Lab 2 (Java Generics)

CSC 172 (Data Structures and Algorithms)

University of Rochester

Due Date: 01/30 11:59 PM

Objective

The objective of this lab is exploring Java Generics in more details. After completing the lab, you should be confident to undertake any project/assignment involving Java generics. Also, this lab will teach you how you can convert any non-generic method into a generic one.

Tasks

Consider the following code snippet, where the `printArray()` method is used to print out arrays of different types:

```
Integer [] intArray = {1, 2, 3, 4, 5 };
Double [] doubArray = {1.1, 2.2, 3.3, 4.4};
Character [] charArray = {'H','E','L', 'L', 'O' };
String [] strArray = {"once", "upon", "a", "time" };
printArray(intArray);
printArray(doubArray);
printArray(charArray);
printArray(strArray);
```

You will program variations of this functionality, among others, to help solidify your understanding of how types operate in Java.

Requirements

Write a Java program named Lab2.java with a `main` method and implementations of the following:

1. A static `printArrayNonGen()` method with an array of Objects as parameter.
2. A static `printArray()` method using method overloading. Implement four versions of `printArray()`, one for each array type.
3. Implement a single static method `printArrayGen()` that uses the generic programming technique to support all 4 array types and maintain type safety.

4. Using non-generic techniques, implement a static method `getMax()` that takes an array of type Comparable and returns the maximum element in the array.
(i.e. "`public static Comparable getMax(Comparable [] anArray)`").
5. Using the generic techniques to specify super-class relationships, implement a type safe version of the method in 4 named `getMaxGen()`.

Submission

Hand in the source code from this lab at the appropriate location on the Blackboard system at learn.rochester.edu. You should hand in a single **zip** (compressed archive) named yourNetID_Lab2.zip containing all the following files:

1. A plain text file named *README* that includes your first name and last name in the first line, partner's first name and last name in the second line (if any), an empty line and then contact information, a brief explanation of the lab. Include information identifying what class and lab number your files represent and one sentence explaining the contents of any other files you hand in.

Additionally, please briefly describe the steps you took to build and run your code.

2. A single Java source code file representing the work accomplished for sections 1-5 of this lab. All source code files should contain author and partner identification in the comments at the top of the file.

Important note: In attachment you will find a sample input and respective output. The aim of the sample is to keep a common standard format in the answers. Graders will use their private test cases for testing the correctness of your code. You may choose to read/write input/output from/to a text file or use the standard input/output. However, you should describe the chosen method in the README file.

Grading **(10 pts)**

Each section (1-5) accounts for 2 pts. Total 10 pts.

Important final note

All labs are open book. The code you submit has to be **mainly** your own code. You can get code snippets from the internet if you want (make sure you cite those properly). But that is not the purpose. We want you to work together, think about an algorithm, and then implement it together with your partner.