

Die Aufgaben der 2. Runde

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 ein Exemplar des Buches 'Fit fürs Studium - Informatik' des Rheinwerk Verlags gewinnen;
- hast du am Ende eine Arbeit fertig gestellt, die du als Besondere Lernleistung in die Abiturwertung einbringen kannst;
- kannst du dich (wenn du jünger bist) um die Teilnahme an einer Deutschen Schülerakademie bewerben;
- hast du die Chance auf eine Einladung zu den „Führungstagen 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 28. April 2025.

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 Aufgabebearbeitung 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. Auf unserer Webseite findest du außerdem Pflichtbeispiele, die zu dokumentieren sind. 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 beigelegt werden.

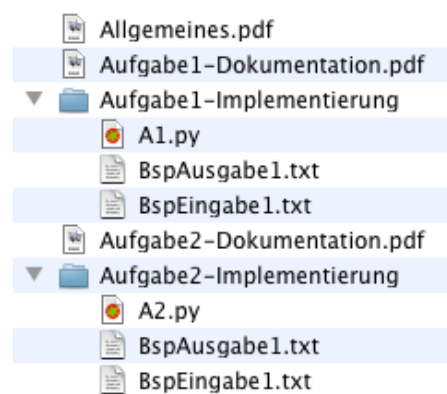
Die Dokumentation zu einer Aufgabe mit allen oben genannten Bestandteilen muss als PDF-Dokument eingereicht werden. **Es kann sein, dass für die Bewertung deiner Einsendung nur die Dokumentation herangezogen wird.** Sie sollte also einen lückenlosen und verständlichen Nachweis des Leistungsumfangs und 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 73400 und 78000.

Weitere Hinweise

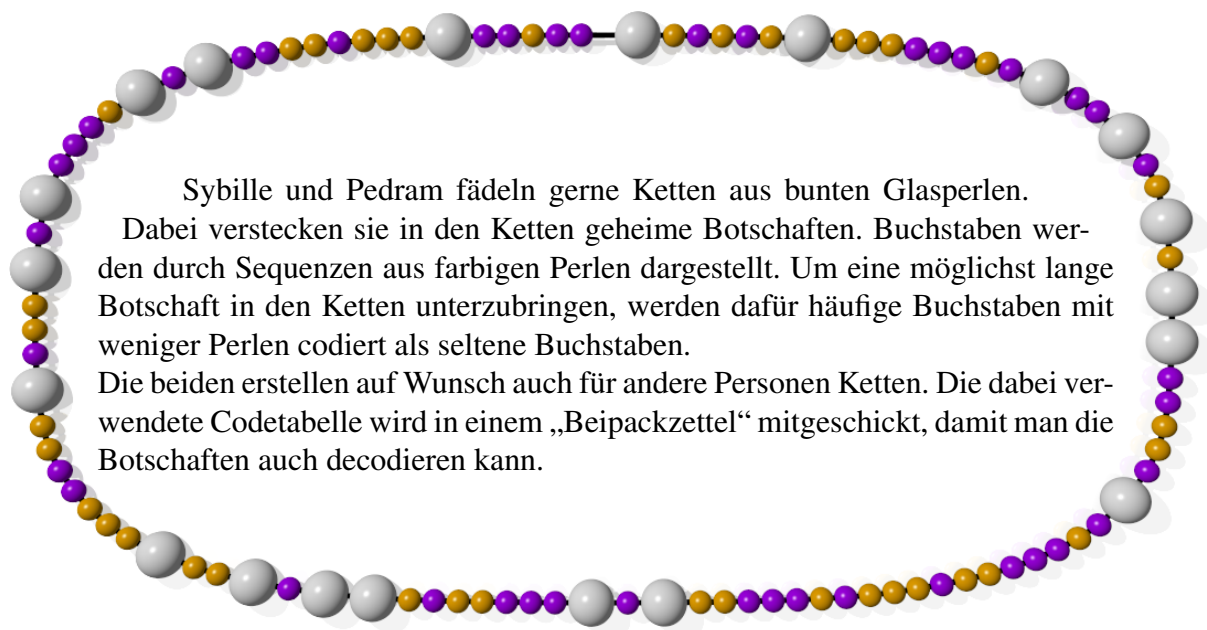
Bei der Bewertung können Programme unter Windows (10), Linux, Mac OS X (13.5) und Android ausgeführt werden.

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. Dort findest du auch einen Link zu unserer BWINF-Online-Community auf Discord; dort werden sicher wieder viele Teilnehmerinnen und Teilnehmer über die Aufgaben diskutieren – ohne Lösungsideen auszutauschen.

Allen Teilnehmern der 2. Runde wird bis Mitte Juni 2025 die Bewertung mitgeteilt. Die Besten werden zur Endrunde eingeladen, die 16.-19. September 2025 von der Technischen Universität München 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: Schmucknachrichten



Mit dem folgenden Beipackzettel kommt man zum Beispiel für die Botschaft **DIE SONNE SOLL DIR IMMER SCHEINEN** auf genau 113 Perlen. Mit einem Durchmesser von 1mm pro Perle kann man die Botschaft also in 11,3 cm unterbringen.

A	C	D	E	H	I
—●●●—	—●●●●—	—●●●●—	—●●●—	—●●●●●—	—●●●—
L	M	N	O	R	S
—●●●●—	—●●●●—	—●●●—	—●●●●—	—●●●●—	—●●●—

Die beiden erhalten von den Interessenten mehrere zu codierende Sätze. Auf Grundlage dieser Sätze wird eine Tabelle mit einem präfixfreien Code¹ erstellt, mit dem diese Sätze in einer möglichst kurzen Kette dargestellt werden können.

Aufgabe

- a) Sybille und Pedram möchten die Botschaften mit Hilfe von mehr als zwei Farben noch kürzer auffädeln. Alle Perlen haben denselben Durchmesser.

Schreibe ein Programm, das die Anzahl der Perlenfarben und einen Text mit einer Botschaft einliest, die aus beliebigen Unicode-Symbolen bestehen kann. Anschließend gibt das Programm eine für diese Botschaft optimierte Codetabelle aus. Außerdem soll die Gesamtlänge der Botschaft ausgegeben werden. Die Gesamtlänge soll möglichst klein sein!

¹In einem präfixfreien Code gibt es kein Codewort, das Präfix eines anderen Codewortes ist. Dadurch ist garantiert, dass jede Botschaft eindeutig dekodiert werden kann. <https://de.wikipedia.org/wiki/Pr%C3%A4fixcode>

- b) Die Perlenfirma hat die Bestellung nicht ordnungsgemäß ausgeführt. Die Perlen verschiedener Farben haben auch unterschiedliche Durchmesser. Ändere Dein Programm so, dass die Ketten aus diesen Perlen möglichst kurz sind.

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

Aufgabe 2: Simultane Labyrinth

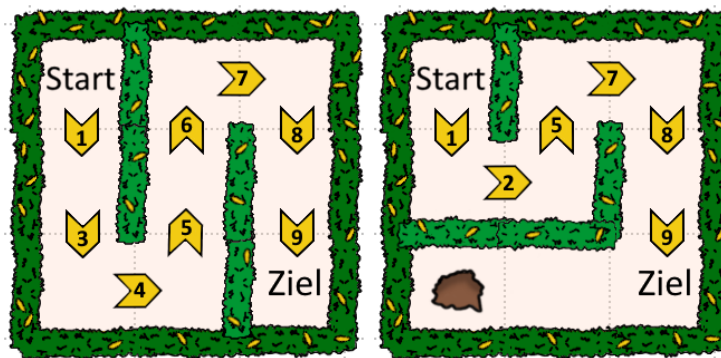
Anton, Bea und Chris haben von der neuesten Attraktion im Nachbardorf gehört: Bauer Knuth hat in seinem Maisfeld zwei Labyrinth angelegt. Beide haben dieselbe Grundgröße, ein rechteckiges Gitter mit vorgegebenen Seitenlängen n und m . Zwei Personen sollen in beiden Labyrinth gleichzeitig von Feld $(1, 1)$ starten und sich zum Zielfeld (n, m) bewegen.

Dabei werden sie von einer dritten Person mit ein und derselben Anweisungsfolge geleitet. Eine Anweisung gibt eine Richtung vor, in die sich beide Personen in einem Schritt um ein Gitterfeld bewegen sollen. Würde eine Person dabei gegen eine Wand laufen, so ignoriert sie die entsprechende Anweisung. Erreicht eine Person das Zielfeld vor der anderen, so bleibt sie dort stehen und wartet.

Die drei sind ins Nachbardorf aufgebrochen, und Chris wird Anton und Bea durch die Labyrinth leiten. Er hat sich eine Folge aus neun Anweisungen ausgedacht, mit denen beide zum Zielfeld gelangen:

1) \downarrow 2) \rightarrow 3) \downarrow 4) \rightarrow 5) \uparrow 6) \uparrow 7) \rightarrow 8) \downarrow 9) \downarrow

Die Bilder zeigen, wie Bea und Anton sich anhand dieser Anweisungsfolge durch die beiden Labyrinth bewegen:



Chris hätte stattdessen Bea mit sechs Anweisungen direkt zum Zielfeld leiten können. Dann wäre Anton jedoch im linken Labyrinth noch fast am Start und Chris bräuchte weitere sechs Anweisungen, damit auch Anton das Zielfeld erreicht. Insgesamt wären dann 12 Anweisungen erforderlich.

Bauer Knuths Art, Gäste anzulocken, macht die Runde, und bald sprießen die Labyrinthpaare aus den Ackerböden.

Aufgabe

- Entwirf einen Algorithmus, der für ein gegebenes Labyrinthpaar eine möglichst kurze Anweisungsfolge berechnet, mit der beide Personen das Zielfeld erreichen.
- Bald gibt es auch Labyrinth mit Gruben: Würde eine Person durch eine Anweisung in eine Grube fallen, so geht sie stattdessen zum Startfeld zurück. Erweitere Deinen Algorithmus so, dass er auch für solche Labyrinth funktioniert.

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

Aufgabe 3: Konfetti

Anna möchte an Ihrer Schule eine neue AG gründen. Da die Nachfrage hoch ist, hat sie einen Aushang erstellt, in dem sich Interessierte eintragen können. Dabei geben sie an, wann sie an der AG teilnehmen können. Also zum Beispiel so:

	16:00	17:00	18:00	19:00
Anna	✓	×	×	×
Bob	✓	✓	✓	×
Charlie	×	×	✓	✓
David	×	✓	✓	×

Anna hat den ausgefüllten Aushang mit nach Hause genommen, um die AG zu planen. Leider hat ihr kleiner Bruder den Aushang in die Hände bekommen, erst die Zeile mit den Uhrzeiten abgeschnitten und dann alle Spalten auseinander geschnitten. Kann Anna herausfinden, wie der Aushang ursprünglich aussah, obwohl sie die Anordnung der Spalten nicht mehr kennt?

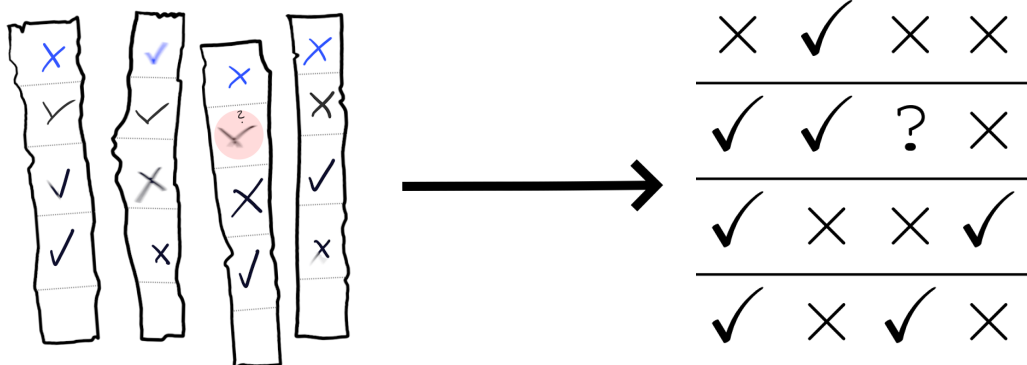
Leider hat Anna vergessen, was sie selbst eingetragen hatte. Zum Glück kann sie sich wenigstens daran erinnern, dass in jeder Zeile ein zusammenhängender Zeitblock mit '✓' markiert war. Für Anna ist eine Anordnung der Spalten „zulässig“, wenn sie diese Eigenschaft hat. Die folgende Anordnung ist zum Beispiel unzulässig:

Anna	×	×	×	✓
Bob	✓	✓	×	✓
Charlie	✓	×	✓	×
David	✓	✓	×	×

Aufgabe

- Schreibe ein Programm, das überprüft, ob sich die Spalten so umordnen lassen, dass eine zulässige Anordnung entsteht, und gegebenenfalls eine solche Anordnung ausgibt.
- Anna fragt sich, ob eine von deinem Programm gefundene Anordnung die einzige zulässige ist. Ändere dein Programm so, dass es diese Frage beantwortet.
- Was ändert sich, wenn Anna sich doch wieder an ihre eigene Zeile erinnert? Ändere dein Programm entsprechend.

- d) Leider hat Anna die Spaltenschnipsel im Garten gelassen und einen Regenschauer zu spät bemerkt. Jetzt sind auch noch einige Einträge unleserlich geworden. Sie liest also zum Beispiel:



Ändere dein Programm so, dass es auch mit unleserlichen Einträgen ? umgehen kann, indem es jedes davon in ✓ oder ✗ so ändert, dass eine zulässige Anordnung möglich wird.

- e) Obwohl dein Programm aus Teilaufgabe d) korrekt ist, findet es keine zulässige Anordnung. Anna vermutet daher, auch einige der Zeichen ✓ oder ✗ falsch abgelesen zu haben. Schreibe ein Programm, das für eine gegebene Menge von gleich langen Spalten aus den Zeichen ✓, ✗ und ? möglichst wenige Einträge so ändert, dass eine zulässige Anordnung existiert, und eine solche auch ausgibt.

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