

Meine Aufgaben Ratings:



- Best Case: Ihr macht alle Aufgaben (, wenn die Zeit reicht)
- Falls nicht: Ich mache Ratings zur Wichtigkeit der einzelnen Aufgaben
- (Keine offizielle Empfehlung, meine subjektive Meinung auf Basis meiner eigenen Erfahrung als Student in diesem Kurs)
- ! Sagen nichts über die Schwierigkeit der Aufgaben aus !

Wichtig für die Prüfung:
(Prüfungs- oder
prüfungsähnliche Aufgaben)



Wichtig für euer Verständnis:
(Aber nicht in Prüfungsform)



Wichtig für tiefes Verständnis/
Zusatzaufgaben:
(Bspw. viele vom gleichen Typ)



Aufgabe 1: Binärdarstellung



Schreiben Sie ein Programm “Binaer.java”, das eine positive Zahl Z einliest und dann die Binärdarstellung druckt. (Hinweis: Finden Sie zuerst die grösste Zahl k , so dass die Zweierpotenz $K = 2^k$ kleiner als Z ist.)

Beispiel: Für $Z = 14$ sollte Ihr Programm 1110 ausgeben.



Aufgabe 2: Grösster gemeinsamer Teiler

Schreiben Sie ein Programm “GGT.java”, das den grössten gemeinsamen Teiler (ggT) zweier ganzer Zahlen mithilfe des Euklidischen Algorithmus berechnet. Hierbei handelt es sich um eine iterative Berechnung, die auf folgender Beobachtung basiert:

1. Wenn x grösser oder gleich y ist und sich x durch y teilen lässt, dann ist der ggT von x und y gleich y ;
2. andernfalls ist der ggT von x und y der gleiche wie der ggT von y und $x \% y$.

Üben Sie diesen Algorithmus zuerst von Hand an ein paar Beispielen und schreiben Sie dann das Java Programm.

Beispiel: Für $x = 36$ und $y = 44$:

$$\text{ggT}(\underbrace{36}_x, \underbrace{44}_y) \stackrel{2.}{=} \text{ggT}(44, \underbrace{36 \% 44}_{36}) \stackrel{2.}{=} \text{ggT}(36, \underbrace{44 \% 36}_8) \stackrel{2.}{=} \text{ggT}(8, \underbrace{36 \% 8}_4) \stackrel{1.}{=} 4$$

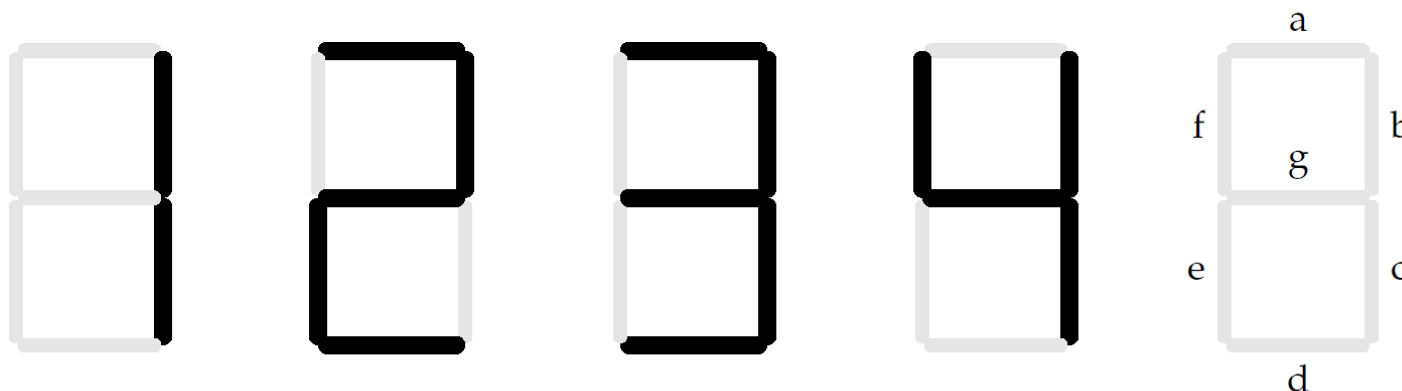
Aufgabe 3: Zahlenerkennung



Für diese Aufgabe verwenden wir einen String um die erleuchtenden Segmente einer **Sieben-segmentanzeige** zu kodieren. Die Segmente sind, wie im Bild gezeigt, von a bis g nummeriert. Die Kodierung einer möglichen Anzeige ist ein String, in welchem der Buchstabe 'x' genau dann vorkommt, wenn das 'x'te Segment der Anzeige erleuchtet ist. Zum Beispiel wird die Zahl 2 kodiert durch 'abged'. Zur Einfachheit darf angenommen werden, dass kein Buchstabe mehr als einmal in der Kodierung vorkommt und dass nur die Zahlen 0 bis 9 kodiert werden.

Schreiben Sie ein Programm "Zahlen.java", das einen String, der eine Anzeige kodiert, einliest und die kodierte Zahl als Integer ausgibt. Überlegen Sie wie viele IF Blöcke benötigt werden um jede Zahl zu erkennen.

Tipp: Sie können `str.contains("a")` verwenden, um zu überprüfen, ob ein String `str` den Buchstaben 'a' enthält.





Aufgabe 4: Berechnungen

2. Implementieren sie die Methode `Calculations.magic7(int a, int b)`. Die Methode gibt einen Boolean zurück. Die Methode soll `true` zurückgeben, wenn einer der Parameter 7 ist oder wenn die Summe oder Differenz der Parameter 7 ist. Ansonsten soll die Methode `false` zurückgeben.

Beispiele

- `magic7(2,5)` gibt `true` zurück.
- `magic7(7,9)` gibt `true` zurück.
- `magic7(5,6)` gibt `false` zurück.

Hinweis: Mit der Funktion `Math.abs(num)` können Sie den absoluten Wert einer Zahl `num` erhalten.

Aufgabe 4: Berechnungen



3. Implementieren Sie die Methode `Calculations.fast12(int z)`. Das Argument `z` ist nicht negativ. Die Methode gibt einen Boolean zurück. Die Methode soll `true` zurückgeben, wenn `z` nahe an einem Vielfachen von 12 ist. Eine Zahl `x` ist nahe an einer Zahl `y`, wenn eine der Zahlen um maximal 2 grösser oder kleiner ist als die andere Zahl. Ansonsten soll die Methode `false` zurückgeben.
- `fast12(12)` gibt `true` zurück.
 - `fast12(14)` gibt `true` zurück.
 - `fast12(10)` gibt `true` zurück.
 - `fast12(15)` gibt `false` zurück.

Aufgabe 5: Scrabble



In dieser Aufgabe sollen Sie Scrabble-Steine legen, mittels ASCII-Art auf der Konsole. Vervollständigen Sie die Methode `drawNameSquare` in der Klasse `Scrabble`. Diese Methode nimmt einen Namen als String-Parameter und soll den Namen als in einem Quadrat angeordnete Scrabble-Steine auf der Konsole (`System.out`) ausgeben. Wenn z.B. der String `Alfred` übergeben wird, sollte folgendes Bild ausgegeben werden:

```
+---+---+---+---+---+---+
| A | L | F | R | E | D |
+---+---+---+---+---+---+
| L |                               | E |
+---+                               +---+
| F |                               | R |
+---+                               +---+
| R |                               | F |
+---+                               +---+
| E |                               | L |
+---+---+---+---+---+---+
| D | E | R | F | L | A |
+---+---+---+---+---+---+
```

(Alte Prüfungsaufgabe:

<https://exams.vis.ethz.ch/exams/w0cikcqb.pdf>)

(Achtung: Auch 1 Sterne Prüfungsaufgaben sind mittlerweile schwieriger geworden)

Aufgabe 5: Scrabble

Ihr Code braucht nur Namen der Länge 3 oder länger unterstützen. Für einen Namen mit Länge 3, z.B. Jim, sollte die Ausgabe so aussehen:

```
+---+---+---+
| J | I | M |
+---+---+---+
| I |   | I |
+---+---+---+
| M | I | J |
+---+---+---+
```

Beachten Sie, dass Ihr Programm keinerlei andere Ausgabe als das Scrabble-Quadrat machen darf.