1. Test –Einführung in JAVA - BITEC-01

09.10.2024 – Cammerlander

- Sie haben 120 Minuten Zeit die Aufgaben zu lösen
- Sie können maximal 50 Punkte erreichen
- Es sind zur Prüfung zugelassen:
    - Taschenrechner (wenn erwünscht)
    - Wasserflasche
    - Geodreieck, Stifte, usw.
    - **am Computer sind alle Unterlagen sowie die Nutzung des Internets erlaubt**.
    - Die Nutzung des Internets umfasst nicht
        - Chatbots
        - Teilen der eigener Lösungen im Netz
        - sonstige Kommunikation mit anderen Usern
    - Die Nutzung von allen anderen Dingen, muss vorher mit mir abgesprochen werden (z.B. Nutzung von Ohropax), ansonsten wird dies als schummeln gewertet.
    - Die Folge des Schummeln ist eine Bewertung mit 0 Punkten.
- Die Abgabe des Programmcodes erfolgt über Teams (mir ein ZIP File als Nachricht schicken).
- Viel Erfolg! :)

Notenschlüssel:

[0-50): N5; [50-62.5%): G4; [62.5-75%): B3; [75-87.5%): G2; [87.5-100%]: S1., (Schulnotensystem)

Name: _____

Bewertung (vom LV-Leiter auszufüllen):

Aufgabe 1.1	__ / 22
Aufgabe 2	__ / 28
Summe:	__ / 50

Aufgabe 1.1 - (22 Punkte)

Programmverständnis:

Erstelle eine JAVA Klasse mit einer main Methode und füge den unten gegebenen Code dort ein. Lösen Sie dort die folgenden Aufgaben:

```
Integer[] zahlen = {28, 26, 6, 4, 2};

Integer platzhalter;

for (int j = 0; j < zahlen.length - 1; j++) {

   for (int i = 0; i < zahlen.length - 1; i++) {

      if (zahlen[i] > zahlen[i+1]) {

         platzhalter = zahlen[i];
         zahlen[i] = zahlen[i+1];
         zahlen[i+1] = platzhalter;

      }

   }

}
```

a) Gegeben ist ein Sortieralgorithmus (Bubble Sort). Beantworten Sie die folgenden Fragen in eigenen Worten:
   i) Was ist die Aufgabe der äußeren For-Schleife?
   ii) Was ist die Aufgabe der inneren For-Schleife?
   iii) Was ist die Aufgabe der If-Anweisung?
   iv) Was ist die Aufgabe der Variable Platzhalter?
   v) Was würde passieren wenn wir ohne dem Platzhalter arbeiten würden? Also innerhalb der IF-Anweisung
   `zahlen[i+1]= zahlen[i];`
   `zahlen[i] = zahlen[i+1];`
   schreiben würden?

b) Ersetze in der inneren Schleife `zahlen.length - 1` durch `zahlen.length`. Beschreibe warum es zu einem Fehler kommt.

c) Beide Schleifen zählen in unseren Beispiel von 0 bis 4. Funktioniert der Algorithmus noch wenn wir diese vertauschen? Erkläre warum ja oder nein.

d) Mache nun b) Rückgängig. Wie oft wiederholen wir die innere und äußere Schleife zusammen?
   Erstelle dazu eine Variable **wiederholungen** vom Typ **Integer** und berechne mit dieser die Anzahl der Wiederholungen beider Schleifen.
   Optional: Geht das auch „im Kopf“? Wie schaut die Rechnung dafür aus?

e) Gib die Zählvariable i für jeden Schleifendurchlauf der inneren Schleife zusammen mit dem zu sortierenden Array aus. Was fällt bei der Ausgabe auf?

f) Ersetze nun bei der inneren Schleife `zahlen.length - 1` mit `zahlen.length - 1 - j`. Was fällt nun bei der Ausgabe auf? Versuche in Worten zu beschreiben warum wir hiernicht `zahlen.length - 1` Schleifendurchgänge brauchen, sondern nur `zahlen.length - 1 - j`.

Aufgabe 2.1 – (28 Punkte)

Schleifen, **Verzweigungen** und **User Input** und **Arrays** über Konsole:

Das folgende Programm generiert zufällige Kommentare eines Online-Warenhauses.

(Lesen Sie zuerst die Puntke 1.-3. der Aufgabe durch)

1. **Kurze Kommentare**
    a. Hier sollen Kommentare erzeugt werden welche folgendermaßen aufgebaut sind.
    Ein **Kommentar** ist eine **Ansammlung von Sätzen**.
    Dieses soll **zuerst** aus **einem Satz** bestehen.
    Dieser Satz besteht aus einem **Prefix** (z.B. „Was aber nicht vergessen werden darf, “) und **Postfix.** Beide werden durch **concatenation (+)** zuammengefügt.
    (a) **neutraler Postfix** (z.B. „es tut was es soll.“), ein
    (b) **positiver Postfix** (z.B. „es ist genial, dass es zusätzlich einen Regler hat um das Licht zu dämmen!“) und ein
    (c) **negativer Postfix** (z.B. „die Staatsanwaltschaft ermittelt wegen Betrug.“) sein.

    Das **Kommentar** soll aus einem **Satz** bestehen welcher:
    1. einen **zufälligen Prefix** aus einem Pool von Kandidaten hat und
    2. **zufällig einen neutral**, **positiv** oder **negativ Postfix** hat.
    3. **Dazu** kommt nach Wahl ob dieser Postfix neutral, positiv oder negativ ist, eine **zufällige Wahl** einses **Postfix** aus einem Pool von mehreren Kandidaten.

    Eine Vorlage ist weiter unten zu finden. Es muss also nichts erfunden werden.

    Geben Sie das generierte Kommentar auf der Konsole aus.

    b. Fragen Sie am Schluss den User ob er mit diesem generierten Kommentar zufrieden ist, wenn ja beendet das Programm, wenn nein, wird neu generiert.

2. Längere Kommentare

a. Zusätzlich soll der **User nun die Anzahl der Sätze** in einem Kommentar **bestimmen** können. Diese Sätze sind folgendermaßen aufgebaut:

i. Einem neutralen Satz folgen positive, neutrale, oder negative Sätze.
ii. Einem positiven Satz folgen nur mehr positive Sätze.
iii. Einem negativen Satz folgen nur mehr negative Sätzen.

Beschränken Sie sich auf eine Anzahl der Sätze [1 bis 4].

Optional: Beliebige Anzahl an Sätzen möglich.

Optional: Doppelte Sätze im Kommentar ausschließen.

Anzahl	1	2	3
Sätze	**neutral** ODER **positiv** ODER **negativ**	**neutral + neutral** ODER **neutral + positiv** ODER **neutral + negativ** ODER **negativ + negativ** ODER **positiv + positiv**	**neutral + neutral + neutral** ODER **neutral + neutral + positiv** **ODER** **neutral + neutral + negaiv** ODER **neutral + positiv + positiv** ODER **neutral + negativ + negativ** ODER **positiv + positiv + positiv** ODER **negativ + negativ + negativ**

b. Schreibe die Möglichkeiten für 4 Sätze als Kommentar in deinem JAVA Code?

3. „User Experience“

a. Verwenden Sie die **Aufgabe 1 oder 2** um ein **Kommentar** zu **generieren**. Fordern Sie den **User** auf eine **Beschreibung des Produktes** einzugeben.
(z.B. „Staubsauger Diedsoon Aftershock, HIER link zur Hompage, am 20.06.2024 gekauft.“). **Fragen** Sie am Schluss noch ob der **User zufrieden** ist, **ansonsten** erlauben Sie dem User eine **Neue Eingabe der Beschreibung** (das Kommentar wird aber gespeichert und nicht neu eingegeben!).

b. Fragen Sie danach den User ob dieser, mit dem **Kommentar,** der **Beschreibung** oder **beidem zufrieden** ist.

i. **Wenn** dieser mit **beidem zufrieden** ist, **terminiert das Programm** und das Kommentar wird „veröffentlicht“ (sout auf die Konsole).
ii. Wenn dieser mit dem **Kommentar zufrieden ist**, dann wird kann die **Beschreibung geändert werden.**
iii. Wenn dieser mit der **Beschreibung zufrieden ist**, dann wird kann das **Kommentar geändert werden.**

Aufgabe 2.2 - (OPTIONAL)

Schleifen, **Verzweigungen** und **User Input** und **Arrays** über Konsole:

Versuchen Sie die Aufgabe 7 und 8 des Probetests. Hier nochmal die gesamte Angabe.

Generiere Muster, welche vom User gewählt werden, sowie die benötigten Parameter der Muster.

Zuerst soll der User gefragt werden welches Muster zu generieren wünscht. Die Eingabe des Users soll der Name des Musters sein (z.B. **Dreieck, Pyramide, Raute**). Weitere Kategorien der Muster sind nach Eingabe des „Hauptmusters“ zu erfolgen
(z.B. **Pyramide** dann **spitze rechts**, usw.). Falls danach weitere Usereingaben nötig sind, sind diese danach abzufragen (z.B. **Pyramide** dann **spitze rechts**, dann **~** und **%**).

1. **Dreieck** (5 Punkte)
   Hier soll der User die Höhe des Dreieicks steuern können. Die Höhe ist hier die **Anzahl der Zeilen.** Lesen Sie dazu den benötigten Userinput ein.

```
//          #
//          ##
//          ###
//          ####
//          #####
//          ######
```

2. **Pyramide - spitze rechts** (5 Punkte)
   Hier soll der User die Höhe der Pyramide steuern können. Die Höhe ist hier die **Anzahl der Spalten.** Lesen Sie dazu den benötigten Userinput ein.

```
//          #
//          ##
//          ###
//          ####
//          #####
//          ######
//          #####
//          ####
//          ###
//          ##
//          #
```

3. **Pyramide - spitze oben** (6 Punkte)
   Hier soll der User die Höhe der Pyramide steuern können. Die Höhe ist hier die **Anzahl der Zeilen.** Lesen Sie dazu den benötigten Userinput ein.

```
//               #
//              ###
//             #####
//            #######
//           #########
//          ###########
```

4. **Pyramide - spitze links** (6 Punkte)
   Hier soll der User die Höhe der Pyramide steuern können. Die Höhe ist hier die **Anzahl der Spalten.** Lesen Sie dazu den benötigten Userinput ein.

5. **Raute** (6 Punkte)
   Hier soll der User die **Länge einer der Diagonalen** der Raute steuern können. Lesen Sie dazu den benötigten Userinput ein.

6. **Muster** (6 Punkte)
   Hier soll der User die **Länge einer der Diagonalen** der Raute steuern können. Lesen Sie dazu den benötigten Userinput ein.
7. **Füllung** (6 Punkte)
   Es soll dem User möglich sein, die „Füllung“ der Aufgaben 1.-6. bestimmen zu können (bis jetzt wars #). Zudem soll es dem User möglich sein jede n’te Zeile mit einer gewählten „Füllung“. Hier bedeutet z.B. „n’te = 3“, dass der User 3 eingibt, und dadurch jede 3. Zeile mit einem anderen Symbol zu befüllen ist.

8. „Steigung“ der Aufgaben 1-6. (BONUS)

Hier soll durch die Eingabe der „Steigung“ gesteuert werden wie „spitz“ das generierte Muster ist. Vergessen Sie nicht, Sie können das Internet verwenden!

z.B.

Steigung 1 bedeutet dass $\frac{\Delta y}{\Delta x} = 1$, y bedeutet hier die vertikale (Zeilen) und x die horizontale (Spalten).

- Lösen Sie zuerst das Problem mit Steigung kleiner als 1 und danach größer als 1.
- Achtung! Da die Höhe (auf der y Achse) vom User fixiert ist, müss solange die Schritte in x gegangen werden, bis diese Höhe erreicht ist!
- Wählen Sie frei ob sie wenn die Steigung nicht genau dargestellt werden kann, ob sie floor, ceiling oder round verwenden. Dies beeinflusst das generierte Muster, sie sind jedoch alle richtig. Es wird round empfohlen.
- Achtung! Gehen Sie davon aus dass Δx und Δy eines Symbols das gleiche ist! Damit ist folgendes gemeint.
  z.B. die Eingabe „Steigung 1“ hat folgendes Bild zur folge, obwohl in echt, hier nicht die Steigung 1 vorliegt da ein Symbol höher als breit ist. Wir ignorieren das aber, da das nur ein Datstellungsproblem ist!

```
//               #
//              ##
//             ###
//            ####
//           #####
//          ######
```

Hier ein Beispiel zur Steigung $0.67 = \frac{\Delta y}{\Delta x}$ und ein möglicher Lösungsversuch:

Stellen Sie sich auf die linke ecke, dort ist dort ist die koordinate $x = 1$. Wir gehen hier nach rechts weiter. Wenn wir wissen wollen welches feld von der Linie berührt wird, sagt uns $y = \frac{\Delta y}{\Delta x} * x$ das Feld. $\frac{\Delta y}{\Delta x}$ kommt vom user als Eingabe, x und y ist die Position eines Arrays.
Also $0.67 * 1 = 0.67$. Bedeutet gerundet $x = 1$ und $y = 1$.
Für $x = 2$ $y = 0.67 * 2 = 1.33$ und gerundet $y = 1$. Wenn wir das weiter machen, haben wir

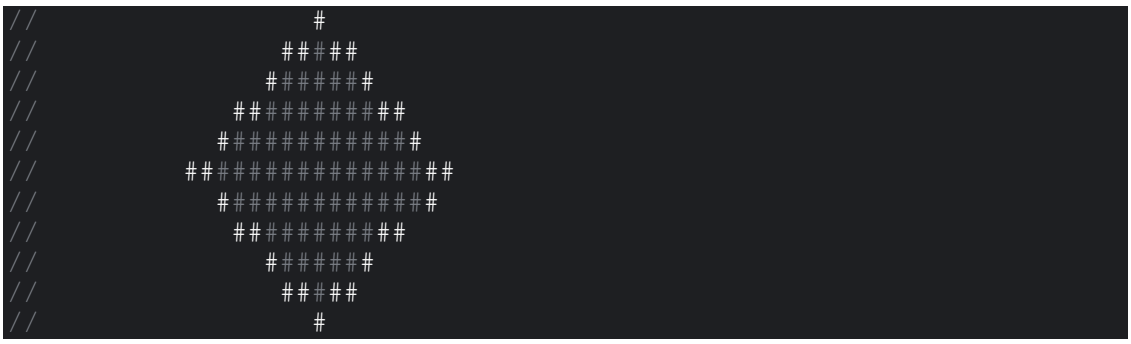
x	1	2	3	4	5	6	7	8	**9**
y	0.67	1.33	2	2.67	3.33	4	4.67	5.33	6
runden	1	1	2	3	3	4	5	5	**6**

Wir haben also in der letzten Reihe **9** Symbole und **6** Symbole als höhe, was wieder $\frac{\Delta y}{\Delta x} = \frac{\mathbf{6}}{\mathbf{9}} = 0.67$ ergibt. Die Zeile „runden“ in der Tabelle ergibt die weißen ‚#‘.

Verwende ein 2d-Array um diese Linie anzulegen. Danach suche das Symbol ‚#‘ in jeder Zeile und fülle nach rechts auf.

Für kompliziertere Muster, wie die Raute, teile das Problem in 4 kleinere Probleme (das wir und gerade angeschaut haben ist eines davon) und füge diese danach zusammen. Also

Ergibt



Beachte die Eingabe des Users (länge er Diagonale, nicht die Höhe).