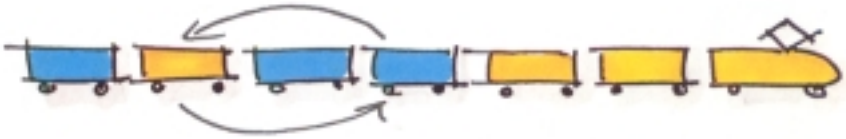
Informatik...

...ist Banane.

nein

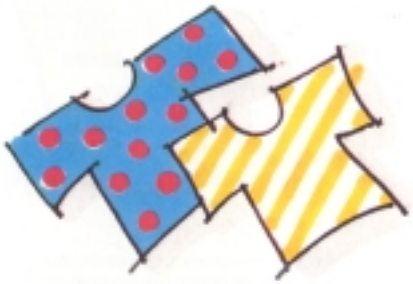
ja

...macht Autos rund.



...regelt einiges.

...verrät wie es morgen wird.

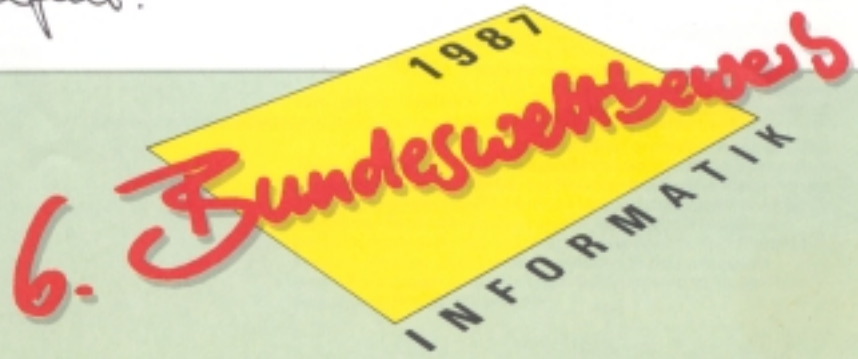


...schafft Vielfalt.

...macht Gleiches

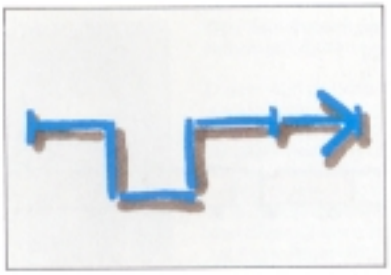
gleich gut.

Das Aufgabenblatt



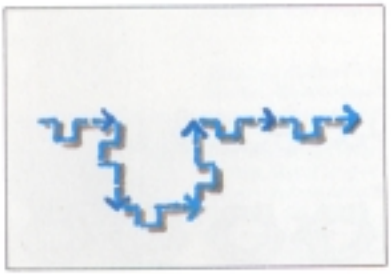
Aufgabe 1:
Tanz der Schildkröte

Die Schildkröte hat einen neuen Tanz erfunden. Das Grundmuster besteht aus gleich langen Schritten, die vorwärts oder seitwärts ausgeführt werden können. Es enhält genauso viele Seitschritte nach rechts wie nach links. Das Grundmuster „vor-rechts-vor-links-vor-vor" sieht zum Beispiel so aus:



Nach jedem Grundmuster ändert die Schildkröte die Tanzrichtung um einen immer gleichen Winkel, der ein Teiler von 360 Grad ist, nach links. Dann beginnt sie, ein neues Grundmuster zu tanzen. Dies tut sie so lange, bis sie den Startpunkt wieder erreicht hat. Bei einem Winkel von 72 Grad wäre das beispielsweise nach fünf Grundmustern. Damit ist der erste Umlauf fertig.

Beim zweiten Umlauf tanzt die Schildkröte über jeder Strecke des ersten Umlaufs ein komplettes Grundmuster, natürlich mit entsprechend zierlicheren Schritten. Im Beispiel:



Beim dritten Umlauf tanzt die Schildkröte über jeder Strecke des zweiten Umlaufs ein komplettes Grundmuster, beim vierten Umlauf über jeder Strecke des dritten, usw.

Aufgabe:

Schreibe ein Programm, das als Eingabe ein Grundmuster, einen Tanzrichtungsänderungswinkel und eine Umlaufnummer erwartet, und anschließend den Tanz der Schildkröte in diesem Umlauf zeichnet.

Schicke uns mindestens fünf Tänze mit mehreren Umläufen. Eines der Beispiele sei: „vor-rechts-vor-links-vor-vor", 72 Grad, dritter Umlauf.

Aufgabe 2:
Computer-Kurzschrift

Tiro schreibt auf seinem kleinen Computer, den ihm Cicero geschenkt hat, viele deutsche Texte und speichert sie als Dateien auf Disketten ab. Da Tiro ein armer Sklave ist, muß er mit dem Speicherplatz sehr sparsam umgehen. Disketten mit 18 KB kosten leider mindestens 18 Sesterzen. Daher überlegt er sich 18 Kürzel für die in deutschen Texten häufigen Wörter. Er meint, wenn er ein Programm hätte, das eine Volltextdatei und eine Kurzschriftdatei entsprechend ineinander umsetzte, könnte er seine Texte platzsparender abspeichern. Cicero, dem er davon erzählt, stimmt dem zu und meint: „Wie wäre es, wenn das Programm sich für jeden einzelnen Text zusätzlich noch weitere 18 Kürzel für solche Wörter berechnet, die dort besonders oft vorkommen oder besonders lang sind? Das würde noch weiteren Platzgewinn bedeuten!" Leider kann Tiro nicht programmieren.

Aufgabe:

Stelle eine Liste von 18 sehr häufigen deutschen Wörtern zusammen, deren Abkürzung Platzgewinn verspricht, und schreibe für Tiro drei Programme:

1. Das Programm „Textuntersuchung" soll eine Volltextdatei lesen und die 18 Wörter, die den größten Platzgewinn versprechen, wenn man dafür Kürzel einführt, in eine Kürzeldatei schreiben. Diese 18 Wörter sind im allgemeinen für jede Volltextdatei verschieden.
2. Das Programm „Kurzschrift" soll eine Volltextdatei in eine Kurzschriftdatei umsetzen, die alle Informationen zur Rekonstruktion der Volltextdatei enthält. Dazu benutzt es Deine Liste von 18 sehr häufigen deutschen Wörtern und die spezielle Kürzeldatei des Programms „Textuntersuchung". Die Volltextdatei und die Kürzeldatei werden anschließend gelöscht, und das Programm druckt den Platzgewinn in Byte aus.
3. Das Programm „Volltext" soll eine Kurzschriftdatei in die ursprüngliche Volltextdatei umsetzen, damit Tiro den Text wieder lesen kann.

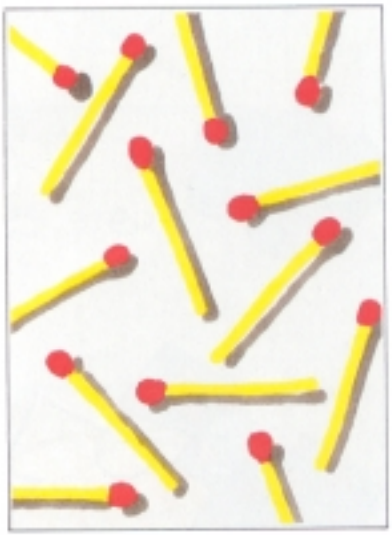
Du darfst annehmen, daß in den Texten das Dollarzeichen niemals vorkommt, weil man in der Tiro bekannten Welt mit Sesterzen bezahlte.

Schicke uns mindestens fünf Beispiele mit Volltext, Kurzschrift und Platzgewinn. Ein Beispiel sei genau der Text dieser Aufgabe – von der Überschrift „Computer-Kurzschrift" bis zum nun folgenden letzten Punkt.

Aufgabe 3:
Streichholzspiel

Am Streichholzspiel nehmen zwei Spieler teil. Zu Beginn liegen dreizehn Streichhölzer auf dem Tisch. Die Spieler nehmen abwechselnd Streichhölzer weg, mindestens eins und höchstens drei. Wer das letzte Streichholz nehmen muß, der hat verloren.

Das Streichholzspiel kann aber auch mit beliebig vielen Streichhölzern gespielt werden. Die Mitspieler einigen sich dann vor dem Spiel, mit wievielen Streichhölzern sie spielen wollen und wieviele Streichhölzer in einem Zug höchstens weggenommen werden dürfen.



Aufgabe:

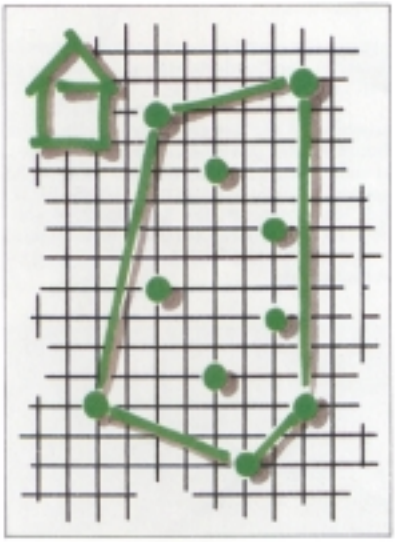
Schreibe ein Programm, so daß Du das Streichholzspiel gegen Deinen Computer spielen kannst. Eingabe für das Programm sind die Gesamtzahl der Streichhölzer zu Beginn und die Höchstzahl der Streichhölzer, die pro Zug weggenommen werden dürfen. Das Programm soll versuchen, zu gewinnen.

Schicke uns mindestens fünf Spielprotokolle mit verschiedenen Eingabewerten.

Aufgabe 4:
Goldgräberclaim

Ein Goldgräber hat nach Streifzügen durch das weite, unerforschte Land einige vielversprechende Schürfplätze entdeckt. Diese Plätze hat er auf einer geheimgehaltenen Karte vermerkt.

Bevor er jedoch mit seinen Grabungen beginnen kann, muß er das von ihm beanspruchte Gebiet, seinem Claim, nach alter Goldgräbersitte abstecken. Dazu will er einen Zaun errichten, der den Claim umschließt. Weil das Material teuer ist, soll der Zaun möglichst kurz sein. Zum Beispiel ist der kürzeste Zaun um den in der Abbildung gezeigten Claim aus zehn Schürfplätzen 34,51 Rastereinheiten lang.



Aufgabe:

Schreibe ein Programm, das nach Eingabe beliebig vieler Schürfplätze die Streckenbeschreibung und die Länge des kürzesten Claim-Zaunes ermittelt und ausgibt. Jeder Schürfplatz wird durch ein Paar (x,y) ganzzahliger Koordinaten angegeben. Zur Vereinfachung kannst Du davon ausgehen, daß ein Schürfplatz keine Ausdehnung besitzt und auch dann schon zum Claim gehört, wenn er genau unter dem Zaun zu liegen kommt.

Schicke uns mindestens fünf Zaunberechnungen. Einer der Zäune sei für einen Claim mit folgenden Schürfplätzen: (8,17) (9,16) (12,18) (9,19) (10,17) (7,16) (9,15).

Aufgabe 5:
Ampel im Streß

Die Ampel an einer Hauptverkehrsstraße soll nach den Ergebnissen einer Simulation so eingestellt werden, daß möglichst geringe Wartezeiten entstehen. Eine Verkehrsanalyse ergibt:

1. Auf der Hauptstraße kommen pro Sekunde entweder kein, ein, zwei oder drei Wagen mit gleicher Wahrscheinlichkeit an. Bei Grün passieren pro Sekunde mit gleicher Wahrscheinlichkeit zwei bis sieben Wagen die Ampel.
2. Ein Ampelzyklus besteht aus den Phasen rot, rotgelb, grün, gelb und dauert 60 Sekunden, davon fünf Sekunden gelb und fünf Sekunden rotgelb.

Aufgabe:

Schreibe ein interaktives Programm, das die kürzeste Grünphase ermittelt, für welche die Warteschlange vor der Ampel auf lange Sicht 72 wartende Autos nicht überschreitet. Gib die durch Dein Programm ermittelten Zeiten für folgende drei Fälle an und protokolliere dabei den zeitlichen Verlauf der Warteschlangenlänge während fünf stabiler Zyklen:

a) Normaler Verkehr wie oben beschrieben.

b) Simulation einer Spitzenzeit, bei der die Wahrscheinlichkeit
für 3 pro Sekunde ankommende Wagen 40%,
für 4 pro Sekunde ankommende Wagen 30%,
für 5 pro Sekunde ankommende Wagen 20%,
für 6 pro Sekunde ankommende Wagen 10% beträgt.

c) In der Spitzenzeit wird der Verkehr nicht durch eine starre, sondern durch eine flexible Grünphase gesteuert. Dazu sind am Straßenrand Sensoren montiert, welche der Ampelanlage melden, wann weniger als 12 Autos und wann mehr als 72 Autos warten. Gib zur Lösung an, in welchem Bereich jetzt die Grünphase variiert.

Allgemeine Hinweise

Alle Einsendungen sollten aus den folgenden Teilen bestehen,
getrennt nach Aufgaben:

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 Rechenzeiten und Speicherbedarf.

Programmablauf-Protokoll:
Kommentierte Probeläufe des Programms. Manchmal genügt ein Protokoll, manchmal müssen es mehrere sein.

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

Deine Einsendungen werden danach bewertet
- ob sie vollständig und richtig sind,
- ob die Ausarbeitungen gut strukturiert und verständlich sind,
- ob die Programmunterlagen übersichtlich und lesbar sind.

Bitte schicke Deine Ergebnisse in einem Exemplar, auf einseitig beschriebenen oder bedruckten DIN-A4-Blättern. Alle Blätter sind rechts oben durchzunumerieren und mit Deinem vollständigen Namen zu versehen. Die Einsendungen müssen in deutscher Sprache abgefaßt sein. Sende uns keine Disketten ein.

Bitte fülle das Begleitformular (die abtrennbare Klappe am Aufgabenblatt oder eine Kopie davon) vollständig aus. Bei Gruppen muß jeder Teilnehmer ein Begleitformular ausfüllen und unterschreiben. Die statistischen Angaben haben keinen Einfluß auf die Bewertung.

Stecke das Exemplar Deiner Ergebnisse zusammen mit dem Begleitformular in einen DIN-A4-Umschlag mit der Adresse:

Bundeswettbewerb Informatik 1987
GMD-Schloß Birlinghoven
5205 Sankt Augustin 1

Einsendeschluß ist der 27. November 1987 (Datum des Poststempels gilt). Bei mehreren Einsendungen wird nur die vor Einsendeschluß zuletzt abgeschickte bewertet. 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 27. November 1965 geboren wurden. Sie dürfen jedoch im Sommersemester 1987 noch nicht studiert oder vor dem 1. September 1987 ihre Ausbildung beendet und einen Beruf ergriffen haben (näheres siehe unter „Teilnahmeberechtigung").

Musteraufgabe: Baum des Pythagoras

Der „Baum des Pythagoras" setzt sich aus lauter Quadraten zusammen, die so um rechtwinklige Dreiecke angeordnet sind, daß sie den Satz des Pythagoras illustrieren. Ihre Seitenverhältnisse sind 3 : 4 : 5. Die ersten Verästelungen des Baumes zeigt Abb. A.

Schreibe ein Programm, das den Baum so weit zeichnet, wie die Auflösung des Bildschirms es zuläßt.

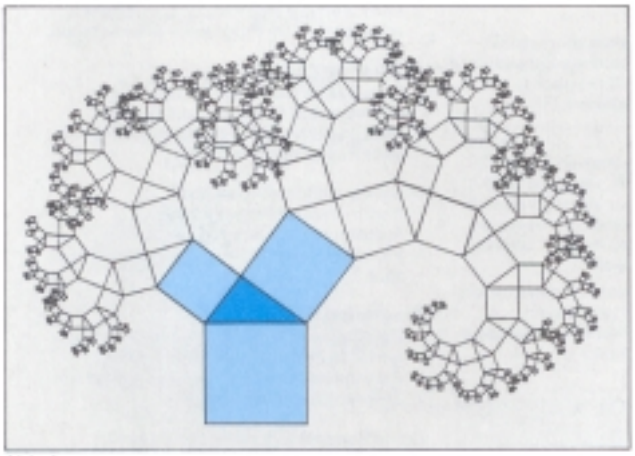


Abb. A

Musterprogramm: „Baum des Pythagoras"

```
1   PROGRAM Baum_des_Pythagoras;
2   (* 5. Bundeswettbewerb Informatik, 1. Runde *)
3
4   {$I GRAPH.P}
5
6   CONST  startpunkt_x                    =  0;
7          startpunkt_y                    =  -99;
8          Startrichtung                   =  north;
9          seitenlaenge_kleinstes_quadrat  =  2;
10
11  VAR    seitenlaenge_startquadrat : integer;
12
13  PROCEDURE   baum_zeichnen
14              (seitenlaenge_hypotenusenquadrat : integer);
15
16          VAR    quadratseitenzaehler,
17                 seitenlaenge_rechtes_kathetenquadrat,
18                 seitenlaenge_linkes_kathetenquadrat : integer;
19
20          BEGIN
21                 FOR quadratseitenzaehler := 1 TO 4 DO
22
23                 BEGIN
24                        forwd (seitenlaenge_hypotenusenquadrat);
25                        turnleft (90)
26                 END;
27
28                 seitenlaenge_rechtes_kathetenquadrat :=
29                 round (4/5 * seitenlaenge_hypotenusenquadrat);
30                 seitenlaenge_linkes_kathetenquadrat :=
31                 round (3/5 * seitenlaenge_hypotenusenquadrat);
32
33                 IF     seitenlaenge_linkes_kathetenquadrat >=
34                        seitenlaenge_kleinstes_quadrat
35                 THEN
36                 BEGIN  forwd (seitenlaenge_hypotenusenquadrat);
37                        turnright (36.87);
38
39                        baum_zeichnen( seitenlaenge_rechtes_kathetenquadrat);
40
41                        turnleft (36.87);
42                        turnleft (53.13);
43                        forwd (seitenlaenge_rechtes_kathetenquadrat);
44
45                        baum_zeichnen( seitenlaenge_linkes_kathetenquadrat);
46
47                        back ( seitenlaenge_rechtes_kathetenquadrat);
48                        turnright (53.13);
49                        back (seitenlaenge_hypotenusenquadrat)
50                 END
51          END;
52
53  BEGIN  writeln;    write ('Welche Seitenlaenge soll das Startquadrat haben?  ');
54                     read  (seitenlaenge_startquadrat);
55
56          graphmode;     setposition (startpunkt_x, startpunkt_y);
57                         setheading (startrichtung);
58
59          baum_zeichnen( seitenlaenge_startquadrat)
60  END.
```

Musterlösung: Nr. 1 vom Aufgabenblatt 1986

Lösungsidee

Da der Baum sich nur aus Quadraten zusammensetzt, die in konstanten Winkeln mit den Ecken aneinanderstoßen, bietet sich eine Programmlösung an, die auf der Turtlegrafik aufbaut.

Die zentrale Prozedur („baum_zeichnen") zeichnet ein Hypotenusenquadrat und ruft sich selbst zweimal mit den Vorbedingungen je eines der beiden Kathetenquadrate auf (doppelte Rekursion).

Programm-Dokumentation

Das Zeichnen beginnt mit dem größten Quadrat, dem Stamm des Baumes, dessen Seitenlänge der Benutzer eingeben kann.

Die Rekursionen brechen jeweils ab, wenn sie Quadrate zeichnen sollen, die nur noch aus einem Bildschirmpunkt bestehen (daher „seitenlaenge_kleinstes_quadrat = 2"). Dies tritt stets beim linken Kathetenquadrat zuerst auf und braucht deshalb nur dort getestet werden (Programmzeile 33).

Es ist logisch, daß die Rekursionen nach links mit weniger Schritten abbrechen müssen als nach rechts. Die Quadratseiten werden links schneller kleiner (Faktor = 3/5) als rechts (Faktor = 4/5).

Abb. B beschreibt die Bewegungen der Turtle für einen Durchlauf von „baum_zeichnen". Der Baum wächst von innen nach rechts außen und dann nach links. Damit er sich richtig entfaltet, muß die Turtle vor und nach den beiden Rekursionen (7, 11) die Schritte (1, 2, 3, 4, 5, 10, 12, 14) und die Drehungen (6, 8, 9, 13) vornehmen. Die Schritte 12, 14 und die Drehung 13 sind notwendig, damit die Turtle am Ende jedes Durchlaufs von „baum_zeichnen" wieder die gleiche Position und Richtung einnimmt wie zu Beginn des Durchlaufs.

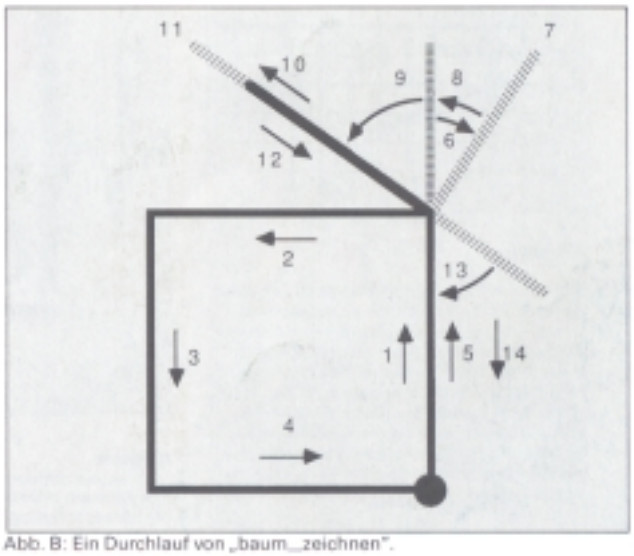


Abb. B: Ein Durchlauf von „baum_zeichnen".

Programmablauf-Protokoll

Sinnvolle Programmablauf-Protokolle sind bei dieser Aufgabe die gezeichneten Bäume selbst, mit Angabe des eingegebenen Wertes für „seitenlaenge_startquadrat" und der benötigten Zeichenzeit.

Programm-Text

Das Programm ist in Turbo-Pascal für einen Mikrocomputer mit Betriebssystem MS-DOS formuliert. In Programmzeile 4 werden die standardmäßig vorhandenen Funktionen der Turtlegrafik zugeladen.