INTE 12213– Object Oriented Programming

Lab Exercises 06

You are given the following tasks. Perform all of them. You can use either IDE or text editor to complete the tasks. Prepare a report with the experiment, your observations on each task and conclusions.
Submit the files by 14th, March 11.59 pm No marks will be given to any late submissions.

Lab06_Task01

Given the following class:

```
public abstract class Cake{
        protected String name;
        protected double rate;
        public Cake(String n,double r){
                name =n;
                rate=r;
        }
        public abstract double calcPrice();
        public String toString(){
                return name +"\t"+rate;
        }
}
```

Based on class Cake and the following table, define TWO (2) subclasses named as orderCake and readymadeCake. (Hint : Use Polymorphism Concepts)

	orderCake	readymadeCake
Additional attribute	weight(kg)	quantity
Price calculation	rate*weight	rate*quantity

By using classes definition from a), write an application program that will:

I. declare an array of 20 cake objects
II. input data from cake objects and store them into the array
III. display the total price for all types of cakes
IV. display the total price and the quantity sold for ready-made cakes

V. display the information for the cake that has been sold for the highest price

Lab06_Task02

Compile and run the following Java programming code and observe the result.

```
public class ExceptionCheck {
    public static void main(String args[]) {
        try{
            System.out.println ("statement 1");
            //System.out.println (5/0);
            System.out.println ("statement 2");
            System.out.println ("statement 3");
        }
        catch (ArithmaticException e) { // catch (FileNotFoundException e)
            System.out.println(10/2);
            //System.out.println(10/0);
        }
        System.out.println ("statement 4");
    }
}
```

What can you say about the termination of the program in following cases?

(a) If no exception in the program,

(b) If statement 2 has the exception, (replace statement 2 with System.out.println (5/0);)

(c) If corresponding catch block is not available, // catch (FileNotFoundException e)

(d) If the exception occurs at the catch block, (remove comment //System.out.println(10/0);)

(e) If the exception occurs at statement 4

Lab06_Task03

Compile and run the following Java programming codes and observe the result. Write down your observation in each case.

(i)

```
import java.io.*;
public class TestChecked1{
        public static void main(String args[]){
                PrintWriter pw=new PrintWriter("ABC.txt");
        pw.println("Hello");
        }
}
```

(ii)

```
import java.io.*;
public class TestChecked2 {
        public static void main(String args[]) throws FileNotFoundException {
                PrintWriter pw=new PrintWriter("ABC.txt");
                pw.println("Hello");
        }
}
```

(iii)

```
import java.io.*;
public class TestChecked3 {
        public static void main(String args[]) throws FileNotFoundException{
                PrintWriter pw=new PrintWriter("ABC.txt");
                pw.println("Hello");
                System.out.println(10/0);
        }
}
```

Lab06_Task04

To print exception information to console, there are multiple methods. Compile and run the following Java programming code and observe the result. Observe the corresponding output of each method. State what method would be used by default exception handler internally to inform the exception to the console?

```
public class ExceptionMethod {
        public static void main(String args[]) {
        try{
                System.out.println (5/0);
        }
        catch (Exception e) {
                e.printStackTrace();
                System.out.println(e.toString());
                System.out.println(e);
                System.out.println(e.getMessage());
        }
        System.out.println ("statement 4");
        }
}
```

Lab06_Task05

Compile and run the following Java programming code and observe the result.

```
public class Multiple_Catch {
        public static void main(String args[]) {
                try {
                        int array_size = args.length;
                        int result = 42/ array_size;
                        int new_array[] = { 1 };
                        new_array[90] = 99;
                } catch (ArithmeticException e) {
                        System.out.println("\n\tDivision by zero.");
                        System.out.println("\n\tException: "+ e);
                } catch (ArrayIndexOutOfBoundsException e) {
                        System.out.println("Array index : "+e);
                }
        }
}
```

(a) How many exceptions are there in the try block?

(b) All exceptions in the try block are captured by catch block?

(c) If all exceptions are not captured, change the program to capture all exceptions.

Lab06_Task06

Compile and run the following Java programming code and observe the result.

```
public class TestFinally {
        public static void hello(){
                try{
                        System.out.println("hi");
                        // System.out.println(5/0);
                }
                catch(RuntimeException e){
                        System.out.println(e);
                }
                finally{
                        System.out.println("finally");
                }
        }
        public static void main(String[] args){
                hello();
        }
}
```

Remove the comment and recompile and run the programming code.

What is your observation?

Lab06_Task07

Sometimes a situation may arise where a part of a block may cause one error and the other part may cause another error. In such cases, exception handlers have to be nested. You can change the program with nested try block to capture all exceptions.

```
public class Multiple_Catch {
        public static void main(String args[]) {
                try {
                        int array_size = args.length;
                        int result = 42/ array_size;
                        int new_array[] = { 1 };
                        new_array[90] = 99;
                } catch (ArithmeticException e) {
                        System.out.println("\n\tDivision by zero.");
                        System.out.println("\n\tException: "+ e);
                } catch (ArrayIndexOutOfBoundsException e) {
                        System.out.println("Array index : "+e);
                }
        }
}
```

Lab06_Task08

Sometimes it will be necessary to define your own exception class that deals with specific conditions that occur within your program. Such a class should be derived from the Exception class or one of its subclasses.

Create your own exception class called InvalidAgeException by extending Exception class as follows.

```
public class InvalidAgeException extends Exception{
        InvalidAgeException(String s){
                super(s);
        }
}
```

The following program is to throw an InvalidAgeException if the age is below 18.

```
public class TestCustomerException1{
        public static void validate(int age)throws InvalidAgeException{
                if(age<18)
                        throw new InvalidAgeException("not valid");
                else
                        System.out.println("welcome to vote");
        }

        public static void main(String args[]){
                try{
                        validate(8);
                }catch(Exception e){System.out.println("Exception occured: "+e);}
                        System.out.println("rest of the code...");
                }
        }
}
```

Based on your experience of using throw and throws keywords in question 3, question 4 and last week question 2 & question 7:

(a) Explain the purpose of throw, how and when do we need to use it

(b) Explain the purpose of throws, how and when do we need to use it