United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam, Trimester: Spring 2024

Course Code: CSE-1115, Course Title: Object Oriented Programming

Total Marks: 40, Duration: 2 Hours

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

QUESTION 1(a) [5 MARKS]

Write an interface named ***Inf1*** containing two methods ***m1()*** and ***m2()***. Write another interface named ***Inf2*** containing one method ***m3().*** Write an abstract class named ***Abs*** containing one abstract method ***m4().***

Now, you have to write another concrete class named ***Concrete*** that inherits both of the interfaces, ***Inf1*** and ***Inf2*** and the abstract class ***Abs***. Print the name of the method in each method body while completing.

QUESTION 1(b) [5 MARKS]

Imagine you are developing a program that involves inputting values into an array. However, there are several potential issues that could occur during this process, such as inputting invalid array indices or providing incorrect input data types. To illustrate how to handle various types of exceptions that may occur during the execution of the program, you are tasked with creating a program that does the following:

Create a class named "**ExceptionHandling**" and initialize an integer array, named **myArray** of size **n** where the value of **n** will be assigned by user. User will give an index position at first and then insert value in that particular array index. When a user tries to assign an invalid index, your code will handle "Type A Exception" and will print "**Type A Exception occurred**". When a user tries to assign an invalid value at array index, your code will handle "Type B Exception" and will print "**Type B Exception occurred**". For all other exceptions, your code will handle with generic exception and will print "**Some other exception occurred!**". Whether an exception arises or does not and whether an exception is handled or not, your program will always show this message "**Exception handling is amazing**".

N. B: You have to replace "Type A Exception", "Type B Exception" or "Generic Exception" with appropriate exception names.

Sample Input	**Sample Output**	
Enter Array Size: 3 Enter index position: 4	Type A Exception Occurred! Exception handling is amazing	**N.B:** Replace "Type A Exception" with appropriate exception name.

QUESTION 2(a) [5 MARKS]

Consider the following **MovieTheater** class

```
public class MovieTheater {
   int availableSeats;
   MovieTheater(int s){
      availableSeats=s;
   }
   public synchronized int bookTickets(int numOFseats){
      int numOfTicketsBooked = 0;
      // The tickets are booked one by one
      for(int i = 1; i <= numOFseats; i++){
         if(availableSeats > 0){
            availableSeats--;
            numOfTicketsBooked ++;
         }
      }
      return numOfTicketsBooked;
   }
}
```

- Create a thread class named "User" extending the "Thread" class.
- "User" must have the following 2 member variables:
  - MovieTheater m
  - int NumOfTickets

- "User" thread class must have a constructor to initialize the member variables m, NumOfTickets and the name of the thread which will be passed to the parent "Thread" class.
- Inside the "run" method, it should invoke the "bookTickets" method of MovieTheater with the parameter NumOfTickets. See the expected output for details.
- Create another class named Movie in which write the main method. In the main method, create an object of "MovieTheater" class with 15 availableSeats.
- Consider that Mina wants to book 6 tickets, Nabil wants to book 8 tickets and Farhan wants to book 4 tickets. Now, create 3 "User" threads and start the threads. After all the users have completed their booking, print the number of tickets the users booked and the remaining avaiableSeats of MovieTheater.

Expected output (depends on the completion of concurrent runs of the threads) of the program may be given as follows:

```
Mina has booked 6 tickets
Nabil has booked 5 tickets
Farhan has booked 4 tickets
Available tickets:0
```

QUESTION 2(b) [5 MARKS]

Consider the following "Appliance" class

```
public class Appliance {
   String name,category;
   double powerConsumption;//watt
   Appliance(String n,String c,double d){
      name=n;category=c;
      powerConsumption=d;
   }
   public String toString(){
      return name+" "+category+" "+powerConsumption;
   }
}
```

Now, only write the missing codes in the following main method of "ApplianceTest" class.

```
class ApplianceTest{
   public static void main(String[] args) {
      //task-1: create an empty arrayList of "Appliance" type
      /*task-2: add the following appliances into the list
         "Television","Entertainment",150
         "Washing machine","Laundry",2000
         "Refrigerator","Kitchen",100  */
      //task-3: Find the "kitchen" appliances from the list and print their details
      /*task-4: Sort the list based on their power consumption in descending order. You must use
comparator or comparable interfaces for comparing objects of Appliance type and sort method of
Collections class  */
      /*task-5: Now that you have a sorted list, print the details of the appliances with highest and
lowest power consumption */
   }
}
```

QUESTION 3 [10 MARKS]

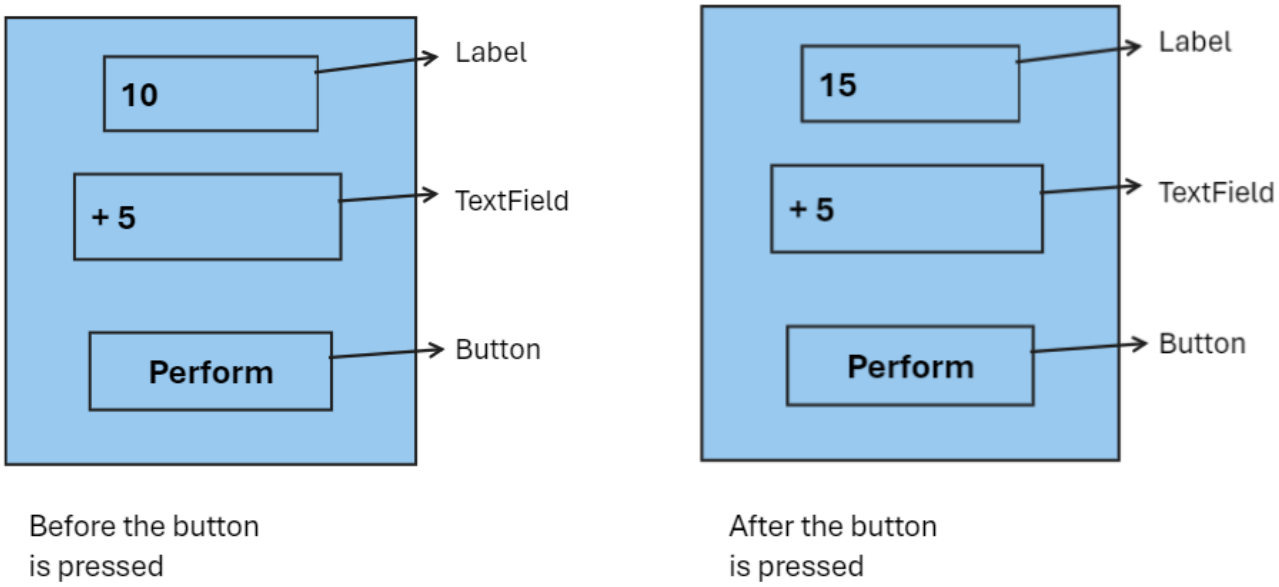
You have an input file named "input.txt" with multiple lines, some of which contain only numbers. Your task is to find the minimum value among those numbers and write it to an "output.txt" file. Both the input and output files should be located in the "src" directory.

In the following example, the minimum value is 5.

```
Abcd
10
20
Xyz
5
Ijk
```

QUESTION 4 [10 MARKS]

Write code to implement the following GUI application.



Before the button is pressed

After the button is pressed

The names of the label instance, textfield instance and button instance are *Output*, *Input* and *Perform*, respectively. *Output* initially contains 10. *Input* contains two strings separated by a space, where the first string is an operator(+, -, *, or /), and the second string is a number.

You have to perform the operation using the operator on the *output* number with the *input* number given in the textfield when *Perform* button is pressed. As an example in the given GUI, you have to add 5 with 10 and show 15 in *Output* label. Note that you only **have to write *actionPerformed()*** method.