41. Bundeswettbewerb Informatik 2022/2023

Die Aufgaben der 2. Runde

Bundeswettbewerb Informatik

Allgemeine Hinweise

Herzlichen Glückwunsch zum Erreichen der 2. Runde! Hier sind die Aufgaben. Sie sind anspruchsvoll, und ihre Bearbeitung ist aufwendig. Aber die Mühe lohnt sich, denn durch Teilnahme an der 2. Runde

- wirst du sicher sehr viel lernen;
- kannst du dich für die Endrunde qualifizieren;
- kannst du einen Buchpreis der Verlage O'Reilly oder dpunkt.verlag gewinnen;
- hast du am Ende eine Arbeit fertig gestellt, die du als 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" 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 dieser Aufgaben enthalten**, deren Bewertung dann das Gesamtergebnis ausmacht. Sollte eine Einsendung Bearbeitungen zu mehr als zwei Aufgaben enthalten, werden wir zwei davon zufällig auswählen und nur diese bewerten.

An dieser Runde dürfen nur Einzelpersonen teilnehmen, die in der 1. Runde in drei Aufgaben insgesamt mindestens 12 Punkte erreicht oder einem Team angehört haben, dem dieses gelungen ist. Gruppenarbeit ist in der 2. Runde nicht zulässig.

Einsendeschluss ist Montag, der 17. April 2023.

Bearbeitung

Die Bearbeitung einer Aufgabe sollte zunächst eine nachvollziehbare und vollständige Lösung aller Teilaufgaben enthalten. **Zusatzpunkte** 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, die praktisch realisiert werden; uninteressant sind aufwendige 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 Beschreibung der Umsetzung mit Verweisen auf die entsprechenden Quellcode-Elemente.

In die Dokumentation gehören auch aussagekräftige *Beispiele* (Programmeingaben/-ausgaben, ggf. inklusive Zwischenschritte/-ergebnisse), die zeigen, wie das Programm sich in unterschiedlichen Situationen verhält. Komplettiert wird die Dokumentation durch *Auszüge aus dem Quelltext*, die alle wichtigen Module, Methoden, Funktionen usw. enthalten. Die Beschreibung des Lösungswegs und der Umsetzung sollte jedoch keinen oder nur wenig Quellcode enthalten.

Weiteres Ergebnis der Aufgabenbearbeitung ist die **Implementierung**. Sie besteht aus dem zur Lösung der Aufgabe geschriebenen lauffähigen *Programm* und dem vollständigen *Quelltext*. Außerdem können Beispieleingabe/-ausgaben oder weiteres hilfreiches Material der Implementierung beigefügt werden.

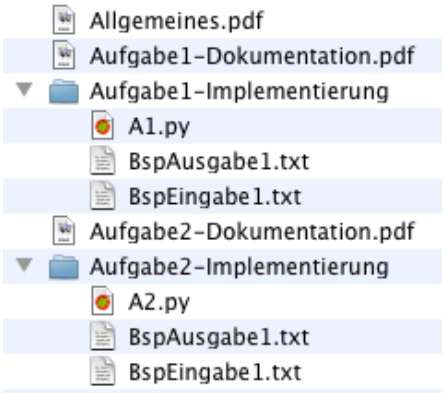
Die Dokumentation zu einer Aufgabe mit allen oben genannten Bestandteilen muss als PDF-Dokument eingereicht werden. **Es kann sein, dass für die Bewertung deiner Einsendung nur die Dokumentation herangezogen wird.** Sie sollte also einen lückenlosen und verständlichen Nachweis des Leistungsumfangs und der Funktionstüchtigkeit der Programme geben – und unbedingt die vorgegebenen Beispiele neben eigenen enthalten!

Der Umfang der Dokumentation 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 Umsetzung nachzuvollziehen.

Entscheidend für eine gute Bewertung sind zwar richtige (und sauber umgesetzte) Lösungswege, aber die Dokumentation hat schon oft den Ausschlag für oder gegen das Weiterkommen gegeben. Das Erstellen der Dokumentation sollte die Arbeit an Lösungsideen und Umsetzung eng begleiten. Wer zunächst die Lösungsidee verständlich formuliert, dem fällt anschließend eine fehlerlose Implementierung leichter. Abbildungen tragen in der Regel zur Verständlichkeit bei, und es schadet nicht, die Dokumentation von Dritten prüfen zu lassen, selbst wenn diese fachfremd sind.

Einsendung

Die Einsendung erfolgt wieder über das BWINF AMS (login.bwinf.de). Hochladen kannst du ein max. 40 MB großes ZIP-Archiv (z. B. `VornameNachname.zip`). Sein Inhalt sollte so strukturiert sein wie rechts abgebildet. Die Dokumentationen der bearbeiteten Aufgaben müssen als PDF-Dokumente enthalten sein; Dateien in anderen Formaten werden möglicherweise ignoriert. Ein Dokument `Allgemeines.pdf` ist nur dann nötig, wenn du allgemeine, von den Aufgabenbearbeitungen unabhängige Bemerkungen zu deiner Einsendung machen willst. Die Schriftgröße einer Dokumentation muss mindestens 10 Punkt sein, bei Quelltext mindestens 8 Punkt. Auf jeder Seite einer Dokumentation sollen in der Kopfzeile die Teilnahme-ID, Vorname, Name und Seitennummer stehen; hierfür sind auf den BWINF-Webseiten Vorlagen zu finden. Die Teilnahme-ID steht auf der Teilnahmebescheinigung der 1. Runde, und du findest sie auch im AMS; es handelt sich um eine Zahl zwischen 64.000 und 69.000.



Weitere Hinweise

Bei der Bewertung können Programme unter Windows (10), Linux, Mac OS X (12.1) und Android ausgeführt werden.

Fragen zu den Aufgaben können per Mail an `bundeswettbewerb@bwinf.de` gestellt werden. Die Antwort auf E-Mail-Anfragen kann sich leicht verzögern. Informationen zur 2. Runde finden sich auf unseren Webseiten: `bwinf.de/bundeswettbewerb`. Unsere Online-Community findest du unter `einstieg-informatik.de`; dort werden im Forum sicher wieder viele Teilnehmerinnen und Teilnehmer über die Aufgaben diskutieren – ohne Lösungsideen auszutauschen.

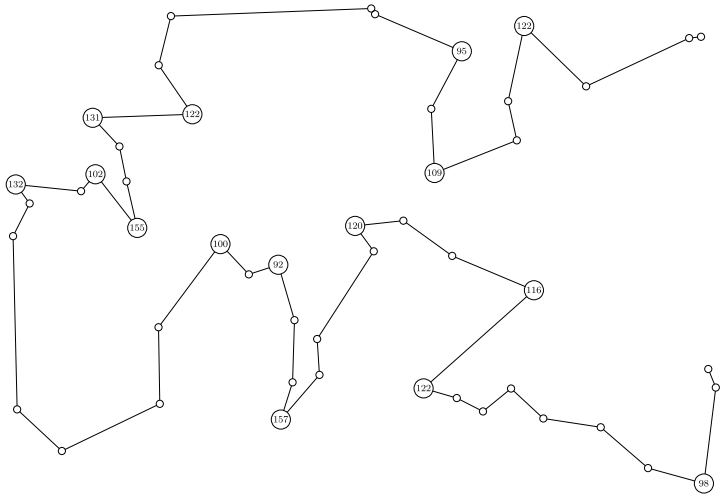
Allen Teilnehmern der 2. Runde wird bis Mitte Juni 2023 die Bewertung mitgeteilt. Die Besten werden zur Endrunde eingeladen, die 12.-15. September 2023 von der Fakultät für Informatik des Karlsruher Instituts für Technologie ausgerichtet werden wird. Dort wird entschieden, wer mit einem Bundessieg oder mit einem zweiten Preis ausgezeichnet wird. Bundessiegerinnen und Bundessieger werden in der Regel ohne weiteres Auswahlverfahren in die Förderung der Studienstiftung des deutschen Volkes aufgenommen. Außerdem werden Geldpreise vergeben. Der Rechtsweg ist ausgeschlossen.

Viel Spaß und viel Erfolg!

Aufgabe 1: Weniger krumme Touren

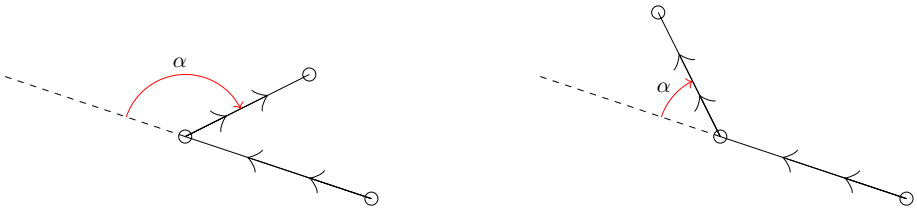
Ein Anteil von Antons Arbeit ist das Abfliegen aller Außenstellen in Australien. Auf seiner Tour muss er vorgegebene Orte besuchen, kann sich aber aussuchen, in welcher Reihenfolge dies geschieht.

Anton möchte dies schnell erledigen. Deshalb hat er sich für seine Tour die folgende Route mit möglichst kurzer Flugstrecke überlegt. Sie ist 2649 Kilometer lang. Bei dieser Route muss er an manchen Orten sehr scharf abbiegen. Für diese Orte hat er die Abbiegewinkel in seine Karte eingetragen:



In Zukunft möchte er alle Abbiegewinkel von mehr als 90 Grad vermeiden. Kannst du ihm helfen und eine neue Route für ihn berechnen? Diese neue Route muss natürlich weiterhin alle Außenstellen enthalten. Sie kann an beliebigen Orten beginnen und enden. Desweiteren muss der Flug von einer Außenstelle zur nächsten geradlinig verlaufen, und Anton muss an keinem Ort mit einem Winkel von mehr als 90 Grad abbiegen.

Hier siehst du zwei verschiedene Abbiegewinkel α. Links ist $\alpha > 90°$ und in der Route nicht gewünscht, im Gegensatz zum Winkel $\alpha \leq 90°$ rechts.



Aufgabe

Schreibe ein Programm, das Koordinaten einlesen kann und eine entsprechende Route berechnet. Kann das Programm immer eine solche finden? Versuche die Flugstrecke möglichst kurz zu machen, aber es ist nicht verlangt, die allerbeste Route zu finden.

Wende dein Programm mindestens auf alle Beispiele an, die du auf den BWINF-Webseiten findest, und dokumentiere die Ergebnisse.

Aufgabe 2: Alles Käse

Während eines tristen Lockdowns spazierte Antje mal wieder durch ihre Wohnung und telefonierte dabei mit Bernd. Das Gespräch dauerte viel länger als gedacht, es dämmerte bereits, und schon bei der Begrüßung hatte Antje ziemlichen Appetit auf ein Brot mit Käse …

So landete sie schließlich in der Küche, legte das Stück ihres Lieblingskäses auf ein Brettchen und begann, nicht so ganz bei der Sache, weil auf das Gespräch konzentriert, Scheibe für Scheibe vom Käsequader abzuschneiden.

Die Scheiben waren zwar immer ein Millimeter dick, sodass die Quaderform nach jedem Schnitt erhalten blieb, aber zwischen zwei Schnitten drehte und wendete sie das Käsestück irgendwie mal hier mal dorthin, drehte es um, usw. So entstanden eine Menge Käsescheiben und der Quader wurde nach und nach kleiner. Nachdem sie sich von Bernd verabschiedet hatte, realisierte sie, dass der ursprüngliche Quader verschwunden war, da sie alles in Scheiben geschnitten hatte.

Sie stellte ihren Hunger, ehemals Appetit, hintenan und versuchte, den ursprünglichen Käsequader wieder zusammen zu setzen.

Das war schwierig.

Noch mit fettigen Fingern erstellte sie ein Programm, das dann wie geschmiert lief.

Aufgabe

a) Tue es Antje gleich und schreibe ein Programm, das prüft, ob eine gegebene Menge von Käsescheiben wieder zu einem vollständigen Käsequader zusammengesetzt werden kann.

   Die Käsescheiben sind als Rechtecke angegeben; sie sind immer einen Millimeter dick.

   Das Programm soll ausgeben, ob die Käsescheiben zu einem Quader zusammengesetzt werden können und falls ja, in welcher Reihenfolge.

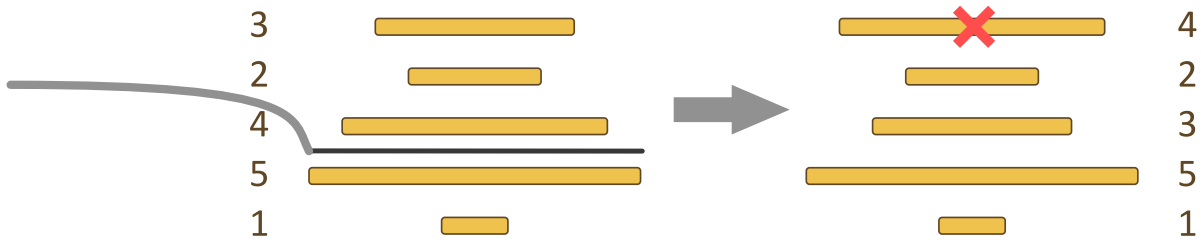
b) Kannst du dein Programm erweitern, sodass es allgemeinere Fragestellungen lösen kann? Zum Beispiel könnte ja eine Scheibe fehlen, weil Antje sie mittendrin aufaß, oder Antje könnte mit mehreren statt einem Käsequader angefangen haben.

Wende dein Programm mindestens auf alle Beispiele an, die du auf den BWINF-Webseiten findest, und dokumentiere die Ergebnisse.

Aufgabe 3: Pancake Sort

Bei dieser Aufgabe geht es um Sortieren von Pfannkuchen unter erschwerten Bedingungen. Die Pfannkuchen liegen auf einem Stapel und sollen nach ihrer Größe sortiert werden. Sortiert werden darf ausschließlich mit einem Pfannenwender, und zwar auf folgende Weise: Der Pfannenwender wird in den Stapel hineingeschoben oder unter ihn, dann wird der auf dem Wender liegende Teilstapel umgedreht, und anschließend wird der oberste Pfannkuchen aufgegessen. Das Ziel ist ein sortierter Stapel, bei dem der kleinste Pfannkuchen oben liegt. Wie viele Pfannkuchen übrig sind, ist dabei unerheblich.

Der Einfachheit halber nehmen wir an, dass die Größen unserer n Pfannkuchen gerade den Zahlen 1 bis n entsprechen (jede Größe kommt nur einmal vor). Enthält der Stapel nun die Größen $(3,2,4,5,1)$ (von oben nach unten), und wir schieben den Wender zwischen den Pfannkuchen mit Größe 4 und den mit Größe 5, so erhalten wir als Ergebnis unserer Wende-und-Ess-Operation den Stapel $(2,3,5,1)$.



Man kann sich nun einen Algorithmus überlegen, der für einen gegebenen Stapel eine möglichst kurze Folge von Wende-und-Ess-Operationen berechnet, die zu einem aufsteigend sortierten Stapel führt. Bei der Pfannkuchen-Wende-Und-Ess-Zahl (PWUE-Zahl) wird es noch etwas komplizierter: Wir nennen die kleinste mögliche Anzahl an Wende-und-Ess-Operationen, die einen gegebenen Stapel S in einen sortierten Stapel überführt, $A(S)$. Im obigen Beispiel mit $S = (3,2,4,5,1)$ ist $A(S) = 2$. Die PWUE-Zahl $P(n)$ ist nun die größte Zahl, die $A(S)$ für irgendeinen Startstapel S mit n Pfannkuchen annimmt.

Ein Stapel mit nur einem Pfannkuchen kann nicht falsch sortiert sein, also ist $P(1) = 0$. Außerdem ist $P(2) = 1$, da maximal eine Operation notwendig ist. Für drei Pfannkuchen wird es schon etwas schwieriger.

Es sind aber einige PWUE-Zahlen bekannt:

n	1	2	3	4	5	6	7	8	9	10	11	12	13
$P(n)$	0	1	2	2	3	3	4	?	?	?	?	?	?

Aufgabe

a) Schreibe ein Programm, das bei Eingabe eines Stapels S eine möglichst kurze Liste an Wendeoperationen ausgibt, die S in einen aufsteigend sortierten Stapel überführen.

b) Schreibe ein Programm, das bei Eingabe von n die PWUE-Zahl $P(n)$ berechnet sowie ein Beispiel ausgibt, für das tatsächlich $P(n)$ Wendeoperationen notwendig sind. Berechne mindestens $P(8)$ bis $P(11)$.