33. Bundeswettbewerb Informatik 2014/2015

Die Aufgaben der zweiten Runde

Allgemeine Hinweise

Herzlichen Glückwunsch zum Erreichen der zweiten Runde! Hier sind die Aufgaben. Sie sind anspruchsvoll, und ihre Bearbeitung ist aufwändig. Aber die Mühe lohnt sich, denn durch Teilnahme an der zweiten Runde

- wirst du sicher sehr viel lernen;
- kannst du dich für die Endrunde qualifizieren;
- kannst du mit einer guten Leistung einen Buchpreis des Verlags O'Reilly gewinnen;
- hast du am Ende eine Arbeit fertig gestellt, die du als so genannte „Besondere Lernleistung" in die Abiturwertung einbringen kannst;
- kannst du dich (als jüngerer Teilnehmer) um die Teilnahme an einer Deutschen Schülerakademie bewerben;
- hast du die Chance auf eine Einladung zu den „Forschungstagen Informatik 2015" des Max-Planck-Instituts für Informatik in Saarbrücken.

Wir wünschen also viel Spaß und viel Erfolg bei der Bearbeitung!

Es gibt drei Aufgaben. **Eine Einsendung darf Bearbeitungen zu höchstens zwei Aufgaben enthalten**, deren Bewertung dann das Gesamtergebnis ausmacht. Sollte eine Einsendung Bearbeitungen zu allen drei Aufgaben enthalten, werden wir zwei davon zufällig auswählen und nur diese bewerten.

An dieser Runde dürfen nur Einzelpersonen teilnehmen, die in der ersten Runde in drei Aufgaben insgesamt mindestens 12 Punkte erreicht oder einem Team angehört haben, dem dieses gelungen ist. Gruppenarbeit ist in der zweiten Runde nicht zulässig.

Einsendeschluss ist der 20. April 2015.

Bearbeitung

Die Bearbeitung einer Aufgabe sollte zunächst eine nachvollziehbare und vollständige Lösung aller Teilaufgaben enthalten. **Pluspunkte** für eine höhere Bewertung kannst du erreichen, wenn du die Aufgabe dort, wo es möglich und sinnvoll ist, eigenständig weiterentwickelst. Sinnvoll sind inhaltliche Erweiterungen und Verbesserungen, etwa von Datenstrukturen und Algorithmen; uninteressant sind aufwändige Tricks, z. B. zur reinen Verschönerung der Benutzungsoberfläche. Begründe für jede Erweiterung, weshalb sie sinnvoll ist und ihre Realisierung eine eigene Schwierigkeit darstellt.

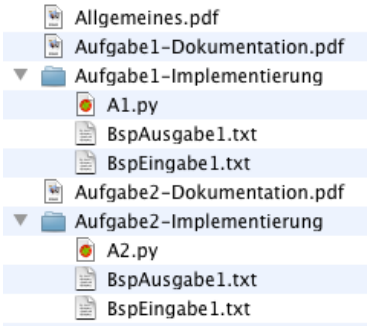
Grundsätzlich gelten die Vorgaben der 1. Runde weiter. Wesentliches Ergebnis der Aufgabenbearbeitung ist also eine **Dokumentation**, in der du den *Lösungsweg* sowie die *Umsetzung* des Lösungswegs in das dazugehörige Programm beschreibst. Die Beschreibung des Lösungswegs kann mit Hilfe (halb-)formaler Notationen präzisiert werden, die der Umsetzung mit Verweisen auf die entsprechenden Quellcode-Elemente. In die Dokumentation gehören auch *Beispiele* (Programmein- und -ausgaben oder Zwischenschritte) die zeigen, wie das Programm sich in unterschiedlichen Situationen verhält. Komplettiert wird die Dokumentation durch den *Quelltext*, wobei unwichtige und automatisch generierte Teile weggelassen werden sollten.

Das zweite Ergebnis der Bearbeitung ist die **Implementierung**. Sie besteht aus dem zur Lösung der Aufgabe geschriebenen lauffähigen *Programm* und dem kompletten *Quelltext*. Außerdem können Beispielein- und ausgaben oder weiteres hilfreiches Material der Implementierung beigefügt werden.

Die komplette Dokumentation zu einer Aufgabe muss als ein Dokument eingereicht werden. Dieses Dokument wird für die Bewertung ausgedruckt. **Es kann sein, dass für die Bewertung nur die Dokumentation herangezogen wird! Die Dokumentation muss also unbedingt auch Beispiele enthalten!** Diese sollte also einen lückenlosen und nachvollziehbaren Nachweis des Leistungsumfangs und der Funktionstüchtigkeit der Programme geben. Ihr Umfang soll sich in Grenzen halten; eine gute Dokumentation vermittelt kurz und präzise alles Nötige, insbesondere die wesentlichen Ideen beim Lösungsweg. Nötig ist alles, was Interessierte mit guten Informatikkenntnissen, die die Aufgabenstellung kennen, wissen müssen, um den Lösungsweg zu verstehen und seine Realisierung nachzuvollziehen. Entscheidend für eine gute Bewertung sind zwar richtige (und sauber umgesetzte) Ansätze, aber die Dokumentation hat schon oft den Ausschlag für oder gegen das Weiterkommen gegeben.

Einsendung

Die Einsendung erfolgt wieder über das BWINF-PMS (pms.bwinf.de). Hochladen kannst du ein max. 20MB großes ZIP-Archiv (z.B. VornameNachname.zip); seine Inhalte sollten so strukturiert sein wie im Bild rechts. Ein Dokument `Allgemeines.pdf` ist aber nur nötig, wenn du allgemeine, von den Aufgabenbearbeitungen unabhängige Anmerkungen zu deiner Einsendung machen willst. Die Dokumentationen müssen als PDF-Dokumente enthalten sein. Dateien in anderen Formaten werden möglicherweise ignoriert. Die Schriftgröße einer Dokumentation muss mindestens 10 Punkt sein, bei Quelltext mindestens 8 Punkt. Auf jeder Seite einer Dokumentation sollen in der Kopfzeile Verwaltungsnummer, Vorname, Name und Seitennummer stehen. Die Verwaltungsnummer steht auf der Teilnahmebescheinigung der ersten Runde.



Weitere Hinweise

Bei der Bewertung können Programme unter Windows (7 / 8), Linux, Mac OS X (10.10) und Android ausgeführt werden.

Fragen zu den Aufgaben können per E-Mail an `bwinf@bwinf.de` oder telefonisch unter 0228–378646 (zu üblichen Arbeitszeiten) gestellt werden. Die Antwort auf E-Mail-Anfragen kann sich leicht verzögern. Informationen zur 2. Runde finden sich auf unseren Webseiten (`www.bundeswettbewerb-informatik.de`). Auf `einstieg-informatik.de` werden in der Community sicher wieder viele Teilnehmer über die Aufgaben diskutieren – ohne Lösungsideen auszutauschen.

Allen Teilnehmern der zweiten Runde wird bis Mitte Juni 2015 die Bewertung mitgeteilt. Die Besten werden zur Endrunde eingeladen, die im September 2015 vom Fachbereich Informatik der Technischen Universität Darmstadt ausgerichtet wird, unterstützt durch das Fraunhofer-Institut für Graphische Datenverarbeitung, das Fraunhofer-Institut für Sichere Informationstechnologie und Lufthansa Systems. Dort werden die Bundessieger und Preisträger ermittelt und ausgezeichnet. Bundessieger werden in der Regel ohne weiteres Auswahlverfahren in die Förderung der Studienstiftung des deutschen Volkes aufgenommen. Außerdem werden Geld- und Sachpreise vergeben. Der Rechtsweg ist ausgeschlossen.

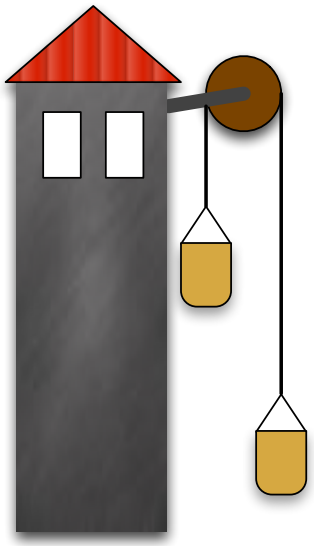
Zum Schluss noch einmal: Viel Spaß und viel Erfolg!

Aufgabe 1: Seilschaften

Lewis Carrol, bekannt als Autor von „Alice in Wonderland“, beschäftigte sich auch mit Rätseln. Eines davon ist besonders bekannt geworden:

> *Eine Königin war mit Sohn und Tochter in einem Turmzimmer gefangen. Die Tür war verschlossen, aber vor dem offenen Fenster gab es einen Seilzug, an dessen Enden je ein Korb befestigt war. Diesen Mechanismus konnten die drei zur Flucht nutzen. Wer in einem der gleich schweren Körbe abwärts fahren wollte, musste diesen Korb schwerer beladen als den anderen Korb, aber weniger als 15 Pfd. schwerer, sonst wäre die Fahrt zu rasant gewesen. Die Königin wog 195 Pfd., die Tochter 105 Pfd. und der Sohn 90 Pfd. Im Turmzimmer lag außerdem ein Stein, der 75 Pfd. wog. Wie konnten die drei sicher entkommen?*

Wir wollen eine Verallgemeinerung dieses Rätsels betrachten: Es gibt n *Personen* mit Gewichten $p_1, \ldots, p_n$ und m *Steine* mit Gewichten $s_1, \ldots, s_m$. Jede Person und jeder Stein befindet sich entweder *oben* oder *unten*. Steine können nur durch anwesende Personen in die Körbe geladen werden. Die Körbe können dann fahren, wenn die Ladung des oberen Korbs schwerer ist als die des unteren Korbs. Personen fahren sicher, wenn die Gewichte der Ladungen sich höchstens um d unterscheiden.



Mithilfe des Seilzugs und der Körbe sollen alle Personen sicher entkommen.

Aufgabe

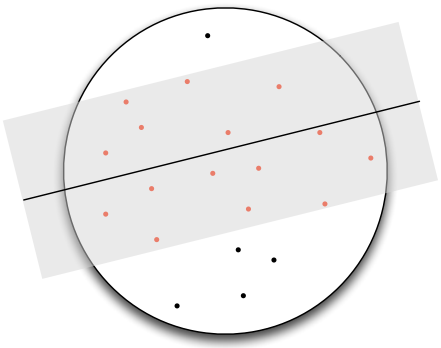
Schreibe ein Programm, welches die Anzahl der Personen n, die Anzahl der Steine m, alle Gewichte $p_1, \ldots, p_n, s_1, \ldots, s_m$, die Positionen der Personen und Steine sowie die Schranke d einlesen kann. Anschließend soll dein Programm feststellen, ob es eine Folge von Fahrten gibt, mit der alle Personen sicher entkommen können. Ist dies der Fall, soll dein Programm eine kürzeste solche Folge in übersichtlicher Form ausgeben.

Wende dein Programm auf die unter `bundeswettbewerb-informatik.de` abgelegten Beispieleingaben an und dokumentiere die Ergebnisse. Unter den Beispieleingaben befinden sich auch die Daten des originalen Rätsels.

Aufgabe 2: Panorama-Kegeln

Anna langweilt sich im Urlaub mit den Eltern. Da sie heute keine Kirche und kein Museum mehr sehen kann, bleibt sie lieber in der Ferienwohnung und programmiert ein simuliertes Kegelspiel. Ihr Spiel soll aber etwas anders sein, und es soll nicht auf Behändigkeit, sondern auf Überlegung und Planung ankommen.

Am Anfang einer Runde werden 20 Kegel auf einer Kreisscheibe mit Radius 2 platziert. Die Kegel sind punktförmig, und jeder wird, unabhängig von den anderen, an einem Punkt platziert, der zufällig und gleichmäßig aus der gesamten Kreisscheibe gezogen wird. Dann geht das eigentliche Spiel los, bei dem die Spieler abwechselnd eine Kugel mit Radius 1 werfen. Das Zentrum einer geworfenen Kugel bewegt sich auf einer Geraden ℓ, und die Kugel trifft jeden Kegel, dessen Abstand zu ℓ höchstens 1 beträgt. Jeder von der Kugel eines Spielers getroffene Kegel wird aus dem Spiel entfernt und gibt dem Spieler einen Punkt.



Zu Hause will Anna gegen ihre Klassenkameraden spielen; aber im Urlaub ist sie allein, und so beschließt sie, gegen einen gedachten Gegner Randy zu üben, der zufällig spielt. Genauer wählt Randy vor jedem seiner Kugelwürfe einen zufällig und gleichmäßig verteilten Winkel v mit $0 \leq v \leq 360$ und einen zufällig und gleichmäßig verteilten Punkt x in der Kreisscheibe und wirft seine Kugel dann aus der Himmelsrichtung v so, dass ihr Zentrum x trifft.

Um kein zu leichtes Spiel gegen Randy zu haben, gibt ihm Anna einen buchstäblichen Startvorteil: Randy wirft immer die erste Kugel. Danach spielen abwechselnd Anna und Randy. Nach jedem ihrer Würfe kann Anna wählen, die Runde zu beenden und eine neue anzufangen. Ihr Ziel ist, über viele Spielrunden hinweg mehr Punkte als Randy zu sammeln.

Aufgabe

1. Überlege dir, wie man in einfacher und effizienter Weise die zufällige Platzierung der Kegel am Anfang einer Runde vornehmen kann. Implementiere dein Verfahren und eine graphische Ausgabe des Ergebnisses.
2. Implementiere den Rest des Spiels so, dass Anna am Rechner möglichst bequem gegen Randy im Rechner spielen kann.
3. Suche eine Strategie für Anna, mit der sie tatsächlich gegen Randy gewinnen kann. Implementiere deine Strategie so, dass man am Bildschirm verfolgen kann, wie Anna Randy nach und nach in die Knie zwingt.
4. Wie erfolgreich ist deine Strategie bei anderen Kegelanzahlen und Radien?

Wenn du Lust hast, eine Strategie gegen andere auszuprobieren, kannst du dies auf dem BWINF-Turnierserver machen (`turnier.bundeswettbewerb-informatik.de`). Dort ist ein Spielmodus „Panoramakegeln“ eingerichtet. Ob und wie du dieses Angebot nutzt, hat keinen Einfluss auf die Bewertung.

Aufgabe 3: Mississippi

Die Erbinformation aller Lebewesen befindet sich in der Desoxyribonukleinsäure (DNS, engl. DNA) in Form einer Folge der vier Basen Adenin (A), Guanin (G), Cytosin (C) und Thymin (T). Lange Folgen von Basen bilden die Chromosomen. Jedes Chromosom ist ein Doppelstrang aus verdrillten Basenfolgen. Eine derartige „Strickleiter“ als verdrillter Doppelstrang wird auch Doppelhelix genannt.

Das Genom, die Folge der Basen in den Chromosomen, ist für den Menschen inzwischen vollständig bekannt. Für weitergehende Forschungen bezüglich der Heilung von Krankheiten ist es allerdings notwendig, das Genom genauer zu untersuchen, etwa auf wiederholt vorkommende Basenfolgen, auch Repetitionen genannt.

Aufgabe

Schreibe ein Programm für eine solche Untersuchung, das eine Zeichenkette sowie zwei Zahlen k und l einliest und diejenigen Teilzeichenketten ausgibt, die

- *lang* sind (mindestens l Zeichen),
- *häufig* (mindestens k-mal) in der Zeichenkette auftreten *und*
- *maximal* (nicht Teil einer längeren und gleich häufigen Teilzeichenkette) sind.

Wir betrachten als Beispiel die folgende Zeichenkette: CAGGAGGATTA

- Häufige Teilzeichenketten ($k = 2$): A (kommt 4-mal vor), AG (2), AGG (2), AGGA (2), G (4), GA (2), GG (2), GGA (2), T (2)
- Maximale häufige Teilzeichenketten ($k = 2$): A, AGGA, G, T
- Ausgabe ($k = 2, l = 1$): A, AGGA, G, T
- Ausgabe ($k = 2, l = 2$): AGGA

Wende dein Programm auf die unter `bundeswettbewerb-informatik.de` abgelegten Beispieleingaben an und dokumentiere die Ergebnisse. Dein Programm sollte mindestens Eingaben der Länge 10.000 erfolgreich verarbeiten können.