TU Berlin
Software and Business Engineering
Prof. Dr. Ingo Weber
Dr. Tobias Heindel
Richard Hobeck

WiSe 2021/22
Aufgabe 4
Block 2
15. November 2021

Aufgabenblatt 4 *Programmieren I*

Hinweise

- Die Abgabe dieser Übungsaufgaben muss bis spätestens Sonntag, den 21. November 2021 um 23:59 Uhr im ISIS-Kurs erfolgt sein. Es gelten die Ihnen bekannten Übungsbedingungen.
- Lösungen zu diesen Aufgaben sind als gezippter Projektordner abzugeben. Eine Anleitung zum Zippen von Projekten finden Sie auf der Seite des ISIS-Kurses. *Bitte benutzen Sie einen Dateinamen der Form* `VornameNachname.zip`.
- Bitte beachten Sie, dass Abgaben im Rahmen der Übungsleistung für die Zulassung zur Klausur relevant sind. Durch Plagiieren verwirken Sie sich die Möglichkeit zur Zulassung zur Klausur in diesem Semester.

Aufgabe 4.1 *Klassenhierarchie für Personen* (4 Punkte)

Erstellen Sie ein neues Projekt mit Namen Aufgabe4 und darin eine Klasse `Person` mit weiteren Unterklassen, die von dieser direkt oder indirekt in der Vererbungsrelation stehen, nämlich je eine Klasse für folgende Personengruppen:

- Student*in
- Bachelorstudent*in
- Masterstudent*in
- Gast
- Angestellte*r
- Wissenschaftler*in
- Verwaltungsangestellte*r

Desweiteren gibt es Variablen für folgende Informationen:

- Matrikelnummer
- Personalnummer
- höchste wissenschaftliche Qualifikation
- Studienabschlüsse
- Schulabschlüsse
- Abteilung
- Fachgebiet
- Name

1. Erstellen Sie eine Klassenhierarchie in der Sie diese Eigenschaften durch geeignete Variablen repräsentieren. Achten Sie dabei darauf, dass Sie später Methoden zur Immatrikulation und Exmatrikulation schreiben müssen.

2. Fügen Sie zusätzlich Klassenvariablen und Code ein, damit die Personalnummer eindeutig ist für neu eingestellte.

3. Fügen Sie zusätzlich Klassenvariablen und Code ein, damit die Matrikelnummer eindeutig ist für neu immatrikulierte.

4. Schreiben Sie eine Methode `obtainDegree`, die Studierenden den entsprechenden Abschluss verleiht.

5. Schreiben Sie eine Methode `exmatriculate` für die Exmatrikulation von Studierenden.

Aufgabe 4.2 *Zufälliges Partikel – objektorientiert* **(3 Punkte)**

Erstellen Sie eine Klasse `ZufallsPartikel` mit folgenden Anforderungen:

- Jede Instanz dieser Klasse hat eine Referenz auf ein `Position`-Objekt und eine Variable `direction` für die Richtung (in Grad) wobei die Klasse `Position` wie folgt implementiert ist:

```
public class Position {
    int x;
    int y;

    public Position (int x, int y){
        this.x = x;
        this.y = y;
    }
}
```

- Es gibt einen Konstruktor mit einem `Position`-Objekt als Formalparameter, der die initiale Position angibt; die initiale Richtung sei immer 90 Grad.
- Es gibt getter-Methoden für die x- und y-Koordinate der Instanzen.
- Es gibt eine Instanzmethode `move` *ohne* Formalparameter, die eine Referenz auf ein `Position`-Objekt zurückgibt. Bei Aufruf der Methode `move` wird—jeweils mit Wahrscheinlichkeit ½—entweder im Uhrzeigersinn rotiert (um 90 Grad), oder das Partikel bewegt sich einen Schritt in die derzeitige Richtung weiter. Die Rückgabe ist die neue Position des Partikels.
- Es gibt eine Klassenvariable, die ein `ZufallsPartikel`-Objekt referenziert.
- Es gibt eine Klassenvariable `arena` für ein zweidimensionales Array von Zeichen.
- Es gibt eine Klassenmethode mit Namen initialize mit zwei Formalparametern (nämlich die Größe eines zweidimensionalen Arrays von Zeichen), und diese Methode
  - erzeugt ein neues Array von Zeichen der entsprechenden Größe
  - belegt alle Zeichen des Arrays als Leerzeichen (also `' '`)
  - setzt die Referenz `arena` auf dieses Array,
  - erzeugt eine neues Partikel und setzt sie „in die Mitte des Feldes“ (wobei ggf. beliebig auf- oder abgerundet werden darf).

- Es gibt eine Klassenmethode `run` ohne Formalparameter, die das Partikel "laufen lässt" bis es sich außerhalb des Feldes begibt, also wiederholt die `move`-Methode aufruft, solange die zurückgegebene Position noch einem Zeichen im Array `arena` entspricht. Wann immer das Partikel ein Feld betreten hat soll dort ein `'x'` stehen und ein großes `'X'`, wo sich das Partikel gerade befindet.
- Es gibt eine Klassenmethode `printToConsole` ohne Parameter, die den derzeitigen Zustand des Arrays von `char`-Werten angibt.
- In der `main`-Methode der Klasse `ZufallsPartikel` erfragen Sie geeignete Parameter per Konsole, starten den Lauf des Partikels und geben am Ende des Laufes der Ameise den Zustand des Feldes aus (bevor das Partikel die Arena verlässt).

Testen Sie Ihr Programm. Ein Ablauf des Programms könnte wie folgt aussehen.

```
Bitte geben Sie die Breite des Feldes ein:21
Bitte geben Sie die Höhe des Feldes ein:21
---------------------

          xxxx
       xxxxxx  x
              xxX

---------------------

Process finished with exit code 0
```

Bemerkung Sie können auch gerne etwas experimentieren indem Sie mehrere Partikel "gleichzeitig" los lassen, wobei jedes Partikel dann wohl noch eine anderen Buchstaben verwenden könnte.

Zusatzaufgabe 4.3 Generieren von fraktalen Kurven

Schreiben Sie (rekursiv) eine Methode für die Malanweisungen einer Variante der Koch-Kurve.

Zusatzaufgabe 4.4 "Turtle graphics" Zeichnen (in Arrays)

Führen Sie die Malanweisungen wie in der Aufgabe zuvor aus, und zwar indem Sie entsprechende `char`-Arrays ausgeben. Beachten Sie, dass Sie zuerst die Größe des Arrays bestimmen müssen.