4. Bundeswettbewerb Informatik 1985

Das Aufgabenblatt

Aufgabe 1:
Figuren nachprogrammieren

Die Abbildungen 1a, 1b und 1c zeigen drei Figuren, die nach dem gleichen Prinzip, aber mit unterschiedlichen Werten zweier Parameter WINKEL und ANZAHL entstanden sind. Schreiben Sie ein Programm, das die abgebildeten und weitere Figuren dieser Art für beliebige ganzzahlige Werte von WINKEL und ANZAHL erzeugen kann. Legen Sie dem Programm bitte die Figur bei, die es bei WINKEL = 70° und ANZAHL = 9 erzeugt (Plot, Hardcopy, Fotografie oder Handzeichnung).

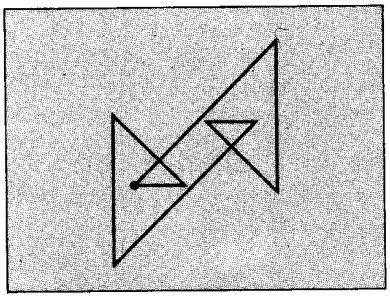


Abb. 1a Winkel: 135° · Anzahl: 4

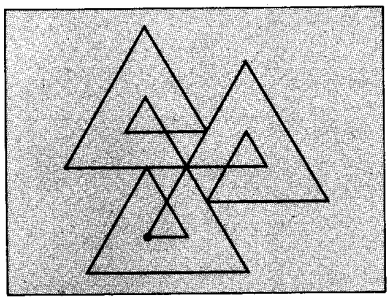


Abb. 1b Winkel: 120° · Anzahl: 5

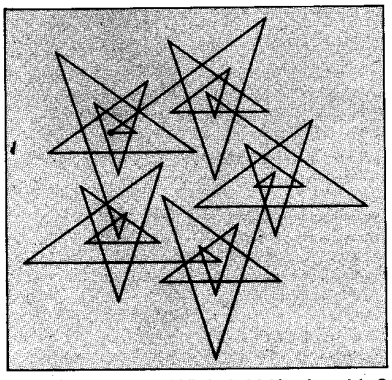


Abb. 1c Winkel: 144° · Anzahl: 8

Aufgabe 2:
Entschlüsseln einer Nachricht

Die verschlüsselte Nachricht

SLES SLEZSIES
VSRNCHIUSNNSITS ZSLIS LNT
NCHWSR ZU SETNCHIUSNNSIE

soll entschlüsselt werden.

Über den Text und seine äußerst einfache Verschlüsselung ist bekannt:

a) Es handelt sich um deutschen Text.

b) Umlaute und scharfes S sind geschrieben: AE, OE, UE, SS, statt Ä, Ö, Ü, ß.

c) Die Nachricht enthält nur Großbuchstaben und Leerzeichen.

d) Die fünf Buchstaben, welche in der Nachricht am häufigsten vorkommen, sind untereinander vertauscht (permutiert).

e) Alle anderen Buchstaben sind unverschlüsselt und stehen an der richtigen Position.

Beispiel:
Im Wort BUNDESHAUPTSTADT kommen die Buchstaben A, D, S, T und U am häufigsten vor. Die willkürliche Vertauschung A→T, D→U, S→D, T→A und U→S ergibt die Verschlüsselung BSNUEDHTSPADATUA.

Schreiben Sie ein Programm für folgenden Zweck: Der Benutzer soll die Entschlüsselung durch Experimentieren herausfinden. Das Programm soll ihn dabei unterstützen.

Aufgabe 3:
Steuerung eines Roboterarms

Die Spitze eines Roboterarms soll von einem Punkt P_1 des dreidimensionalen Raums zu einem Punkt P_2 geführt werden. Der Roboterarm besteht aus drei Teilarmen (je 180 cm lang) und drei Drehgelenken. Das Drehgelenk G_1 kann horizontal auf einen Winkel zwischen 0 und 90 Grad eingestellt werden. Die Drehgelenke G_2 und G_3 können vertikal jeweils auf einen Winkel zwischen 0 und 270 Grad für G_2 beziehungsweise zwischen 0 und 180 Grad für G_3 eingestellt werden (Abbildung 2).

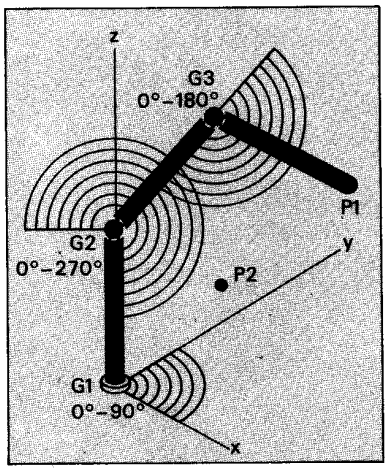


Abb. 2

Die Steuerung des Roboterarms erfolgt durch eine Folge von Befehlen der Form TWIST (i, j, k) mit i, j, k ∈ {–1°, 0°, +1°}. Zum Beispiel bewirkt TWIST (0°, –1°, +1°), daß der Winkel des Gelenks G_1 unverändert bleibt, während der Winkel von G_2 um 1 Grad vermindert und der von G_3 um 1 Grad vergrößert werden (sofern die Winkelgrenzen dabei nicht überschritten werden – sonst wird das entsprechende Gelenk nicht bewegt).

a) Schreiben Sie ein Programm, das die in cm gemessenen Koordinaten (x_1, y_1, z_1) und (x_2, y_2, z_2) zweier Punkte P_1 und P_2 einliest und darauf eine Folge von TWIST-Befehlen ausgibt, durch welche die Spitze des Roboterarms von P_1 möglichst nahe an P_2 heran bewegt wird. Legen Sie Ihrer Lösung mehrere Programm-Protokolle bei, eines davon mit $P_1 = (100, 0, 0)$ und $P_2 = (100, 100, 0)$.

b) Für Jugendliche, die nicht mehr der Sekundarstufe I angehören: Schreiben Sie das Programm so, daß die Spitze des Roboterarms durch die Folge der TWIST-Befehle möglichst gut entlang der Geraden von P_1 nach P_2 bewegt wird.

Aufgabe 4:
Aktienkurs-Analyse

Ein großes Wirtschaftsmagazin will seinen Lesern eine Analyse der Börsenentwicklung der letzten fünf Jahre präsentieren. Dazu sollen unter anderem die Kurse der wichtigsten Aktien in diesem Zeitraum untersucht werden. Für jede Aktie soll nachträglich ein bester Einkaufstag festgestellt werden.

Dabei wird angenommen, daß ein Kapitalanleger jede Aktie höchstens einmal eingekauft hätte, und zwar in einer beliebigen Stückzahl, und daß er zum Ende des betrachteten Zeitraums alle Stücke wieder verkauft hat. Der beste Einkaufstag für eine Aktie wäre dann derjenige gewesen, der zu einem eingesetzten Betrag den höchsten Gewinn geliefert hätte (Steuern, Gebühren und alternative Anlagemöglichkeiten sollen außer Betracht bleiben).

Das Wirtschaftsmagazin hat von einem Börsendienst Informationen über die Notierungen jeder Aktie für alle Börsentage der letzten fünf Jahre gekauft. Für jede Aktie erhält es eine Zahlenfolge. Die erste Zahl ist der Kurs der Aktie am ersten Börsentag und jede folgende Zahl gibt die absolute Kursveränderung gegenüber dem Vortag an, in der Reihenfolge der Börsentage. Der Kurs, der sich für einen gewissen Tag ergibt, gilt für alle Käufe und Verkäufe dieses Tages.

Unterstützen Sie die Börsenanalyse, indem Sie ein Programm schreiben, das für eine Aktie aus der gegebenen Zahlenfolge nachträglich einen besten Einkaufstag, einen besten Verkaufstag und den dabei höchsten erzielbaren Gewinn (in Prozent vom eingesetzten Betrag) ermittelt. Da Ihr Programm sehr lange Zahlenfolgen verarbeiten muß, ist es außerordentlich wichtig, daß die Laufzeit bei zunehmender Zahlenfolgenlänge nicht stärker als nötig wächst.

Beispiel:

Eingabe
127,5 –0,5 +2 –1 +1 +3,5 –13 +7 –2
–6 –9 –21 –17 –5 +0,5 +4 –7 –12 +2,5
–3 +2

Ausgabe
„Ein bester Einkaufstag wäre der 14. Börsentag gewesen, ein dazugehöriger Verkaufstag der 16. Börsentag. Der so realisierbare Gewinn wäre 6.7669% vom eingesetzten Betrag gewesen.

Aufgabe 5:
Simulation von Widerstandsnetzen

Ein elektrisches Widerstandsnetz der gezeigten Art (Abbildung 3) habe folgende Eigenschaften:

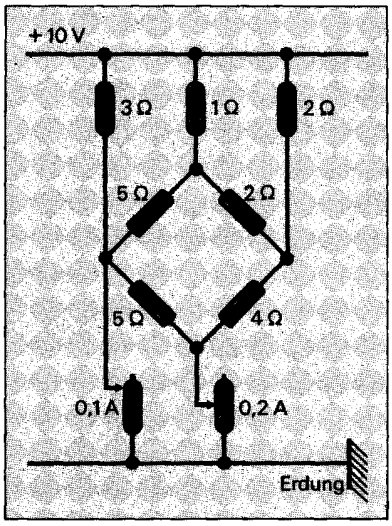


Abb. 3

a) Das Netz ist in der Ebene überschneidungsfrei.

b) An seinen Knoten sind im allgemeinen Stromstärken für Verbraucherabflüsse durch Potentiometereinstellungen definiert.

c) Mehrere Einspeisungen sind möglich, ebenso verschiedene Widerstandswerte.

d) Maschen können mehr als drei Leitungen enthalten.

Schreiben Sie ein Programm, das Ihnen alle in solchen Netzen fließenden Ströme und alle an den Knoten vorhandenen Potentiale in Tabellenform ausgibt (ein Fremdprogramm zur Lösung linearer Gleichungssysteme darf als Werkzeug benutzt werden). Legen Sie als Testfall Ihrem Programm die Lösung des Beispiels in Abbildung 3 bei.

Allgemeine Hinweise

Ihre Lösung sollte im Prinzip aus den folgenden Teilen bestehen:

Lösungsidee:
Eine Beschreibung der Lösungsidee. Die Form und die Begriffe der Lösungsidee müssen sich im Programm wiederfinden.

Programm-Dokumentation:
Eine Beschreibung des Programms. Hinweise auf Besonderheiten und Nutzungsgrenzen. Angaben zu Rechenzeit und Speicherbedarf.

Programm-Protokoll:
Kommentierte Probeläufe des Programms. Manchmal genügt ein Protokoll.

Programm-Liste:
Das Programm selbst in einer der gängigen Programmiersprachen. Keinen Assembler benutzen. Die Liste sollte eine Zeilennumerierung haben.

Ihre Lösung wird danach bewertet
- ob sie vollständig und richtig ist,
- ob die Ausarbeitung strukturiert und nachvollziehbar ist,
- ob die Programmunterlagen übersichtlich und lesbar sind.

Bitte reichen Sie die Lösung auf einseitig schwarz beschriebenen oder bedruckten DIN A4-Blättern ein. Alle Blätter sind rechts oben durchzunumerieren und mit Ihrem vollständigen Namen zu versehen. Die Einsendungen müssen in deutscher Sprache abgefaßt sein. Senden Sie keine Disketten ein.

Füllen Sie bitte das Begleitformular (die abtrennbare Klappe am Aufgabenblatt oder eine Kopie davon) vollständig aus. Die statistischen Angaben haben keinen Einfluß auf die Bewertung.

Stecken Sie alle Lösungsblätter und das Begleitformular in einen DIN A4-Umschlag mit der Adresse:

Bundeswettbewerb Informatik 1985
GMD – Schloß Birlinghoven
5205 Sankt Augustin 1

Einsendeschluß ist der 15. November 1985 (Datum des Poststempels gilt). Verspätete Einsendungen werden nicht geöffnet. Der Rechtsweg ist ausgeschlossen.

Die Einsendungen werden nicht an die Teilnehmer zurückgegeben. Mit der Einsendung wird den Veranstaltern des Wettbewerbs das Recht übertragen, die Beiträge in geeigneter Form zu veröffentlichen.

Teilnehmen können Jugendliche, die nach dem 15. November 1963 geboren wurden. Sie dürfen jedoch im Sommersemester 1985 noch nicht studiert oder vor dem 1. September 1985 ihre Ausbildung beendet und einen Beruf ergriffen haben. (Näheres siehe unter „Teilnahmeberechtigung".)

Musteraufgabe: Ein Rangierproblem

a) Auf dem Gleis E stehen n Lokomotiven, die von 1 bis n durchnumeriert sind. Folgende Rangieroperationen sind erlaubt:

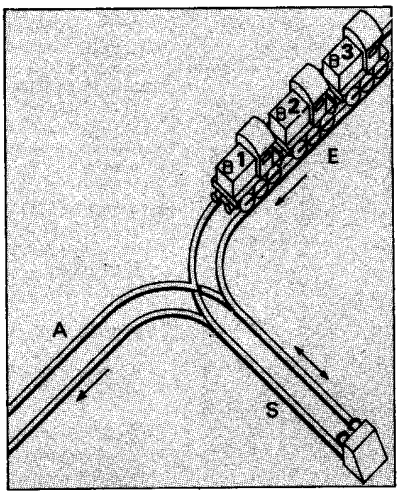
1. Fahren von Gleis E auf Gleis S,
2. Fahren von Gleis S auf Gleis A.

Von Gleis S darf also nicht auf Gleis E gefahren werden.

Nach genau 2n Rangieroperationen ist das Gleis E leer und die Loks stehen in einer gewissen Anordnung auf Gleis A.

Schreiben Sie ein Programm, das zu gegebenem n alle möglichen Anordnungen der Loks auf Gleis A ausgibt, sowie deren Anzahl a(n) bestimmt.

Beispiel:
Bei n=3 Loks sind die Anordnungen 321, 231, 213, 132 und 123 möglich; also ist a(3)=5.



b) Es sei b(n) die Anzahl aller aus je n öffnenden und n schließenden Klammern korrekt gebildeten Klammerausdrücke.

Beispiel:
(()()), ((()())) und (()()()) sind korrekt gebildet, ())(() und)((()) dagegen nicht.

Schreiben Sie ein Programm, das zu gegebenem n alle korrekt gebildeten Klammerausdrücke dieser Art ausgibt und deren Anzahl b(n) bestimmt. Zur Kontrolle: Es ist b(4)=14.

c) Welche Beziehung besteht zwischen den Teilaufgaben a) und b)?

Musterprogramm: Ein Rangierproblem

```
1   PROGRAM  Rangierproblem;
2     (* Bundeswettbewerb Informatik 1985, Musteraufgabe *)
3     VAR
4       LokAnzahl, KlammerAnzahl, Nummer :  integer ;
5       Antwort                          :  char    ;
6
7     PROCEDURE  Interpretiere( Rangierfolge : string);
8       (* Interpretiert einen Klammerausdruck als Folge von Rangieroperationen *)
9       VAR
10        Stelle                         :  integer ;
11        Gleis-E, Gleis-S, Gleis-A, Lok :  string  ;
12      BEGIN
13        Gleis-E := '123456789'; Gleis-S := ''; Gleis-A:= '';
14        FOR  Stelle:= 1 TO  length(Rangierfolge) DO
15          CASE Rangierfolge[Stelle] OF
16            '(' :  BEGIN
17                     Lok     :=  copy   (Gleis-E, 1, 1);
18                                 delete (Gleis-E, 1, 1);
19                     Gleis-S :=  concat (Gleis-S, Lok);
20                   END;
21            ')' :  BEGIN
22                     Lok     :=  copy   (Gleis-S,  length(Gleis-S), 1);
23                                 delete (Gleis-S,  length(Gleis-S), 1);
24                     Gleis-A :=  concat (Gleis-A,  Lok)
25                   END
26          END (* of CASE *)
27        writeln( Gleis-A: LokAnzahl+5)
28      END; (* of Interpretiere *)
29
30    PROCEDURE Erzeuge( Ausdruck : string; Oeffnende, Schliessende : integer);
31      (* Erzeugt rekursiv alle korrekten Klammerausdruecke *)
32      BEGIN
33        IF  Oeffnende + Schliessende = KlammerAnzahl THEN
34          BEGIN  Nummer := Nummer+1;
35            write( Nummer : 6, Ausdruck : Klammeranzahl+5);
36            Interpretiere( Ausdruck)
37          END; (* of THEN *)
38        IF  Oeffnende < KlammerAnzahl DIV 2  THEN
39            Erzeuge( concat( Ausdruck, '('), Oeffnende+1, Schliessende);
40        IF  Schliessende < Oeffnende         THEN
41            Erzeuge( concat( Ausdruck, ')'), Oeffnende  , Schliessende+1);
42      END; (* of Erzeuge *)
43
44
45    BEGIN (* Hauptprogramm *)
46      page(output); writeln;
47      writeln ( 'Rangierproblem');
48      writeln ( '--------------');
49
50      REPEAT
51        writeln; writeln; write( 'Anzahl der Lokomotiven? (1..9) ');
52        readln( LokAnzahl); writeln;
53
54        Nummer := 0; KlammerAnzahl := 2 * LokAnzahl;
55
56        Erzeuge( '', 0, 0);
57
58        REPEAT
59          writeln; write( 'Noch einmal? (j/n) '); read( Antwort)
60        UNTIL Antwort IN [ 'j', 'J', 'n', 'N']
61
62      UNTIL Antwort IN [ 'n', 'N']
63    END.
```

Musterlösung: Ein Rangierproblem

Lösungsidee

Ich bezeichne die erste Rangieroperation im Aufgabentext mit E und die zweite Operation mit A. Dann wird beispielsweise die Anordnung „132" der Lokomotiven auf Gleis A durch die Rangierfolge EAEEAA erzeugt. Jede mögliche Anordung kann durch eine geeignete Rangierfolge erzeugt werden, und zu verschiedenen Rangierfolgen gehören verschiedene Anordnungen der Lokomotiven.

Welche Gestalt haben nun die Rangierfolgen? Zunächst ist klar, daß sie aus je n der Operationen E und A bestehen. Denn aus Gleis S kann man nicht Lokomotiven herausfahren, die nicht zuvor hereingefahren wurden. Damit sind die zulässigen Rangierfolgen offensichtlich durch folgende Bedingungen charakterisiert: 1. Es gibt gleich viele E-Operationen wie A-Operationen, 2. In jeder von links beginnenden Rangierfolge müssen immer mindestens so viele E-Operationen wie A-Operationen vorkommen. Im Fall n=5 sind das beispielsweise die fünf Rangierfolgen EEEAAA, EEAEAA, EEAAEA, EAEEAA, EAEAEA.

Nach dem, was ich im Algebra-Unterricht über korrektes Klammern gelernt habe, gelten folgende Regeln: 1. Es gibt gleich viele öffnende wie schließende Klammern, 2. Baut man einen Klammerausdruck von links her auf, müssen immer mindestens so viele öffnende wie schließende Klammern vorkommen.

Lösung von c): Faßt man jede öffnende Klammer als E-Operation und jede schließende Klammer als A-Operation auf, erhält man eine umkehrbar eindeutige Zuordnung zwischen den korrekt gebildeten Klammerausdrücken und den zulässigen Rangierfolgen. Da die Rangierfolgen aber auch umkehrbar eindeutig den verschiedenen möglichen Anordnungen der Lokomotiven auf Gleis A entsprechen, ist für die zugehörigen Anzahlen nunmehr bewiesen: a(n)=b(n) für n=1, 2, 3,...

Zur Lösung der Teilaufgaben a) und b) benötige ich also nur ein Programm!

Programm-Dokumentation

Mein Programm ist in UCSD-Pascal geschrieben und arbeitet folgendermaßen: Nach Eingabe der Anzahl der Lokomotiven (Programmzeile 52) wird diese zur Anzahl der Klammern verdoppelt (Zeile 54); dann wird die Prozedur *Erzeuge* aufgerufen (Zeile 56), die alle korrekten Klammerausdrücke rekursiv erzeugt (ab Zeile 30).

Die zweite Bedingung für die Korrektheit eines Klammerausdrucks ist in Zeile 40 realisiert. Eine schließende Klammer wird nur dann angefügt, wenn die Anzahl *schließender* Klammern kleiner als die *öffnender* Klammern ist. Die erste Bedingung findet sich in den Zeilen 38 und 33. Eine öffnende Klammer wird nur dann angefügt, wenn die Anzahl der *öffnenden* Klammern kleiner als die *halbe Klammeranzahl* ist (Zeile 38); ist die Summe der *öffnenden* und *schließenden* Klammern gleich der *Klammeranzahl* insgesamt (Zeile 33), wird der Klammerausdruck am Bildschirm ausgegeben und als Folge von Rangieroperationen interpretiert (Zeile 36).

Dies erledigt die Prozedur *Interpretiere* (ab Zeile 7). Die drei Gleise sind im Programm Zeichenketten (Zeile 11). Zu Beginn stehen auf *Gleis E* neun Lokomotiven; *Gleis S* und *Gleis A* sind leer (Zeile 13). Nun wird der gegebene Klammerausdruck, als *Rangierfolge* aufgefaßt, Zeichen für Zeichen verarbeitet (Zeile 14); ist das Zeichen eine öffnende Klammer (Zeile 16), so wird die vorderste Lokomotive vom *Gleis E* weggenommen (Zeilen 17, 18) und auf *Gleis S* gebracht (Zeile 19). Handelt es sich um eine schließende Klammer (Zeile 21), wird die zuletzt hereingefahrene Lokomotive vom *Gleis S* genommen (Zeilen 22, 23) und auf das *Gleis A* gebracht (Zeile 24). Schließlich wird der Inhalt von *Gleis A* ausgegeben (Zeile 27).

Die Eingabe der Anzahl der Lokomotiven habe ich nicht extra abgesichert, der Benutzer kann also auch Zahlen n>9 eingeben. In diesem Fall arbeitet die Prozedur *Interpretiere* (wegen Zeile 13) nicht mehr korrekt, wohl aber die Prozedur *Erzeuge*. Vorsicht: Für n>9 werden die Laufzeiten sehr lang!

Programm-Protokoll (hier aus Platzgründen nur verkürzt)

Das Programm lieferte die Anzahlen:
a(1)=1, a(2)=2, a(3)=5, a(4)=14, a(5)=42, a(6)=132, a(7)=429, a(8)=1430 und a(9)=4862.

Für den Fall n=3 gab es beispielsweise die folgenden fünf Klammerausdrücke aus:
((())), (()()), (())(), ()((()), ()()().

Bundeswettbewerb Informatik:
Eine gemeinsame Initiative von GMD und GI.

Design: KUHNS KÖRNER Kürten/Köln · Titelillustration: Uwe Behrens