Object-Oriented Programming

Lab session #8

Question 0: Practice a basic Generic coding problem with Hackerrank:
https://www.hackerrank.com/challenges/java-generics/problem

Question 1: Get familiar with generic types

Given the following class

```java
public class MyPair<T, U> {
    public final T Fst;
    public final U Snd;

    public MyPair(T fst, U snd) {
        this.Fst = fst;
        this.Snd = snd;
    }

    public String toString() {
        return "(" + Fst + ", " + Snd + ")";
    }

}
```

a. In a new source file, write a Java program that includes this declaration and a class with an empty Main method. Compile it to check that the program can run without any error.

b. Declare a variable of type `MyPair<String, Integer>` and create some values, for instance new `MyPair<String, Integer>("Anders", 13)`, and assign them to the

variable.

c. Declare a variable of type `MyPair<String, Double>`. Create a value such as new `MyPair<String, Double>("Phoenix", 39.7)` and assign it to the variable. d. Can you assign a value of type `MyPair<String, Double>` to a variable of type `MyPair<String, Integer>`? Should this be allowed?

e. Declare a variable grades of type `MyPair<String, Integer>[]`, create an array of length 5 with element type `MyPair` and assign it to the variable. Create a few `MyPairs` and store them into grades[0], grades[1] and grades[2].

f. Use the `foreach` statement to iterate over `grades` and print all its elements. What are the values of those array elements you did not assign anything to?

g. Declare a variable `appointment` of type

$$MyPair<MyPair<Integer, Integer>, String>$$

and create a value of this type and assign it to the variable.
What is the type of `appointment.Fst.Snd`? This shows that a type-argument may itself be a constructed type.

h. Declare a method `Swap()` in `MyPair<T, U>` that returns a new value of type `MyPair` in which the components have been swapped.

Question 2: Try to apply generic methods or classes to solve the problem
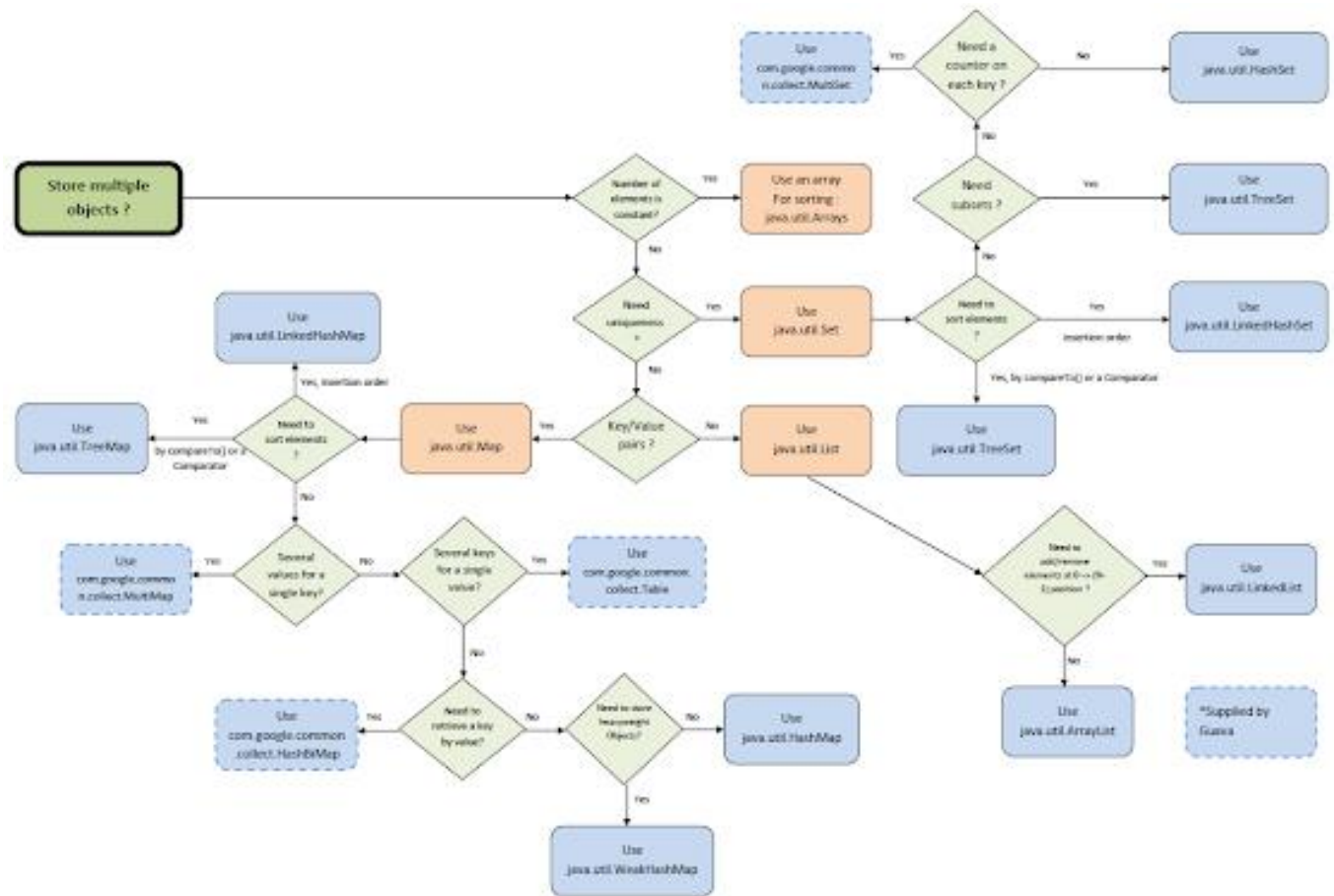
Consider the class below:

```
class MyList {
   private List<String> values;

   void add(String value) {
      values.add(value);
   }

   void remove(String value) {
      values.remove(value);
   }
}
```

MyList can be used to store a list of Strings only.

```
MyList myList = new MyList();
myList.add("Value 1");
myList.add("Value 2");
```

Extra knowledge - The more you know: As you know that Generic Collections are the prebuilt data structures which are also generic classes containing generic methods. However, which one do we use to solve each problem? Here is the diagram which answers this question:

Store multiple objects ?

That's it! Pretty short lab, enjoy! Now, you can work on the final projects with your teammates!

In short, use set when you don't need duplicates, use List when you need order with duplicates, and use Map when you need to store key-value pair.

CONGRATULATIONS

KEEP UP THE GREAT WORK