**Object-Oriented programming**

Lab #7 – **Generic classes and methods**

**I. Get familiar with generic types** Given the following class

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 + ")"; }

}

1. In a new source file, write a program that includes this declaration and also a class with an empty Main method. Compile it to check that the program is well-formed.
2. 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.
3. 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.
4. Can you assign a value of type MyPair<String, Double> to a variable of type MyPair<String, Integer>? Should this be allowed?
5. 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].
6. 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?
7. 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.

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

**II. Differences between Object, generic and generic raw types**

As we want to reimplement the Map class, implement a class named MyMap to manage (store and get back) any object by its ID.

* User can put an object ***obj*** to a Map ***m*** by calling

*m.put(obj.getID(), obj);*

* User can get back an object from the map ***m*** by invoking

# m.get(id);

1. Implement the MyMap in two different ways:
   1. Use Object as the type for both the Key and the Value parameters of the ***put*** and ***get*** methods
   2. Use generic type
2. With your implementations, write a main function to
   1. Test these two implementations
   2. To show advantage of generic type over Object
   3. To show advantage of parameterized type over generic raw type

Reference: textbook “Java How to Program”, chapter 20

public class MyPair<T, U> {

public final T First;

public final U Second;

public MyPair(T first, U second)

{

this.First = first;

this.Second = second;

}

public String toString()

{

return "(" + First + ", " + Second + ")";

}

//Question H

public MyPair Swap()

{

return new MyPair<>(Second, First);

}

public static void main(String[] args) {

//Question B

MyPair<String, Integer> people1 = new MyPair<>("Anders", 13);

//Question C

MyPair<String, Double> people2 = new MyPair<>("Phoenix", 39.7);

/\*Question D : I cannot assign a value of type

MyPair<String, Double> to a variable

of type MyPair<String, Integer>\*/

//Question E

MyPair<String, Integer>[] grades = new MyPair[5];

grades[0] = new MyPair<String, Integer>("Ly", 26);

grades[1] = new MyPair<String, Integer>("Minh", 05);

grades[2] = new MyPair<String, Integer>("Trung", 2001);

//Question F

for(MyPair<String, Integer> hihi : grades)

{

System.***out***.print(" " + hihi);

}

//Question G

MyPair<MyPair<Integer, Integer>, String> appointment = new MyPair<>(new MyPair<>(26, 05).Swap(), "LyMinhTrung");

System.***out***.print("\n\n " + appointment);

System.***out***.println(" " + appointment.First.Second);

//Question H

System.***out***.print("\n Swapped: " + appointment.Swap());

}

}

import java.util.HashMap;

import java.util.Map;

public class MyMap <K extends Object, V extends Object>{

private Map<K, V> map = new HashMap<>();

public get(K keyword) {

return map.get(keyword);

}

public void put(K keyword, V value) {

map.put(keyword, value);

}

public void PrintObject() {

for(K keyword : map.keySet())

{

System.***out***.printf("Key: \"%s\"\t - Value: \"%s\"\n", keyword, map.get(keyword));

}

}

public static void main(String[] args) {

MyMap map1a = new MyMap();

map1a.put('1', 1);

map1a.put(2, "Hello");

map1a.put("Hole", 3.14);

map1a.PrintObject();

MyMap<String, Integer> map1b = new MyMap();

map1b.put("Hello", 7);

map1b.PrintObject();

}

}