Sofia University
Department of Mathematics and Informatics

Course : Applied OO Programming part 1
Date: May 19, 2020
Student Name:

Lab No. 12

Problem 1

Revise the attached code for class **GenericStack** to implement it using an array rather than an **ArrayList<E>,** where **the type of the array elements are generics**. Implement the revised class **GenericStack as a nested class of a class named GenericStackProvider. Test the revised GenericStack class with different values of the generic parameter- Integer, String.** You should check the array size before adding a new element to the stack. If the array is full, create a new array that doubles the current array size and copy the elements from the current array to the new array.

Problem 2

The attached code for class **GenericStack** is implemented using composition of ArrayList. Define a new **GenericStack** class that extends **ArrayList as an inner static class of a class GenericStackAsArrayListTest**. Write a **main**() method in **class GenericStackAsArrayListTest** program that prompts the user to enter five strings and displays them in reverse order using the nested **GenericStack** class.

Problem 3

Write the following method that returns a new **ArrayList**. The new list contains the non-duplicate elements from the original list.

```
public static <E> ArrayList<E> removeDuplicates(ArrayList<E> list)
```

Problem 4

Implement the following generic method for linear search.

```
public static <E extends Comparable<E>> int linearSearch(E[] list,
                                                          E key)
```

Problem 5

Write a generic method that returns the maximum element in a two-dimensional array.

```
public static <E extends Comparable<E>> E max(E[][] list)
```

Problem 6

Write the following method that sorts an **ArrayList** using **the Insertion Sort algorithm**:

```
public static <E extends Comparable<E>> void sort(ArrayList<E> list)
```

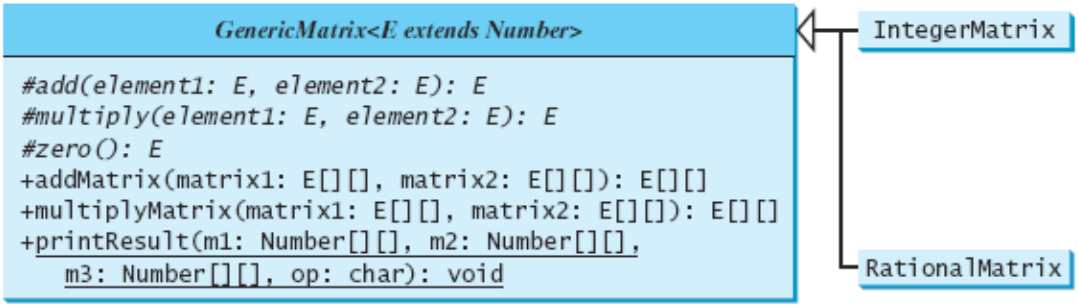
Problem 7

Write the following method that shuffles an **ArrayList**:

```
public static <E> void shuffle(ArrayList<E> list)
```

Problem 8

Implement the classes in the following UML class diagram, where the code for **class Rational** is attached



Write a **GenericMatrixTest** application to test methods **addMatrix**() and **multiplyMatrix**() for instances of classes **IntegerMatrix** (derived from GenericMatrix<Integer>) and **RationalMatrix(**derived from GenericMatrix<Rational>, where class rational is provided in the attached to Lab 12 source code**)**.

- Why are the **add**, **multiple**, and **zero** methods defined abstract in the **GenericMatrix** class?
- How are the **add**, **multiple**, and **zero** methods implemented in the **IntegerMatrix** class?
- How are the **add**, **multiple**, and **zero** methods implemented in the **RationalMatrix** class?
- What would be wrong if the **printResult** method defined as follows?

```
public static void printResult( E[][] m1, E[][] m2,
                                E[][] m3, char op)
```

Problem 9a

Use the attached code for **class Complex** to develop the **ComplexMatrix** class for performing matrix operations involving complex numbers. The **ComplexMatrix** class should extend the **GenericMatrix** class introduced in Problem 8.

Note, that class **Complex (*see attached code to Lab12*)** is not a subtype of **Number. Therefore** you have to modify the abstract class **GenericMatrix** by implementing **interface CanCompute,** where you declare the **add**, **multiple**, and **zero** methods.

Add classes **IntegerMatrix, Rational** and **RationalMatrix** from Problem 8 in this inheritance hierarchy and their respective test programs **TestIntegerMatrix** and **TestRationalMatrix**. Write a test program **ComplexMatrixTest** that creates the following two matrices and displays the result of addition and multiplication of the matrices by invoking the **printResult** method. Run all the three test programs to verify the obtained results.

Problem_9b

Write the UML class diagram for the classes involved in the solution for Problem 9a.

Problem_9c

Add a class **BigDecimalMatrix** the same way to the inheritance hierarchy in Problem 9a. Write a test program **BigDecimalMatrixTest** that creates the following two matrices and displays the result of addition and multiplication of the matrices by invoking the **printResult** method.

Problem_10

Suppose in a particular business all documents are given a **twocharacter** designation starting with either **U**, **C**, or **P,** standing for unclassified, confidential, or proprietary. Create an exception class called **`InvalidDocumentCodeException`**, designed to be thrown when an improper designation for a document is encountered during processing. If a document designation is encountered that doesn't fit that description, the exception is thrown. Create a driver program to test the exception, allowing it to terminate the program.
Catch and handle the exception if it is thrown. Handle the exception by printing an appropriate message, and then continue processing

Problem_11a

Write a program that creates an exception class called **`StringTooLongException`**, designed to be thrown when a string is discovered that has too many characters in it. In the main driver of the program, read strings from the user until the user enters "**`DONE`**". If a string is entered that has too many characters (say **`20`**), throw the exception. Allow the thrown exception to terminate the program.

Problem_11b

Modify the solution to **Problem 11a** such that it catches and handles the exception as a **checked** `Exception` if it is thrown. Handle the exception by printing an appropriate message, and then continue processing more strings.

Problem_12

Напишете generic **`class Pair`**, който има две данни, чиито тип се определя от два параметъра за тип **`F`** и **`S`**. Напишете **`get`** и **`set`** методи за тези данни на класа .

Упътване: Заглавието на класа да бъде

```
public class Pair< F, S >
```