Der 17. Bundeswettbewerb Informatik _ Das Aufgabenblatt

Terror on Cheese Planet

Allgemeine Hinweise:

Zu den Aufgaben sende uns folgendes:

Lösungsidee:
Eine Beschreibung der Lösungsidee. Die Idee und die zu ihrer Beschreibung verwendeten Begriffe müssen sich im Programm wiederfinden.
Programm-Dokumentation:
Eine Beschreibung, wie die Lösungsidee im Programm realisiert wurde. Hinweise auf Besonderheiten und Nutzungsgrenzen, verlangte Eingabeformate usw.
Programmablauf-Protokoll:
Kommentierte Probeläufe des Programms, d.h. wann wird welche Eingabe erwartet, was passiert dann, welche Ausgabe erscheint usw. Mehrere unterschiedliche Beispiele, die die Lösung der Aufgabe verdeutlichen. Bildschirm-Fotos sind auch zulässig.
Programm-Text:
Das kommentierte Programm selbst in einer der gängigen höheren Programmiersprachen wie z. B. Pascal. Maschinensprache ist nicht zulässig..

Einsendungen werden danach bewertet:

- ob sie vollständig und richtig sind,
- ob die Ausarbeitungen gut strukturiert und verständlich sind,
- ob die (Programm-) Unterlagen übersichtlich und lesbar sind.

Bitte schicke Deine Arbeit nach Aufgaben geordnet und geheftet auf einseitig bedrucktem DIN-A4-Papier. Endlospapier schneide bitte entsprechend zu. Numeriere alle Blätter rechts oben und versieh sie mit Deinem Namen. Die Texte sollen in Deutsch abgefaßt sein. Falls Du einige DIN-A4-Klarsichthüllen **mit Heftrand** zur Hand hast, stecke bitte jeweils alles, was zu einer Aufgabe gehört, in eine Sichthülle. Andernfalls loche die Blätter bitte.
Schicke uns außerdem die Programmtexte und die lauffähigen Programme auf einer 3.5“ DOS-Diskette. (Die Programmtexte müssen trotzdem noch ausgedruckt werden!)
Fülle das Begleitformular (Klappe des Aufgabenblattes oder eine Kopie davon) **vollständig** und **leserlich!** aus. Insbesondere das Geburtsdatum muß angegeben sein, denn sonst kann die Einsendung nicht korrigiert werden. Bei Gruppen muß jeder Teilnehmer ein Formular ausfüllen, **Gruppenmitglieder ohne Teilnahmeformular werden nicht gewertet!** Teilnehmer, die die Schule bereits verlassen haben, geben bei „Klassenstufe“ bitte an, was sie jetzt machen. Einsendungen per email oder nur auf Diskette ohne Ausdruck können wir leider nicht akzeptieren.

Sende alles an:

Bundeswettbewerb Informatik
Ahrstraße 45, 53175 Bonn
Für Fragen zu den Aufgaben gibt es eine Hotline:
Jeweils von 17-19 Uhr
am 3. 9., 28. 9., 8. 10. und 6. 11.
Tel: 0228 / 37 86 46
oder email an: bwinf@gmd.de
oder schriftlich an die obige Adresse
bzw. per Fax an 0228 / 37 29 000.
Informationen (FAQ) gibt’s auch im Internet:
http://www.bwinf.de

Einsendeschluß ist der 09.11.98

(Datum des Poststempels). Verspätete Einsendungen können nicht berücksichtigt werden. Der Rechtsweg ist ausgeschlossen. Die Einsendungen werden nicht zurückgegeben. Der Veranstalter erhält das Recht, die Beiträge in geeigneter Form zu veröffentlichen. Wer wissen möchte, ob seine Einsendung angekommen ist, kann eine an ihn selbst adressierte und frankierte Postkarte mitschicken.

Unter der Schirmherrschaft des Bundespräsidenten

Träger des Wettbewerbs:
Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie,
Gesellschaft für Informatik e.V. (GI),
GMD-Forschungszentrum Informationstechnik GmbH

Musteraufgabe:

Sechseckfärbung

In einem sechseckigen Feld von dicht gepackten Kreisen sollen möglichst viele Kreise gefärbt werden, allerdings so, daß auf jeder zu einer Sechseckseite parallelen Reihe höchstens ein Kreis eingefärbt ist.

Die Abbildung zeigt ein Feld mit Kantenlänge 3 und einem rot eingefärbten Kreis. Die gelben Kreise geben die Reihen an, auf die der rote Kreis ausstrahlt. In diesen Reihen können also keine weiteren Kreise gefärbt werden.

Aufgabe:
Schreibe ein Programm, welches die Kantenlänge des Feldes einliest und dazu die maximale Anzahl der Kreise bestimmt und einschließlich ihrer Position ausgibt, die auf diesem Feld rot eingefärbt werden können, ohne daß eine zu einer Sechseckseite parallele Reihe mehr als einen roten Kreis enthält.

▶ Sende uns mindestens drei Beispiele mit unterschiedlichen Kantenlängen.

(aus Platzgründen entfallen die Programmdokumentation und die Bildschirmausgabe)

Lösungsidee:
Zunächst muß das Sechseck konstruiert werden; die einzelnen Punkte des Sechsecks werden als Objekte implementiert und kennen sowohl ihre Koordinaten (relativ zum mittleresten Punkt, der die Koordinaten 0,0 erhält) als auch ihre Nachbarn. Das heißt, die Punkte werden sowohl gemäß ihren Positionen in einer Matrix abgelegt, was bei der Konstruktion und Ausgabe des Sechsecks nützlich ist, als auch untereinander mit Zeigern verbunden, was die Färbung entlang der Sechseckkanten sehr einfach macht.

Um dann die eigentliche Lösung zu finden, werden die Steine mit Hilfe eines Backtracking-Algorithmus auf freie Felder plaziert. Das Programm beginnt mit der Plazierung eines Steines auf dem Feld 0.0. Nach der Plazierung werden die Felder in der gleichen Reihe und den Diagonalen markiert (gefärbt). Nun wird versucht, auf gleiche Weise weitere Steine zu setzen. Voraussetzung dafür ist, daß mindestens ein unmarkiertes Feld vorhanden ist. Ist dies nicht der Fall, werden der zuletzt gesetzte Stein und die durch ihn verursachten Markierungen entfernt und der Stein, wenn möglich, anders plaziert.
Das Programm numeriert die Felder bei ihrer Erzeugung durch und verwendet diese Nummern im weiteren Verlauf bei der Reihenfolge, in der Felder besetzt werden. Prinzipiell ist die die Reihenfolge, in der die Felder besetzt werden, aber beliebig, sie hat keinen Einfluß auf die maximal setzbare Anzahl Steine.
Die Steine werden auf die Felder in der Reihenfolge wachsender Nummern gesetzt. Dies reicht aus, alle übrigen möglichen Reihenfolgen (absteigend, alternierend) werden nicht ausprobiert.
Das Programm bricht ab, wenn es alle möglichen Positionierungen gepruft oder 2 * Anzahl Kantenlänge - 1 Steine plaziert hat. (Das ist die maximal mögliche Zahl.)

(Alternative Lösungsidee: Die Felder werden nicht explizit markiert, sondern man prüft beim Setzen eines Steins, ob in den entsprechenden Zeilen und Diagonalen bereits Steine gesetzt sind.)

Halbformale Programmbeschreibung:
Lies die Kantenlänge ein
Erzeuge den mittleren Punkt des Sechsecks
{rekursiver Aufruf: Wenn ein Punkt erzeugt
wird, werden auch alle neuen
Nachbarpunkte, die noch im Sechseck
liegen, erzeugt}

wiederhole:
Ist die Anzahl der gesetzten Steine größer als
die bisher beste Lösung?
Wenn ja, übernimm dies als neue Lösung
und gib sie aus
{ nicht unbedingt erforderlich }
Für alle Felder:
{ in der vorgegebenen Reihenfolge }
Könnte auf dieses Feld ein Stein gesetzt
werden?
Wenn ja:
Setze den Stein, markiere die Zeile
und Diagonalen
Rufe die Schleife { wiederhole }
rekursiv auf
Entferne den Stein und die von ihm
erzeugten Markierungen
bis alle Kombinationen getestet oder das
Maximum erreicht wurde.

Programmtext: (zur Abwechslung in C++)
// Sechseck.cpp

```
#include <stdio.h>
#include <stdlib.h>

class Sechseck;

class Punkt
{
  protected:
    int     iX, iY, iTyp, iZaehler;
    Punkt   *Rechts, *Links, *RechtsOben,
    *LinksUnten, *RechtsUnten, *LinksOben;

  void SetzeTyp(int iNeuerTyp);
  void EntferneTyp(int iNeuerTyp)

  public:
    Punkt(int X,int Y);

    void BildeVerbindungen(Sechseck *Basis);
    void SetzePunkt();
    void EntfernePunkt();
    int  LiefereTyp();
}
typedef Punkt* PunktZeiger;

class Sechseck
{
  protected:
    PunktZeiger* Punkte;
    PunktZeiger* PunktListe;
    int   iKantenLaenge, iBreite, iNrPunkte;

  public:
    Sechseck(int iKante);
    ~Sechseck();
    int   NrPunkte();
    Punkt* PunktNr(int iNr);
    Punkt* LieferePunkt(int iX,int iY);
    Punkt* LiefereOderErzeugePunkt(int iX,int iY);
}


Punkt::Punkt(int X,int Y)
{iX = X; iY = Y; iTyp = 0; iZaehler = 0;}


void Punkt::BildeVerbindungen (Sechseck *Basis)
{
  Rechts = Basis->LiefereOderErzeugePunkt(iX+2,iY);
  if (Rechts != NULL) Rechts->Links = this;
  Links = Basis->LiefereOderErzeugePunkt(iX-2,iY);
  if (Links != NULL)   Links->Rechts = this;
  RechtsOben = Basis->LiefereOderErzeugePunkt(iX+1,iY+1);
  if (RechtsOben != NULL)  RechtsOben->LinksUnten = this;
  LinksUnten = Basis->LiefereOderErzeugePunkt(iX-1,iY-1);
  if (LinksUnten != NULL) LinksUnten->RechtsOben = this;
  RechtsUnten = Basis->LiefereOderErzeugePunkt(iX+1,iY-1);
  if (RechtsUnten != NULL) RechtsUnten->LinksOben=this;
  LinksOben = Basis->LiefereOderErzeugePunkt(iX-1,iY+1);
  if (LinksOben != NULL) LinksOben->RechtsUnten = this;
}


void Punkt::SetzeTyp(int iNeuerTyp)
{
  if (iTyp == iNeuerTyp) iZaehler++;
  else {
       iTyp = iNeuerTyp;
       iZaehler = 1;
    }
}


void Punkt::EntferneTyp(int iNeuerTyp)
{
  if (iZaehler < 2)
    {
       iZaehler = 1;
       iTyp = iNeuerTyp;
    } else iZaehler--;
}


int Punkt::LiefereTyp()
{return iTyp}

void Punkt::SetzePunkt()
{
  Punkt *t;
  l = Rechts;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->Rechts; };
  l = Links;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->Links; };
  l = RechtsOben;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->RechtsOben; };
  l = RechtsUnten;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->RechtsUnten; };
  l = LinksOben;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->LinksOben; };
  l = LinksUnten;
  while (l != NULL)
    { l->SetzeTyp(2); l = l->LinksUnten; };
  SetzeTyp(1);
}


void Punkt::EntfernePunkt()
{
  Punkt *t;
  l = Rechts;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->Rechts; };
  l = Links;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->Links; };
  l = RechtsOben;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->RechtsOben; };
  l = RechtsUnten;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->RechtsUnten; };
  l = LinksOben;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->LinksOben; };
  l = LinksUnten;
  while (l != NULL)
    { l->EntferneTyp(0); l = l->LinksUnten; };
  SetzeTyp(0);
}


Sechseck::Sechseck(int iKante)
{
  int iA;
  iKantenLaenge = iKante;
  iBreite = 2 * iKante - 1;
  iBreite* = 2;
  iNrPunkte = 1;
  for (iA = 0; iA < iKante; iA++)
      iNrPunkte + = 6 * iA;
  PunktListe = new PunktZeiger[iNrPunkte];
  iNrPunkte = 1;
  Punkte = new PunktZeiger[iBreite * iBreite];
  for (iA = 0; iA < iBreite * iBreite; iA++)
      Punkte[iA] = NULL;
  PunktListe[0] = new Punkt(0,0);
  (Punkte[iKante*iBreite+iKante] =
      PunktListe[0]).-BildeVerbindungen(this);
}

Sechseck:~Sechseck()
{
  int iA;
  for (iA = 0; iA < iNrPunkte; iA++)
      delete PunktListe[iA];
}


Punkt* Sechseck::LieferePunkt(int iX, int iY)
{
  if ((abs(iX) >= 2*iKantenLaenge) || (abs(iY) >=
      iKantenLaenge)) return NULL;
  if (abs(iX) >= ((2*iKantenLaenge) - abs(iY))) return NULL;
  return Punkte[(iY+iKantenLaenge) * iBreite + iX +
      iKantenLaenge];
}

Punkt* Sechseck::LiefereOderErzeugePunkt(int iX, int iY)
{
  int iAJB;
  if ((abs(iX) >= 2*iKantenLaenge) || (abs(iY) >=
      iKantenLaenge)) return NULL;
  if (abs(iX) >= ((2*iKantenLaenge) - abs(iY))) return NULL;
  iA = iX+iKantenLaenge;
  iB = iY+iKantenLaenge;
  if (Punkte[iB*iBreite+iA] == NULL)
    {
      (Punkte[iB*iBreite+iA] =
          new Punkt(iX,iY)).-BildeVerbindungen(this);
      Punkte[iB*iBreite+iA]++ = Punkte[iB*iBreite+iA];
    };
  return Punkte[iB*iBreite+iA];
}

int Sechseck::NrPunkte()
{return iNrPunkte}

Punkt* Sechseck::PunktNr(int iNr)
{return PunktListe[iNr]}

Sechseck *Form;
int   iBest, iMaximum, iKantenLaenge, iNrPunkte;

void ZeigeSechseck()
{
  int iX, iY;
  for (iY = -iKantenLaenge+1; iY < iKantenLaenge; iY++)
    {
      for (iX = -((iKantenLaenge-1)*2);
        iX<2*iKantenLaenge-1; iX++)
        if (Form->LieferePunkt(iX,iY) == NULL)  printf(" ");
        else
          switch (Form->LieferePunkt(iX,iY)->LiefereTyp()) {
            case 0: printf(" "); break;
            case 1: printf("O"); break;
            case 2: printf(" + "); break;
          };
      printf("\n");
    }
}

int Suche(int iPunkte,int iStart)
{
  int   iA;
  Punkt *P;
  if (iPunkte>iBest)
    {
      iBest = iPunkte;
      printf(" % ifn ", iBest);
      ZeigeSechseck();
      if (iBest == iMaximum) return 1;
    };
  for (iA = iStart; iA < iNrPunkte; iA++)
    if ((P = Form->PunktNr(iA)->LiefereTyp() == 0)
      {
        P->SetzePunkt();
        if (Suche(iPunkte+1,iA+1)) return 1;
        P->EntfernePunkt();
      };
  return 0;
}

int main()
{
  printf("Kantenlaenge: ");
  scanf("%i",&iKantenLaenge);
  iBest = 0;
  iMaximum = 2 * iKantenLaenge - 1;
  Form = new Sechseck(iKantenLaenge);
  iNrPunkte = Form->NrPunkte();
  Suche(0,0);
  delete Form;
  return 0;
}
```

Design: Kohn, Karmann & Kohn, Köln Illustration: Gunnar Colbe, Köln Druck: DFS Druck und Verlag, Köln

Grußwort des Bundesministers für Bildung, Wissenschaft, Forschung und Technologie

Dr. Jürgen Rüttgers

Auch in diesem Jahr – zum 17. Mal – rufe ich Schülerinnen und Schüler auf, ihre Kenntnisse und Fähigkeiten auf dem Gebiet der Informatik in einem bundesweiten Wettbewerb unter Beweis zu stellen. Hier können junge Leute, die sich mit Talent und Leidenschaft der Informatik verschrieben haben, ihr Können testen, vergleichen und ihre Kenntnisse weiter verbessern. Das Aufgabenblatt des 17. Bundeswettbewerbs ist einmal mehr gefüllt mit Knobeleien und Spielen, Rätselhaftem und Verblüffendem aus der Welt der Kopfnüsse.

Die neuen Informations- und Kommunikationstechniken prägen direkt oder indirekt 60 % unserer Arbeitsplätze. Sie sind Schlüsselfaktoren für eine gute wirtschaftliche Zukunft unseres Landes. Mit den neuen Medien entwickelt sich auch eine neue Kulturtechnik. Ihre Beherrschung entscheidet zunehmend über persönliche Zukunftschancen junger Menschen.

Die Vermittlung von Medienkompetenz muß deshalb bereits in der Schule beginnen. Mit dem Projekt „Schulen ans Netz" wollen wir dazu beitragen, in den nächsten Jahren die 44.000 Schulen in Deutschland mit PC's auszustatten und ans Netz zu bringen.

Unser Land braucht helle Köpfe, die bereit sind, ihr Talent einzubringen und sich dem Wettbewerb zu stellen. Aber ein Schülerwettbewerb soll vor allem auch Freude machen, Freude am geistigen Schlagabtausch, Freude an der Herausforderung zu Spitzen- und Höchstleistungen. Ich wünsche allen Teilnehmerinnen und Teilnehmern Witz, Verstand und auch Glück beim Lösen der anspruchsvollen und vielseitigen Aufgaben. Wer in der Endrunde gewinnt, der darf sich Bundessieger nennen.

Grußwort der Präsidentin der Kultusministerkonferenz

Anke Brunn

Zum 17. Male wird in diesem Jahr der Bundeswettbewerb Informatik durchgeführt. Wir alle wissen, daß die neuen Informations- und Kommunikationstechniken zu den wichtigen Schlüsselqualifikationen in unserer modernen Industriegesellschaft gehören.
Diese Techniken verändern unser gesellschaftliches Leben in allen Bereichen nachhaltig und tiefgreifend, so daß auch der Lernort Schule sich mehr und mehr diesem Gegenstand geöffnet hat: Die neuen Informations- und Kommunikationstechniken, die neuen Medien nehmen inzwischen einen gesicherten Platz an den Schulen ein. Computerarbeitsplätze und lokale Vernetzungen sind in vielen Schulen verfügbar. Konzepte für die unterrichtliche Umsetzung des wichtigen Themas „Telekommunikation" werden mit Erfolg erprobt. Eine vernetzte Kommunikationsstruktur für die Schulen in Deutschland ist im Aufbau.

Die Schule übernimmt hier eine wichtige Aufgabe. Sie soll die Schülerinnen und Schüler zur Teilhabe an der globalen Information und Kommunikation befähigen und sie dabei in die Lage versetzen, Daten und Informationen selbständig zu gewinnen, zu überprüfen und sie auch zu bewerten. Dabei verfolgt sie insbesondere auch das Ziel, zu einem verantwortungsvollen Umgang mit den neuen Medien zu erziehen.

Der Bundeswettbewerb Informatik leistet nach Auffassung der Kultusminister und -senatoren der Länder in der Bundesrepublik Deutschland dazu einen wertvollen Beitrag. Er spornt mit seinen Aufgabenstellungen Jugendliche an, sich mit den Inhalten und Methoden der Informatik, aber auch mit den Fragen eines verantwortlichen Umgangs mit Informationen zu befassen. Er dient damit zugleich der Leistungsförderung auf einem für die Zukunft unserer Gesellschaft besonders wichtigen Gebiet.

Gesellschaft für Informatik e.V.

Die Gesellschaft für Informatik e.V. (GI) wurde 1969 als gemeinnütziger Verein zur Förderung der Informatik gegründet. Die 19 000 persönlichen und die 300 korporativen Mitglieder kommen aus der Informatik-Forschung, -Ausbildung, -Industrie, -Anwendung und -Studentenschaft. Träger der wissenschaftlichen Arbeit innerhalb der GI sind die Fachausschüsse und Fachgruppen, die in folgenden neun Fachbereichen zusammengeschlossen sind: Grundlagen der Informatik, Künstliche Intelligenz, Softwaretechnologie und Informationssysteme, Technische Informatik und Architektur von Rechensystemen, Informationstechnik und technische Nutzung der Informatik, Wirtschaftsinformatik, Informatik in Recht und öffentlicher Verwaltung, Ausbildung und Beruf, Informatik und Gesellschaft.

GMD Forschungszentrum Informationstechnik GmbH

Die GMD - Forschungszentrum Informationstechnik GmbH, eine der 16 Großforschungseinrichtungen der Bundesrepublik Deutschland, betreibt mit ihren ca. 1200 wissenschaftlich-technischen Mitarbeitern Forschung, Entwicklung und Transfer im Bereich der Informationstechnik, Informatik, Kommunikationstechnik und Medien. Die Forschungs- und Entwicklungsaufgaben reichen von der Grundlagenforschung bis zur konkreten Zusammenarbeit mit der Industrie in gemeinsamen Kooperationsvorhaben.

Bundeswettbewerb Informatik

Der Bundeswettbewerb Informatik wurde 1980 von der Gesellschaft für Informatik (GI) auf Initiative von Prof. Dr. Volker Claus ins Leben gerufen. Ziel des Wettbewerbs ist es, Interesse an der Informatik zu wecken und zu intensiver Beschäftigung mit ihren Inhalten und Methoden sowie den Perspektiven ihrer Anwendung anzuregen. Er gehört zu den bundesweiten Schülerwettbewerben, die von den Kultusministern der Länder unterstützt werden. Gefördert wird er vom Bundesminister für Bildung, Wissenschaft, Forschung und Technologie. Er steht unter der Schirmherrschaft des Bundespräsidenten. Die Träger des Wettbewerbs sind die GI (Gesellschaft für Informatik e. V.) und GMD - Forschungszentrum Informationstechnik GmbH. Die Gestaltung des Wettbewerbs und die Auswahl der Sieger werden vom Auswahlausschuß vorgenommen; Vorsitzender: Prof. Dr. Ingo Wegener, Universität Dortmund. Die Auswahl und Entwicklung von Aufgaben und die Festlegung von Bewertungsverfahren übernimmt ein Aufgabenausschuß; Vorsitzende: OStR Vera Reineke, Hannover. Eine Geschäftsstelle erledigt die fachlichen und organisatorischen Arbeiten; Geschäftsführerin: Dr. Gabriele Reich, Bonn.

Start und Ziel im September

Der Wettbewerb beginnt und endet im September, dauert etwa ein Jahr und besteht aus drei Runden. In der ersten und zweiten Runde sind fünf bzw. drei Aufgaben zu Hause selbständig zu bearbeiten. Dabei können die Aufgaben der ersten Runde ohne größere Informatikkenntnisse gelöst werden; die Aufgaben der zweiten Runde sind deutlich schwieriger. Die Bearbeitungszeit beträgt circa zwei Monate. In der ersten Runde ist Gruppenarbeit zugelassen und erwünscht. An der zweiten Runde dürfen jene, die allein oder zusammen mit anderen wenigstens drei Aufgaben weitgehend richtig gelöst haben, teilnehmen.
In der zweiten Runde ist dann selbständige Einzelarbeit gefordert. Die Bewertung erfolgt durch eine relative Plazierung der Arbeiten. Die ca. dreißig bundesweit Besten werden zur dritten Runde, einem Kolloquium, eingeladen. Darin führt jeder ein Gespräch mit je einem Informatiker aus Schule und Hochschule und analysiert und bearbeitet im Team zwei Informatik-Probleme.

Wer ist teilnahmeberechtigt?

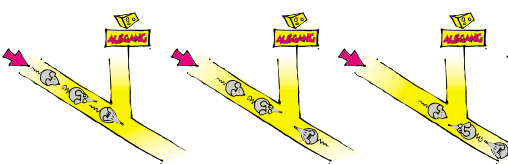
Teilnehmen können Jugendliche, die nach dem 9.11.1976 geboren wurden. Sie dürfen jedoch zum 1.9.1998 noch nicht ihre Ausbildung abgeschlossen oder eine Berufstätigkeit aufgenommen haben. Ebenfalls ausgeschlossen sind Personen, die zum Wintersemester 1998/99 oder früher ihr Studium an einer Hochschule/Fachhochschule aufnehmen bzw. aufgenommen haben. Jugendliche, die nicht deutsche Staatsangehörige sind, müssen wenigstens vom 1.9. - 9.11.98 ihren Wohnsitz in Deutschland haben oder eine staatlich anerkannte deutsche Schule im Ausland besuchen.

Als Anerkennung . . .

Jeder Teilnehmer der ersten Runde erhält eine Urkunde für die Teilnahme bzw. die erfolgreiche Teilnahme bei richtiger Lösung von wenigstens drei der fünf Aufgaben. Erfolgreiche Teilnahme berechtigt zum Eintritt in die zweite Runde. Die Besten der zweiten Runde werden zur dritten Runde, einem Kolloquium, eingeladen, das im Herbst 1999 von der Deutschen Bank ausgerichtet werden wird. Die dort ermittelten Bundessieger werden in der Regel ohne weiteres Aufnahmeverfahren in die Studienstiftung des deutschen Volkes aufgenommen. Zusätzlich sind für besondere Leistungen Geld- und Sachpreise sowie Einladungen zu Sommerakademien vorgesehen.

. . . Teilnahme an der Informatik-Olympiade

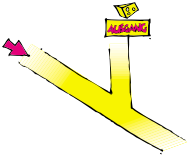
Aus dem Teilnehmerkreis der Endrunde werden die Kandidaten festgelegt, aus denen im Laufe mehrerer Trainingsrunden das vierköpfige deutsche Team ausgewählt wird, das an der Internationalen Olympiade in Informatik 2000 in China teilnimmt.



Aufgabe 1

Cheese.Champions

Prof. Grips will die Durchsetzungsfähigkeit einzelner Mäuse einer Mäusekolonie feststellen. Dazu hat er die folgende Versuchsanordnung entworfen: Ausgehend von einem unterirdischen Gang gräbt er in einem spitzen Winkel einen Ausgang zur Oberfläche, an dessen Ende er ein Stück duftenden Käse deponiert.
Die Mäuse der Kolonie kommen direkt hintereinander von links in den unterirdischen Gang und wollen durch den Abzweig zum Käse. Da der Gang zum Käse in einem spitzen Winkel abzweigt, muß jede Maus erst über den Abzweig hinaus weiter in den Gang hineinlaufen. Dann dreht sie sich um und hat möglicherweise ein Problem: Sie stößt auf die nachfolgende Maus, die ebenfalls in den rechten Abschnitt möchte.



Präziser gelten die folgenden Regeln für das Erreichen des Käses:
▶ Jede Maus muß zuerst an der Abzweigung vorbei.
▶ Wenn sie im rechten Teil nach der Abzweigung ist, kann sie entweder den Zweikampf gegen die nachfolgende Maus gewinnen und so in die Abzweigung gelangen oder sie wird weiter in den unterirdischen Gang geschoben. Die nachfolgende Maus wird dann in einen Zweikampf mit der ihr nachfolgenden Maus verwickelt.

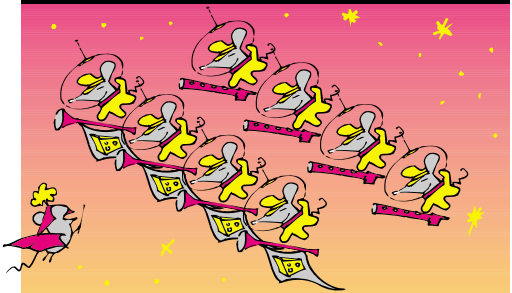
Prof. Grips numeriert die Mäuse in der Reihenfolge, in der sie in den Gang hineinlaufen, mit 1, 2, 3, ... und beobachtet, in welcher Reihenfolge sie beim Käse ankommen.

Aufgabe:
Schreibe ein Programm, das aus der beobachteten Reihenfolge, in der die Mäuse beim Käse ankommen, die Anzahl der gewonnenen Zweikämpfe für jede Maus berechnet und ausgibt. Zusätzlich soll Dein Programm nicht realisierbare Austrittsreihenfolgen erkennen und zurückweisen (wie z.B. die Austrittsreihenfolge 3, 1, 2).

Teste Dein Programm an den folgenden Notizen von Prof. Grips:
a) Ankommen am Käse:
7, 6, 5, 4, 3, 2, 8, 9, 1;
b) Ankommen am Käse:
1, 3, 4, 6, 7, 10, 9, 11, 12, 8, 5, 2;
c) Ankommen am Käse: 4, 5, 3, 1, 2, 6.

Aufgabe 2

Liiinks Umm!



Sergeant Jackson ist Dirigent des berühmtesten Marschier-Musik-Corps der Welt. Unter den Soldaten seiner Armee werden jedes Jahr die besten Musiker ausgesucht und dann einem rigorosen Training im Formationsmarschieren unterzogen. Jackson ist ein Perfektionist und sein Corps hat bei internationalen Wettbewerben schon viele Pokale gewonnen. Die ausgeklügelte Marsch-Choreographie sorgt jedes Jahr für großes Aufsehen und Bewunderung.

Leider geht dieses Jahr ausgerechnet bei einer Vorführung vor Staatschefs aus 17 Ländern etwas schief: In Takt 27 sollen sich die Musiker, die in einer Linie auf der großen Plaza der Stadt stehen, auf Taktschlag 3 um 90° drehen. Nur haben einige Soldaten vergessen, ob es nach links oder nach rechts gehen soll, und es kommt, wie es kommen muß: Manche Soldaten drehen sich nach rechts, andere nach links. Doch alle Musiker erinnern sich noch an den Leitsatz, den ihnen Jackson eingebleut hat:

„Frauen! Männer! Wenn ihr euch dreht und ihr schaut eurem Nachbarn ins Gesicht, dann habt ihr Mist gebaut! Aber keine Sorge! Wenn das passiert, dreht ihr euch beim nächsten Schlag in die andere Richtung! Das sind 180°! Und wenn dann immer noch wer seinem Nachbarn ins Gesicht schaut, dann macht ihr das nochmal! Und so weiter, bis Ruhe einkehrt! Alles klar?“

„Sir! Yes! Sir!!“

Jawoll...Jawoll...Jawoll...

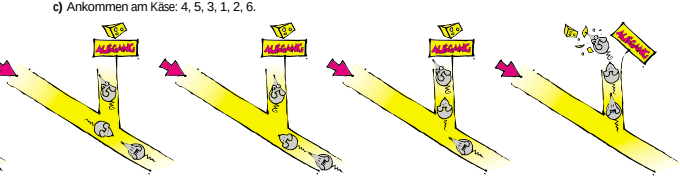
Das ist nicht so schwierig, und die Soldaten folgen den Anweisungen ihres Sergeanten und drehen sich und drehen sich wieder. Einige der Soldaten kommen nach der dritten Drehung ins Schwitzen und fragen sich, ob sie jemals zum Stillstand kommen.

Aufgabe:
▶ Kommen die Musiker jemals zur Ruhe?

▶ Wenn ja, wie lange dauert es?

▶ Schreibe ein Programm, welches die Anzahl der Soldaten einliest und, für jeden Soldaten, ob er sich bei der ersten Drehung nach links oder rechts dreht. Das Programm soll dann für jeden folgenden Taktschlag anzeigen, wie sich die Soldaten gedreht haben. Es soll anhalten, wenn keine weiteren Drehungen mehr stattfinden. Schicke uns drei Beispiele, darunter das mit der folgenden „Mist-gebaut-Situation“:
lrllrlrrrrrllirrrlrrl

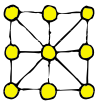
▶ Erweitere das Programm so, daß es für eine gegebene Anzahl der Soldaten deren Anfangsdrehung eigenständig zufällig auswählen kann.



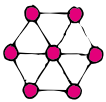
Aufgabe 3

Trixi Bilder

Trixi hat einen schwarz-weißen Computermonitor geschenkt bekommen, aber ach - kein herkömmliches Modell: Die Bildpunkte sind nicht quadratisch angeordnet (Rasterbild), sondern in gleichseitigen Dreiecken (Wabenbild). Die Abstände der Zeilen sind zwar gegenüber einem herkömmlichen Modell unverändert, aber Spalten gibt es nicht mehr.



Rasterstruktur



Wabenstruktur

Nach einigem Nachdenken findet Trixi aber, daß der Monitor für ihr Hobby "Bildverarbeitung" doch sehr praktisch ist: Viele Algorithmen könnten übersichtlicher und schneller werden, da jetzt jedes Pixel 6 gleichweit entfernte Nachbarn hat und nicht mehr 8 Nachbarn in zwei verschiedenen Entfernungen.

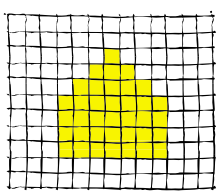
Weil aber alle Welt noch mit den unpraktischen Rasterbildern operiert, braucht Trixi Transformatoren für Rasterbilder in Wabenbilder und zurück. Die Bildinformation soll natürlich so gut wie möglich erhalten bleiben!

Aufgabe:

1. Vergleiche die Anzahl Pixel pro Fläche des herkömmlichen mit der des neuen Bildschirms (Prozentangabe).

2. a) Schreibe ein Programm, das schwarz-weiße Rasterbilder in schwarz-weiße Wabenbilder überführt,
b) und ein Programm, das schwarz-weiße Wabenbilder in schwarz-weiße Rasterbilder überführt,
so daß jeweils Größe und Aussehen der Bilder weitgehend erhalten bleiben.

Sende uns drei Rasterbilder im Originalzustand, nach Anwendung einer Hin- und Rücktransformation und nach Anwendung einer weiteren Hin- und Rücktransformation. Darunter muß auch folgendes Rasterbild sein:



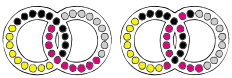
Trixi dankt !!!

Aufgabe 4

Rubiks:Ringe

Rubiks Ringe enthalten 10 schwarze, 9 gelbe, 10 rote und 9 graue Kugeln. Die Kugeln eines Ringes können linksdrehend oder rechtsdrehend verschoben werden. Man kann jeden Ring einzeln drehen. Durch die beiden Schnittpositionen können Kugeln von einem Ring in den anderen übergehen.

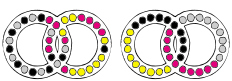
Beispiel:
Der Zustand rechts ist aus dem links entstanden durch:



1. Drehung des rechten Ringes um zwei Kugeln nach rechts (im Uhrzeigersinn),
2. Drehung des linken Ringes um eine Kugel ebenfalls nach rechts.

Ziel des Spiels ist, durch Drehen der Ringe aus einer bestimmten Anordnung der Kugeln eine andere bestimmte Anordnung der Kugeln herzustellen.

Beispiel:
Ausgangsstellung Zielstellung



Aufgabe:

1. Entwirf eine Notation für Ringe, Kugeln und Züge.

2. Entwirf eine Datenstruktur zum Aufnehmen der Zustände der Ringe und der Kugeln.

3. Schreibe ein Programm, welches zunächst Anfangs- und Zielzustand einliest. Anschließend wird jeweils ein Zug eingelesen und der Zug und der erreichte Zustand auf dem Bildschirm dargestellt (nicht notwendig graphisch).
Bei Beendigung soll das Programm ausgeben ob das Ziel erreicht wurde und, wenn ja, mit wievielen Zügen.

4. Lege der Aufgabenlösung 3 Protokolle bei, bestehend aus Ausgangsstellung, Zielstellung, Zugfolge und Ausgabe bei Beendigung.

Aufgabe 5

Linguistisk.Tool

Unser Gastschüler Earnest Younger schreibt seine Hausaufgaben auf einem Computer. Da er mit der deutschen Sprache noch nicht so vertraut ist, aber vermutet, ein größerer aktiver Wortschatz könne sich allgemein positiv auf seine Bewertungen auswirken, möchte er seine Fortschritte mit einem Programm messen. Ihn interessieren zu jedem seiner Texte:

▶ die Größe des Wortschatzes absolut,
▶ die Textlänge in Wörtern,
▶ die Größe des Wortschatzes relativ zur Textlänge,
▶ die einzelnen Anzahlen der im Wortschatz vorkommenden Wörter mit der Buchstabenlänge 1, 2, 3, usw.
(das Wortschatz-Histogramm).

Regelmäßig konjugierte oder deklinierte Wörter, die in mehreren Formen vorkommen, zählt Earnest nur einmal. Bei unregelmäßig konjugierten oder deklinierten Wörtern gilt ihm dagegen jede Form als eigenes Wort. Wort-Zusammensetzungen akzeptiert er als neue Wörter nur dann, wenn sie bindestrich-frei sind – sonst zählen die Teil-Wörter einzeln. Earnest geht von rechtschreibfehlerfreien Texten ohne Trennungsstriche aus; das stellt er vorher mit seinem elektronischen Lexikonprogramm sicher.

Earnest scheitert übrigens mit dem Versuch einer vollautomatischen Lösung der Abtrennung von Wortendungen. Also läßt er sich von seinem Programm in allen Zweifelsfällen interaktiv um eine Entscheidung bitten. Gleiches geschieht auch bei der Großschreibung kleiner Wörter am Satzanfang und ein paar anderen Spezialitäten, z.B. Abkürzungen.

Aufgabe:

▶ Implementiere E.Y.s Programm für die deutsche Schriftsprache (ob nach der alten, neuen oder privaten Rechtschreibung, ist uns gleich).

▶ Schicke uns vier Textanalysen, bestehend aus:
– dem Text selbst,
– Wortschatzgröße absolut und relativ, Textlänge in Wörtern,
– Wortschatz-Histogramm,
– einer Liste, bei welchen Wörtern das Programm um eine Entscheidung bitten mußte.

Die vier zu analysierenden Texte sind:
1. Ein eigener für die Schule geschriebener Text (mindestens 500 Wörter),
2. ein fremder Text (mindestens 500 Wörter),
3. der komplette Quelltext Deines Programms,
4. der Text dieser Aufgabe (siehe auch unter "http://www.bwinf.de/bwi17/Earnest.txt"), angepaßt an Deine Rechtschreibung.

Bitte in Druckschrift leserlich ausfüllen und Zutreffendes ankreuzen.

Geburtsdatum Name, Vorname

Straße männlich weiblich

Postleitzahl Wohnort

Telefon: Vorwahl / Durchwahl Bundesland der Schule

Name der (ehemaligen) Schule Schultyp

Informatiklehrer/in

Straße der Schule

Postleitzahl Schulort Klassenstufe (1-13)

Namen anderer Gruppenmitglieder

Wieviele Stunden hast Du gebraucht ?

Programmiersprache ? Rechner ? eigener

Hast Du an anderen Wettbewerben teilgenommen?

Mathematik Informatik sonstige

Wie hast Du vom Wettbewerb erfahren?

schon mal teilgenommen Schule ehemalige Teilnehmer Messen sonstiges

Diese Daten werden an niemanden weitergegeben, haben keinen Einfluß auf die Bewertung, aber dienen statistischen Zwecken.

Ich bin mit der Computerspeicherung dieser Daten einverstanden und versichere, daß ich die Aufgaben selbständig bzw. mit den angegebenen Gruppenmitgliedern bearbeitet habe.

Datum Unterschrift