Advanced Object-Oriented Programming

CPT204 – Lab 11

Erick Purwanto

Xi'an Jiaotong-Liverpool University

西交利物浦大学

CPT204 Advanced Object-Oriented Programming

Lab 11

Comparable, Equals, Set 2

Welcome !

- Welcome to Lab 11 !
  - We are going to make our classes and data structures support equality, and their objects can be compared to each other!

- You will find in this lab
  1. Lab Exercise 11.1, and 11.2, and their hints
  2. Exercise 11.1, and 11.2

- Download **lab11** zip files from Learning Mall
- Don't forget to import the **lab11** files and the library into an IntelliJ project
  - Read **lab1** again for reference

Lab Exercise 11.1 Duration COMPARETO

- Complete the method int **compareTo**(Duration other) of Duration which implements Comparable.

- It returns a positive, zero, negative number if its duration is larger than, equal to, smaller than other's duration, respectively.

- Test case 1:

```
Duration d1 = new Duration(2, 5);
Duration d2 = new Duration(1, 2);
d1.compareTo(d2) > 0            →     true
d2.compareTo(d1) < 0            →     true
```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Lab Exercise 11.1 Duration COMPARETO Hints

- Use the trick in the lecture for a one-liner code

Lab Exercise 11.2 Dog EQUALS

- Complete the method boolean **equals**(Object that) of Dog that overrides the one in Object.

- To be the same, they must share the same name and weight.
- The method must satisfy the Object Contract.

- Test case 1:

```
Dog d1 = new Dog("Baobei", 5);
Dog d2 = new Dog("Baobei", 5);
Dog d3 = new Dog("Jiaozi", 7);
d1.equals(d2);                    →     true
d1.equals(d3);                    →     false
```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Lab Exercise 11.2 Dog EQUALS Hints

- Similar to the equals implementation example in the lecture

Triple

- We want to write a generic class `Triple<T, S, U>` that stores triples of objects, each of which can have an arbitrary type
- In addition, we also want to be able to compare two Triple objects
  - To compare two triples, compare the first elements of each; if they are the same, compare the second elements; and if they are the same, compare the third
- Of course, it is required that the first, second and third element in the triple are comparable
  - we declare it using `extends` on the generic types as follows:

```
public class Triple<T extends Comparable<T>, S extends Comparable<S>,
        U extends Comparable<U>> implements Comparable<Triple<T, S, U>>
```

Exercise 11.1 Triple COMPARETO

- Complete the method int **compareTo**(Triple<T, S, U> other) of a comparable Triple.
- It must compare the first element first, followed by the second and third as in the following test case:
- Test case 1:

```
Triple<Integer, String, Double> t1 = new Triple<>(1, "b", 2.0);
Triple<Integer, String, Double> t2 = new Triple<>(1, "a", 5.0);
Triple<Integer, String, Double> t3 = new Triple<>(1, "b", 2.0001);
t1.compareTo(t2) > 0                    →       true
t1.compareTo(t3) < 0                    →       true
```

Exercise 11.2 ARSet EQUALS

- Complete the method boolean **equals**(Object that) of ARSet that overrides the one in Object.
- To be the same, they must share the same items.
- The method must satisfy the Object Contract.
- Test case 1:

```
ARSet<String> set1 = new ARSet<>();
set1.add("a");  set1.add("b");  set1.add("c");
ARSet<String> set2 = new ARSet<>();
set2.add("b");  set2.add("c");  set2.add("a");
set1.equals(set2);                  →      true
set2.add("d");
set1.equals(set2);                  →      false
```

Thank you for your attention !

- In this lab, you have learned:
  - To make your data structure comparable
  - To equip your data structure with an equality method that adheres to the Object Contract