Date: 06-02-2024

Experiment - 2.1

Aim:

Write a program to create a class named shape. It should contain 2 methods- draw() and erase() which should print “Drawing Shape” and “Erasing Shape” respectively. For this class we have three sub classes- Circle, Triangle and Square and each class override the parent class functions- draw () and erase (). The draw() method should print “Drawing Circle”, “Drawing Triangle”, “Drawing Square” respectively. The erase() method should print “Erasing Circle”, “Erasing Triangle”, “Erasing Square” respectively. Create objects of Circle, Triangle and Square in the following way and observe the polymorphic nature of the class by calling draw() and erase() method using each object.

Code:

```
class Shape {
    public void draw() {
        System.out.println("Drawing Shape");
    }
    public void erase() {
        System.out.println("Erasing Shape");
    }
}
class Circle extends Shape {
    @Override
    public void draw() {
        System.out.println("Drawing Circle");
    }
    @Override
    public void erase() {
        System.out.println("Erasing Circle");
    }
}
class Triangle extends Shape {
    @Override
    public void draw() {
        System.out.println("Drawing Triangle");
    }
    @Override
    public void erase() {
        System.out.println("Erasing Triangle");
    }
```

```
}
class Square extends Shape {
    @Override
    public void draw() {
        System.out.println("Drawing Square");
    }
    @Override
    public void erase() {
        System.out.println("Erasing Square");
    }
}
public class Main {
    public static void main(String[] args) {
        Shape c = new Circle();
        Shape t = new Triangle();
        Shape s = new Square();
        c.draw();
        c.erase();
        t.draw();
        t.erase();
        s.draw();
        s.erase();
    }
}
```

Output:

```
[Running] cd "c:\Users\ankus\OneDrive\Desktop\java test\" && javac
Main.java && java Main
Drawing Circle
Erasing Circle
Drawing Triangle
Erasing Triangle
Drawing Square
Erasing Square
```

Date: 06-02-2024

Experiment - 2.2

Aim:

Write a Program to take care of Number Format Exception if user enters values other than integer for calculating average marks of 2 students. The name of the students and marks in 3 subjects are taken from the user while executing the program. In the same Program write your own Exception classes to take care of Negative values and values out of range (i.e. other than in the range of 0-100)

Code:

```
import java.util.Scanner;
class NegativeValueException extends Exception {
    public NegativeValueException(String message) {
        super(message);
    }
}
class OutOfRangeException extends Exception {
    public OutOfRangeException(String message) {
        super(message);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Enter the name of the first student: ");
            String name1 = scanner.nextLine();
            System.out.print("Enter marks for three subjects for " + name1 + ": ");
            int[] marks1 = readMarks(scanner);
            System.out.print("Enter the name of the second student: ");
            String name2 = scanner.nextLine();
            System.out.print("Enter marks for three subjects for " + name2 + ": ");
            int[] marks2 = readMarks(scanner);
            double average1 = calculateAverage(marks1);
            double average2 = calculateAverage(marks2);
            System.out.println("Average marks for " + name1 + ": " + average1);
            System.out.println("Average marks for " + name2 + ": " + average2);
        } catch (NumberFormatException e) {
            System.out.println("Number Format Exception: Please enter valid integer values for
marks.");
        } catch (NegativeValueException e) {
```

```
            System.out.println("Negative Value Exception: " + e.getMessage());
        } catch (OutOfRangeException e) {
            System.out.println("Out of Range Exception: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
    private static int[] readMarks(Scanner scanner) throws NegativeValueException,
OutOfRangeException {
        int[] marks = new int[3];
        for (int i = 0; i < 3; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            int mark = Integer.parseInt(scanner.nextLine());
            if (mark < 0) {
                throw new NegativeValueException("Marks cannot be negative.");
            } else if (mark < 0 || mark > 100) {
                throw new OutOfRangeException("Marks should be in the range of 0-100.");
            } else {
                marks[i] = mark;
            }
        }
        return marks;
    }
    private static double calculateAverage(int[] marks) {
        int sum = 0;
        for (int mark : marks) {
            sum += mark;
        }
        return (double) sum / marks.length;
    }
}
```

OUTPUT:

```
PS C:\Users\ankus\OneDrive\Desktop\java test> javac Main.java
PS C:\Users\ankus\OneDrive\Desktop\java test> java Main
Enter the name of the first student: Ankush
Enter marks for three subjects for Ankush: Enter marks for subject 1: 70
Enter marks for subject 2: 102
Out of Range Exception: Marks should be in the range of 0-100.
```

Date: 06-02-2024

Experiment - 2.3

Aim:

Write a program that takes as input the size of the array and the elements in the array. The program then asks the user to enter a particular index and prints the element at that index. Index starts from zero. This program may generate Array Index Out Of Bounds Exception or NumberFormatException . Use exception handling mechanisms to handle this exception.

Code:

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      try {
         System.out.print("Enter the size of the array: ");
         int size = Integer.parseInt(scanner.nextLine());
         int[] array = new int[size];
         System.out.print("Enter the elements of the array (separated by space): ");
         String[] elements = scanner.nextLine().split(" ");
         for (int i = 0; i < size; i++) {
            array[i] = Integer.parseInt(elements[i]);   }
         System.out.print("Enter the index to retrieve the element: ");
         int index = Integer.parseInt(scanner.nextLine());
         int element = getElementAtIndex(array, index);
         System.out.println("Element at index " + index + ": " + element);
      } catch (NumberFormatException e) {
         System.out.println("NumberFormatException: Please enter valid integers for array
elements and index.");
      } catch (ArrayIndexOutOfBoundsException e) {
         System.out.println("ArrayIndexOutOfBoundsException: The entered index is out of
bounds for the array.");
      } finally {
         scanner.close();
      }  }
   private static int getElementAtIndex(int[] array, int index) {
      return array[index];
   }
}
```

Output:

```
PS C:\Users\ankus\OneDrive\Desktop\java test> javac Main.java
PS C:\Users\ankus\OneDrive\Desktop\java test> java Main
Enter the size of the array: t
NumberFormatException: Please enter valid integers for array elements and index.
```

Date: 06-02-2024

Experiment - 2.4

Aim:

Write a Java Program to demonstrate the concept of multi-threading.

Code:

```
class MyThread extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(Thread.currentThread().getId() + " Value " + i);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        t1.start();

        MyThread t2 = new MyThread();
        t2.start();
    }
}
```

OUTPUT:

```
[Running] cd "c:\Users\ankus\OneDrive\Desktop\java test\" && javac
Main.java && java Main
16 Value 1
16 Value 2
15 Value 1
15 Value 2
16 Value 3
15 Value 3
15 Value 4
16 Value 4
16 Value 5
15 Value 5
```