Prof. Dr. Christoph Bockisch
MSc Steffen Dick
Fachbereich Mathematik und Informatik
AG Programmiersprachen und -werkzeuge

Philipps Universität Marburg

Übungen zur Vorlesung
Objektorientierte Programmierung: Wintersemester 2021/2022

Nr. 11, Abgabe bis 07.02.2022

Aufgabe 11.1: Deep Dive: Above Zero — 8 Punkte

Für den neuen Ableger eines Tauchspiels sollen wir Kreaturen erstellen. Außerdem soll es in *Deep Dive: Above Zero* die Möglichkeit geben, dass der Spieler ein Aquarium anlegt. In diesem Aquarium soll es möglich sein, Herbivoren und Carnivoren aufzuziehen. Aufgrund ihrer Größe sollen allerdings keine Leviathan-Klasse Lebensformen in dieses Aquarium setzbar sein können.

a) Implementieren Sie zunächst die abstrakte Klasse `Seacreature`. `Seacreature` soll ein Feld für die Größe einer Kreatur in Centimetern halten. — 1

   Erstellen Sie außerdem drei abstrakte Unterklassen von `Seacreature`:
   - `Herbivore`
   - `Carnivore`
   - `Leviathan`

b) Erstellen Sie Klassen für die in der Tabelle angegebenen Kreaturen. Stellen Sie in deren Konstruktor sicher, dass die vorgegebenen Längen eingehalten werden. — 2

Kreatur	Oberklasse	Größe
Pea Dragon Leviathan	`Leviathan`	110 - 116 Meter
Keeper Leviathan	`Leviathan`	54 - 56 Meter
Pampeel	`Carnivore`	20 - 22 Meter
Quidshark	`Carnivore`	11 - 12 Meter
Bellyray	`Herbivore`	7 - 9 Meter
Huddlefish	`Herbivore`	80 - 90 Centimeter

c) Schreiben Sie nun eine Klasse `FishTank`. `FishTank` soll ein Feld `ArrayList<SeaCreature> creatures` beinhalten, welches zunächst als leere Liste im Konstruktor initialisiert werden soll. — 0.5

d) Fügen Sie `FishTank` eine Methode `void addCreature(Seacreature)` hinzu, mit welcher Sie eine Kreatur in Ihr Aquarium setzen können. Sollte ein Spieler versuchen, einen Leviathan in das Aquarium zu setzen, soll eine eigene `Exception` geworfen werden. Implementieren Sie eine geeignete `Exception` mit einem geeigneten Text. — 1

   Hinweis: Mit `a instanceof B` können Sie überprüfen, ob `a` eine Instanz von `B` ist.

e) Schreiben Sie eine Methode `<T ...> List<T> filter(T creature)` in `FishTank`, die eine Liste aller Kreaturen zurückgibt, die sich aktuell in Ihrem Aquarium befinden und vom Typ T sind. — 1.5

   Hinweis: Mit `a.getClass().isInstance(b)` können Sie überprüfen, ob `b` eine Instanz des Typen von `a` ist.

f) Testen Sie jede Methode ausführlich mit JUnit-Tests. Befüllen Sie Ihren Tank mit mindestens 10 Kreaturen verschiedener Längen und mindestens 2 verschiedenen Arten. — 2

Aufgabe 11.2: Yolanda Montez

4 Punkte

Geben Sie für jeden der untenstehenden Code-Blöcke den correcten Wildcard-Typen von `l` an. Sollte ein Code-Block nicht möglich sein, geben Sie zusätzlich eine Begründung an. Lösen Sie diese Aufgabe am Besten, ohne den Code zu kompilieren oder die JShell zu verwenden. Geben Sie Ihre Lösung als *.txt*-Datei in Ihrer Abgabe ab.

a) Fügt einen `Double` und einen `Integer` in die Liste `l` ein. [1]

```
1 void listOperationsA(List<...> l) {
2   l.add(new Double(3.14));
3   l.add(new Integer(42));
4 }
```

b) Gibt alle Elemente in der Liste auf der Konsole aus. [1]

```
1 void listOperationsB(List<...> l) {
2   for(int i = 0; i < l.size(); i++){
3     System.out.println(l.get(i));
4   }
5 }
```

c) Vergleicht das erste und zweite Element in der Liste und gibt das Ergebnis zurück. [1]

```
1 int listOperationsC(List<...> l) {
2   return l.get(0).compareTo(l.get(1));
3 }
```

d) Entfernt das erste Element aus der Liste und hängt es am Ende an. [1]

```
1 void listOperationsD(List<...> l) {
2   Integer i = l.get(0);
3   l.remove(i);
4   l.add(i);
5 }
```