Ohmsches Gesetz **H1**

Nach Eingabe von Strom und Spannung soll der Widerstand berechnet und am Terminal ausgegeben werden. Dieses Programm besteht hauptsächlich aus Ein- und Ausgabeanweisungen, die mit Hilfe von `FORMAT`-Anweisungen zu gestalten sind. Für die Berechnung des Widerstandes ist eine einzige Zuordnungsanweisung notwendig.

Hinweis: Dieses Programm dient als erstes Beispiel im wesentlichen dem Kennenlernen der Tastatur. Bitte beachten Sie, daß in `FORTRAN` keine Kombination von Ein- und Ausgabe in EINER Anweisung möglich ist, und daß im allgemeinen die `FORMAT`-Anweisungen für Ein- und Ausgabe verschieden sein werden (Carriage control character!). Es wird häufig gewünscht, daß der Cursor nach einer Text-Ausgabe für die nachfolgende Eingabe in derselben Bildschirmzeile stehen bleiben soll: In diesem Fall muß im Ausgabeformat als Zeilenvorschubzeichen ein $-Zeichen verwendet werden.

Polarkoordinaten B5

Umrechnung von kartesischen Koordinaten x, y in ebene Polarkoordinaten r, ϕ. Es ist zu berücksichtigen, daß die Argumente von Winkelfunktionen und die zugehörigen Umkehrfunktionen im Bogenmaß definiert sind. Das Programm soll die Winkel im Gradmaß angeben sowie die einzelnen Quadranten und etwaige Spezialfälle unterscheiden können.

Hinweis: In der `FORTRAN`-Bibliothek sind zwei Umkehrfunktionen für den Tangens enthalten: `ATAN` und `ATAN2`. Erprobe beide Funktionen und vergleiche die Ergebnisse.

Geburtstagsparty	**F2**

Auf einer Geburtstagsparty sind N Personen anwesend. Berechne die Wahrscheinlichkeit p dafür, daß zumindest (!) zwei davon am gleichen Tag Geburtstag haben und gib das Ergebnis für verschiedene Werte von N in Form einer Tabelle aus. Es gibt insgesamt 365^N Möglichkeiten wie N Geburtstage verteilt sein können. Es ist leichter die Wahrscheinlichkeit q dafür zu berechnen, daß alle N Personen an verschiedenen Tagen Geburtstag haben:

$$q = \frac{365.364\ldots(366 - N)}{365^N}$$

Hinweis: Die gesuchte Wahrscheinlichkeit p ergibt sich dann durch $p = 1 - q$

Dreiecksbedingung	**B3**

Koppelt man zwei Drehimpulse mit den Quantenzahlen J_1 und J_2 zu einem dritten Drehimpuls, dann müssen die zugehörigen Quantenzahlen J_3 zusammen mit J_1 und J_2 die Dreiecksbedingung erfüllen:

$$J_3 \quad \in \quad \{|J_1 - J_2|, |J_1 - J_2| + 1, \ldots, J_1 + J_2\}$$

Nach Eingabe von J_1 und J_2 sollen die erlaubten Werte von J_3 ausgegeben werden. Es soll berücksichtigt werden, daß in der Quantenmechanik auch halbzahlige Drehimpulse (Spin !) auftreten.

Binär - Dezimal - Umwandlung	H3

Erstelle ein Programm, das aus einer einzugebenden Dualzahl (zum Beispiel 1001011) eine Dezimalzahl mit dem entsprechenden Wert erzeugt. Das Programm soll so eingerichtet sein, daß die Dualzahl mit einem Eingabevorgang eingelesen wird, nachdem die Anzahl der Stellen festgelegt wurde.

Hinweis: Die einzelnen binären Ziffern werden in ein eindimensionales Feld gespeichert und der Index (=Position im Feld) in die entsprechende Potenz von 2 umgerechnet .

Eulersche Zahl **J1**

Berechne die Euler'sche Zahl e aus der Reihenentwicklung

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots = \sum_{0}^{\infty} \frac{1}{n!}$$

Bei der Berechnung soll die endliche numerische Genauigkeit berücksichtigt und die Konvergenz der Reihe kontrolliert werden.

Hinweis: Bei einer Berechnung mit `DOUBLE PRECISION`-Variablen müssen auch die auftretenden Konstanten als doppelt genau definiert werden. Es ist auch möglich ein `FUNCTION`-Unterprogramm zu definieren, das jeweils aus $n!$ den folgenden Wert $(n+1)!$ berechnet.

Integral zur Berechnung von π	**J2**

Bestimme die Zahl π aus dem folgenden Integral

$$\int\limits_0^1 \frac{2}{\sqrt{1-x^2}} dx = \pi$$

Da der Integrand an der oberen Grenze singulär wird, können weder die Simpson'sche Regel noch die Trapezregel angewendet werden. Es empfiehlt sich daher die Summation von Intervall-Rechtecken als Näherung für das Integral. Dabei wird der Funktionswert in der Mitte jedes Intervalls mit der zugehörigen Intervall-Breite multipliziert.

Hinweis: Es ist zu erwarten, daß das Ergebnis infolge der Unendlichkeitsstelle bei $x = 1$ sehr ungenau ist. Eine mögliche Verbesserung besteht in der Verwendung unterschiedlicher Intervall-Breiten, wobei die endliche numerische Genauigkeit aber beliebig kleine Intervalle verbietet.

Konversion von Zeichen und Zahlen — R1

Erprobe das Eingeben einer `INTEGER`-Variablen, die dann mit einer `FORMAT`-Angabe für Zeichen ausgegeben wird. Vergleiche das Ergebnis mit der Wirkung der Bibliotheksfunktion `CHAR(.)`. Erstelle eine Tabelle für die ASCII-Werte der Großbuchstaben `A-Z`.

Hinweis: Die verwendete Variable muß dabei vom Datentyp `LOGICAL` sein.

Pythagoräische Tripel **A1**

Mit Hilfe von `INTEGER`-Variablen sollen Zahlentripel i, j, k gesucht werden, für die die Beziehung $i^2 + j^2 = k^2$ exakt erfüllt ist. Der Name des Beispiels deutet an, daß die entsprechenden Seitenlängen ein rechtwinkeliges Dreieck ergeben.

Hinweis: Der Suchvorgang soll soweit eingeschränkt werden, daß nach Möglichkeit jedes Zahlenpaar i, j nur einmal getestet wird. Die Anzahl der nötigen Tests für $k \leq 50$ soll ausgegeben werden.

Positive Potenzen von 2	**A2**

Berechne den dezimalen Wert der Zahl 2^{193}. Zur Vereinfachung der Kontrolle des Ergebnisses soll die Ausgabe so durchgeführt werden, daß Tausendergruppen jeweils durch einen Zwischenraum getrennt sind.

Hinweis: Das Ergebnis besteht aus 59 Ziffern: `12␣554␣203...025␣792`

Nullstelle (Intervallschachtelung)	B2

Suche die Nullstelle der Funktion $y = x^2 + e^x - 2$ im Intervall zwischen 0 und 1. Das Programm soll nicht nur für monoton ansteigende Funktionen anwendbar sein.

Hinweis: Die Nullstelle soll durch fortschreitende Halbierung von Intervallen in Verbindung mit geeigneten `IF`-Entscheidungen angenähert werden. Es ist zweckmäßig, für die Berechnung der Funktion die Form eines `FUNCTION`-Unterprogramms oder einer `FUNCTION`-Anweisung zu benützen.

Vieleck (π)	**B6**

Berechne π durch schrittweise Annäherung des (halben) Umfanges eines Einheitskreises durch Vielecke. Ausgehend von einem 6-Eck soll die Anzahl der Ecken in jedem Schritt verdoppelt werden und danach die Anzahl der Ecken und der (halbe) Umfang ausgedruckt werden.

Hinweis: Die folgende Formel für die Seitenlänge a_{2n} eines $2n$-Ecks, wenn a_n die Seitenlänge des n-Ecks gegeben ist, kann leicht mit elementaren geometrischen Formeln überprüft werden: $a_{2n}^2 = 2 - \sqrt{4 - a_n^2}$

Strahlungsfunktion	B7

Bestimme auf 5 Dezimalstellen genau die Lage des Maximums x_{max} und den zugehörigen Funktionswert $f(x_{max})$ für die PLANCKsche Strahlungsfunktion

$$f(x) = \frac{1}{x^5} \frac{1}{e^{\frac{1}{x}} - 1}$$

Hinweis: Das Programm soll einen Suchvorgang enthalten, der in der Lage ist, ein Maximum zu erkennen und gegen dieses Maximum zu konvergieren. Die Genauigkeit des Ergebnisses soll durch Eingabe beeinflußt werden können.

Nullstelle (Regula falsi)	**B8**

Die Lage der Nullstellen der Funktion $g(x) = e^{-x^2} + (5-x)e^x - 5$ soll durch fortgesetzte lineare Interpolation angenähert werden. Dazu wird für zwei Argumente, bei denen die Funktion g verschiedenes Vorzeichen hat, eine Gerade durch die Punkte $x_i, g(x_i)$ und $x_{i+1}, g(x_{i+1})$ gelegt und der zugehörige Schnittpunkt mit der x-Achse wird zur oberen oder unteren Grenze des neuen Intervalls.

Hinweis: Das Programm soll solange nach zwei neuen Argumenten fragen, bis die zugehörigen Funktionswerte verschiedenes Vorzeichen haben, und dann mit der Suche nach der Nullstelle beginnen. Für die Berechnung der Funktion soll ein Funktions-Unterprogramm oder eine Anweisungsfunktion verwendet werden.

Quotientenpyrometer **B9**

Bei einer Temperaturmessung mit dem Quotientenpyrometer wird das Verhältnis $R = \frac{I_1}{I_2}$ zweier Strahlungsintensitäten bei den Wellenlängen $\lambda_1 = 530\text{nm}$ und $\lambda_2 = 620\text{nm}$ bestimmt. Nach dem PLANCK-schen Strahlungsgesetz gilt für R:

$$R = \frac{\lambda_2}{\lambda_1}\left(\frac{e^{\frac{a}{\lambda_2 T}} - 1}{e^{\frac{a}{\lambda_1 T}} - 1}\right)$$

Es gilt $a = 1.438\text{Kcm}$. Bestimme aus dieser Formel die Temperatur in Kelvin für die Werte $R = 0.305$ und $R = 0.0425$.

Dreiecksfläche **C1**

Die Lage eines Dreiecks im Raum sei durch die drei Eckpunkte P_1, P_2 und P_3 gegeben. Das Programm soll die zugehörige Fläche bestimmen. Für die Koordinaten der drei Punkte bietet sich die Verwendung eines zweidimensionalen Feldes `P(3,3)` an, doch muß bei der Programmierung sauber unterschieden werden, welcher Index sich auf die Punkte und welcher sich auf die Koordinaten bezieht.

Hinweis: Die Verwendung des Ex-Produktes von Vektoren ist eine Möglichkeit zur Bestimmung der Fläche.

Ferromagnet	**D1**

Die Molekularfeldtheorie ergibt unterhalb der CURIE-Temperatur für den Verlauf der (reduzierten) Magnetisierung σ als Funktion der (reduzierten) Temperatur τ ($\tau \equiv \frac{T}{T_{Curie}}$) die Formel

$$\sigma = \mathrm{B}_S \left(\frac{3S}{S+1} \frac{\sigma}{\tau} \right)$$

wobei die BRILLOUIN-Funktion B_S auf folgende Weise definiert ist:

$$\mathrm{B}_S(x) = \frac{2S+1}{2S} \coth \left(\frac{2S+1}{2S} x \right) - \frac{1}{2S} \coth \left(\frac{1}{2S} x \right)$$

Da in der obigen Gleichung σ auch im Argument der BRILLOUIN-Funktion enthalten ist, handelt es sich um eine transzendente Gleichung für σ. Für diese Gleichung läßt sich eine Lösung durch fortgesetzte Iteration finden: Ein Wert für σ wird in die BRILLOUIN-Funktion eingesetzt und das resultierende σ dann mit dem vorhergehenden Wert verglichen, wieder eingesetzt, wieder verglichen ... usw., bis die beiden Werte im Rahmen einer Fehlerschranke übereinstimmen.

Hinweis: Es sollen Anweisungsfunktionen für $\cosh(x)$ und $\sinh(x)$ definiert werden.

Negative Potenzen von 2	**D2**

Erstelle ein Programm, das die negativen Potenzen 2^{-n} mit n zwischen 1 und 100 ermittelt und am Schnelldrucker in Form von Dezimalzahlen untereinander ausdruckt:

$$\begin{aligned} 2**(-1) &= 0.5 \\ 2**(-2) &= 0.25 \\ 2**(-3) &= 0.125 \\ &\ \ \ldots \text{usw.} \ldots \end{aligned}$$

Hinweis: Vielleicht verlangt das Beispiel eine Methode, die sich nicht nur auf das Rechnen mit rellen Zahlen beschränkt (numerische Genauigkeit !?). Man kann bei der Lösung des Problems versuchen, die Rechenregeln für Integerzahlen oder die Funktion `MOD(I,J)` einzusetzen.

Näherung für die Wurzelfunktion **E5**

Benütze den folgenden rekursiven Ausdruck zum näherungsweisen Berechnen der der Wurzel einer positiven, reellen Zahl a:

$$x_0 = a \quad ; \qquad x_{n+1} = 0.5 \left(\frac{a}{x_n} + x_n \right)$$

Das Programm soll zwei Abbruchbedingungen enthalten, eine maximale Iterationszahl und eine Schranke für die Differenz $x_{n+1} - x_n$. Die Ausgabe soll in Form einer Tabelle die Iterationszahl, den Näherungswert, das Ergebnis der Bibliotheksfunktion und die Differenz der beiden Ergebnisse enthalten.

Numerische Differentiation **E6**

Die analytisch berechnete Ableitung einer Funktion $f'(x)$ soll mit dem Näherungswert für die Ableitung entsprechend der Formel

$$f'(x) \simeq \frac{f(x+h) - f(x)}{h}$$

verglichen werden. Drucke eine Tabelle, in der die Differenz zwischen Näherungswert und analytischem Ergebnis für abnehmende Schrittweite h ausgegeben wird.

Hinweis: Funktionsnamen können *nicht* erst bei Ablauf des Programms durch Eingabe ausgewählt werden, denn das zugehörige Unterprogramm muß während des Link-Vorganges aus den Routinen der `FORTRAN`-Bibliothek ausgewählt worden sein.

Normreihe für Ohmsche Widerstände H2

Aus Spannung und Strom kann sehr leicht der zugehörige Widerstand berechnet werden. Das Programm soll aber den der Normreihe (E12) entsprechenden, nächstliegenden Widerstandswert finden und dazu den veränderten Strom und die am Widerstand abgegebene Leistung angeben.

Hinweis: Die Normreihe enthält die Stufen (E12) = 1.0, 1.2, 1.5, 1.8, 2.2, 2.7, 3.3, 3.9, 4.7, 5.6, 6.8, 8.2, 10.0

Zahlen Ordnen	**B1**

Verlangt ist ein Programm zum Einlesen und Ordnen von `N` reellen Zahlen. Zuerst wird die Anzahl der einzulesenden Zahlen festgelegt und danach die Zahlenfolge eingegeben. Als Ergebnis soll die ursprüngliche Zahlenfolge und die (auf- oder absteigend) geordnete Folge am Terminal ausgegeben werden.

Hinweis: Es ist lehrreich, die Ordnung durch Austauschen von Wertepaaren innerhalb *eines* einzigen eindimensionalen Feldes vorzunehmen.

Faktorielle bis 99!	**B4**

Berechne die Werte von n! und stelle die Ergebnisse für $0 \leq n \leq 99$ in Form einer Liste dar. Dabei sollen 5 Spalten auf einem Blatt ausgedruckt werden, wobei die erste Spalte die Ergebnisse von 0! bis 19! etc. enthält.

Hinweis: Um den aufgrund der Größe der Ergebnisse zu erwartenden Schwierigkeiten auszuweichen, empfielt sich die Berechnung von $\log_{10}(n!)$, um auf indirekte Weise zum Ziel zu kommen. Die Werte für die Logarithmen sollen in einer Feldvariablen abgespeichert werden.

Drehung eines Koordinatensystems — C4

Die Koordinaten eines Punktes im Koordinatensystem S seien durch p_x und p_y gegeben. Derselbe Punkt (Vektor) in einem um den Winkel α gedrehten Koordinatensystem S' hat die Koordinaten q_x, q_y, mit

$$\begin{aligned} q_x &= p_x \cos(\alpha) + p_y \sin(\alpha) \\ q_y &= -p_x \sin(\alpha) + p_y \cos(\alpha) \end{aligned}$$

Berechne q_x, q_y in Abhängigkeit von einem einzugebenden Drehwinkel.

Hinweis: Bei der Durchführung soll für die Transformationsmatrix ein zweidimensionales Feld und für die Punktkoordinaten ein eindimensionales Feld verwendet werden.

Simpson-Integration **L8**

Quadratische Gleichung **O1**

Schreibe ein Programm zur Lösung der quadratischen Gleichung

$$x^2 + bx + c = 0$$

und unterscheide die drei Fälle, in denen die Diskriminante $>$, $<$ oder $= 0$ ist.

Hinweis: Suche die Lösungen für $b = 4$ und $c = 2, 4, 8$.

Für die numerische Integration nach der Simpson-Regel gilt folgende Formel:

$$\int_{x_0}^{x_n} y(x) = \frac{h}{3}(y_0 + 4y_1 + y_2) + \frac{h}{3}(y_2 + 4y_3 + y_4) + \ldots \qquad , \qquad h = x_i - x_{i-1}$$

Es soll ein Unterprogramm `SIMPS` erstellt werden, das die Integration einer Funktion `FUNC` zwischen den Grenzen `XU` und `XO` durchführt.

Hinweis: Da beim Aufruf von `SIMPS` im Hauptprogramm der aktuelle Name `FUNC` der zu integrierenden Funktion als Parameter an das Unterprogramm `SIMPS` übergeben wird, muß dem Hauptprogramm mit einer `EXTERNAL`-Deklaration mitgeteilt werden, daß `FUNC` der Name eines Funktions-Unterprogrammes und keiner Variablen ist.

Bessel-Funktionen D3

Erstelle ein `FUNCTION`-Unterprogramm `BESMX`, das den Wert der BESSEL-Funktion $J_m(x)$ aus der Potenzreihe

$$J_m(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{n!(m+n)!} \left(\frac{x}{2}\right)^{m+2n}$$

berechnet und vergleiche diese Werte in einer kleinen Tabelle ($0 \leq m \leq 10$) mit jenen Werten, die sich mit Hilfe einer Rekursionsformel ergeben:

$$J_{m+1}(x) = \frac{2m}{x} J_m(x) - J_{m-1}(x)$$

Hinweis: Um die Rekursionsformel benutzen zu können, müssen die ersten beiden Ergebnisse für $m = 0$ und $m = 1$ bekannt sein, sie werden aus der Summenformel berechnet. Vergleiche die Tabellen für $x = 0.5$, 1.0 und 2.0.

Nagelbrett	**F1**

Auf einem Nagelbrett sind `NR` Reihen von Nägeln so angebracht, daß jede Reihe gegenüber der vorhergehenden um einen halben Nagelabstand versetzt ist. Ausgehend von derselben Stelle fallen `NK` Kugeln durch die Nagelreihen, wobei bei jedem Nagel die Entscheidung, ob die jeweilige Kugel in den rechten oder linken Zwischenraum fällt, zufällig getroffen werden soll... usw. Die durch die Zwischenräume der letzten Nagelreihe fallenden Kugeln werden gezählt und auf dem Zeilendrucker graphisch dargestellt, wobei jede zweite Zeile leer bleiben soll (Nagel).

Hinweis: Bei der graphischen Darstellung soll am Anfang jeder Zeile auch die Zahl der gezählten Kugeln angegeben werden. Es kann in diesem Beispiel die Definition der Felder so erfolgen, daß die Kugeln anfänglich im Element mit dem Index 0, nach der ersten Reihe in den Elementen -1 und 1, nach der nächsten Reihe in -2, 0, 2 anzutreffen sind ...usw. Die Initialisierung des Zufallszahlengenerators soll vom Terminal aus geschehen. Für große Kugelzahlen soll von Zeit zu Zeit eine Meldung über die schon geworfenen, bzw. noch zu werfenden Kugeln vorgesehen werden!!!

Potenzen einer Matrix **O3**

Erstelle ein Programm, das zu einer m-reihigen, quadratischen Matrix $\mathcal{A}$ die Matrizenprodukte $\mathcal{A}^2$, $\mathcal{A}^3$, $\mathcal{A}^4$ und $\mathcal{A}^5$ berechnet ($m \leq 10$). Die Formel für ein Matrizenprodukt lautet:

$$\mathcal{C} = \mathcal{A}\mathcal{B} \qquad : \qquad c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$$

$$\mathcal{A} = \begin{pmatrix} 1. & -5. & 4. \\ 7. & 2. & 0. \\ 3. & 0. & 6. \end{pmatrix}$$

Hinweis: Verwende zum Einlesen und Ausdrucken die implizite `DO`-Anweisung.

Ableitung eines Polynoms **O4**

Schreibe ein Programm zur Bestimmung der Koeffizienten aller Ableitungen eines Polynoms n-ter Ordnung $P_n(x) = \sum a_i x^i$. Nach jeder Ableitung sollen die Koeffizienten aller Potenzen von x ausgegeben werden.

Hinweis: Überprüfe das Programm mit $P_{12}(x) = x^{12} + 4x^{11} - 2x^{10} + 5x + 2$.

Monte Carlo-Methode (Integral)	Y3

Anwendung der Monte Carlo-Methode zur Integration der Funktion $z = x^2$ im Intervall $[0, 1]$. Da die Funktion z in diesem Intervall zwischen 0 und 1 variiert, kann man auf folgende Weise zu einem Näherungswert für das Integral kommen: Mit Hilfe eines Zufallszahlengenerators werden viele Wertepaare $x, y \in [0, 1]$ bestimmt. Aus dem Verhältnis der Treffer (= wenn das 'gewürfelte' y kleiner ist als der Funktionswert z an der Stelle x) zur Zahl der Versuche kann man den gesuchten Flächenbereich bestimmen.

Hinweis: In der `FORTRAN`-Bibliothek ist ein Pseudozufallszahlengenerator in Form eines `FUNCTION`-Unterprogramms `RAN(I)` enthalten. Die Integervariable `I` ist (nur!) anfänglich auf einen großen, ungeraden Wert zu setzen. Sie verändert sich bei jedem Aufruf der Funktion `RAN`.

Lineare Regression	**Y1**

Erstelle ein Programm, das von einem Datenfile `[FORPRAK]REGRESS.DAT` alle darin enthaltenen Records mit je einem Zahlenpaar x, y einliest und die Steigung und den Achsenabschnitt der Ausgleichs-(Regressions)-Geraden berechnet. Unter der Annahme, daß n Wertepaare eingelesen wurden, gelten folgende Formeln für die Steigung k und den Achsenabschnitt d:

$$k = \frac{n\Sigma xy - \Sigma x.\Sigma y}{n\Sigma x^2 - (\Sigma x)^2} \qquad , \qquad d = \frac{1}{n}(\Sigma y - k.\Sigma x)$$

Hinweis: Die beiden Zahlen sind mit einer Feldlänge von 10 einzulesen.

Pascal-Graphik	E1

Die Binomialkoeffizienten des PASCALschen Dreiecks sollen durch zwei (wählbare) Zeichen, zum Beispiel '.' und '0', dargestellt werden, je nachdem ob der Koeffizient gerade oder ungerade ist. Die Ausgabe soll bis zu einer (wählbaren) Ordnung auf dem Schnelldrucker erfolgen. Um eine symmetrische Anordnung zu erreichen, werden zwischen die einzelnen Zeichen Leerstellen gesetzt.

```
      0
   0     0
0     .     0
0     0     0     0
  ...usw. ...
```

Hinweis: Die Entscheidung ob ein Binomialkoeffizient gerade oder ungerade ist, kann auch ohne Berechnung des Koeffizienten (das heißt, ohne daß man seinen Wert kennt) durchgeführt werden.

Relativistische Faktoren	**E3**

Im Rahmen der speziellen Relativitätstheorie treten eine Reihe von Funktionen auf, die durch das Verhältnis $\beta = \frac{v}{c}$ bestimmt werden. Berechne eine Tabelle für die Funktionen $(1-\beta^2)^{\frac{1}{2}}$, $(1-\beta^2)^{-\frac{1}{2}}$, $(1-\beta^2)^{-\frac{3}{2}}$ und $\beta(1-\beta^2)^{-\frac{1}{2}}$. Dabei sollen die Schritte für β abgestuft gewählt werden, etwa $\Delta\beta = 0.05$ für $0 < \beta < 0.6$, $\Delta\beta = 0.02$ für $0.6 < \beta < 0.96$, $\Delta\beta = 0.01$ für $0.96 < \beta < 0.99$ und als letzter Wert $\beta = 0.995$.

Julianisches Datum C3

Man berechne für ein einzugebendes Datum das Julianische Datum. Beim Julianischen Datum werden die Tage seit dem Mittag des 1.1.4713 v.Chr. gezählt. `JD` = 1 entspricht dem 1.1.4713 v.Chr.(12 Uhr) und `JD` = 2436934 dem 1.1.1960 (12 Uhr).

Hinweis: Das Programm soll für Kalenderdaten zwischen dem 1.1.1960 und dem 31.12.1999 geeignet sein. Der Unterschied des Tagesanfangs soll nicht berücksichtigt werden.

Pascalsches Dreieck	**L2**

Man schreibe ein Programm, welches die ersten n Zeilen des PASCAL'schen Dreiecks berechnet und auf ein Datenfile schreibt. Dieses File soll dann auf dem Schnelldrucker ausgegeben werden:

$$
\begin{array}{ccccccc}
 & & & 1 & & & \\
 & & 1 & & 1 & & \\
 & 1 & & 2 & & 1 & \\
1 & & 3 & & 3 & & 1 \\
 & & \ldots & \text{usw.} & \ldots & &
\end{array}
$$

Hinweis: Durch die in einer Druckerzeile zur Verfügung stehende Anzahl von Zeichen (132) ergibt sich bei der obigen Anordnung eine natürliche Grenze, bis zu der gerechnet werden kann. Man kann aber auch eine andere Anordnung wählen, ...

Bildschirm-Plot einer Funktion — H4

Das Unterprogramm `TEPLOT(XANF,XEND,XSTEP,AFUN)` soll eine Funktion `AFUN` zwischen einem Anfangswert und einem Endwert mit den Schritten `XSTEP` am Bildschirm in der unten angegebenen Form ausgeben. Der Maßstab soll so festgelegt werden, daß Maximum und Minimum innerhalb einer Zeile dargestellt werden können. Die graphische Darstellung soll den Funktionswert durch '`*`', die Begrenzung mit '`I`' und eine Nullinie durch '`.`' andeuten.

```
x-Wert    <y-Min                        0                        y-Max>
4.000     I           *                 .                             I
4.020     I                       *     .                             I
4.040     I                             *                             I
4.060     I                             .    *                        I
4.080     I                             .       *                     I
4.100     I                             .         *                   I
4.120     I                             .          *                  I
```

Hinweis: In jeder Zeile am Bildschirm sollen nur 79 Zeichen ausgegeben werden. Das Unterprogramm kann mit der Funktion $y = e^{-x}\sin(5x)$ getestet werden.

Runge-Kutta Approximation **C5**

Man löse die gewöhnliche Differentialgleichung $y' = -x.y$ mit der Runge-Kutta-Methode. Als Anfangsbedingung für $y(x)$ soll $y(0) = 1$ gelten. An der Stelle $x_0 = 0$ beginnend, soll die Integration mit der Schrittweite h bis zu x_{end} durchgeführt werden. Man vergleiche das Ergebnis mit der exakten Lösung $y = exp(-x^2/2)$. Die gesuchten Wertepaare x_i, y_i werden nach den Beziehungen $x_i = x_{i-1} + h$, $y_i = y_{i-1} + k$ für $i = 1, 2, \ldots$ bestimmt. Die Größe k wird für eine Differentialgleichung $y' = f(x, y)$ wie folgt berechnet:

$$
\begin{aligned}
k &= (k_1 + 2k_2 + 2k_3 + k_4)\tfrac{1}{6} \\
k_1 &= h.f(x_{i-1}, y_{i-1}) \\
k_2 &= h.f(x_{i-1} + \tfrac{h}{2}, y_{i-1} + \tfrac{k_1}{2}) \\
k_3 &= h.f(x_{i-1} + \tfrac{h}{2}, y_{i-1} + \tfrac{k_2}{2}) \\
k_4 &= h.f(x_{i-1} + h, y_{i-1} + k_3)
\end{aligned}
$$

Hinweis: Das Unterprogramm `RUNKUT` soll allgemein verwendbar sein. Dazu wird die zu integrierende Funktion als `FUNCTION`-Unterprogramm gestaltet, im aufrufenden Programm mittels `EXTERNAL` deklariert und als Parameter an `RUNKUT` übergeben.

Quadroschneck **D5**

Die Elemente eines zweidimensionalen Feldes (Typendeklaration `CHARACTER*1`) sollen so durch *-Zeichen besetzt werden, daß beim Ausdrucken eine Spirale entsteht:

```
*************
*  *******  *
*  *  ****  *
*  **********
********  . . .
```

Hinweis: Um eine annähernd quadratische Form zu erreichen, muß man den Unterschied zwischen dem Zeilen- und dem Zeichenabstand des Schnelldruckers berücksichtigen.

Matrix-Inversion **D4**

Ausgehend von einer vorgegebenen Matrix $\mathcal{A}$ und einer zweiten Matrix $\mathcal{B}$, in der anfänglich die Einheitsmatrix gespeichert ist, soll mit Hilfe des Gaußschen Eliminationsverfahrens die inverse Matrix $\mathcal{A}^{-1}$ ermittelt werden. Dabei werden die gleichen linearen Operationen jeweils an beiden Matrizen vorgenommen: Zunächst erhält das Diagonalelement a_{11} den Wert 1 (Division der (ersten) Zeile durch a_{11}) und dann werden die anderen Elemente dieser Spalte eliminiert, indem man geeignete Linearkombinationen der (ersten) Zeile mit den anderen bildet. Nach der schrittweisen Durchführung dieser Operationen (immer für $\mathcal{A}$ und $\mathcal{B}$!) für alle Spalten enthält $\mathcal{B}$ die inverse Matrix zu $\mathcal{A}$. Ein Produkt der beiden Matrizen $\mathcal{A}$ und $\mathcal{A}^{-1}$ sollte die Einheitsmatrix ergeben.

$$\mathcal{A} = \begin{pmatrix} 3. & 2. & 1. \\ 2. & 3. & 1. \\ 1. & 1. & 4. \end{pmatrix}$$

Hinweis: Um die Kontrolle der schrittweisen Ausführung zu erleichtern, sollen beide Matrizen nach jedem Schritt ausgedruckt werden. Es wird empfohlen, die Elemente der Matrix $\mathcal{A}$ mit Hilfe der `DATA`-Anweisung einzugeben.

Entschlüsseln eines Geheimcodes S1

Ein Text kann leicht verschlüsselt werden, indem man jeden Buchstaben durch seinen n-ten Nachfolger im Alphabet ersetzt, wobei nach `Z` wieder mit `A` begonnen wird, etc. Der österreichische Geheimdienst hat im File [FORPRAK]CODE.TXT einige Botschaften zusammengestellt. Leider wurde der Zettel mit den entschlüsselten Texten zum Pfeifenanzünden verwendet, sodaß von jeder Zeile nur mehr einige Buchstaben erhalten geblieben sind:
`WAHR ...TOPF ...KEIN ...AUCH ...COOL ...IHRE ...HIER ...DAS ...DEN ...ABER`.
Da jede Zeile anders verschlüsselt ist, brauchen wir ein Programm, das eine (verschlüsselte) Textzeile einliest und dann den entschlüsselten Text ausschreibt.

Hinweis: Es werden nur Großbuchstaben des (englischen) Alphabets und das Leerzeichen verwendet. Den Großbuchstaben `A-Z` entsprechen die ASCII-Werte 65-90. Die Leerzeichen werden bei der Verschlüsselung *nicht* verändert. Das Leerzeichen hat den ASCII-Wert 32.

Mandelbrot-Menge	**E2**

Wenn der Betrag einer komplexen Zahl z_i, die mit der Iterationsformel $z_1 = c$; $z_{i+1} = z_i^2 + c$ berechnet wird, den Wert 2 übersteigt, wächst der Betrag bei Fortsetzung der Iteration über jede Grenze. Jene Zahlenmenge in der komplexen Ebene, für die der Betrag 2 auch bei beliebiger Iteration nicht erreicht wird, wird als MANDELBROT-Menge bezeichnet. Erzeuge mit dem Schnelldrucker ein Bild von der Grenze dieser Menge durch Abbildung des Bereichs $-1.5 \leq \mathrm{Re}(z) \leq 1$, $-1 \leq \mathrm{Im}(z) \leq 1$. Es wird dabei nur bis zu einer maximalen Anzahl von Iterationsschritten gerechnet, und die Mitglieder der Menge werden durch Leerzeichen wiedergegeben. Die übrigen Zahlen können durch verschiedene Zeichen dargestellt werden, je nachdem nach wieviel Schritten sie aus der Menge ausscheiden.

Hinweis: Welche Zahlen aus der komplexen Ebene auf Zugehörigkeit zur MANDELBROT-Menge überprüft werden, hängt von der Anzahl der Zeichen einer Schnelldruckerzeile (132) ab. Jede Druckposition wird in den entsprechenden Wert $\mathrm{Re}(z)$ (oder $\mathrm{Im}(z)$ umgerechnet.

Verzögertes Lesen **E4**

Gesucht ist ein Programm mit dessen Hilfe man eine Zeichenkette vom Terminal in die Variable `ALINE` (`CHARACTER*80`) einlesen kann, in der neben Worten auch die Werte von reellen Größen vorkommen sollen:

```
DIE ERSTE ZAHL IST 537.75 UND DIE ZWEITE 462.25 UND JETZT IST SCHLUSS
```

Die Zeichenkette muß also in Hinblick auf enthaltene Ziffern analysiert werden, und wenn die Ziffern gefunden sind, können sie durch eine interne Leseanweisung in die zugehörigen `REAL`-Variablen übertragen werden.

Hinweis: Es wird vorteilhaft sein, ein `SUBROUTINE`-Unterprogramm zu schreiben, das ab einer gegebenen Position die Lage eines beliebigen Zeichens (etwa eines Leerzeichens) innerhalb der Zeichenkette `ALINE` feststellt.

ææ