE DER MATRIX
```

```
80 DIM A(N,N),B(N,N)
90 FORI=1TON:FORJ=1TON
100 PRINTCHR$(147):PRINT:PRINT"EINGABE D
ER MATRIX (TERM FUER TERM)"
110 PRINTCHR$(19):PRINT:PRINT:PRINT:PRIN
T"REIHE";I,
130 PRINT"SPALTE";J;:INPUT A(I,J)
140 NEXT:NEXT
150 REM RECHNUNG
160 PRINTCHR$(147)"DIE MATRIX":PRINT
165 P=7:IFN>6THENP=INT(40/(N+1))
170 FORI=1TON:O=0:FORJ=1TON:O=O+P:PRINTA
(I,J)TAB(O):NEXT:PRINT:NEXT
180 PRINT:PRINT"RECHNUNG"
190 FORI=1TON:B(I,I)=1:NEXT
200 X=0:GOSUB300
210 REM SCHLUSS
220 PRINT"NEUER START (J/N) ?"
230 GETG$:IFG$<>"N"ANDG$<>"J"THEN230
240 IFG$="J"THENRUN
250 PRINTCHR$(147)"AUF WIEDERSEHEN":END
300 REM RECHNUNG
310 X=X+1:Z=X:PRINT"*";
320 IF A(Z,X)=0 THEN Z=Z+1:IFZ<=N THEN32
0
330 IFZ>NTHENPRINT:PRINT:PRINT"KEINE INV
ERSE MATRIX":RETURN
340 IFZ<>XTHENR=1/A(Z,X):I=X:K=Z:GOSUB50
0
350 IFA(X,X)<>1THENR=1/A(X,X):I=X:GOSUB5
50
360 FORI=1TON
370 IFI=XTHENI=I+1:IFI>NTHEN390
380 IFA(I,X)<>0THENR=-A(I,X):K=X:GOSUB50
0
390 NEXT
400 IFX<NTHEN310
410 PRINT:PRINT"DIE INVERSE MATRIX"CHR$(
5):PRINT
415 P=7:IFN>6THENP=INT(40/(N+1))
```

```
420 FORI=1TON:O=0:FORJ=1TON:O=O+P:PRINTB
(I,J)TAB(O):NEXT:PRINT:NEXT
430 PRINTCHR$(154):RETURN
500 REM RECHNUNG
510 FORJ=1TON
520 A(I,J)=A(I,J)+R*A(K,J):B(I,J)=B(I,J)
+R*B(K,J)
530 NEXT
540 RETURN
550 REM RECHNUNG
560 FORJ=1TON
570 A(I,J)=R*A(I,J):B(I,J)=R*B(I,J)
580 NEXT
590 RETURN
```

GLEICHUNGSSYSTEME

Matrizen lassen sich zum Lösen von Gleichungssystemen verwenden. Das nächste Beispiel zeigt ein System zweier Gleichungen mit zwei Unbekannten:

$$3*X + 1*Y = 7$$
$$5*X + 2*Y = 9$$

Dasselbe sieht in Matrixschreibweise so aus:

$$\begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} 7 \\ 9 \end{bmatrix}$$

Eine mögliche Lösung der Aufgabe besteht darin, die Inverse der 2-mal-2-Matrix auf der linken Seite zu finden und dann die Gleichung mit dieser Inversen zu multiplizieren. Im Beispiel ist die Inverse die Matrix

$$\begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix}$$

Wir erhalten also die Gleichungskette

$$\begin{bmatrix} X \\ Y \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix}$$

$$= \begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix} * \begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix}$$

$$= \begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix} * \begin{bmatrix} 7 \\ 9 \end{bmatrix}$$

$$= \begin{bmatrix} 2*7-1*9 \\ -5*7+3*9 \end{bmatrix}$$

$$= \begin{bmatrix} 5 \\ -8 \end{bmatrix}$$

Also lösen X = 5 und Y = −8 das Gleichungssystem.
Das unten abgedruckte Programm ahmt im wesentlichen diese Schritte nach, um ein System von N Gleichungen in N Unbekannten zu lösen. Zu Anfang des Verfahrens wird wie oben das Gleichungssystem in zwei Matrizen A(I,J) und B(I) geschrieben. Das obige Beispiel würde wie folgt dagestellt:

$$\begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix} \qquad \begin{bmatrix} 7 \\ 9 \end{bmatrix}$$

Dann werden Zeilenoperationen ausgeführt, bis die Matrix auf der linken Seite eine Einheitsmatrix geworden ist. Die rechte Matrix geht dabei in die Lösung des Gleichungssystems über.

```
10 REM 34GLEICHUNGSSYSTEME
20 POKE53280,6:PRINTCHR$(147)"       *** G
LEICHUNGSSYSTEME ***"
30 PRINT:PRINT"DIESES PROGRAMM LOESST GL
EICHUNGSSYSTEMEMIT N UNBEKANNTEN";
40 PRINT:PRINT"ANZAHL DER UNBEKANNTEN"
50 PRINT:INPUT"ZAHL";N
60 IFN<2ORN<>INT(N)THENPRINT:PRINT"NOCHM
```

```
AL VERSUCHEN":GOTO50
70 REM EINGABE KOEFFIZENTEN
80 DIM A(N,N),B(N)
90 FORI=1TON:FORJ=1TON
100 PRINTCHR$(147)"EINGABE MATRIX DER KO
EFFIZENTEN           (TERM FUER TERM)"
110 PRINT:PRINT"REIHE";I,"SPALTE";J;:INP
UTA(I,J)
120 NEXT
130 PRINT:INPUT"RECHTE SEITE";B(I)
140 NEXT
180 REM RECHNUNG
190 PRINTCHR$(147)"MATRIX DER KOEFFIZENT
EN":PRINT
195 P=INT(40/(N+1.3))
200 FORI=1TON:FORJ=1TON:PRINTA(I,J)TAB(P
*J):NEXT:PRINTTAB(P*(N+.5))B(I):NEXT
210 PRINT:PRINT"RECHNUNG"
220 X=0:GOSUB300
230 REM SCHLUSS
240 PRINT:PRINT"NEUER START (J/N) ?"
250 GETG$:IFG$<>"J"ANDG$<>"N"THEN250
260 IFG$="J"THENRUN
270 PRINTCHR$(147)"AUF WIEDERSEHEN":END
300 REM RECHNUNG
310 X=X+1:Z=X:PRINT"*";
320 IFA(Z,X)=0THENZ=Z+1:IFZ<=NTHEN320
330 IFZ>NTHENPRINT:PRINT:PRINT"KEINE LOE
SUNG":RETURN
340 IFZ<>XTHENR=1/A(Z,X):I=X:K=Z:GOSUB50
0
350 IFA(X,X)<>1THENR=1/A(X,X):I=X:GOSUB5
50
360 FORI=1TON
370 IFI=XTHENI=I+1:IFI>NTHEN390
380 IFA(I,X)<>0THENR=-A(I,X):K=X:GOSUB50
0
390 NEXT
400 IFX<NTHEN310
410 PRINT:PRINT"DIE LOESUNG":PRINT
```

```
420 FORI=1TON
430 IF POS(1)+LEN(STR$(B(I)))>38THENPRIN
T
440 PRINTB(I)
450 NEXT
460 PRINT:RETURN
500 :
510 FORJ=1TON
520 A(I,J)=A(I,J)+R*A(K,J)
530 NEXT
540 B(I)=B(I)+R*B(K):RETURN
550 :
560 FORJ=1TON
570 A(I,J)=R*A(I,J)
580 NEXT
590 B(I)=R*B(I):RETURN
```

KAPITEL 12

CODES

Der Bedarf an gesicherter Datenübertragung wird ständig größer. Dadurch wurde die Forschung auf dem Gebiet der Kryptographie – der Kunst des Verschlüsselns – sehr angeregt. Kryptographie besteht aus zwei Schritten: die Nachricht bzw. die Daten müssen zuerst verschlüsselt und dann wieder entschlüsselt werden.

TAUSCHALPHABETE

Die einfachsten Schlüssel sind diejenigen, in denen jeder Buchstabe durch etwas anderes ersetzt wird, in der Regel durch einen Buchstaben desselben Alphabets. Hier ist ein Beispiel:

verschlüsseln→ ←entschlüsseln	verschlüsseln→ ←entschlüsseln
A – A	B – F
C – K	D – P
E – U	F – Z
G – E	H – J
I – O	J – T
K – Y	L – D
M – I	N – N
O – S	P – X
Q – C	R – H
S – M	T – R
U – W	V – B
W – G	X – L
Y – Q	Z – V

Die Nachricht MESSAGE würde in IUMMAEU übersetzt.
In diesem Schlüssel ist das Tauschalphabet nicht zufällig gewählt worden. Werden die Buchstaben des Alphabets von 0 bis 25 durchnumeriert, dann wird der Buchstabe mit der Nummer NUM durch den Buchstaben mit folgender Nummer ersetzt:

NUM*5 – 26*INT(NUM*5/26)

Diese Zuordnung wird rückgängig gemacht durch die Formel

NUM*21 – 26*INT(NUM*21/26)

Dieses Tauschalphabet kann mit einem kurzen Programm aufgelistet werden:

```
10 REM 35EINFACHCODE
20 PRINTCHR$(147)"CODIERUNG":PRINT
30 FORI=0TO25
40 J=I*5-26*INT(I*5/26)
50 PRINTCHR$(65+I)"-"CHR$(65+J),
60 NEXT
70 PRINT:PRINT:PRINT"ENTSCHLUESSELUNG":P
RINT
80 FORI= 0TO25
100 J=I*21-26*INT(I*21/26)
110 PRINT CHR$(65+I)"-"CHR$(65+J),
120 NEXT
```

Um einen Buchstaben zu verschlüsseln, multiplizieren wir die Nummer des Buchstabens mit 5 und lassen Vielfache von 26 weg. Zum Entschlüsseln des Buchstaben dividieren wir durch 5 und lassen Vielfache von 26 weg. Ebenso könnten wir mit 21 multiplizieren und Vielfache von 26 vernachlässigen, weil gilt:

$$\begin{aligned} 5 * 21 &= 105 \\ &= 1 + 4 * 26 \end{aligned}$$

Mit anderen Worten: Wenn wir Vielfache von 26 weglassen, ist 21 reziprok zu 5. Man sagt auch, daß 21 das Inverse von 5 modulo 26 ist.
Dieser Schlüssel hat den offensichtlichen Nachteil, daß eine Nachricht wie PLEASE COME QUICKLY zu XDUAMU KSIU CWIKYDQ verschlüsselt würde: Leerstellen bleiben Leerstellen. Um diesem Mangel abzuhelfen, sollten wir Leerstellen, Punkt, Komma, Ziffern und vielleicht das Fragezeichen in unser Tauschalphabet mit aufnehmen.
Da uns ein Computer zur Verfügung steht, sollten wir die ASCII-Zeichen verwenden. Es bieten sich die 59 ASCII-Zeichen von 31 bis 90 an; sie umfassen alle gewünschten Buchstaben, und außerdem ist 59 eine Primzahl, was sich später noch als nützlich erweisen wird. CHR$(31) ist ein Farbcode und wird nicht wirklich gebraucht.
Verschlüsselt wird in erster Linie durch Multiplikation mit 5. Der Ausdruck für die Verschlüsselung des einzelnen Zeichens A$ zu C$ lautet präzise:

```
N = 5 * (ASC(A$)–31)
C$ + CHR$(31 + N – 59*INT(N/59))
```

Entschlüsselt wird durch Multiplikation mit 12 (das Produkt von 5 und 12 ist 60, also gleich 1 ohne Vielfache von 59). Um ein einzelnes Zeichen C$ zu entschlüsseln, gehen wir demnach wie folgt vor:

M = 12 * (ASC(C$)−31)
A$ = CHR$(31 + M − 59*INT(M/59))

Hier ist ein kurzes Programm, das auf dem genannten Schlüssel beruht und eine Nachricht verschlüsselt oder entschlüsselt. Als zusätzliche Besonderheit müssen Sie eine der Zahlen von 2 bis 58 zum Verschlüsseln und Entschlüsseln eingeben. Natürlich muß beim Entschlüsseln dieselbe Zahl wie beim Verschlüsseln gewählt werden. Zum Verschlüsseln multiplizieren wir im wesentlichen mit der gewählten Zahl, sagen wir N. Zum Entschlüsseln multiplizieren wir mit dem Inversen von N modulo 59, also mit einer Zahl M, für die N*M bis auf Vielfache von 59 gleich 1 ist.

```
10 REM 36TAUSCHCODE
20 POKE53280,6:PRINTCHR$(147)"          ***
 TAUSCHCODE ***"
30 PRINT:PRINT"DIESES PROGRAMM VER- UND
ENTSCHLUESSELT NACHRICHTEN"
40 PRINT:PRINT"BITTE CODE SCHLUESSEL EIN
GEBEN"
50 PRINT:INPUT"ZAHL ZWISCHEN 2 UND 58 ";
C
60 IFC<>INT(C)ORC<2ORC>58THEN PRINT:PRIN
T"FALSCHE EINGABE":GOTO50
70 PRINT:PRINT"SOLL VER- ODER ENTSCHLUES
SELT WERDEN ?"
80 PRINT:INPUT"V ODER E ";A$
90 IFA$<>"E"ANDA$<>"V"THEN80
100 D=1:I=C:B$="ENT":IFA$="V"THEND=C:B$=
"VER":GOTO130
110 C=C+I:D=D+1:IFC-59*INT(C/59)<>1THEN1
10
130 PRINTCHR$(147)"NACHRICHT EINGEBEN (B
IS ZU 254 ZEICHEN)":PRINT
140 M$="":N$="":L=0
150 GETG$:IFG$=""THEN150
170 G=ASC(G$)
180 IFG>31ANDG<91THENM$=M$+G$:PRINTG$;:L
=L+1
```

```
190 IFG=20ANDL>0THENPRINTG$;:L=L-1:M$=LE
FT$(M$,L)
200 IFG<>13ANDL<255THEN150
210 PRINT" "
220 REM VER- ENTSCHLUESSELN
230 FORI=1TOL
240 N=ASC(MID$(M$,I,1))-31:N=D*N
250 N$=N$+CHR$(31+N-59*INT(N/59))
260 NEXT
270 PRINTCHR$(147)"DIE "B$"SCHLUESSELTE
NACHRICHT LAUTET":PRINT:PRINTN$
280 PRINT:PRINT:PRINT:PRINT"NEUER START
(J/N) ?"
290 GETG$:IFG$<>"J"ANDG$<>"N"THEN290
300 IFG$="J"THENRUN
310 PRINTCHR$(147)"AUF WIEDERSEHEN":END
```

All unseren Bemühungen zum Trotz ist es (für Experten) leicht, unseren Code zu knacken, d. h. zu entschlüsseln. Das Problem beim Tauschalphabet ist, daß jedes Zeichen prinzipiell durch ein festgelegtes anderes Zeichen dargestellt wird. Manche Buchstaben und Buchstabenpaare kommen jedoch häufiger vor als andere. Beispielsweise macht der Buchstabe E in einem normalen englischen Text ungefähr 13% der Zeichen aus, T etwa 9%, P etwa 2% und Q etwa 0.2%. Mit solchen Informationen kann man ein Tauschalphabet dechiffrieren, obwohl man mit 59 Zeichen (ASCII-Codes 31 bis 90) ungefähr $1.4*10^{80}$ Tauschalphabete bilden kann. Daran zeigt sich, wie sehr die große Fülle von Möglichkeiten täuschen kann.
Im nächsten Abschnitt werden Codes vorgestellt, die ein gegebenes Zeichen nicht immer durch dasselbe Zeichen ersetzen.

MATRIXSCHLÜSSEL

Zum Verschlüsseln von Nachrichten kann man Matrizen verwenden. Dieses Verfahren erläutern wir an einem Beispiel: Angenommen, wir wollen die Nachricht PLEASE COME QUICKLY verschlüsseln. Dazu ordnen wir sie zunächst in zwei Zeilen an:

```
P E S   O E Q I K Y
L A E C M   U C L .
```

Daraus bilden wir dann eine zweizeilige Matrix, indem wir die Buchstaben in ihre ASCII-Codes minus 31 verwandeln:

$$\begin{bmatrix} 49 & 38 & 52 & 1 & 48 & 38 & 50 & 42 & 44 & 58 \\ 45 & 34 & 38 & 36 & 46 & 1 & 54 & 36 & 45 & 15 \end{bmatrix}$$

Nun multiplizieren wir die Matrix von links mit der folgenden Matrix:

$$\begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix}$$

Das Ergebnis ist:

$$\begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix} * \begin{bmatrix} 49 & 38 & 52 & 1 & 48 & 38 & 50 & 42 & 44 & 58 \\ 45 & 34 & 38 & 36 & 46 & 1 & 54 & 36 & 45 & 15 \end{bmatrix}$$

$$= \begin{bmatrix} 53 & 42 & 66 & -34 & 10 & 75 & 46 & 48 & 43 & 101 \\ -110 & -88 & -146 & 103 & -102 & -187 & -88 & -102 & -85 & -245 \end{bmatrix}$$

Dann wandeln wir die Elemente der Matrix in Zahlen zwischen 0 und 58 um, indem wir Vielfache von 59 addieren oder subtrahieren, und zwar nach der Regel:

N = N – 59*INT(N/59)

Es ergibt sich diese Matrix:

$$\begin{bmatrix} 53 & 42 & 7 & 25 & 50 & 16 & 46 & 48 & 43 & 42 \\ 8 & 30 & 31 & 44 & 16 & 49 & 30 & 16 & 33 & 50 \end{bmatrix}$$

Schließlich addieren wir 31 zu allen Zahlen und fragen mit PRINT CHR$(X) nach den zugehörigen Zeichen. Das Ergebnis sieht so aus:

```
T  I  &  8  Q  /  M  O  J  I
'  =  >  K  /  P  =  /  @  Q
```

Die endgültige verschlüsselte Botschaft ist

T'I=&>8KQ//PM=O/J@IQ

Beachten Sie, daß in der ursprünglichen Form der Buchstabe E dreimal vorkommt. Diese drei E sind zu I, > und / verschlüsselt. Der Schlüssel ist also subtiler als ein einfaches Tauschalphabet.
Ein Code taugt nichts, wenn Nachrichten nicht wieder entschlüsselt werden können. Der Trick bei dem gerade beschriebenen Code besteht darin, das Verfahren umzukehren. Auf den ersten Blick sieht das schwierig aus, aber wir können etwas Mathematik zu Hilfe nehmen.
Wir wollen folgende Nachricht entziffern:

I=M?"S?$:8M@>'<M

Zunächst schreiben wir die verschlüsselte Nachricht in zwei Zeilen:

I	M	"	?	:	M	>	<
=	?	S	$	8	@	'	M

Dann bestimmen wir die ASII-Codes minus 31, um eine Matrix zu erzeugen:

$$\begin{bmatrix} 42 & 46 & 3 & 32 & 27 & 46 & 31 & 29 \\ 30 & 32 & 52 & 25 & 5 & 33 & 8 & 46 \end{bmatrix}$$

Diese multiplizieren wir von links mit der Matrix

$$\begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix}$$

Beachten Sie, daß diese Matrix nicht dieselbe ist, die zum Verschlüsseln benutzt wurde. Die Entschlüsselungsmatrix ist vielmehr die Inverse der Verschlüsselungsmatrix, wie die nachstehende Rechnung zeigt.

$$\begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix} * \begin{bmatrix} 2 & -1 \\ -5 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} 3*2 + 1*-5 & 3*-1 + 1*3 \\ 5*2 + 2*-5 & 5*-1 + 2*3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Wir multiplizieren also die Entschlüsselungsmatrix mit der Matrix, in der die Nachricht verschlüsselt ist:

$$\begin{bmatrix} 3 & 1 \\ 5 & 2 \end{bmatrix} * \begin{bmatrix} 42 & 46 & 3 & 32 & 27 & 46 & 31 & 29 \\ 30 & 32 & 52 & 5 & 25 & 33 & 8 & 46 \end{bmatrix}$$

$$= \begin{bmatrix} 156 & 170 & 61 & 101 & 106 & 171 & 101 & 133 \\ 270 & 294 & 119 & 170 & 185 & 296 & 171 & 237 \end{bmatrix}$$

Dann addieren (oder subtrahieren) wir Vielfache von 59, um Zahlen zwischen 0 und 58 zu erhalten:

$$\begin{bmatrix} 38 & 52 & 2 & 42 & 47 & 53 & 42 & 15 \\ 34 & 58 & 1 & 52 & 8 & 1 & 53 & 1 \end{bmatrix}$$

Schließlich addieren wir 31 zu diesen Zahlen und suchen die zugehörigen Zeichen:

E	S	!	I	N	T	I	.
A	Y		S	'		T	

Auf diese Weise erhalten wir die Mitteilung EASY! ISN'T IT.
Das nächste Programm verwendet solche Matrizen, um Nachrichten zu verschlüsseln und/oder zu entschlüsseln.

```
10 REM 37MATRIXCODE
20 POKE53280,6:PRINTCHR$(147)"            *
** MATRIX CODE ***"
30 PRINT:PRINT"DIESES PROGRAMM VER- UND
ENTSCHLUESSELT NACHRICHTEN
40 S=2:FORI=1TOS:FORJ=1TOS:READA(I,J):NE
XT:NEXT
50 PRINT:INPUT"ENT- ODER VERSCHLUESSELN
(E/V) ";A$
70 IFA$<>"E"ANDA$<>"V"THEN50
80 B$="VER"
90 IFA$="E"THENB$="ENT":FORI=1TOS:FORJ=1
TOS:READA(I,J):NEXT:NEXT
100 PRINTCHR$(147)"NACHRICHT EINGEBEN (M
AX 254 ZEICHEN)":PRINT
110 M$="":L=0
```

```
120 GETG$:IFG$=""THEN120
130 G=ASC(G$)
140 IFG>31ANDG<91THENM$=M$+G$:PRINTG$;:L
=L+1
150 IFG=20ANDL>0THENPRINTG$;:L=L-1:M$=LE
FT$(M$,L)
160 IFG<>13ANDL<225THEN120
170 PRINT" ":M=INT(L/S+0.9):DIMB(S,M),C(
S,M)
180 REM NACHRICHT VER/ENTSCHLUESSELN
190 FORI=1TOS:FORJ=1TOM
200 K=S*J+I-S
210 IFK<=LTHENN=ASC(MID$(M$,K,1))-31:B(I
,J)=N-59*INT(N/59)
220 NEXT:NEXT
230 FORI=2TOS:IFB(I,M)<1THENB(I,M)=1
240 NEXT
250 FORI=1TOS:FORJ=1TOM
260 FORK=1TOS:C(I,J)=C(I,J)+A(I,K)*B(K,J
):NEXT
270 C(I,J)=C(I,J)-59*INT(C(I,J)/59)
280 NEXT:NEXT
290 PRINTCHR$(147)"DIE "B$"SCHLUESSELTE
NACHRICHT LAUTET":PRINT
300 FORJ=1TOM:FORI=1TOS
310 PRINTCHR$(31+C(I,J));
320 NEXT:NEXT:PRINT
330 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
340 GETG$:IF G$<>"J"ANDG$<>"N"THEN340
350 IF G$="J"THENRUN
360 PRINTCHR$(147)"AUF WIEDERSEHEN":END
400 DATA 2,-1,-5,3
410 DATA 3,1,5,2
```

Für diesen Code haben wir eine 2-mal-2-Matrix zum Verschlüsseln und die Inverse der Matrix zum Entschlüsseln gebraucht. Grundsätzlich kann man eine beliebige Matrix und ihre Inverse verwenden, sofern beide Matrizen nur ganze Zahlen als Elemente besitzen. Hier ist eine weitere Matrix, die zum Verschlüsseln verwendet werden könnte:

$$\begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix}$$

Die zugehörige Entschlüsselungsmatrix sieht so aus:

$$\begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}$$

Dieses Verschlüsselungsverfahren könnte man verfeinern, indem man 3-mal-3-Matrizen nimmt. Die Botschaft müßte dann in drei Zeilen geschrieben werden und das oben beschriebene Verfahren entsprechend übertragen werden. Hier sind zwei Matrizen, die man zum Ver- und Entschlüsseln verwenden kann:

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 1 & 3 \\ 4 & 2 & 5 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & -1 \\ -2 & -9 & 5 \\ 0 & 2 & -1 \end{bmatrix}$$

Das untenstehende Programm benutzt diese Matrizen zum Ver- und Entschlüsseln. Wenn Sie wollen, können Sie Ihre eigenen Matrizen einsetzen. Prüfen Sie aber nach, ob sie invers zueinander sind und nur ganzzahlige Elemente enthalten.

```
10 REM 38MATRIXCODE 3
20 POKE53280,6:PRINTCHR$(147)"             *
** MATRIX CODE ***"
30 PRINT:PRINT"DIESES PROGRAMM VER- UND
ENTSCHLUESSELT NACHRICHTEN
40 S=3:FORI=1TOS:FORJ=1TOS:READA(I,J):NE
XT:NEXT
50 PRINT:INPUT"ENT- ODER VERSCHLUESSELN
(E/V)  ";A$
70 IFA$<>"E"ANDA$<>"V"THEN50
80 B$="VER"
90 IFA$="E"THENB$="ENT":FORI=1TOS:FORJ=1
TOS:READA(I,J):NEXT:NEXT
100 PRINTCHR$(147)"NACHRICHT EINGEBEN (M
AX 254 ZEICHEN)":PRINT
110 M$="":L=0
120 GETG$:IFG$=""THEN120
130 G=ASC(G$)
140 IFG>31ANDG<91THENM$=M$+G$:PRINTG$;:L
=L+1
```

```
150 IFG=20ANDL>0THENPRINTG$;:L=L-1:M$=LE
FT$(M$,L)
160 IFG<>13ANDL<225THEN120
170 PRINT" ":M=INT(L/S+0.9):DIMB(S,M),C(
S,M)
180 REM NACHRICHT VER/ENTSCHLUESSELN
190 FORI=1TOS:FORJ=1TOM
200 K=S*J+I-S
210 IFK<=LTHENN=ASC(MID$(M$,K,1))-31:B(I
,J)=N-59*INT(N/59)
220 NEXT:NEXT
230 FORI=2TOS:IFB(I,M)<1THENB(I,M)=1
240 NEXT
250 FORI=1TOS:FORJ=1TOM
260 FORK=1TOS:C(I,J)=C(I,J)+A(I,K)*B(K,J
):NEXT
270 C(I,J)=C(I,J)-59*INT(C(I,J)/59)
280 NEXT:NEXT
290 PRINTCHR$(147)"DIE "B$"SCHLUESSELTE
NACHRICHT LAUTET":PRINT
300 FORJ=1TOM:FORI=1TOS
310 PRINTCHR$(31+C(I,J));
320 NEXT:NEXT:PRINT
330 PRINT:PRINT:PRINT"NEUER START (J/N)
?"
340 GETG$:IF G$<>"J"ANDG$<>"N"THEN340
350 IF G$="J"THENRUN
360 PRINTCHR$(147)"AUF WIEDERSEHEN":END
400 DATA 1,0,-1,2,1,3,4,2,5
410 DATA 1,2,-1,-2,-9,5,0,2,-1
```

OFFENE SCHLÜSSEL

Die im vorigen Abschnitt beschriebenen Codes haben einen Nachteil: Wenn man einmal weiß, wie eine Nachricht zu verschlüsseln ist, weiß man auch, wie sie zu entschlüsseln ist. Das gilt nicht für offene Schlüssel. Sie bestehen aus zwei Teilen: dem Verschlüsselungsverfahren, das veröffentlicht wird und jedermann das Verschlüsseln von Nachrichten ermöglicht, und dem Entschlüsselungsverfahren, das geheimgehalten wird und nur dem Hersteller des Codes das Entschlüsseln von Nachrichten erlaubt.

Im folgenden beschreiben wir einen solchen offenen Schlüssel. Wählen Sie zunächst zwei sehr große Primzahlen P und Q. Jede sollte ungefähr 50 Dezimalstellen haben; deshalb läßt sich die Suche danach kaum mit dem Commodore 64 realisieren. Sei N das Produkt von P und Q. Wählen Sie nun eine ganze Zahl A, die kleiner als N ist und keinen gemeinsamen Faktor mit (P−1)*(Q−1) hat. Die Zahlen N und A dürfen dann jedem zugänglich sein.
Wie wird eine Nachricht mit den Zahlen N und A verschlüsselt? Man geht dabei so vor:

1. Übersetze die Nachricht in Zahlen (Leerstelle = 01, A = 34, etc.); sie ist dann eine große Zahl.
2. Zerlege die in eine Zahl übersetzte Nachricht in Blöcke übersichtlicher Länge.
3. Verschlüssele jeden Block B wie folgt:

C = B↑A−N*INT((B↑A)/N)

Die Zahl B wird zur Potenz A erhoben, und Vielfache von N werden weggelassen, um eine Zahl zwischen 0 und N zu erhalten.
Wie wird die entstandene Nachricht wieder entschlüsselt? Da der größte gemeinsame Teiler von A und (P−1)*(Q−1) die Zahl 1 ist, gibt es zwei Zahlen X und Y, so daß A*X + (P−1)*(Q−1)*Y = 1 ist. Mit Hilfe von X kann man die Nachricht wie folgt entschlüsseln:

1. Zerlege die verschlüsselte Nachricht in Blöcke.
2. Berechne für jeden Block C:

C↑X−N*INT((C↑X)/N).

3. Füge die entstandenen Blöcke wieder zusammen und übersetze die Zahlen wieder in Zeichen zurück.
   Das Verfahren funktioniert, weil

(B↑A)↑X = B↑(A*X)
= B↑(1−(P−1)*(Q−1)*Y)
= B + Vielfache von N.

Die letzte Aussage folgt aus einem Satz, der von dem Mathematiker Fermat bewiesen worden ist.
Warum ist dieser Code schwer zu knacken? Beachten Sie, daß wir zum Entschlüsseln der Nachricht den Wert von X kennen müssen. Dieser Wert kann aus dem Wert von (P−1)*(Q−1) berechnet werden. Um (P−1)*(Q−1) zu kennen, müssen wir P und Q kennen. Bekannt sind nur A und N. Wenn man N kennt, ist es theoretisch möglich, N zu faktorisieren und P und Q zu bestimmen. Die Zahl N hat aber

ungefähr 100 Dezimalstellen, und es sind ungeheure Rechenzeiten (mehrere Millionen Jahre) erforderlich, um solche Zahlen zu faktorisieren. Aus diesem Grund ist der Code sicher.
Um einen sehr sicheren Code herzustellen, brauchen Sie jetzt nur noch ein Programm für Ihren Commodore 64 zu schreiben, das mit sehr großen Zahlen fehlerfrei umgehen kann. Siehe dazu das Kapitel 'Vermischtes'.

KAPITEL 13

ZUFALL!

KOPF ODER ZAHL

Wirft man eine Münze, so sind Kopf oder Zahl als Ergebnis gleich wahrscheinlich, zumindest, wenn die Münze nicht verfälscht ist. Ihr Commodore 64 kann das Werfen von Münzen mit der eingebauten Funktion RND simulieren.
Für jede positive Zahl X liefert RND(X) eine (Pseudo-)Zufallszahl zwischen 0 und 1 (die 1 nicht eingeschlossen). Wenn Sie Ihren Commodore 64 einschalten und folgendes tippen:

```
FOR I=1 TO 5:PRINT RND(1):NEXT I
```

erhalten Sie diese Zahlenfolge:

```
.185564016
.0468986348
.827743801
.554749226
.897233831
```

Falls Sie Ihren 64er aus- und wieder einschalten (nicht empfehlenswert) und die Anweisung wiederholen:

```
FOR I=1 TO 5:PRINT RND(1):NEXT 1
```

so erhalten Sie dieselbe Zahlenfolge. Um das zu vermeiden, beginnt man gewöhnlich ein Programm, in dem Zufallszahlen vorkommen, mit einer Zeile wie

```
Y = RND(−TI)
```

die dazu führt, daß eine neue Folge von Zufallszahlen ausgelöst wird. Allgemein gilt: Für negatives X beginnt RND(X) eine neue Folge von Zufallszahlen. Ist X = 0, so ergibt sich dieselbe Zahl wie vorher.
Das folgende kurze Programm simuliert das Werfen einer unverfälschten Münze. Es wird eine Tabelle ausgedruckt, die zeigt, ob Kopf (K für Kopf) oder Zahl (Z für Zahl) erscheint. Nachdem 100 solcher Buchstaben ausgedruckt sind, wird angezeigt, wie häufig Kopf und Zahl erschienen sind.

```
10 REM 39KOPF ODER ZAHL
20 POKE53280,6:PRINTCHR$(147)"         ***
 KOPF ODER ZAHL ***":K=0
30 PRINT:PRINT"DIESES PROGRAMM SIMULIER
T DAS WERFEN   EINER MUENZE ";
40 PRINT:PRINT:INPUT"ANZAHL DER WUERFE "
;N
50 IFN<1ORN<>INT(N)THEN40
60 Y=RND(-TI)
70 K=K+1:J=0:PRINTCHR$(147)STR$(K)". DUR
CHLAUF MIT "N"WUERFEN"CHR$(158):PRINT
80 FORI=1TON
90 A$="K":IF RND(1)>=.5THENA$="Z":J=J+1
100 PRINT" "A$;
110 NEXT
120 PRINT:PRINT:PRINTCHR$(154)N "WUERFE"

130 PRINT:PRINT" DAVON "N-J" KOEPFE,  "J
" ZAHLEN"
140 PRINT:PRINT:PRINT" NEUER DURCHLAUF (
J/N) ?"
150 GETG$:IFG$<>"N"ANDG$<>"J"THEN150
160 IFG$="J"THEN200
170 PRINTCHR$(147)"AUF WIEDERSEHEN":END
200 PRINTCHR$(147)"ANZAHL  DER WUERFE VE
RAENDERN (J/N) ?"
210 GETG$:IFG$<>"J"ANDG$<>"N"THEN210
220 IFG$="J"THENRUN
230 GOTO60
```

Geben Sie statt Zeile 60 diese Zeile ein:

60 Y = RND(−1)

Nun müßten Sie feststellen, daß Sie bei jedem Programmdurchlauf dieselbe Folge von Kopf und Zahl erhalten. Deshalb verwenden wir

60 Y = RND(−TI)

um die Folge von Zufallszahlen zufällig zu machen.

WÜRFELN

Wird ein unverfälschter sechsseitiger Würfel geworfen, so erscheint eine der sechs Zahlen 1, 2, 3, 4, 5 und 6, keine häufiger als eine andere. Wiederum kann der Commodore 64 benutzt werden, um das Würfeln zu simulieren. Das nächste Programm listet das Ergebnis von 120 Würfen auf. Am Schluß wird ausgedruckt, wie häufig jede Augenzahl vorgekommen ist.

```
10 REM 40WUERFEL
20 POKE53280,6:PRINTCHR$(147)"
*** WUERFEL ***":K=0
30 PRINT:PRINT"DIESES PROGRAMM SIMULIER
T DAS WERFEN   EINES WUERFELS"
40 PRINT:PRINT:INPUT"ANZAHL DER WUERFE "
;N
50 IFN<1ORN<>INT(N)THEN40
60 Y=RND(-TI)
70 K=K+1:FORI=1TO6:A(I)=0:NEXT
80 PRINTCHR$(147)STR$(K)". DURCHLAUF MIT
 "N"WUERFEN"CHR$(158):PRINT
90 FORI=1TON
100 L=INT(RND(1)*6)+1:PRINTL"  ";:A(L)=A(
L)+1
110 NEXT
120 PRINT:PRINT:PRINTCHR$(154)" BEI "N"W
UERFEN FIEL"
130 PRINT:FORI=1TO6:PRINTA(I)"MAL DIE"I,
:NEXT
140 PRINT:PRINT:PRINT" NEUER DURCHLAUF (
J/N) ?"
150 GETG$:IFG$<>"N"ANDG$<>"J"THEN150
160 IFG$="J"THEN200
170 PRINTCHR$(147)"AUF WIEDERSEHEN":END
200 PRINT:PRINT" ANZAHL DER WUERFE AENDE
RN (J/N) ?"
210 GETG$:IFG$<>"N"ANDG$<>"J"THEN210
220 IFG$="J"THENRUN
230 IFG$="N"THEN60
```

Als nächstes folgt eine andere Version des Programms. Es liefert auch eine graphische Darstellung der Würfelseiten. Die Gestaltung der Seiten ist im Feld D$(I,J) enthalten.

```
10 REM 41WUERFEL MIT
20 POKE53280,6:PRINTCHR$(147)"          ***
FALLENDER WUERFEL ***"
30 PRINT:PRINT"DIESES PROGRAMM SIMULIERT
 DAS FALLEN    EINES WUERFELS"
40 FORI=0TO6:FORJ=0TO6:READA:A$(I)=A$(I)
+CHR$(A):NEXT:NEXT
50 FORI=1TO6:FORJ=0TO6:READA:D$(I,J)=A$(
A):NEXT:NEXT
60 PRINT:PRINT:PRINT:PRINT"BITTE LEER-TA
STE DRUECKEN"
70 GETG$:IFG$<>" "THEN70
80 Y=RND(-TI)
90 K=K+1:PRINTCHR$(147)K". WURF"CHR$(158
):PRINT
100 L=INT(RND(1)*6)+1:A(L)=A(L)+1
110 FORJ=0TO7:PRINTTAB(16) D$(L,J):NEXT
115 PRINT:PRINT:PRINT:PRINTCHR$(154)"IN"
K" WUERFEN FIEL"
120 PRINT:FORI=1TO2:PRINT"DIE"I;A(I)"MAL
   ";:NEXT
125 PRINT:FORI=3TO4:PRINT"DIE"I;A(I)"MAL
   ";:NEXT
130 PRINT:FORI=5TO6:PRINT"DIE"I;A(I)"MAL
   ";:NEXT
140 PRINT:PRINT:PRINT:PRINT"NEUER WURF (
J/N) ? "
150 GETG$:IFG$<>"J"ANDG$<>"N"THEN150
160 IFG$="J"THEN80
170 PRINTCHR$(147)"AUF WIEDERSEHEN":END
200 REM DATEN FUER'S AUSSEHEN
210 DATA 213,192,192,192,192,192,201
220 DATA 221,32,32,32,32,32,221
230 DATA 221,113,32,32,32,32,221
240 DATA 221,32,32,113,32,32,221
250 DATA 221,32,32,32,32,113,221
260 DATA 221,113,32,32,32,113,221
```

```
270 DATA 202,192,192,192,192,192,203
300 REM DATEN FUER WUERFEL SEITE
310 DATA 0,1,1,3,1,1,6
320 DATA 0,4,1,1,1,2,6
330 DATA 0,4,1,3,1,2,6
340 DATA 0,5,1,1,1,5,6
350 DATA 0,5,1,3,1,5,6
360 DATA 0,5,1,5,1,5,6
```

Gleichzeitiges Werfen zweier Würfel kann ebenso leicht simuliert werden. Die mögliche Augenzahl bei jedem Wurf ist eine Zahl von 2 bis 12. Wie Sie sicher wissen, kommen manche Augenzahlen häufiger als andere vor. Das sollte auch mit jedem der beiden nächsten Programme deutlich werden. Werden zwei Würfel mehrmals geworfen, so ergibt sich die erwartete Häufigkeit (oder die Wahrscheinlichkeit) jeder Augenzahl aus dieser Tabelle:

Augenzahl	Wahrscheinlichkeit
2	1/36
3	2/36
4	3/36
5	4/36
6	5/36
7	6/36
8	5/36
9	4/36
10	3/36
11	2/36
12	1/36

```
10 REM 42 2WUERFEL
20 POKE53280,6:PRINTCHR$(147)"          ***
 ZWEI WUERFEL ***"
30 PRINT:PRINT"DIESES PROGRAMM SIMULIER
T DAS WERFEN    VON ZWEI WUERFELN":K=0
40 PRINT:PRINT:INPUT"WIEVIELE WUERFE ";N

50 IFN<1ORN<>INT(N)THEN40
60 Y=RND(-TI)
70 K=K+1:PRINTCHR$(147)K".DURCHLAUF MIT
"N"WUERFEN":PRINT
80 FORI=1TON
```

```
90 L=INT(RND(1)*6)+1:M=INT(RND(1)*6)+1:A
(L+M-2)=A(L+M-2)+1
100 PRINTCHR$(150)"*"CHR$(158)L;MCHR$(15
0)"*";
110 NEXT
115 PRINT:PRINTCHR$(154)
120 PRINT" IN"N"WUERFEN FIEL":PRINT
130 FORI=0TO10
140 PRINTA(I)"X"(I+2),
150 NEXT
160 PRINT:PRINT:PRINT" NEUER WURF (J/N)
?"
170 GETG$:IFG$<>"N"ANDG$<>"J"THEN170
180 IFG$="N"THENPRINTCHR$(147)"AUF WIEDE
RSEHEN":END
190 PRINT:PRINT" ANZAHL DER WUERFE VERAE
NDERN (J/N) ?"
200 GETG$:IFG$<>"N"ANDG$<>"J"THEN200
210 IFG$="N"THEN60
220 IFG$="J"THENRUN
```

```
10 REM 43 2WUERFEL MIT
20 POKE53280,6:PRINTCHR$(147)"███*** ZWE
I FALLENDE WUERFEL ***  "
30 PRINT:PRINT"DIESES PROGRAMM SIMULIER
T DAS FALLEN   VON ZWEI WUERFELN"
40 FORI=0TO6:FORJ=0TO6:READA:A$(I)=A$(I)
+CHR$(A):NEXT:NEXT
50 FORI=1TO6:FORJ=0TO6:READA:D$(I,J)=A$(
A):NEXT:NEXT
60 PRINT:PRINT:PRINT:PRINT"BITTE LEER-TA
STE DRUECKEN"
70 GETG$:IFG$<>" "THEN70
80 Y=RND(-TI)
90 K=K+1:PRINTCHR$(147)K". WURF":PRINT:P
RINT:PRINTCHR$(158)
100 L=INT(RND(1)*6)+1:M=INT(RND(1)*6)+1:
A(L+M-2)=A(L+M-2)+1
110 FORJ=0TO7:PRINTTAB(10)D$(L,J)SPC(6)D
$(M,J):NEXT
```

```
120 PRINT:PRINT:PRINTCHR$( 154)"BEI"K"WUE
RFEN FIEL":PRINT
130 FORI=0TO10
140 PRINTA( I)"*"( I+2),
150 NEXT
160 PRINT:PRINT:PRINT"NEUER WURF (J/N) ?
"
170 GETG$:IFG$<>"N"ANDG$<>"J"THEN170
180 IFG$="J"THEN80
190 PRINTCHR$( 147)"AUF WIEDERSEHEN":END
200 REM DATEN FUER'S AUSSEHEN
210 DATA 213,192,192,192,192,192,201
220 DATA 221,32,32,32,32,32,221
230 DATA 221,113,32,32,32,32,221
240 DATA 221,32,32,113,32,32,221
250 DATA 221,32,32,32,32,113,221
260 DATA 221,113,32,32,32,113,221
270 DATA 202,192,192,192,192,192,203
300 REM DATEN FUER WUERFEL SEITE
310 DATA 0,1,1,3,1,1,6
320 DATA 0,4,1,1,1,2,6
330 DATA 0,4,1,3,1,2,6
340 DATA 0,5,1,1,1,5,6
350 DATA 0,5,1,3,1,5,6
360 DATA 0,5,1,5,1,5,6
```

KARTENSPIELE

Ein normales Kartenspiel hat 52 Karten. In einem gut gemischten Spiel kann jede Karte mit gleicher Wahrscheinlichkeit oben liegen. Das nächste Programm zeigt, wie der Commodore 64 das Ziehen einer Karte aus einem gut gemischten Kartenspiel simulieren kann. Bei jedem neuen Ziehen wird angenommen, daß die vorher gezogene Karte zurückgelegt und das Spiel wieder gut gemischt worden ist.

```
10 REM 44 KARTEN
20 POKE53280,15:POKE53281,15:PRINTCHR$( 1
47)CHR$( 31)"              *** KARTEN ***"
30 PRINT:PRINT"DIESES PROGRAMM SIMULIER
T DAS ZIEHEN   EINER KARTE AUS EINEM ";
35 PRINT"GUT GEMISCHTEM     HAUFEN"
```

```
50 DIM A$(12):FORI=0TO12:READA$(I):NEXT
60 FORI=0TO3:READA:B$(I)=CHR$(A):NEXT
70 FORI=0TO3:READA:C$(I)=CHR$(A):NEXT
80 PRINT:PRINT:PRINT:PRINT:PRINT"BITTE L
EER-TASTE DRUECKEN"
90 GETG$:IFG$<>" "THEN90
100 Y=RND(-TI)
110 K=K+1:PRINTCHR$(147)K"ZUG":PRINT:PRI
NT
120 L=INT(RND(1)*13):M=INT(RND(1)*4)
130 PRINTTAB(16+(L=9))C$(M) B$(M)"    "A$
(L)
140 PRINT:PRINT:PRINT:PRINT:PRINTCHR$(31
)" NEUES MISCHEN (J/N) ?"
150 GETG$:IFG$<>"J"ANDG$<>"N"THEN150
160 IFG$="J"THEN100
170 POKE53281,6:POKE53280,14:PRINTCHR$(1
47)CHR$(154)"AUF WIEDERSEHEN":END
200 REM DATEN
210 DATA A,2,3,4,5,6,7,8,9,10,B,D,K
220 DATA 120,122,115,97
230 DATA 144,28,28,144
```

Im Programm ‘Karten’ wählt der Computer erst eine Zahl zwischen 0 und 12, die (um 1 erhöht) den Wert der Karte bestimmt. Dann wählt er eine Zahl von 0 bis 3, die für eine der vier möglichen Farben der Karte steht. Statt dessen wäre es auch möglich, eine Zahl zwischen 0 und 51 zu wählen:

K = INT(RND(1)*52)

und Wert und Farbe der Karte aus dieser Zahl zu bestimmen. Das leistet diese Zeile:

L = INT(K/4):M = K – 4*L

Jetzt bestimmt die Zahl L den Wert der Karte, während M die Farbe angibt.
Das Programm ‘Karten’ zieht eine Karte aus einem gut gemischten Spiel. Vor jedem neuen Ziehen wird die Karte zurückgelegt und das Spiel wieder gemischt. Was wäre, wenn wir nun das Spiel einmal mischen und dann die Karten der Reihenfolge nach von oben nach unten auflisten wollten? Dazu wird eine andere Routine benötigt. Im wesentlichen numerieren wir die Karten von 0 bis 51 und ordnen diese

52 Zahlen dann zufällig um. Dieses Umordnen geschieht systematisch. Zunächst wird die erste Zahl zufällig mit einer der 51 anderen vertauscht. Dann wird die zweite Zahl mit einer der 50 übrigen Zahlen vertauscht, und so weiter. Das Programm 'Kartenmischer' veranschaulicht die Methode.

```
10 REM 45KARTEN MISCHER
20 POKE53281,15:POKE53280,15:PRINTCHR$(1
47)CHR$(31)"        *** KARTEN MISCHER ***
"
30 PRINT:PRINT"DIESES PROGRAMM ILLUSTRIE
RT DAS MISCHEN VON KARTEN"
50 DIMA$(12):FORI=0TO12:READA$(I):NEXT
60 FORI=0TO3:READA:B$(I)=CHR$(A):NEXT
70 FORI=0TO3:READA:C$(I)=CHR$(A):NEXT
80 DIMD%(51):FORI=0TO51:D%(I)=I:NEXT
90 PRINT:PRINT:PRINT:PRINT"BITTE LEER-TA
STE DRUECKEN"
100 GETG$:IFG$<>" "THEN100
110 Y=RND(-TI)
120 K=K+1:PRINTCHR$(147):PRINTK". MISCHE
N":PRINT:PRINT
130 REM MISCHEN
140 FORI=0TO50
150 L=INT(RND(1)*(52-I))+I
170 T=D%(I):D%(I)=D%(L):D%(L)=T
180 NEXT
190 FORI=1TO52
200 L=INT(D%(I-1)/4):M=D%(I-1)-4*L
210 PRINTTAB(O)C$(M)B$(M)"    "A$(L);:O=O
+10:IFINT(I/4)=(I/4)THENPRINT:O=0
220 NEXT
230 PRINT:PRINT:PRINTCHR$(31)" NEUES MIS
CHEN (J/N) ?"
240 GETG$:IFG$<>"J"ANDG$<>"N"THEN240
250 IFG$="J"THEN120
260 POKE53281,6:POKE53280,14:PRINTCHR$(1
47)CHR$(154)"AUF WIEDERSEHEN":END
270 REM DATEN
300 DATA A,2,3,4,5,6,7,8,9,10,B,D,K
310 DATA 120,122,115,97
320 DATA 144,28,28,144
```

UNTERSCHIEDLICH WAHRSCHEINLICHE EREIGNISSE

In den meisten bisher behandelten Beispielen ist jedes mögliche Ereignis ebenso wahrscheinlich wie ein anderes. Beim nächsten Beispiel ist das nicht der Fall.
Ein Topf enthält 100 farbige Knöpfe, und zwar 6 rote, 54 blaue und 40 grüne. Nachstehende Zeilen simulieren die Auswahl eines Knopfes aus dem Topf:

```
X = RND(1)
R$ = "ROT"
IF X> = 0.06 THEN R$ = "BLAU"
IR X> = 0.60 THEN R$ = "GRUEN"
PRINT R$
```

Das Programm 'Knöpfe' simuliert diese Entnahme von Knöpfen aus dem Topf. 100 mal wird ein Knopf gewählt, und nach jeder Ziehung wird er wieder zurückgelegt.

```
10 REM 46KNOEPFE
20 POKE53281,15:POKE53280,15:PRINTCHR$(1
47)CHR$(144)"              *** KNOEPFE ***"
30 PRINT:PRINT"IN DIESEM PROGRAMM WIRD E
IN KNOPF AUS   EINEM SACK MIT 6 ROTEN";
40 PRINT", 40 GRUENEN UND  54 BLAUEN KNO
EPFEN GEZOGEN"
60 PRINT:PRINT:PRINT:PRINT"BITTE LEER-TA
STE DRUECKEN"
70 GETG$:IFG$<>" "THEN70
80 Y=RND(-TI)
90 K=K+1:PRINTCHR$(147)K". DURCHLAUF"
100 X=RND(1):R$=CHR$(28)+CHR$(209)+"  RO
T"
110 IFX>=0.06THENR$=CHR$(31)+CHR$(209)+"
 BLAU"
120 IFX>=0.60THENR$=CHR$(30)+CHR$(209)+"
 GRUEN"
130 PRINT:PRINT:PRINTTAB(16)R$:PRINT:PRI
NT:PRINTCHR$(144)" BEI"K"ZIEHUNGEN";
135 PRINT" WURDE":PRINT
140 S=ASC(R$)-28:A(S)=A(S)+1
150 PRINTCHR$(28)A(0)"* ROT",CHR$(31)TAB
(12)A(3)"* BLAU";
155 PRINTTAB(26)CHR$(30)A(2)"* GRUEN"
```

```
160 PRINT:PRINTCHR$(144)" GEZOGEN"
170 PRINT:PRINT:PRINT" NEUE ZIEHUNG (J/N
) ?"
180 GETG$:IFG$<>"N"ANDG$<>"J"THEN180
190 IFG$="J"THEN90
200 POKE53280,14:POKE53281,6:PRINTCHR$(1
47)CHR$(154)"AUF WIEDERSEHEN":END
```

KAPITEL 14

STATISTIK

Was tut man, wenn man mit einer großen Menge numerischer Daten konfrontiert ist? Hier ist etwa eine Reihe von Zahlen, die z. B. die Ergebnisse eines Experiments oder einer Prüfung darstellen könnten:

23 67 89 45 40 10 5
19 99 40 23 9 11 21
34 34 56 41 42 27 80

Man kann einiges über diese Zahlen erfahren, wenn man sich deskriptive Maßzahlen wie das Mittel, die Varianz und die Standardabweichung ansieht. Aber zunächst muß man die Daten in den Computer eingeben.

Im Commodore 64 kann man die Daten mit Hilfe eines Feldes X(I) abspeichern, wobei X(0) = 23, X(1) = 67, etc. zugewiesen wird. Die Eingabe kann im Dialog mit einem einfachen Programm vor sich gehen. Das veranschaulichen die Programme 'Daten-Eingabe 1' und 'Daten-Eingabe 2'.

```
200 REM DATEN-EINGABE 1
210 POKE53280,6:PRINTCHR$(147)"          *
** DATEN-EINGABE"
220 PRINT:PRINT"MIT DIESEM PROGRAMM KOEN
NEN DATEN IM     COMPUTER GESPEICHERT";
225 PRINT" WERDEN"
230 PRINT:PRINT"WIEVIELE DATEN SOLLEN EI
NGEGEBEN WERDEN?"
240 PRINT:INPUT"WIEVIEL   ";N
250 IFN<2ORN<>INT(N)THENPRINT:PRINT"UNMO
EGLICH":GOTO240
260 N=N-1:DIMX(N)
270 FORI=1TON
280 PRINTCHR$(147)"BITTE JETZT DIE "N+1"
 DATEN EINGEBEN":PRINT
290 PRINT"BITTE "I+1". WERT EINGEBEN ";:
INPUTX(I)
300 NEXTI
310 PRINT:PRINT"WEITER MIT LEERTASTE "
320 GETG$:IFG$<>" "THEN320
```

'Data Entry I' fragt zu Beginn nach der Anzahl der einzugebenden Daten. Das Feld X(I) wird dann entsprechend dimensioniert. 'Data Entry II' nimmt hingegen an, daß nicht mehr als 100 Zahlen eingegeben werden. Sie geben die Daten auf Anforderung ein; wenn Sie fertig sind, geben Sie −99999 ein. Falls nötig, kann die Zahl M in Zeile 240 von 100 in eine beliebige andere Zahl geändert werden.
Dieses Kapitel enthält mehrere kurze Routinen, die an die 'Data Entry'-Programme angehängt werden können. Insgesamt ergibt sich damit ein nützliches Programm zur Auswertung Ihrer Daten.

```
200 REM DATEN-EINGABE 2
210 POKE53280,6:PRINTCHR$(147)"        *
** DATEN-EINGABE"
220 PRINT:PRINT"MIT DIESEM PROGRAMM KOEN
NEN DATEN IM    COMPUTER GESPEICHERT";
225 PRINT" WERDEN"
230 PRINT:PRINT"WEITER MIT LEERTASTE "
240 GETG$:IFG$<>" "THEN240
250 M=100:DIMX(M)
260 FORI=0TOM
270 PRINTCHR$(147)"BITTE DATEN EINGEBEN"

280 PRINT:PRINT"EINGABE MUSS MIT -99999
BEENDET WERDEN"
290 PRINT:PRINT"BITTE "I+1". WERT EINGEB
EN ";:INPUTX(I)
300 IFX(I)=-99999ANDI>1THENN=1:I=M
305 NEXT
310 PRINT:PRINT:PRINT:PRINT"WEITER MIT L
EERTASTE "
320 GETG$:IFG$<>" "THEN320
```

MITTELWERT

Der *Mittelwert* oder Durchschnitt ist eine wichtige statistische Maßzahl. Man erhält ihn, indem man alle Zahlen addiert und die Summe durch die Anzahl der Zahlen teilt.

$$\text{Mittelwert} = \frac{\text{Summe der Daten}}{\text{Anzahl der Daten}}$$

Sind die Zahlen im Feld X(I) für I = 0 bis N − 1 gespeichert, so kann der Mittelwert mit Hilfe der folgenden Programmzeilen berechnet werden:

```
X=0
FOR I=0 TO N−1 : X=X+X(I) : NEXT
XM=X/N
```

Das unten angeführte kurze Programm berechnet den Mittelwert und druckt ihn aus. Es kann in die 'Daten-Eingabe'- Programme eingebaut werden.

```
400 REM MITTELEWERTE
410 POKE53280,6:PRINTCHR$(147)"     *** D
ATEN-ANALYSE SEITE I ***":PRINT
415 PRINT:PRINT"DATEN-ANALYSE VON"N+1"WE
RTEN"
420 X=0:FORI=0TON:X=X+X(I):NEXT:XM=X/(N+
1)
430 PRINT:PRINT:PRINT"MITTELWERT
 = "XM
```

MAXIMUM, MINIMUM, SPANNWEITE

Es ist oft sinnvoll, den größten und kleinsten Wert der Daten zu kennen. Sie können Ihren Commodore 64 danach suchen lassen. Ein solches Suchprogramm ist unten abgedruckt. Es berechnet zusätzlich die *Streubreite* oder *Spannweite* der Daten. Das ist einfach die Differenz zwischen dem größten und kleinsten Wert der Daten.

```
500 REM MAXIMUM,MINIMUM UND SPANNWEITE
510 MAX=-1E+37:MIN=1E+37
520 FORI=0TON
530 IF X(I)>MAX THEN MAX=X(I)
540 IF X(I)<MIN THEN MIN=X(I)
550 NEXT
560 PRINT:PRINT:PRINT"MINIMUM
 = "MIN
570 PRINT:PRINT"MAXIMUM              = "MA
X
580 PRINT:PRINT"SPANNWEITE           = "MA
X-MIN
```

Vielleicht möchten Sie Ihre Daten auch in ansteigender (oder absteigender) Reihenfolge ordnen. Dafür gibt es verschiedene Verfahren (wie Bubble-Sort, Quick-Sort, Shell-Sort). Wir gehen an dieser Stelle jedoch nicht ins Detail.

STANDARDABWEICHUNG UND VARIANZ

Der Mittelwert ist eine einfache, nützliche und aussagekräftige Maßzahl. Aber er sagt uns nicht alles, was wir wissen müssen. Sehen Sie sich etwa diese beiden Datensätze an:

DATEN für X(I) 20, 21, 20, 19
DATEN für Y(I) 38, 26, 14, 2

Die Mittelwerte XM und YM sind beide gleich 20. Aber die Daten für Y(I) streuen viel stärker als die Daten für X(I). Diese Streuung kann man mit der Standardabweichung der Daten messen.
Die *Standardabweichung* eines Datensatzes ist durch die Formel gegeben:

Standardabweichung = Quadratwurzel (Summe der Quadrate der Differenzen zwischen Daten und Mittelwert / (Datenzahl −1)).

Die *Varianz* ist das Quadrat der Standardabweichung. Die Standardabweichung der im Feld X(I) gespeicherten Daten wird nach diesem Verfahren berechnet:

1. Berechne den Mittelwert XM.
2. Bestimme die Abweichungen vom Mittelwert, das heißt die Werte X(I) −XM.
3. Quadriere jede Abweichung, das heißt berechne $(X(I)-XM)^2$.
4. Addiere die Quadrate der Abweichungen.
5. Teile durch die Datenanzahl minus 1. Das ergibt die Varianz XV der Daten.
6. Ziehe die Quadratwurzel. Das ergibt die Standardabweichung XD der Daten.

Die Standardabweichung gibt eine Vorstellung davon, wie weit die Daten um den Mittelwert streuen. Sehen wir uns noch einmal die Beispiele an:

DATEN für X(I) 20, 21, 20, 19
DATEN für Y(I) 38, 26, 14, 2

Die Standardabweichungen ergeben sich aus diesen Berechnungen:

$$
\begin{aligned}
XD &= SQR((0*0 + 1*1 + 0*0 + (-1)*(-1))/3) \\
&= SQR(2/3) \\
&\ 0.816496581 \\
YD &= SQR((18*18 + 6*6 + (-6)*(-6) + (-18)*(-18))/3) \\
&= SQR(240) \\
&= 15.4919334
\end{aligned}
$$

Die Ergebnisse spiegeln deutlich den Unterschied in der Streuung der Daten wider. Das nächste kurze Programm berechnet die Standardabweichung der im Feld X(I) abgelegten Daten.

```
600 REM STANDARDABWEICHUNG
610 X=0:FORI=0TON:Y=X(I)-XM:X=X+Y*Y:NEXT

620 XD=SQR(X/N)
630 PRINT:PRINT:PRINT"STANDARDABWEICHUNG
 = "XD
```

KONFIDENZINTERVALLE

Die Standardabweichung ist nützlich, weil sie anzeigt, wie stark die Daten um den Mittelwert streuen. In vielen Massenherstellungsverfahren variieren die erzeugten Produkte leicht in Größe, Qualität, Länge etc. Man nennt die zu messenden Objekte die Population. Die Abweichungen einer Population sind häufig normalverteilt.
Statistiker haben festgestellt, daß die Normalverteilung in vielen Fällen die tatsächlichen Daten gut approximiert. Wenn die Anzahl der Daten groß ist (mehr als ungefähr 30), dann nimmt man für die Berechnungen oft an, daß die Population normalverteilt ist, auch wenn es sich in Wahrheit nicht ganz so verhält.
Vereinfacht gesagt ist eine Normalverteilung symmetrisch um den Mittelwert, wobei der größte Anteil der Population nahe beim Mittelwert und nur ein sehr geringer Anteil weit davon entfernt liegt.
Etwas genauer: Ungefähr 80% einer normalverteilten Population sind weniger als die Standardabweichung vom Mittelwert entfernt, und ungefähr 96% sind weniger als das Doppelte der Standardabweichung vom Mittelwert entfernt. Die folgende Tabelle enthält – noch ausführlicher – die zu verschiedenen Vielfachen der Standardabweichung gehörigen Prozentsätze.

% der Population	Vielfaches der Standardabweichung
50%	0.6745
68.27%	1
80%	1.28
90%	1.645
95%	1.96
95.45%	2
99%	2.575
99.73%	3

Diese Tabelle zeigt, daß 95% der Population weniger als das 1,96fache der Standardabweichung vom Mittelwert entfernt sind. Anders gesagt liegen 95% der Population im Bereich von

XM − 1.96*XD bis XM + 1.96*XD,

wobei XM der Mittelwert und XD die Standardabweichung ist. Man nennt dieses Intervall das 95%-Konfidenzintervall für die Population. Ähnlich reicht das 99%-Konfidenzintervall von

XM − 2.575*XD bis XM + 2.575*XD.

Wir wollen jetzt an einem Beispiel erläutern, wie Konfidenzintervalle angewandt werden können. Angenommen, Sie haben den Verdacht, daß Ihr Händler Ihnen nicht ganz volle Zweiliterflaschen Limonade verkauft. Sie kaufen zehn Flaschen und messen sorgfältig den Inhalt. Das Ergebnis in Litern ist:

2.001, 2.040, 2.020, 2.000, 2.015
2.006, 2.005, 2.031, 2.008, 2.018

Alle Flaschen enthalten mindestens 2 Liter. Lassen Sie uns jedoch Mittelwert und Standardabweichung dieser Daten berechnen. Das Ergebnis:

XM = 2.0144
XD = 0.0132

Unter der Annahme, daß unsere Stichprobe aus einer normalverteilten Population stammt, können wir einige Konfidenzintervalle berechnen. Das 95%-Konfidenzintervall reicht von

2.0144 − 1.96*0.0132 bis 2.0144 + 1.96*0.0132, also von

1.989 bis 2.040.

Wir erwarten also, daß 95% der Flaschen zwischen 1.989 und 2.040 Liter enthalten. Das heißt, wir erwarten, daß 2,5% mehr als 2.040 Liter enthalten, während 2,5% weniger als 1.989 Liter enthalten. Wir können auch das 90%-Konfidenzintervall ausrechnen. Es reicht von

2.0144 – 1.645*0.0132 bis 2.0144 + 1.645*0.0132,

oder von

1.993 bis 2.036.

Also erwarten wir, daß 90% der Flaschen zwischen 1.993 und 2.036 Liter enthalten. Folglich würden mindestens 5% der Flaschen weniger als die geforderten 2 Liter enthalten. (Entsprechend enthalten 5% der Flaschen mehr als 2.036 Liter.)
In dieses Beispiel sind viele Annahmen eingegangen; die daraus gezogenen Schlüsse wären deshalb keine ausreichende Grundlage für eine Strafanzeige.
Der Mittelwert unserer Population ist aus einer Stichprobe berechnet worden. Woher wissen wir, daß dies der tatsächliche Mittelwert der Population ist? Der Mittelwert hängt von der gezogenen Stichprobe ab. Wir können aber mit Hilfe der Standardabweichung schätzen, wie weit unser Stichprobenmittelwert vom wahren Mittelwert entfernt ist. Wir können mit 95% Vertrauen sagen, daß der wahre Mittelwert zwischen folgenden Werten liegt:

XM – 1.96*XD/SQR(N–1) bis XM + 1.96*XD/SQR(N–1),

wobei XM der aus einer Stichprobe vom Umfang N berechnete Mittelwert ist. Dieses Intervall heißt 95%-Konfidenzintervall oder -Vertrauensintervall für den Mittelwert. Das 99%-Vertrauensintervall für den Mittelwert reicht von

XM – 2.575*XD/SQR(N–1) bis XM + 2.575*XD/SQR(N–1).

Genau genommen sind diese Berechnungen nur gültig, wenn der Stichprobenumfang groß ist (sagen wir größer als 30). Für kleinere Stichproben sollten wir eigentlich die sogenannte *Studentsche t-Verteilung* statt der Normalverteilung verwenden. Aber das würde den Rahmen dieses Buches sprengen.
Nehmen Sie die Konfidenzintervalle nicht zu ernst, und verwechseln Sie nicht die beiden Arten von Konfidenzintervallen.

Das nächste Programm berechnet 95%- und 99%-Vertrauensintervalle für die Population und den Mittelwert.

```
700 REM VERTRAUENSINTERVALLE
710 C1=1.96*XD:M1=C1/SQR(N):C2=2.575*XD:
M2=C2/SQR(N)
720 PRINT:PRINT:PRINT"VERTRAUENSINTERVAL
 95 %"
730 PRINT:PRINT"POPULATION      VON "XM-
C1
735 PRINT"                      BIS "XM+C1
740 PRINT:PRINT"MITTELWERT ZWISCHEN "XM-
M1
745 PRINT"                      UND "XM+M1
750 PRINT:PRINT:PRINT"VERTRAUENSINTERVAL
 99 %"
760 PRINT:PRINT"POPULATION      VON "XM-
C2
765 PRINT"                      BIS "XM+C2
770 PRINT:PRINT"MITTELWERT ZWISCHEN "XM-
M2
775 PRINT"                      UND "XM+M2
```

DER LETZTE SCHLIFF

Verbindet man die Programme dieses Kapitels miteinander, so erhält man ein nützliches Programm zur Analyse von Daten. Das nachstehende Programm enthält die erforderlichen Verbindungsglieder.

```
10 REM 47DATEN-ANALYSE
20 POKE53280,6:PRINTCHR$(147)"        **
* DATEN-ANALYSE ***"
30 PRINT:PRINT"DIESES PROGRAMM FUEHRT EI
NE ANALYSE VON DATEN DURCH, ";
40 PRINT"DIE SIE DEM COMPUTER EIN-  GEBE
N."
50 PRINT:PRINT"DIE ANALYSE WIRD IN FOLGE
NDEN PUNKTEN   DURCHGEFUEHRT"
60 PRINT:PRINT"1) MITTELWERT DER DATEN
70 PRINT:PRINT"2) MAXIMUM,MINIMUM UND SP
ANNWEITE
```

```
80 PRINT:PRINT"3) STANDARTABWEICHUNG
90 PRINT:PRINT"4) VERTRAUENSINTERVALLE (
95 UND 99 %)
100 PRINT:PRINT:PRINT"BITTE DIMENSIONIER
EN SIE JETZT DEN SPEI-CHER. ";
110 PRINT"SIE KOENNEN DIE EINGABE JEDERZ
EIT MIT '-99999' BEENDEN."
120 PRINT:INPUT"ZAHL ";M:M=M-1
130 IFM<2ORM<>INT(M)THEN120
150 REM DATEN EINGABE 2
160 DIMX(M)
170 FORI=0TOM
180 POKE53280,6:PRINTCHR$(147)"         **
* DATEN EINGABE ***":PRINT
200 PRINT:PRINT"EINGABE VORZEITIG BEENDE
N MIT  -99999 "
220 PRINT:PRINT"SPEICHERKAPAZITAET TOTAL
 "M+1
230 PRINT:PRINT"                NOCH VORHANDEN
 "M-I+1
270 PRINTCHR$(144):PRINT:PRINT:PRINT:PRI
NTI+1". WERT ";
280 INPUTX(I):PRINTCHR$(154):IFX(I)=-999
99ANDI>1THENN=I-1:I=M:GOTO310
290 NEXT
300 N=I-1
310 PRINT:PRINT:PRINT"WEITER MIT LEER-TA
STE"
320 GETG$:IFG$<>" "THEN320
400 REM MITTELEWERTE
410 POKE53280,6:PRINTCHR$(147)"      *** D
ATEN-ANALYSE SEITE I ***":PRINT
415 PRINT:PRINT"DATEN-ANALYSE VON"N+1"WE
RTEN"
420 X=0:FORI=0TON:X=X+X(I):NEXT:XM=X/(N+
1)
430 PRINT:PRINT:PRINT"MITTELWERT
 = "XM
500 REM MAXIMUM,MINIMUM UND SPANNWEITE
510 MAX=-1E+37:MIN=1E+37
```

```
520 FORI=0TON
530 IF X(I)>MAX THEN MAX=X(I)
540 IF X(I)<MIN THEN MIN=X(I)
550 NEXT
560 PRINT:PRINT:PRINT"MINIMUM
 = "MIN
570 PRINT:PRINT"MAXIMUM              = "MA
X
580 PRINT:PRINT"SPANNWEITE           = "MA
X-MIN
600 REM STANDARDABWEICHUNG
610 X=0:FORI=0TON:Y=X(I)-XM:X=X+Y*Y:NEXT

620 XD=SQR(X/N)
630 PRINT:PRINT:PRINT"STANDARDABWEICHUNG
 = "XD
640 PRINTCHR$(19):FORI=1TO20:PRINT:NEXT:
PRINT
645 PRINTCHR$(18)"SEITE 1 (E)  SEITE 2 (
Z)  SCHLUSS (S)"
650 GETG$:IFG$<>"E"ANDG$<>"Z"ANDG$<>"S"T
HEN820
655 IFG$="E"THEN400
660 IFG$="Z"THEN690
665 IFG$="S"THEN900
690 POKE53280,6:PRINTCHR$(147)"     *** D
ATEN-ANALYSE SEITE II ***"
700 REM VERTRAUENSINTERVALLE
710 C1=1.96*XD:M1=C1/SQR(N):C2=2.575*XD:
M2=C2/SQR(N)
720 PRINT:PRINT:PRINT"VERTRAUENSINTERVAL
 95 %"
730 PRINT:PRINT"POPULATION        VON "XM-
C1
735 PRINT"                     BIS "XM+C1
740 PRINT:PRINT"MITTELWERT ZWISCHEN "XM-
M1
745 PRINT"                     UND "XM+M1
750 PRINT:PRINT:PRINT"VERTRAUENSINTERVAL
 99 %"
```

```
760 PRINT:PRINT"POPULATION       VON "XM-
C2
765 PRINT"                     BIS "XM+C2
770 PRINT:PRINT"MITTELWERT ZWISCHEN "XM-
M2
775 PRINT"                     UND "XM+M2
800 PRINTCHR$(19):FORI=1TO20:PRINT:NEXT:
PRINT
810 PRINTCHR$(18)"SEITE 1 (E)  SEITE 2 (
Z)  SCHLUSS (S)"
820 GETG$:IFG$<>"E"ANDG$<>"Z"ANDG$<>"S"T
HEN820
830 IFG$="E"THEN400
840 IFG$="Z"THEN690
850 IFG$="S"THEN900
900 PRINTCHR$(147):PRINT:PRINT:PRINT:PRI
NT"NEUER START (J/N) ?"
910 GETG$:IFG$<>"J"ANDG$<>"N"THEN910
920 IFG$="J"THENRUN
930 PRINTCHR$(147)"AUF WIEDERSEHEN":END
```

Mathematik mit dem C 64 — Diskette

Inhalt:

```
"01RECHTW.DREIECK"
"02DREIECKE"
"03BRECHUNG"
"04ERDTRIGON."
"05POTENZEN 1"
"06POTENZEN 2"
"07KOMPLEXE ZAHL."
"08QUAD.GLEICH"
"09POLYNOME"
"10POLYNOME 2"
"11WURZELN"
"12FOLGEN ARITHM."
"13FOLGEN GEOMETR"
"14ZINSESZINS"
"15ZINSESZINS VAR"
"16FIBONACCI FOL."
"17BASIS UMWANDL."
"18BINAER UMWANDL"
"19WOCHENTAGE"
"20KALENDER"
"21DATEN MANGE."
"22GROESST.TEILER"
"23ERASTOSTHENES1"
"24PRIMZAHLEN 1"
"25PRIMZAHLEN 2"
"26ERASTOSTHENES2"
"27PRIMZAHLEN 3"
"28PRIMFAKTOREN"
"29PRIMZAHLEN 4"
"30TRIPEL"
"31AUF 40 STELLEN"
"32DEMO MATRIZEN"
"33INVER.MATRIZEN"
"34GLEICH.SYSTEM"
"35EINFACHCODE"
"36TAUSCHCODE"
"37MATRIXCODE"
```

Alle Programme dieses Bandes sind auf der beigefügten Diskette gespeichert. Wir empfehlen, sofort ein Backup anzufertigen und dieses sicher zu verwahren.
Die einzelnen Programme können bequem mit z. B.

LOAD "01*", 8

geladen werden.

```
"38MATRIXCODE 3"
"39KOPF O. ZAHL"
"40WUERFEL"
"41WUERFEL MIT"
"42 2WUERFEL"
"43 2WUERFEL MIT"
"44 KARTEN"
"45KARTEN MISCHER"
"46KNOEPFE"
"47DATEN-ANALYSE"
```

NOTIZEN

Finden Sie manchmal einzelne Teile Ihrer Programme schwierig zu schreiben? So daß Sie zwar wissen, was Sie wollen, aber sich nicht genau im klaren sind, wie Sie dabei vorgehen sollen? Kann man mit COS, ABS oder SGN etwas anfangen? Was bedeuten sie, und wie lassen sie sich einsetzen?

Das vorliegende Buch erklärt alle diese mathematischen Hilfsmittel. Sämtliche mathematischen Funktionen des Commodore 64 werden darin beschrieben. Ihre Anwendungen sind in kurzen Programmen erläutert, die Sie wörtlich in Ihre eigenen Programme übernehmen können.

Das Buch enthält eine Fülle von Informationen zu so unterschiedlichen Gebieten wie Kryptographie, Zufallszahlen, Logik, Folgen und Reihen, Trigonometrie, Primzahlen, Felder und Matrizen, Wahrscheinlichkeitstheorie und Statistik.

Czes Kosniowski lehrt an der Universität von Newcastle upon Tyne. Er hat mehrere Bücher und Artikel über Mathematik und Datenverarbeitung geschrieben und ist regelmäßiger Mitarbeiter der Zeitschrift Popular Computing Weekly.

Commodore

Commodore GmbH
Lyoner Straße 38
D-6000 Frankfurt/M. 71

Commodore AG
Aeschenvorstadt 57
CH-4010 Basel

Commodore GmbH
Kinskygasse 40–44
A-1232 Wien

Artikel-Nr. 556430/ 12.84 Änderungen vorbehalten ISBN-Nr. 3-89133-008-1