<iob> .olympiade { fach: “informatik”; bundesland: “brandenburg”; }

9. Informatik-Olympiade vom Land Brandenburg

– Vorrunde –

Hinweise:
Die Aufgaben werden von einzelnen Personen bearbeitet und unter Erbringung der eigenen Leistung.
Die Abgabe (**Abgabetermin = 26.01.2026**) erfolgt (inklusiv der kompletten Namens- und Schul-Nennung) an informatik@blis-brandenburg.de. Häufig kommt es zu Problemen bei dem Empfang von exe-Dateien. In diesem Fall bitte zunächst versuchen, die exe-Datei einfach als txt abzuspeichern mit dem Hinweis, dass es sich um eine exe-Datei handelt. Im Notfall geht auch ein Link zu einem Upload-Ordner von euch, ansonsten schicken wir auf Anfrage einen Link zu einem Upload-Ordner.
Der Abgabe-Ordner soll einen Programm-Ordner mit den kompletten Dateien enthalten, sodass das Programm ausführbar ist, sowie eine erklärende Dokumentation zum Programm (Aufgabe 2) und das Dokument zur Lösung der theoretischen Aufgabe (Aufgabe 1). Bei den Dokumenten sollen möglichst auch visuelle Darstellungen die Erklärung unterstützen.

Auswirkungen der Vorrunde auf den Hauptwettbewerb (Informatik-Olympiade am 28./29.05.2026):
Die beiden Aufgabentypen bereiten euch als Teilnehmer*innen gleichzeitig auf die beiden Olympiaden-Tage vor.

- Die Teilnahme an der Vorrunde erzeugt ein Ranking für die Teilnehmerbewerbung am Hauptwettbewerb, sodass eine vorzeitige Bekanntgabe der persönlichen Teilnahme bei der Olympiade ermöglicht werden kann.
- Die Teilnahme an der Vorrunde ist auch in diesem Jahr weiterhin noch KEINE Voraussetzung zur Teilnahme am Hauptwettbewerb im Juni, allerdings werden die Teilnehmer aus der Vorrunde bevorzugt zur Hauptrunde zugelassen.

Alle Teilnehmer*innen der Vorrunde starten, je nach Platzierungen (Platz 1-3 und dann der Rest), mit unterschiedlichem Punktevorsprung in den Hauptwettbewerb.

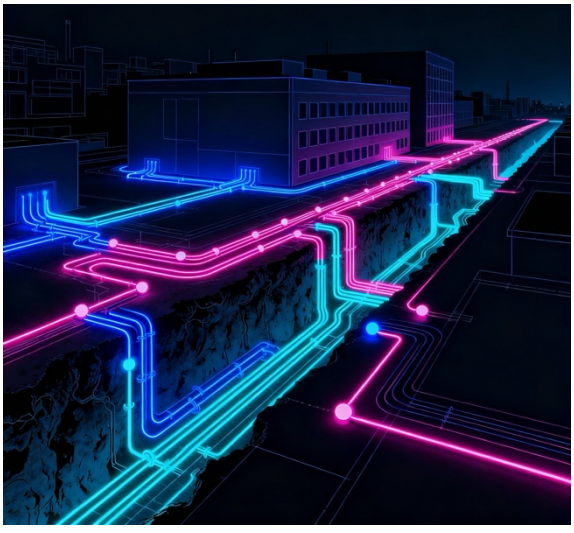
Aufgabe 1: Der mysteriöse Netzwerkplan

Ein Forscherteam plant ein Netzwerk aus unterirdischen Kommunikationskanälen in einer Stadt. Sie haben sieben Stationen A–G. Es sind nur bestimmte direkte Verbindungen möglich (siehe Liste). Jede Station im Netzwerk ist ein Gebäude und die Kanäle mit jeweils einer Länge in Kilometern verbinden bestimmte Gebäude miteinander. Alle Stationen sollen so verbunden werden, dass

- das gesamte Netz **alle Gebäude verbindet**
- **keine geschlossenen Schleifen** entstehen
- **jede einzelne Verbindung darf höchstens 8 km lang** sein (Verbindungen mit Länge > 8 sind verboten)
- die **Gesamtlänge aller verwendeten Verbindungen** soll **minimal** sein, da der Bau der Kommunikationskanäle sehr teuer ist

Liste der möglichen Verbindungen und ihre Längen in km:

- A — B : 4
- A — C : 3
- A — D : 7
- A — G : 11 ← *verboten (länge > 8)*
- B — C : 2
- B — D : 9 ← *verboten*
- B — E : 6
- C — D : 5
- C — E : 4
- C — G : 10 ← *verboten*
- D — F : 8
- E — F : 1
- E — G : 7
- F — G : 3



Aufgabe: Zeichnen Sie einen vollständigen Plan des gültigen Netzwerks, das alle Bedingungen erfüllt, und begründen Sie, warum Ihr Plan die minimalsten Kosten für das Forscherteam erzeugt hat und demnach die optimalste Lösung ist. Überlegen Sie, wie man bei beliebigen Netzwerken allgemein vorgehen kann, um das jeweils minimalste Netzwerk zu ermitteln.

Aufgabe 2: Die rätselhafte Musikbox

2.1 Szenario

Ein alter Tüftler findet auf einem Flohmarkt eine geheimnisvolle Musikbox. Wenn er auf den Knopf drückt, ertönt eine kurze Melodie als Sequenz aus Tönen (repräsentiert durch Buchstaben A–Z). Auf der Rückseite entdeckt er eine Liste mit Tonlängen (in Sekunden), die aber offenbar nicht in der Reihenfolge abgespielt werden, in der sie notiert sind. Die Musikbox spielt scheinbar eine verschlüsselte Melodie, und der Tüftler möchte wissen, welche möglichen ursprünglichen Melodien die Box gespielt haben könnte. Das Problem: Manche Töne werden durch einen defekten Sensor nicht erkannt, und es wird nur die Gesamtdauer der Melodie angezeigt.
Der Tüftler kennt:

- die Liste aller möglichen Töne mit deren Dauer
- die Anzahl der Töne, die gespielt wurden
- die Gesamtdauer der Melodie

2.2 Aufbau der Eingabe und Ausgabe

Das Programm erhält:

1. Ein Dictionary toene mit Tönen → Dauer in Sekunden
   Beispiel: { "A": 1, "B": 2, "C": 3, "D": 4, "E": 5 }
2. Eine Zahl n = Anzahl der Töne in der Sequenz
3. Eine Zahl d = Gesamtdauer in Sekunden

Ausgegeben werden soll:

- Alle möglichen Sequenzen (Listen von Tonbuchstaben), die genau n Töne enthalten und deren Gesamtdauer = d ist.
- Die Sequenzen sollen in lexikographischer Reihenfolge sortiert sein.
- Wenn es keine Lösung gibt → Ausgabe "Keine mögliche Sequenz gefunden"

2.3 Beispiel für die Grundaufgabe

Eingabe: toene = { "A": 1, "B": 2, "C": 3 }; n = 3; d= 6
Ausgabe: [["A","B","C"], ["A","C","B"], ["B","A","C"], ["B","C","A"], ["C","A","B"], ["C","B","A"]]
Erklärung:

- Jede Sequenz muss 3 Töne haben.
- Die Dauer jeder Sequenz muss exakt 6 Sekunden betragen.
- Mögliche Lösungen wären:
  - ["A","B","C"] → 1+2+3=6
  - ["A","C","B"] → 1+3+2=6
  - ["B","A","C"] → 2+1+3=6
  - ["B","C","A"] → 2+3+1=6
  - ["C","A","B"] → 3+1+2=6
  - ["C","B","A"] → 3+2+1=6

2.4 Aufgabe

a) Schreiben Sie ein Programm, welches für beliebige Eingaben (nach dem Eingabeschema) alle möglichen Tonfolgen bestimmt, die die oben genannten Kriterien des Tüftlers erfüllen.
b) Ergänzen Sie ihr Programm so, dass folgendes zusätzlich bestimmt wird: Die Anzahl aller möglichen unterschiedlichen Sequenzen ohne Einschränkungen durch n und d. Die minimalste und maximalste Gesamtdauer und Sequenz, sofern d nicht vorgegeben wird. Die minimalste und maximalste Zahl n inklusive Sequenz, sofern n nicht vorgegeben ist. (Ausgabe bitte nachvollziehbar gestalten!)
c) Erzeugen Sie Beispieldateien zur Eingabe in das Programm mit verschiedenstem Schwierigkeitsgrad, sodass auch mal der Fall eintritt, dass keine mögliche Sequenz gefunden wird.