Day 1: Historian Hysteria

Der Tag 1 ist tatsächlich eine sehr leichte Aufgabe und damit ein perfekter Einstieg für jede/-n in den diesjährigen AoC! Lies dir zunächst die offizielle Aufgabenbeschreibung durch: 2024 - Day 1

Teil 1

Tipps zur Vorgehensweise folgen weiter unten - hier sind zunächst nur wichtige Java-Methoden aufgelistet. Dieser Tag ist so leicht, dass du das auch hinbekommen kannst, ohne viele Tipps zu lesen!

Wichtige Java-Methoden:

- `ArrayList<Integer> varName = new ArrayList();` Eine ArrayList eignet sich am besten, um eine **variable** Anzahl von Elementen zu speichern. Dies trifft generell immer auf **alle** AoC-Aufgaben zu (Arrays werden eher selten benötigt). Denke daran, dass du dazu auch ganz oben `import java.util.ArrayList;` aufrufen musst.
- Mit folgender *for-each* Schleife kannst du am einfachsten über jede Zeile der Eingabedatei iterieren: `for (String line: inputLines) {…}`
- Mit `line.split("…")` bekommst du ein Array von Strings zurückgeliefert, das den ursprünglichen String `line` an denjenigen Stellen aufgeteilt hat, die im Anführungszeichen stehen. Das String-Element, an dem geteilt wird, verschwindet dabei! Nutze das, um jede Zeile in die linke und rechte Zahl zu splitten. An was für einem String-Element musst du die Trennung durchführen?
- Mit `Integer.parseInt(…)` kannst du einen String in einen Integer umwandeln. Damit kannst du pro Schleifendurchlauf folgenden Code verwenden, um an die **linke** Zahl zu gelangen: `Integer.parseInt(line.split("…")[0])` (für die rechte Zahl musst du nur eine Kleinigkeit ändern).
- Mit `liste.add(…)` kannst du ein Element zu einer ArrayList namens `liste` hinzufügen.
- Mit `Collections.sort(liste)` kannst du eine unsortierte `liste` sortieren. Dazu musst du ganz oben `import java.util.Collections;` aufrufen!

Im Wesentlichen besteht die erste Teilaufgabe aus folgenden einzelnen Schritten:

Vorgehensweise

- Initialisiere zwei ArrayLists für Integerzahlen - je eine für die linken und eine für die rechten Zahlen.
- Iteriere über jede Zeile und füge dabei die linke und rechte Zahl in die jeweile ArrayList hinzu.
- Sortiere beide Arrays.
- Merke dir nun die akkumulierenden ("aufaddierenden") Differenzen (beginnend mit 0).
- Iteriere über jedes Listenelement: Addiere zur Lösung (den Differenzen) den **Betrag** (`Math.abs(…)`) der Differenz der linken und der rechten Zahl.
- Nach dem Schleifendurchlauf kannst du die Differenzen ausgeben/zurückgeben.

Lösungsvorschlag

```
public void partOne() {
    ArrayList<Integer> left = new ArrayList();
    ArrayList<Integer> right = new ArrayList();

    for (String line: inputLines) {
        left.add(Integer.parseInt(line.split("   ")[0]));
        right.add(Integer.parseInt(line.split("   ")[1]));
    }

    Collections.sort(left);
    Collections.sort(right);

    int diff = 0;

    for (int i = 0; i < left.size(); i++) {
        diff += Math.abs(left.get(i)-right.get(i));
    }

    System.out.println(diff);
    // Ich habe die Methode zu einer void-Methode umgewandelt, anstatt einen int-Rückgabewert zu verlangen.
    // Die Lösung kann man so einfacher aus der Konsole kopieren.
}
```

Teil 2

Vorgehensweise

- Die ersten Schritte bis inklusive dem zeilenweisen Einlesen alles Zahlen in zwei ArrayLists ist identisch! Anschließend muss die Berechnung aber anders ablaufen.
- Die Zahlen müssen nun **nicht** sortiert werden.
- Iteriere in einer Schleife über alle linken Zahlen. Pro linker Zahl iterierst du in einer weiteren Schleife über **alle** rechten Zahlen und vergleichst, ob die link Zahl mit einer der rechten Zahlen übereinstimmt. **Wichtig:** Du musst hier zum Vergleichen mit `equals(…)` arbeiten - also: `linkeZahl.equals(rechteZahl)`. Zähle die Anzahl der Übereinstimmungen. Anschließend addierst du das Produkt der Übereinstimmungen und der linken Zahl zum Ergebnis.

Lösungsvorschlag

```
public void partTwo() {
    ArrayList<Integer> left = new ArrayList<>();
    ArrayList<Integer> right = new ArrayList<>();

    for (String line: inputLines) {
        left.add(Integer.parseInt(line.split("   ")[0]));
        right.add(Integer.parseInt(line.split("   ")[1]));
```

```
    }

    int similarityScore = 0;

    for (int l = 0; l < left.size(); l++) {
        int times = 0;
        for (int r = 0; r < right.size(); r++) {
            if (right.get(r).equals(left.get(l))) {
                times++;
            }
        }

        similarityScore += times * left.get(l);
    }

    System.out.println(similarityScore);
}
```