Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year:2023), B.Sc. in CSE (Day)

Lab Report NO 03

Course Title : Object Oriented Programming Lab
Course Code : CSE 202
Section : D1

Lab Report on the Topic: Java Exception Handling

Student Details

Name		ID
1.	Jahidul Islam	221002504

Lab Date : 30/05/2023
Submission Date : 06/05/2023
Course Teacher's Name : Ayesha Khatun

Lab Report Status

Marks: …… **Signature:.....**
Comments:..... **Date:.....**

Title:

Exception Handling in Java: Solving a given Lab Task

Question:

Write the code implementation (UML diagram and explanation is given for the desired implementation)-

I. Define a custom built Exception class named NegativeScoreException that throws an exception on negative score.
II. Define a java class JavaProgramming; an object of NegativeScoreException class is thrown by the method setScore(score: double) when a given parameter is negative otherwise initialize the score. setScore(Score: double) will be responsible for setting the individual exam score(quiz, mid, final) Define the method setScore(score: double) of the JavaProgramming class whose UML class diagram is given below. And, also create a getTotal(ob[]: JavaProgramming, numberOfExams: int) method that returns total score if no negative score is given.
III. Create another class "ExceptionError" with main method that receives inputs of individual exam scores and tries to set each of the score using the constructor of JavaProgramming class. Then, Prints the total score using getTotal() method in JavaProgramming Class, if there is no negative score is given as inputs.

JavaProgramming
- score: double
+ JavaProgramming(score: double) + setScore(score: double): void throws NegativeScoreException + getScore(): double + getTotal(ob[]: JavaProgramming, numberOfExams: int): double

Abstract:

The given question revolves around the implementation of a custom Exception class and a Java class called JavaProgramming. The objective is to handle negative scores using exception handling and calculate the total score for a set of exams.

1. Introduction:

Exception handling plays a vital role in creating robust and fault-tolerant Java applications. By anticipating and handling exceptions appropriately, developers can ensure that their programs handle unexpected situations gracefully and continue executing without crashing or producing incorrect results. This lab report presents an overview of Java exception handling mechanisms and demonstrates their practical usage through various examples.

2. Topics that are required to understand and solve this problem:

2.1. Custom Exception class:

- Creating a custom exception class to handle specific types of exceptions.
- Throwing and catching exceptions.
- Defining a custom exception message.

2.2. Java class and object:

- Creating a Java class.
- Instantiating objects from a class.
- Accessing class members (variables and methods) using objects.

2.3. Method overloading:

- Having multiple methods with the same name but different parameters.
- The **setScore()** method is overloaded in the **"JavaProgramming"** class.

2.4. Exception handling:

- Using try-catch blocks to handle exceptions.
- Handling the NegativeScoreException thrown by the setScore() method.

2.5. Encapsulation:

- Encapsulating class members using access modifiers (private, public).
- Accessing and modifying private variables using public methods.

2.6. UML class diagram:

- Understanding and interpreting a UML class diagram.
- Implementing the class and its members based on the UML diagram.

2.7. Arrays:

- Declaring and initializing arrays.
- Accessing array elements using indices.
- Looping through array elements.

2.8.Calculation and accumulation:

- Calculating the total score by accumulating individual scores.
- Using a loop to iterate through an array and perform calculations.

3. Code of the problem:

3.1. Custom Exception Class:

```
// Custom Exception class
class NegativeScoreException extends Exception {
    public NegativeScoreException(String message) {
        super(message);
    }
}
```

3.2. Definition of java class *JavaProgramming*:

```
// JavaProgramming class
class JavaProgramming {
    private double quizScore;
    private double midScore;
    private double finalScore;

    public void setScore(double score) throws NegativeScoreException {
        if (score < 0) {
            throw new NegativeScoreException(
                    message:"Negative value isn't allowed.");
        }
        if (quizScore == 0) {
            quizScore = score;
        } else if (midScore == 0) {
            midScore = score;
        } else if (finalScore == 0) {
            finalScore = score;
        }
    }

    public static double getTotal(JavaProgramming[] ob, int numberOfExams) {
        double tScore = 0;
        for (int i = 0; i < numberOfExams; i++) {
            tScore += ob[i].quizScore + ob[i].midScore + ob[i].finalScore;
        }
        return tScore;
    }
}
```

3.3. “ExceptionError” class with main method:

```
// ExceptionError class
public class ExceptionError {
    public static void main(String[] args) {
        int numberOfExams = 3;
        //array of JavaProgramming objects
        JavaProgramming[] ob = new JavaProgramming[numberOfExams];

        try {
            for (int i = 0; i < numberOfExams; i++) {
                ob[i] = new JavaProgramming();
                double quizScore = 90.0;
                double midScore = 80.0;
                double finalScore = -70.0; // exception happen for this line.

                ob[i].setScore( score:quizScore);
                ob[i].setScore( score:midScore);
                ob[i].setScore( score:finalScore);
            }
            double totalScore = JavaProgramming.getTotal(ob, numberOfExams);
            System.out.println("Total Score: " + totalScore);
        } catch (NegativeScoreException exceptionOBJ) {
            System.out.println("Exception: " + exceptionOBJ.getMessage());
        }
    }
}
```

4. Discussion:

- The problem involves creating a custom Exception class and implementing a Java class called JavaProgramming to handle exam scores.

    - The custom Exception class, NegativeScoreException, extends the Exception class and provides a custom error message.
    - The JavaProgramming class represents an individual exam and has instance variables for quizScore, midScore, and finalScore.
    - The setScore() method in the JavaProgramming class throws the NegativeScoreException if a negative score is encountered.
    - The setScore() method sets the score based on the order in which the scores are received.

    - The getTotal() method in the JavaProgramming class calculates the total score by summing up the scores of each JavaProgramming object in an array.
    - The ExceptionError class creates an array of JavaProgramming objects and sets the individual exam scores.
    - If a NegativeScoreException is thrown, it is caught, and an appropriate error message is displayed.
    - The getTotal() method is called to calculate and print the total score.

- The code can be customized by replacing the example score values with actual input values obtained from the user or other data sources.
- This problem demonstrates the implementation of custom exceptions, exception handling, class design, and calculations on arrays of exam scores.