Faculty of Engineering, University of Jaffna

Department of Computer Engineering

EC5080: Software Construction

Lab 03: Input/output, and error handling

You need to follow the good programming practices learned and it will be considered when grading your program.

Question 01 and parts 1,2,3 of Question 02 must be finished and submitted in the lab time (Lab 03-Inlab). Question 01 should have 2 PDF files for each part. Question 2 must have your source code (IntelliJ project).

Question 01.

Part I

Implement the commented and "....." parts. Submit all your implementations of this part in a PDF file.

```
class ExceptionTest1
{
  public static void main(String args[])
  {
    try{
      int a[]=new int[5];
           a[6] = 9;
    }
    catch(..... e){
      //fill
    }
  }
}
```

```
class ExceptionTest2
{
  public static void main(String args[])
  {
    try{
          int num=Integer.parseInt ("XYZ") ;
                System.out.println(num);
    }catch(..... e){
          //fill

    }
  }
}


class ExceptionTest3
{
  public static void main(String args[])
  {
        try{
                String str=null;
             System.out.println (str.length());
        }
     catch(..... e){
                //fill    }
  }
}
```

Part II

Try different inputs for the following Java programs. Paste the outputs for each Test under each class name and submit as a pdf file.

```
import java.util.Scanner;

class Test1 {

   public static void main(String[] args) {
      Scanner s = new Scanner(System.in);
```

```
        System.out.println("Please enter your roll number");
        int roll = s.nextInt();

        if (roll < 0) {
            throw new ArithmeticException("Roll number can't be negative");
        } else {
            System.out.println("Valid roll number");
        }
    }
}


import java.util.Scanner;

class Test2 {

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.println("Please enter your roll number");
        int roll = s.nextInt();

        try {
            if (roll < 0) {
                throw new ArithmeticException(
                    "The number entered is not positive"
                );
            } else {
                System.out.println("Valid roll number");
            }
        } catch (ArithmeticException e) {
            System.out.println("An exception is thrown");
            System.out.println(e.getMessage());
        }
    }
}


import java.util.Scanner;

class Test3 {

    public static void main(String[] args) {
```

```
        Scanner s = new Scanner(System.in);
        System.out.println("Please enter your age");
        int age = s.nextInt();

        try {
            if (age < 5) {
                throw new ArithmeticException(
                    "Not allowed! Your age is less than 5"
                );
            } else if (age > 20) {
                throw new ArithmeticException(
                    "Not allowed! Your age is greater than 20"
                );
            } else {
                System.out.println("Welcome!");
            }
        } catch (ArithmeticException e) {
            System.out.println("An exception is thrown");
            System.out.println(e.getMessage());
        }
    }
}


import java.util.Scanner;

class Test4 {

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.println("Please enter your roll number");
        int roll = s.nextInt();

        try {
            if (roll < 0) {
                throw new Exception("The number entered is not positive");
            } else {
                System.out.println("Valid roll number");
            }
        } catch (Exception e) {
            System.out.println("An exception is thrown");
            System.out.println(e.getMessage());
```

```
    }
  }
}

class Test5 {

  public static void func() {
    int num = 10 / 0;
  }

  public static void main(String[] args) {
    func();
  }
}


class Test6 {

  public static void func() throws ArithmeticException {
    int num = 10 / 0;
  }

  public static void main(String[] args) {
    func();
  }
}
```

For the following class first run with comments. Then remove the comments, place the comment for int num = 10 / 0; line and run. Paste two outputs for the Test7.

```
class Test7 {

  public static void calculate()
    throws ArithmeticException, ArrayIndexOutOfBoundsException {
    int num = 10 / 0;
    // int a []={1,2,3};

    //System.out.println (a[4]);

}

  public static void main(String[] args) {
    try {
      calculate();
```

```
    } catch (ArithmeticException e) {
        System.out.println("Arithmetic Exception thrown");
        System.out.println(e.getMessage());
    } catch (ArrayIndexOutOfBoundsException e) {
        System.out.println("ArrayIndexOutOfBounds Exception thrown");
        System.out.println(e.getMessage());
    }
  }
}
```

Question 02.

Implement the Accounts class following the questions and statements below. Before implementing party 6 and 7, learn how to create a custom (your own) exception class.

1. First, create an IntelliJ Java project and name it as EC5080_Lab3_RegNo

2. Starting at the topmost line of the file, insert the following under the comments. You need to

choose one of them. If you get any assistance from anyone/book/internet please include that. (This

is to have good practice of writing bibliographical reference).

Certificate of Authenticity: (choose one from below)

// I certify the code of this lab is entirely my own work.

(or)

// I certify the code of this lab is entirely my own work,

// but I received assistance from [insert name (Instructor)].

// Follow this with a description of the type of assistance (Other than given documents).

```
public class Accounts
{
private double balance;
```

```
private int accountNumber;

        public Account( double initialBalance, int accountNumber )
        {
                if ( initialBalance > 0.0 )
                                balance = initialBalance;
        }

         public void credit( double amount )
        {
                balance = balance + amount;

        }

        public void withdraw(double amount)
   {
                balance = balance - amount;

        }

        public double getBalance()
         {
                return balance;

        }

        public int getAccountNumber()

        {

              return this.accountNumber;

        }

        public static void main(String[] args)throws IOException,..... {



        }

}
```

3. Implement a main method considering the following parts.

    a. Read 10 account details of a person (e.g. 1 acc1111).
    b. Write those (of (a)) details in a file.

    (e.g. 1 acc1111

    2 acc121)

4. Implement a method name *findAccount(xxx)* to check array of account numbers according to the given account number considering the following parts. The method should consider the necessary exceptions.

a. Read the file name from the user.
b. Store the account numbers in an array.
c. Check if the given one is there in the array.

5. In your main method continue the following implementations.

a. Get user choice to withdraw or credit until user type "No".

b. For each case (withdraw and credit), write correct try catch parts *if(accountNumber==-1), if(amount==-1), and if (xxxxx. findAccount(xxx) ==null)*. And implement the necessary lines of code for withdraw and credit.

6. Create a custom exception class by inheriting the Exception class to check if a negative amount is credited.

7. Create a custom exception class by inheriting the *Exception* class to check whether there is insufficient balance when withdrawing.

8. In the withdraw and credit methods check the amount with the balance and negative amount considering the appropriate custom exceptions you have created. (Write the correct try catch and throws the appropriate custom exception)

9. In your main methods give all the exceptions and complete the line *throws IOException,……*

Create a zip file in a format of Lab3-Regno-Coursecode including all your code folders and pdf answer sheets.

Upload the zip file on/before given deadline via team.

Any plagiarized work will be given 0 marks.