Lab 9 - Collections Framework

This lab contains in-class exercises related to collections algorithms. Before starting these exercises, one is advised to review `java.util.Arrays` and `java.util.Collections` classes.

Task 1: Develop a class called `ExerciseArrays` to sort alphabetically the array

```
String[] oss = {"Windows", "Unix", "MacOS", "Andorid", "Linux"};
```

Task 2: Develop a class called `ExerciseCollections` to sort the `List`

```
List<String> oss = new ArrayList<>();
oss.add("Windows");
oss.add("Unix");
oss.add("MacOS");
oss.add("Android");
oss.add("Linux");
```

Task 3: Develop a class called `OperatingSystem` that implements `Comparable<OperatingSystem>` The class has two fields:

```
public class OperatingSystem implements Comparable<OperatingSystem> {

   String name;
   float version;
```

and must implement basic methods: `equals, hashCode, toString`

Task 4: Develop a class `ExerciseComparable` that must sort the following array

```
OperatingSystem[] oss = {
       new OperatingSystem("Windows", 8.00f),
       new OperatingSystem("Windows", 7.00f),
       new OperatingSystem("Ubuntu", 12.04f),
       new OperatingSystem("Ubuntu", 14.04f),
       new OperatingSystem("Linux", 6.32f),
       new OperatingSystem("Android", 4.44f),
```

@ Jordan Anastasiade

Comparable implementation compares first the operating system names. If these are the same then it compares the os versions.

Task 5: Develop a class `ExerciseComparator` that must sort the same array as in the task number 4.

```
OperatingSystem[] oss = { ... }
```

However, the class uses the functional interface `Comparator<T>`

You must implement the lambda expression for the invocation:

```
Arrays.sort(oss, (...) -> ...);
```

The result must be the same as for the fourth task.