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

Philipps Universität Marburg

Companion exercises
Objektorientierte Programmierung: Wintersemester 2021/2022

No. 11, due until 07.02.2022

Task 11.1: Deep Dive: Above Zero — 8 Points

We are supposed to create some creatures for the sequel of a diving game. Furthermore, the player is allowed to build their own aquarium in *Deep Dive: Above Zero*. Within this aquarium it is supposed to be possible to raise herbivores and carnivores. Because of their size, Leviathan-class organisms can not be placed within the aquarium.

a) Implement the abstract class `Seacreature`. `Seacreature` should have a field for the size of a creature in centimeter. — 1

Create the following three sub-classes of `Seacreature` as well:

- `Herbivore`
- `Carnivore`
- `Leviathan`

b) Implement classes for the creatures in the table below. Make sure that you check within their constructor that they are within their size range. — 2

Creature	Super-Class	Size
Pea Dragon Leviathan	`Leviathan`	110 - 116 Meters
Keeper Leviathan	`Leviathan`	54 - 56 Meters
Pampeel	`Carnivore`	20 - 22 Meters
Quidshark	`Carnivore`	11 - 12 Meters
Bellyray	`Herbivore`	7 - 9 Meters
Huddlefish	`Herbivore`	80 - 90 Centimeters

c) Write a class `FishTank`. `FishTank` should have a field `ArrayList<SeaCreature> creatures` that should be initialized as an empty ArrayList within the constructor. — 0.5

d) Add the method `void addCreature(Seacreature)` to `FishTank` with which a `SeaCreature` can be added to your aquarium. Should a player try to add a Leviathan to the aquarium, throw an `Exception` of your own making. Implement a useful `Exception` with a useful message. — 1

Attention: You can use `a instanceof B` to check if `a` is an instance of `B`.

e) Write a method `<T ...> List<T> filter(T creature)` within `FishTank` that returns a list of all the creatures within the aquarium that are of type T. — 1.5

Attention: You can use `a.getClass().isInstance(b)` to check if `b` is an instance of the same type as `a`.

f) Test every method in-depth using JUnit-tests. Add at least 10 different creatures of different sizes and at least two different types to the aquarium. — 2

Task 11.2: Theodore Grant | 4 Points

Name the correct wildcard-type for every `l` within the code-blocks below. If any code-block should not be possible, give a reason why. Solve this task without compiling the code or using the JShell. Write your solutions to a *.txt*-file and include it in your submission.

a) Adds one `Double` and one `Integer` to list `l`. | 1

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

b) Prints every element to the console. | 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) Compares the first and the second element of the list and returns the result. | 1

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

d) Removes the first element from the list and adds it at the end. | 1

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