24. Bundeswettbewerb Informatik 2005/2006

Aufgabenblatt 1. Runde

Der 24. Bundeswettbewerb Informatik für Jugendliche bis 21 Jahre.

Einsendeschluss ist der 7. November 2005.

Information und Unterlagen bitte anfordern beim:
Bundeswettbewerb Informatik
Ahrstr 45, 53175 Bonn
bwinf@bwinf.de
www.bwinf.de

Bundeswettbewerb Informatik

Der Bundeswettbewerb Informatik wurde 1980 von der Gesellschaft für Informatik e.V. (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 Bundesministerium für Bildung und Forschung und steht unter der Schirmherrschaft des Bundespräsidenten. Die Träger des Wettbewerbs sind die GI und die Fraunhofer-Gruppe Informations- und Kommunikationstechnik. Die Gestaltung des Wettbewerbs und die Auswahl der Sieger obliegen dem Beirat; Vorsitzender: Prof. Dr. Uwe Schöning, Universität Ulm. Die Auswahl und Entwicklung von Aufgaben und die Festlegung von Bewertungsverfahren übernimmt der Aufgabenausschuss. Die Geschäftsstelle des Wettbewerbs ist in Bonn und ist für die fachliche und organisatorische Durchführung zuständig; Geschäftsführer: Dr. Wolfgang Pohl.

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 selbstständig zu bearbeiten. Dabei können die Aufgaben der ersten Runde mit grundlegenden Informatikkenntnissen gelöst werden; die Aufgaben der zweiten Runde sind deutlich schwieriger. In der ersten Runde ist Gruppenarbeit zugelassen und erwünscht. An der zweiten Runde dürfen jene teilnehmen, die allein oder zusammen mit anderen wenigstens drei Aufgaben weitgehend richtig gelöst haben. In der zweiten Runde ist dann eigenständige Einzelarbeit gefordert; die Bewertung erfolgt durch eine relative Platzierung 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 7.11.1983 geboren wurden. Sie dürfen jedoch zum 1.9.2005 noch nicht ihre (informatikbezogene) Ausbildung abgeschlossen oder eine Berufstätigkeit aufgenommen haben. Ebenfalls ausgeschlossen sind Personen, die keine Schule mehr besuchen und zum Wintersemester 2005/2006 oder früher ihr Studium an einer Hochschule/Fachhochschule aufnehmen bzw. aufgenommen haben. Jugendliche, die nicht deutsche Staatsangehörige sind, müssen wenigstens vom 1.9. bis 7.11.2005 ihren Wohnsitz in Deutschland haben oder eine staatlich anerkannte deutsche Schule im Ausland besuchen.

Junioraufgabe Um die Teilnahme jüngerer Schülerinnen und Schüler am BWINF zu fördern, wird in diesem Wettbewerb zum zweiten Mal eine „Junioraufgabe“ gestellt. Sie darf von bis zu 16-Jährigen bearbeitet werden (geboren nach dem 7.11.1988) bzw. durch Gruppen mit einem solchen Mitglied.

Als Anerkennung ... In allen Runden des Wettbewerbs wird die Teilnahme durch eine Urkunde bestätigt. In der ersten Runde werden darüber hinaus erste und zweite Preise sowie Anerkennungen vergeben; mit einem Preis ist die Qualifikation für die zweite Runde verbunden. Auch in der zweiten Runde gibt es erste und zweite Preise; jüngere Teilnehmer haben die Chance auf eine Einladung zu einer Schülerakademie. Die Gewinner eines ersten Preises in der zweiten Runde werden zur dritten Runde eingeladen. Die dort ermittelten Bundessieger werden in der Regel ohne weiteres Aufnahmeverfahren in die Studienstiftung des deutschen Volkes aufgenommen. Zusätzlich sind für den Bundessieg, aber auch für andere besondere Leistungen Geld- und Sachpreise vorgesehen.

... Teilnahme an der Informatik-Olympiade Ausgewählte Endrundenteilnehmer können sich in mehreren Trainingsrunden für das vierköpfige deutsche Team qualifizieren, das an der Internationalen Informatik-Olympiade 2007 in Kroatien teilnimmt.

... Informatik-Seminare Für erfolgreiche BWINF-Teilnehmer aus Baden-Württemberg wird Anfang 2006 erneut das „Jugendforum Informatik“ auf der Burg Liebenzell vom Kultusministerium des Landes durchgeführt. Für die besten Teilnehmer aus Berlin und Brandenburg wird das Hasso-Plattner-Institut in Potsdam ein Tagesseminar anbieten.

... Preise für Teilnehmer, Lehrer und Schulen Auch beim 24. BWINF gibt es wieder einiges zu gewinnen. Dank der Unterstützung durch die Initiative D21 und die Firma Novell können 30 Pakete SUSE Linux vergeben werden, und zwar u.a. an die besten Erstrundenteilnehmer/innen sowie an Lehrkräfte und Schulen mit besonders vielen Teilnehmerinnen und Teilnehmern. Eine Chance auf ein SUSE-Paket haben auch diejenigen, die schon mal beim BWINF mitgemacht haben, beim 24. BWINF wieder dabei sind und eine Person ohne BWINF-Erfahrung für die Teilnahme am 24. BWINF werben konnten. Der oder die Geworbene muss dazu Namen und Geburtsdatum des/der Werbenden bei der Anmeldung angeben. In der zweiten Runde vergibt der Verlag O’Reilly erneut Buchpreise.

... Sonderpreis für Auszubildende Innerhalb der Altersbegrenzung können auch Auszubildende am BWINF teilnehmen. In diesem Jahr wird ganz ausdrücklich in den Ausbildungsbetrieben dazu aufgerufen, denn zum zweiten Mal schreibt die Gesellschaft für Informatik einen Sonderpreis für Auszubildende aus. Als Gewinn winken attraktive Sachpreise.

Die Sponsoren wünschen allen Teilnehmerinnen und Teilnehmern des 24. Bundeswettbewerbs Informatik viel Erfolg!

Grußwort

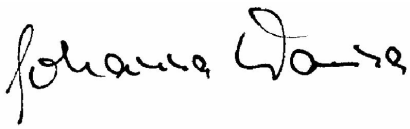
Die Informatik spielt eine Schlüsselrolle bei der Entwicklung der Informations- und Kommunikationstechniken. Überall im beruflichen und privaten Leben, in Wissenschaft und Wirtschaft kommen komplexe Systeme zur Informationsverarbeitung zum Einsatz, deren Leistungsfähigkeit und Weiterentwicklung den Fortschritt maßgeblich mitbestimmen. Durch den Einsatz weltweiter Netze werden Informationen in einem bisher nicht bekannten Maße erschlossen.

Der Bundeswettbewerb Informatik bietet Jugendlichen die Möglichkeit eines Kräftemessens besonderer Art. Dinge zu strukturieren, komplexe Systeme in überschaubare Teile zu zerlegen, zu formalisieren und zu interpretieren – diese Fähigkeiten sind gewissermaßen das Handwerkszeug des Junginformatikers. Auch die Abstraktionsfähigkeit und das Erfassen logischer Zusammenhänge, die Modellbildung und natürlich Sorgfalt, Genauigkeit und Ausdauer sind Eigenschaften, die zum Erfolg bei diesem Wettbewerb führen werden.

Die Erfahrung der vergangenen Jahre zeigt, dass eine große Zahl von Jugendlichen die kreativen und interessanten Aufgabenstellungen als eine motivierende Herausforderung ansieht und sich dieser auch gern stellt. Im letzten Jahr hat die Teilnehmerzahl den höchsten Wert seit fünf Jahren erreicht. Erfreulich ist hier vor allem der überproportionale Zuwachs bei den Auszubildenden und Berufsschülerinnen und -schülern, vor allem in den jüngeren Altersstufen. Besonders möchte ich Mädchen ermuntern, sich bei diesem Wettbewerb stärker zu beteiligen. Leider beobachtet man im Bereich der Informatik – wie insgesamt im mathematisch-naturwissenschaftlich-technischen Bereich – immer noch eine Unterrepräsentanz von weiblichen Jugendlichen.

Auch den Lehrkräften lege ich diesen Wettbewerb ans Herz. Er stellt sicherlich eine geeignete Möglichkeit des Förderns und Forderns dar. Die Aufgaben der ersten Runde können mit grundlegenden Informatikkenntnissen und in Gruppenarbeit gelöst werden. Der Wettbewerb richtet sich also nicht nur an hochtalentierte Jugendliche und „Computergenies“. Das Erarbeiten eigener Lösungswege ist im hohen Maße lehrreich und motivierend.

In diesem Sinne wünsche ich dem diesjährigen 24. Bundeswettbewerb Informatik einen großen Zuspruch und den Teilnehmerinnen und Teilnehmern Freude und Spaß, aber auch Geduld, Ausdauer und gute Ideen bei der Bearbeitung der Aufgaben.



Prof. Dr. Johanna Wanka
Präsidentin der Kultusministerkonferenz

Gesellschaft für Informatik e.V. (GI) Die Gesellschaft für Informatik e.V. (GI) ist mit rund 24.000 Mitgliedern die größte Fachgesellschaft der Informatik im deutschsprachigen Raum. Ihre Mitglieder kommen aus allen Sparten der Wissenschaft, der Informatikindustrie, aus dem Kreis der Anwender sowie aus Lehre, Forschung, Studium und Ausbildung. In der GI wirken Männer und Frauen am Fortschritt der Informatik mit, im wissenschaftlich-fachlich-praktischen Austausch in etwa 120 verschiedenen Fachgruppen und 35 Regionalgruppen. Ihr gemeinsames Ziel ist die Förderung der Informatik in Forschung, Lehre und Anwendung, die gegenseitige Unterstützung bei der Arbeit sowie die Weiterbildung. Die GI vertritt hierbei die Interessen der Informatik in Politik und Wirtschaft. Im Web: **www.gi-ev.de**

Fraunhofer-Gruppe Informations- und Kommunikationstechnik Als größter europäischer Forschungsverbund für Informations- und Kommunikationstechnik (IuK) versteht sich die Fraunhofer-IuK-Gruppe als Anlaufstelle für Industriekunden auf der Suche nach dem richtigen Ansprechpartner in der anwendungsorientierten Forschung. Die Vernetzung von fast 3000 Mitarbeitern an bundesweit 20 Standorten ermöglicht branchenspezifische IT-Lösungen, oft zusammen mit Partnern aus der Industrie, sowie anbieterunabhängige Technologieberatung. Entwickelt werden IuK-Lösungen für die Geschäftsfelder Digitale Medien, E-Business, E-Government, Kommunikationssysteme, Kultur und Unterhaltung, Medizin und Life Sciences, Produktion, Security, Software Engineering sowie Verkehr und Mobilität. Weitere Informationen bei der Geschäftsstelle der IuK-Gruppe: **www.iuk.fraunhofer.de**.

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Fraunhofer IuK Gruppe Informations- und Kommunikationstechnik

Unter der Schirmherrschaft des Bundespräsidenten

Aufgabe 1: Favorites First

Einige MP3-Player haben eine spezielle Zufalls- bzw. Shuffle-Funktion: Beim Anschalten berechnen sie eine zufällige Reihenfolge der abgespeicherten Stücke und spielen sie dann in dieser Folge ab.

Bo Hay, Marketingexperte eines Player-Produzenten, hat nun beobachtet, dass es Nutzerinnen und Nutzer gibt, die die Zufallsfunktion prinzipiell gerne einschalten, es aber schade finden, dass sie dabei ihre Lieblingsstücke relativ selten hören. Eine Ursache dafür ist, dass in der Regel nur die Anfangsteile solch zufälliger Abspielfolgen gehört werden. Bo Hay möchte, dass die Zufallsfunktion auf die folgende Weise abgewandelt wird: Unter den ersten 20 Stücken einer Abspielfolge sollen immer 5 vorkommen, die zufällig aus den 10 bis dahin meistgehörten Stücken ausgewählt werden. Dabei soll berücksichtigt werden, wenn ein Stück nach dem Anspielen übersprungen wird; es gilt dann als nicht gehört. Die Nutzerinnen und Nutzer sollen nicht wissen, dass der Zufall derart beeinflusst wird und sich nur über den für sie angenehmen Effekt freuen. Einen schönen Namen hat er für dieses neue Angebot auch schon gefunden: “Favorites First”.

Aufgabe

1. Realisiere eine Funktion (oder verwende eine von deiner Programmiersprache zur Verfügung gestellte Funktion), die die abgespeicherten Stücke in eine zufällige Reihenfolge bringen kann.
2. Überlege und beschreibe, wie Bo Hays Spezifikation von “Favorites First” umgesetzt werden kann, und schreibe ein Programm, das deine Überlegungen möglichst einfach realisiert.
3. Simuliere den Effekt von “Favorites First”. Gehe davon aus, dass 250 Stücke auf dem Player abgespeichert sind, dass von jeder Abspielfolge nur die ersten L Stücke zumindest angespielt werden (wobei L eine zufällige Zahl zwischen 10 und 20 ist) und dass von diesen L Stücken eine zufällige Auswahl von einem Viertel (abgerundet) der Stücke übersprungen wird. Simuliere mindestens 200 Benutzungen des Players. Ist ein Effekt zu beobachten, den Bo Hay sich vielleicht nicht gewünscht hat? Wenn ja, was ist deiner Meinung nach der Grund für das unerwünschte Verhalten?

Aufgabe 2: Formel-Up

Der Forscher Schinkenfranz ist wieder einmal der Weltformel auf der Spur. Er hat drei neue Messgeräte „H-Wert“,„A-Wert“ und „M-Wert“ gebaut und lässt damit experimentieren. Die ganzzahligen positiven Messwerte werden in eine dreispaltige Messtabelle eingetragen. Jedes Experiment ergibt eine Zeile. Schinkenfranz sucht nach „schönen Formulierungen“ für seine Messtabellen, in der Form:

<faktor-1> * H-Wert <operator> <faktor-2> * A-Wert = <faktor-3> * M-Wert

<faktor-i> sei eine positive ganze Zahl (1, 2, 3, ...).
<operator> sei eine der zwei Grundrechenarten $+$, $-$.

Weil er weiß, dass kleine Messabweichungen unvermeidlich sind und seine völlig überforderten Assistenten gelegentlich auch einen groben Experimentierfehler machen, gibt er diesen drei Hinweise, wie sie eine für ihn „schöne“ Formel finden können:

1. Die Summe der absoluten Abweichungen der M-Werte von ihren durch die Formel geforderten Idealwerten sei klein.
2. Bis zu drei Zeilen, die überhaupt nicht in die Formel passen wollen, dürfen als „fehlerhaft ausgeführte Experimente“ aus der Tabelle gestrichen werden.
3. Kleine Faktoren machen Formeln schöner als große Faktoren.

Die Assistenten stöhnen auf. Da erbarmt sich Schinkenfranz und führt wenigstens noch ein kleines Beispiel vor:

Experiment	H-Wert	A-Wert	M-Wert
1:	22	5	13
2:	57	31	29
3:	44	31	19
4:	42	21	21
5:	128	1	85

Wenn man aus der obigen Messtabelle Experiment 2 als fehlerhaft streicht, dann ist

2 * H-Wert - 1 * A-Wert = 3 * M-Wert

eine schöne Formel, weil die Summe der absoluten M-Wert-Abweichungen 0 ist und die Faktoren (2 1 3) alle klein sind.

Aufgabe

1. Programmiere einen FORMULIERER, der zu einer beliebigen dreispaltigen Messtabelle drei schöne Formeln vorschlägt und seine Vorschläge mit Hilfe einfacher Textbausteine kurz begründet.
2. Nimm die Messtabelle unter **www.bwinf.de/aufgaben/material.php** sowie vier selbst gemachte Messtabellen, um die sinnvolle Arbeit deines FORMULIERERs deutlich zu machen.
3. Gib dem FORMULIERER eine Tabelle mit Zufallszahlen. Was passiert?
4. Beantworte mit ein paar Sätzen: Wie entscheidet dein FORMULIERER, ob eine Formel schöner ist als eine andere? Warum entscheidet er gerade so? Gäbe es dazu vernünftige Alternativen?

Aufgabe 3: Zaras Zauberfolie (Fortsetzung)

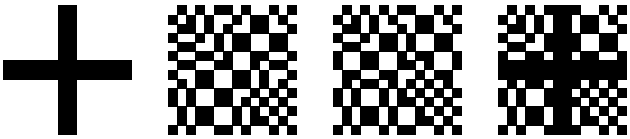
Die Handlungsreisende Zara Zackig wird von ihrer Firma häufig in Länder geschickt, von denen aus sie mit ihrer Firma nur per Fax in Kontakt treten kann.

Für erfolgreiche Verkaufsverhandlungen benötigt Zara immer wieder Informationen, die die Kunden nicht sehen dürfen. Die Firma überlegt sich daher eine Methode, Zara per Fax verschlüsselte Nachrichten zu übermitteln. Dazu wird zunächst eine Transparenz-Folie erzeugt, die Zara mit auf Reisen nimmt. Auf die Nachrichten, die sie unterwegs erhält, legt sie jeweils diese Folie – und siehe da, sie weiß, was sie wissen muss.

Die zu übermittelnde Nachricht entsteht durch Überlagerung des Fax und der Folie als ein schwarz-weißes Rasterbild. Hierzu wird jedes der eigentlichen Rasterquadrate nochmals unterteilt in vier Unterquadrate und tritt sowohl auf der Folie als auch auf dem Fax nur in den folgenden zwei Formen auf: Form A: (entspricht ▚), Form B: (entspricht ▞) Nur wenn A und B genau übereinanderliegen, entsteht ein schwarzes Rasterquadrat (entspricht ■).

Auf der Folie werden Quadrate der Formen A und B zufällig verteilt. In Abhängigkeit von der Folie muss das Fax so gestaltet werden, dass beim Auflegen der Folie auf das Fax das, was Zara sehen soll, schwarz hervortritt.

(Stark vergröbertes) Beispiel mit 7×7 Rasterquadraten (also 14×14 Unterquadraten):



Original; Faxmitteilung + Folie = Nachricht

Ein Konkurrent kann sich Kopien mehrerer von Zara empfangener Faxe aneignen. Er möchte diese nun gerne entschlüsseln. Leider passt Zara gut auf ihre Originalmaske auf und er muss die Entschlüsselung ohne diese vornehmen.

Aufgabe

1. Überlege, ob und wie er die Entschlüsselung vornehmen kann.
2. Schreibe ein Programm, das aus den digitalisierten Kopien der verschlüsselten Faxe die Originale so gut rekonstruiert, dass sie in der Regel lesbar werden.

   Dein Programm bekommt die eingescannten Bilder dabei in folgendem Format (PBM, Portable Bitmap, s. rechts):

   Die Datei beginnt stets mit `P1`, gefolgt von der Breite und Höhe in Pixeln als Dezimalzahl. Danach folgen alle Pixel, die durch 0 und 1 dargestellt werden. Obiges Beispiel zeigt natürlich den Buchstaben A.
3. Erläutere die Arbeitsweise deines Programms und entschlüssele sechs Faxe, welche du von **www.bwinf.de/aufgaben/material.php** herunterladen kannst.

```
P1
9 9
000010000
000010000
000101000
000101000
001000100
001111100
001000100
010000010
111000111
```

Aufgabe 4: Fehlzeitendatenbank

An der Zuse-Schule ist häufiges Fehlen und Zuspätkommen in letzter Zeit zu einem gravierenden Problem geworden. Deshalb hat die Schulkonferenz beschlossen, die Anwesenheit von Schülern und Lehrern elektronisch zu erfassen. Jedem Schüler und jedem Lehrer soll ein RFID-Chip im Nackenbereich implantiert werden. Alle Türrahmen von Unterrichtsräumen sollen mit Erfassungseinrichtungen ausgestattet werden, die Personen beim Betreten und beim Verlassen eines Unterrichtsraumes registrieren.

Zusätzlich fordert der Elternrat der Schule, es solle eine Möglichkeit geschaffen werden, die aktuelle Unterrichtsanwesenheit und die Fehlzeiten der eigenen Kinder und der ihre Kinder unterrichtenden Lehrer via Internet abzufragen.

Aufgabe

1. Beschreibe ein sinnvolles Verfahren für die Abwicklung von Entschuldigungen für Fehlzeiten. Beurteile die Praxistauglichkeit deines Verfahrens.
2. Erstelle ein Entity-Relationship-Modell für eine Datenbank, die eine Abfrage der Fehlzeiten ermöglicht. Beschreibe, in welcher Hinsicht dein Modell eine Vereinfachung der Realität darstellt. Rechtfertige deine Entscheidungen.
3. Gib, deinem Entity-Relationship-Modell entsprechend, ein System von Tabellen in Relationenschreibweise an. Ein Beispiel für die Schreibweise einer Tabelle:
   `Schüler (Schüler-ID, Vorname, Nachname, Geburtsdatum)`
   Unterstreiche Schlüsselattribute und kennzeichne Fremdschlüssel in geeigneter Weise.
4. Was hältst du von dem Beschluss der Schulkonferenz und dem Ansinnen des Elternrates? Begründe deine Meinung.

Hinweis: Literatur und Material zu Datenbanken, Entity-Relationship-Modell und RFID-Chips findest du unter **www.bwinf.de/aufgaben/material.php**.

Aufgabe 5: Kleingeld

Clara ärgert sich über das viele Kleingeld, das ihr Portemonnaie so dick macht. Insbesondere die Cent-Münzen (also die Münzen mit den Werten 1, 2, 5, 10, 20 und 50 Cent) würde sie gerne los werden. Sie überlegt, wie viele Cent-Münzen sie durchschnittlich mit sich trägt, wenn sie stets darauf bedacht ist, die Münzanzahl zu minimieren. Du sollst ihr helfen, diese Frage zu beantworten. Dabei kannst du Folgendes annehmen:

- Clara versucht, nach jedem Einkauf möglichst wenig Cent-Münzen im Portemonnaie zu haben.
- Sie weiß, dass das Kassenpersonal überall freundlich ist und ihr möglichst wenig Cent-Münzen als Wechselgeld gibt.
- Clara hat immer genügend 1-Euro-Münzen dabei.

Wenn sie als einzige Cent-Münze ein 2-Cent-Stück im Portemonnaie hat, wird sie also bei einem Einkauf für 2,46 Euro diese Münze mit auf die Theke legen, um anschließend nur drei Münzen zu haben (50, 5 und 1 Cent) statt vier (50 und drei mal 2 Cent).

Aufgabe

1. Schreibe ein Programm, das die Entwicklung des Cent-Münzen-Gehalts von Claras Portemonnaie simulieren kann. Es soll von einem Anfangsbestand an Cent-Münzen ausgehen und eine Folge von zufälligen Beträgen erzeugen, die Clara bei ihren Einkäufen bezahlen muss. Für jeden Betrag soll es berechnen und geeignet anzeigen, welche und wie viele Cent-Münzen Clara bei der Bezahlung verwendet, wieder herausbekommt und anschließend im Portemonnaie hat.
2. Simuliere eine ausreichend hohe Anzahl von Einkäufen und bestimme die durchschnittliche Anzahl Cent-Münzen in Claras Portemonnaie (zu Beginn hat sie keine Cent-Münzen). Dokumentiere die Ausgaben deines Programms für die ersten 100 Einkäufe.
3. Wird eine Münzsorte besonders selten verwendet? Glaubst du, dass sie einfach weggelassen werden kann?

Junioraufgabe: Der Segelflug der Solinge!

Ein Soling ist ein „Verwandter“ des Lemmings. Im Gegensatz zu einem Lemming tritt ein Soling aber immer alleine auf.

Bei diesem Spiel segeln auf einem gerasterten Bildschirm von oben nach unten derartige Solinge. Auf dem Weg nach unten befinden sich mehrere horizontale Barrieren, die jeweils an bis zu drei Stellen durchbrochen sind.

Die Solinge sollen vom Spieler möglichst schnell zur unteren linken Ecke des Bildschirmes geführt werden. Der Spieler kann dabei die Bewegungsrichtung des Solings nach links oder rechts beeinflussen.

Anschließend segelt der nächste Soling auf den Bildschirm. Dabei verändern sich jeweils auch die Barrieren auf dem Spielfeld.

Ziel des Spieles ist: Möglichst viele Solinge sollen in einer bestimmten Zeit das Ziel erreichen.

Aufgabe

1. Die obige Spielbeschreibung lässt noch viele Punkte offen. Präzisiere diese Beschreibung lückenlos und eindeutig. Begründe dabei deine Entscheidungen.
2. Implementiere das Spiel entsprechend deiner Beschreibung.

Beispiel für ein Spielfeld (das X stellt einen Soling dar):

```
================================================
#Start#                                         #
###  ##                                         #
#                                               #
#                                               #
#-----------------   -------------   ----------#
#                                               #
#                                               #
#                                               #
#                                               #
#-------  -----   -----  ----------------------#
#                   X                           #
#                                               #
#                                               #
#--   ------------------------------------    --#
#                                               #
#                                               #
#----------   ---   --------------------   ----#
#                                               #
#                                               #
#                                               #
##  ##                                          #
#Ziel#                                          #
================================================
```

Mitmachen – Schritt für Schritt

Bearbeitung

Halte dich bei der Bearbeitung der Aufgaben an die verschiedenen Teilaufgaben. Zu den Aufgaben mit Programmierung erarbeite und sende uns jeweils Folgendes:

Lösungsidee:
Eine Beschreibung der Lösungsidee, unabhängig vom eingesandten Programm. Die Idee und die zu ihrer Beschreibung verwendeten Begriffe müssen aber im Programm nachvollziehbar sein.

Programm-Dokumentation:
Eine Beschreibung, wie die Lösungsidee im Programm und seinen Teilen realisiert wurde. Hinweise auf Besonderheiten und Nutzungsgrenzen, verlangte Eingabeformate usw.

Programm-Ablaufprotokoll:
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 und das Funktionieren des Programms verdeutlichen (teilweise in den Aufgabenstellungen vorgegeben). Bildschirm-Fotos sind zulässig.

Programm-Text:
Den kommentierten Quelltext des Programms in einer der gängigen höheren Programmiersprachen wie z.B. Pascal. Skriptsprachen sind zulässig, Maschinensprache nicht. Den Programmtext bitte ausdrucken, dabei aber auf nicht selbst geschriebene Teile (wie z. B. Funktionen der Entwicklungsumgebung und automatisch generierten Programmtext) verzichten.

Programm:
Das lauffähige Programm selbst auf einer CD oder 3,5”-Diskette. Dieser Datenträger muss auch den Programm-Text enthalten und unter Windows-Systemen lesbar sein. Ist kein Programm gefordert, strukturiere deine Bearbeitung der Aufgabenstellung entsprechend.

Bitte schicke deine Arbeit nach Aufgaben geordnet und geheftet auf einseitig bedrucktem DIN-A4-Papier. Nummeriere alle Blätter rechts oben und versieh sie mit deinem Namen. Die Texte sollen in Deutsch abgefasst sein. Verwende DIN-A4-Klarsichthüllen mit Lochrand (pro Aufgabe eine) oder loche die Blätter bitte. Beschrifte den Datenträger, der die Programme und Programm-Texte enthält, mit deinem Namen.

Fragen zu den Aufgaben?

per Telefon: 0228 / 37 86 46 jeweils von 17-19 Uhr am 14.9., 10.10., 26.10. und 4.11.2005
per E-Mail: bwinf@bwinf.de
per Fax: 0228 / 37 29 000
per Brief: an den BWINF (siehe „Einsendung“)

Informationen (allgemeine Tipps und FAQ) gibt es auch im Internet unter www.bwinf.de. Meinungen und Fragen (aber keine Lösungsideen) zu den Aufgaben können auch in der BWINF-Newsgroup fido.ger.bwinf ausgetauscht werden.

Anmeldung

Die Anmeldung erfolgt, indem der Einsendung ein ausgefülltes Formular beigelegt wird. Bei Gruppen muss jede Teilnehmerin und jeder Teilnehmer ein Formular ausfüllen, Gruppenmitglieder ohne Anmeldeformular werden nicht gewertet!

Fülle das Anmeldeformular (Klappe des Aufgabenblattes oder eine Kopie davon) vollständig, korrekt und leserlich! aus. Insbesondere das Geburtsdatum muss angegeben sein, denn sonst kann die Einsendung nicht korrigiert werden. Wer die Schule bereits verlassen hat, gebe bei „Klassenstufe“ bitte an, was sie/er zur Zeit macht. Bei Erstteilnahme kann ggf. der oder die Teilnehmerin genannt werden (mit Namen und Geburtsdatum), der/die zum Mitmachen angeregt hat.

Für die Anmeldung gibt es unter **www.bwinf.de** auch ein Internet-Formular, das vor dem Einsendeschluss ausgefüllt werden kann. Bei dieser Online-Anmeldung wird eine Kennnummer vergeben; bei der Einsendung muss das Papierformular nur noch mit dieser Nummer, dem Namen und einer Unterschrift versehen werden. Wer sich per Internet anmeldet, erhält nach der Einsendung eine Eingangsbestätigung per E-mail.

Einsendung

Einsendungen per E-mail oder nur auf CD/Diskette ohne Ausdruck können wir leider nicht akzeptieren. Auch alle geforderten Beispiele müssen auf Papier dokumentiert sein. Es ist nicht auszuschließen, dass die Gutachterinnen und Gutachter nur auf die Papierunterlagen zugreifen können.

Sende alles an: Bundeswettbewerb Informatik, Ahrstraße 45, 53175 Bonn

Einsendeschluss ist der 7. November 2005 (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 sich selbst adressierte und frankierte Postkarte mitschicken oder die Online-Anmeldung nutzen.

Bewertung

Einsendungen werden danach bewertet,
- ob die Aufgaben vollständig und richtig bearbeitet wurden,
- ob die Ausarbeitungen gut strukturiert und verständlich sind und
- ob die (Programm-) Unterlagen vollständig, übersichtlich und lesbar sind.

Beispiellösung

Aufgabenstellung: tratsCH trATsch

Die Leute vom Planeten Chator schreiben gern Schlechtes übereinander. Wer vielen über andere Schlechtes schreibt, gilt als besonders charmant. Aber natürlich nur, wenn die Kompromittierten nichts davon erfahren.

Chatonen schreiben nur an Leute, die Ihnen sympathisch sind. Doch die können den Tratsch weitertragen, und eventuell genau an den Falschen. Ein Chatone muss also gut aufpassen, dass er keinen Charmefehler macht. Dieses Missgeschick passierte unlängst Ator, als er Btor Schlechtes über Dtor schrieb. Zu dumm: Dtor ist dem Ctor sympathisch, der wiederum Btor sympathisch ist. Und so landete der Tratsch bei Dtor, der über Ator verständlicherweise sehr verärgert war. Dies hätte Ator mit ein wenig Übersicht vermeiden können, denn schließlich wissen alle Chatonen voneinander, wer wem sympathisch ist.

Aufgabe

Programmiere einen Charminator, der einliest, welche Chatonen welchen anderen sympathisch sind. Er soll auf dieser Grundlage möglichst kompakt für alle Chatonen ausgeben, wem der Betreffende über wen Schlechtes schreiben kann, ohne einen Charmefehler zu riskieren.

Teste dein Programm an drei Beispielen für mindestens 5 und höchstens 10 Chatonen; verwende auf jeden Fall das folgende Beispiel:

Dem Chatonen	sind sympathisch:
A	B E
B	C
C	D G
D	C
E	F
F	B E G
G	H
H	D

Bedenke dabei, dass eine Tratschnachricht von Chatone x nach Chatone y weder x noch y betrifft und dass ein Chatone eine Nachricht, die ursprünglich von ihm selbst stammt und wieder bei ihm ankommt, nicht weiterleiten wird – so dumm sind Chatonen nicht.

Lösungsidee *nach Benjamin Berger*

Die Sympathiebeziehungen der Chatonen werden als ein gerichteter Graph aufgefasst: die Chatonen sind die Knoten, die direkten Sympathiebeziehungen die gerichteten Kanten. Dann wird der Weg einer Nachricht von einem Chatonen S (Sender) an einen dem S sympathischen Chatonen E (Empfänger) für alle möglichen Paare (S,E) wie folgt nachvollzogen: Der Graph wird beginnend bei E rekursiv durchsucht (Tiefensuche), bis entweder S selbst erreicht wird (der die Nachricht nicht weiterleitet) oder ein gefundener Chatone schon besucht wurde, die Nachricht

also schon erhalten hat. Über alle Chatonen, die *nicht* besucht wurden, kann S bei E tratschen – da E an sie den Tratsch eben nicht weiterleiten kann.

Programm-Dokumentation

Die Lösungsidee wurde in Java umgesetzt. Das Programm besteht aus zwei Klassen: Die Klasse Chatone stellt Variablen für den Namen eines Chatonen und die Liste der ihm sympathischen Chatonen sowie die Methode für die Suche nach vom Empfänger einer Nachricht aus erreichbaren Chatonen zur Verfügung. Die Klasse Charminator verwaltet alle während des Programmlaufs erzeugten Chatonen-Objekte und enthält die Hauptprozedur.

Die Hauptprozedur liest eine Sympathietabelle als Zeilen von Zeichenketten (Strings) ein, wobei der erste String den Namen eines Chatonen und alle folgenden Strings die Namen der ihm sympathischen Chatonen angibt. Nach dem Einlesen einer Zeile werden für neue Namen die passenden Chatonen-Objekte angelegt; dies regelt die Prozedur `getChatone`. Nach der Eingabe werden für alle Chatonen S die Lästermöglichkeiten berechnet und ausgegeben: Für jeden sympathischen Chatonen E wird die Suchmethode `sucheErreichbareVon` aufgerufen; falls das Komplement der zurückgegebenen Menge nicht leer ist, wird ausgegeben, dass S bei E über die darin enthaltenen Chatonen lästern kann.

Die Suchmethode setzt die Lösungsidee direkt um. Die Variable erreichbar speichert die bei der Suche vom Empfänger E aus besuchten und damit von E aus erreichbaren Chatonen. Zunächst wird der Sender S in dieser Variable gespeichert, damit bei S die Suche endet. Nun wird die rekursive Prozedur `sucheErreichbare` für E aufgerufen. Sie beendet den Suchzweig, wenn der aktuelle Chatone schon besucht wurde. Ansonsten speichert sie diesen als erreichbar und setzt die Suche durch rekursiven Aufruf für alle sympathischen Chatonen fort.

Programm-Ablaufprotokoll

Aus Platzgründen kann hier nur ein Ablaufprotokoll abgedruckt werden. Das Programm liest die Sympathietabelle der Aufgabenstellung ein und gibt die Lästermöglichkeiten aus:

```
$ java Charminator
Gib die Zeilen der Sympathietabelle ein:
Chatone Sympathisch1 Sympathisch2 ...
End-of-File (Ctrl-D) beendet die Eingabe.
A B E
B C
C D G
D C
E F
F B E G
G H
H D
A kann bei B tratschen ueber: E F
E kann bei F tratschen ueber: A
F kann bei E tratschen ueber: A G B D C H
F kann bei G tratschen ueber: A E B
```

```
F kann bei B tratschen ueber: A E
G kann bei H tratschen ueber: A E F B
B kann bei C tratschen ueber: A E F
D kann bei C tratschen ueber: A E F B
C kann bei G tratschen ueber: A E F B
C kann bei D tratschen ueber: A E F G B H
H kann bei D tratschen ueber: A E F B
$
```

Programm-Text

```
import java.util.*;
import java.io.*;

/** Diese Klasse enthält die main()-Methode. **/
public class Charminator{
 //Die Menge aller erzeugten Chatonen-Objekte.
 static Set chatonen=new HashSet();
 //Die Zuordnung von Namen zu Chatonen.
 static Map namen=new HashMap();

 public static void main(String[] args)throws IOException{
  //Begrüßung & Instruktionen
  System.out.println("Gib die Zeilen der Sympathietabelle ein: ");
  System.out.println("Chatone Sympathisch1 Sympathisch2 ...");
  System.out.println("End-of-File (Ctrl-D) beendet die Eingabe.");

  BufferedReader inReader=
    new BufferedReader(new InputStreamReader(System.in));
  //Einleseschleife
  while(true){
   String zeile=inReader.readLine();
   if(zeile==null)break;  //End-Of-File beendet das Einlesen.
   //Zum Zerlegen der Eingabe in die Namen-Strings:
   StringTokenizer stizer=new StringTokenizer(zeile, " ");
   String chatName=stizer.nextToken(); //erster Name: Chatone
   while(stizer.hasMoreTokens()) //alle weiteren: sympathisch
      getChatone(chatName).
        sympathisch.add(getChatone(stizer.nextToken()));
  } //Ende der Einleseschleife

  //Alle Chatonen können Sender sein.
  for(Iterator i=chatonen.iterator(); i.hasNext(); ){
   Chatone sender=(Chatone)i.next();
   //Alle Sympathischen können Empfänger sein.
   for(Iterator j=sender.sympathisch.iterator(); j.hasNext(); ){
    Chatone empfaenger=(Chatone)j.next();
    //Alle vom Empfänger Erreichbaren sollten ...
    Set erreichbar=sender.sucheErreichbareVon(empfaenger);
    Set laesternUeber=new HashSet(chatonen); //(kopieren)
    //... kein Lästerobjekt sein:
    laesternUeber.removeAll(erreichbar);
    //Ausgabe, falls der Sender lästern darf:
    if(laesternUeber.size() != 0){
     System.out.print(sender + " kann bei " +
```

```
                    empfaenger + " tratschen ueber: ");
     for(Iterator k=laesternUeber.iterator(); k.hasNext(); )
                 System.out.print(k.next() + " ");
     System.out.println();
    }
   }
  }
 } //end Main

 /* Gibt ein Chatone-Objekt aus "namen" zurück. Falls nötig,
    wird der Eintrag neu erzeugt (auch in "chatonen"). */
 static Chatone getChatone(String name){
  Chatone ergebnis=(Chatone)namen.get(name);
  if(ergebnis==null){ //Name unbekannt
   ergebnis=new Chatone(name);
   namen.put(name, ergebnis);
   chatonen.add(ergebnis);
  }
  return ergebnis;
 }
}

/** Diese Klasse realisiert die wichtigen Eigenschaften
    eines Chatonen: Name, sympathische Chatonen und
    Erreichbarkeitssuche. **/
class Chatone{
 //Diejenigen, denen dieser Chatone schreiben kann
 HashSet sympathisch=new HashSet();
 //Der Name dieses Chatonen
 final String name;
 public Chatone(String name_){ name=name_; }
 //Gibt den Namen des Chatonen zurück
 public String toString(){ return name; }

 /* Bestimme alle Chatonen, die eine Nachricht dieses
    Chatonen an einen Empfänger erreichen kann. */
 public Set sucheErreichbareVon(Chatone empfaenger){
  Set erreichbar=new HashSet(); //besuchte=erreichbare Chatonen
  erreichbar.add(this); //Beim Sender soll die Suche enden.
  //nun startet die Suche
  empfaenger.sucheErreichbare(erreichbar);
  return erreichbar;
 }

 /* rekursive Suchprozedur */
 void sucheErreichbare(Set erreichbar){
  //Wenn dieser Chatone schon erreicht wurde, endet die Suche.
  if(erreichbar.contains(this)) return;
  //Sonst ist er nun erreicht; die Suche wird bei ihm fortgesetzt.
  erreichbar.add(this);
  for(Iterator i=sympathisch.iterator(); i.hasNext(); )
   ((Chatone)i.next()).sucheErreichbare(erreichbar);
 }
}
```

Anmeldung

24. Bundeswettbewerb Informatik 2005/2006, 1. Runde

Bitte in Druckschrift ausfüllen und Zutreffendes ankreuzen. [X]

Geburtsdatum Name, Vorname [ ] männlich [ ] weiblich

Straße

Postleitzahl Wohnort

Telefon: Vorwahl / Durchwahl E-mail-Adresse

Name der (ehemaligen) Schule Schultyp

Informatiklehrer/in Klassenstufe (1-13)

Straße der Schule

Postleitzahl Schulort Bundesland der Schule

Namen anderer Gruppenmitglieder

Wieviele Stunden hast du gebraucht? _____

Was hast du bei der Bearbeitung verwendet? Welche/s/n ...

Programmiersprache? Betriebssystem? Computer(typ)? [ ] eigener?

Hast du an anderen Wettbewerben teilgenommen? [ ] Informatik [ ] Mathematik [ ] sonstige

Wie hast du vom Wettbewerb erfahren?
[ ] schon mal teilgenommen [ ] Schule/Lehrer [ ] ehemalige Teilnehmer [ ] sonstiges

Geworben durch ehe- und diesmalige(n) Teilnehmer/in:

Name, Vorname, Geburtsdatum

Diese Daten werden an niemanden weitergegeben, haben keinen Einfluss auf die Bewertung, aber dienen statistischen Zwecken. Sie werden ausschließlich für die Zwecke des Bundeswettbewerbs Informatik ausgewertet.

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

Datum, Unterschrift: