RegNo: 23MCA1030 **Name**: Vinayak Kumar Singh

Java Programming Lab (PMCA502P)

Exercise 7: Exception Handling

Question 1: Number Format Exception Handling

Write a Java program that takes user input for two numbers, divides the first number by the second, and prints the result. However, your program should handle potential NumberFormatExceptions if the user inputs non-numeric values.

Your program should:

Prompt the user to enter the first number.

Prompt the user to enter the second number.

Divide the first number by the second number.

Print the result.

If the user enters a non-numeric value for either of the numbers, catch the NumberFormatException, display an error message, and prompt the user to re-enter the number until a valid numeric input is received.

Your program should exit gracefully after printing the result.

Code:

```java
import java.util.Scanner;
public class CalcDivision {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        double firstNumber, secondNumber;
        // Enter the first number
        while (true) {
            System.out.print("Enter first number: ");
            try {
                firstNumber = Double.parseDouble(scanner.nextLine());
                break;
            } catch (NumberFormatException e) {
                System.out.println("Invalid input.Please enter valid number");
            }
        }
        // Enter the second number
        while (true) {
            System.out.print("Enter second number: ");
            try {
                secondNumber = Double.parseDouble(scanner.nextLine());
                if (secondNumber == 0) {
                    System.out.println("Cannot divide by zero.");
                    continue;
                }
                break;
            } catch (NumberFormatException e) {
                System.out.println("Invalid input.Please enter valid number");
            }
        }
        // Divide first number by second number & print result
        double result = firstNumber / secondNumber;
        System.out.println("Result: " + firstNumber + " / " + secondNumber + " = " + result);
        scanner.close();
    }
}
```

Output:

```
PROBLEMS  OUTPUT  PORTS  TERMINAL  COMMENTS

d:\Coding\Java\Ex7\1>cd "d:\Coding\Java\Ex7\1\" && javac CalcDivision.java && java CalcDivision
Enter first number: abc
Invalid input.Please enter valid number
Enter first number: 10
Enter second number: 0
Cannot divide by zero.
Enter second number: 5
Result: 10.0 / 5.0 = 2.0
```

2. Input Validation and Exception Handling

Write a Java program that prompts the user to enter their age. If the user enters a non-numeric value or a negative number, catch the NumberFormatException or IllegalArgumentException accordingly, display an appropriate error message, and prompt the user to re-enter their age until a valid age is provided. Finally, print a message welcoming the user with their age.

Code:

```
import java.util.Scanner;
public class AgeCheck {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); // read user input
        int age = -1; // Initialize age
        while (true) { // Loop until valid age entered
            System.out.print("Enter your age: ");
            try {
                age = Integer.parseInt(scanner.nextLine()); // Parse the user input as an integer
                if (age < 0) { // Check if age is negative
                    throw new IllegalArgumentException("Age cannot be negative"); // Throw an
exception if negative
                }
                break; // Break out of loop if entered age is valid
            } catch (NumberFormatException e) {
                System.out.println("Invalid input kindly enter a valid number for age."); // Print error
            } catch (IllegalArgumentException e) {
                System.out.println(e.getMessage()); // Print the exception message for negative age
            }
        }
        System.out.println("Welcome! Your are " + age + " years old.");
        scanner.close();
    }
}
```

Output:

```
d:\Coding\Java\Ex7\2>cd "d:\Coding\Java\Ex7\2\" && javac AgeCheck.java && java AgeCheck
Enter your age: vinayak
Invalid input kindly enter a valid number for age.
Enter your age: -10
Age cannot be negative
Enter your age: 21
Welcome! Your are 21 years old.
```

3. Input Validation with a Range

Write a Java program that prompts the user to enter a number between 1 and 100 (inclusive). If the user enters a non-numeric value or a number outside this range, catch the NumberFormatException or IllegalArgumentException accordingly, display an appropriate error message, and prompt the user to re-enter the number until a valid input is provided. Finally, print the valid number entered by the user.

Code:

```
import java.util.Scanner;
public class NumValidChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int number = -1;
        while (true) {
            System.out.print("Enter number between 1 and 100: ");
            try {
                number = Integer.parseInt(scanner.nextLine());
                if (number < 1 || number > 100) {
                    throw new IllegalArgumentException("Number must be between 1 and 100
inclusive.");
                }
                break;
            } catch (NumberFormatException e) {
                System.out.println("Invalid input kindly enter a valid number  between 1 and 100.");
```

```
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }
    }
    System.out.println("You entered: " + number);
    scanner.close();
  }
}
```

Output:

```
d:\Coding\Java\Ex7\2>cd "d:\Coding\Java\Ex7\3\" && javac NumValidChecker.java && java NumValidChecker
Enter number between 1 and 100: xyz
Invalid input kindly enter a valid number  between 1 and 100.
Enter number between 1 and 100: 150
Number must be between 1 and 100 inclusive.
Enter number between 1 and 100: 50
You entered: 50
```

4. : Handling ArithmeticException

Write a Java program that takes user input for two numbers and performs division. If the user attempts to divide by zero, catch the ArithmeticException, display an appropriate error message, and prompt the user to re-enter the second number until a non-zero value is provided. Finally, print the result of the division**.**

Code:

```
import java.util.Scanner;
public class DivCalc {
  public static void main(String[] args) {
    Scanner inputScanner = new Scanner(System.in);
    double firstNumber, secondNumber;
    System.out.print("Enter the first number: ");
    firstNumber = Double.parseDouble(inputScanner.nextLine());
    while (true) {
      System.out.print("Enter the second number: ");
```

```
        try {
            secondNumber = Double.parseDouble(inputScanner.nextLine());
            if (secondNumber != 0) {
                break; // Break out of the loop if the second number is non-zero
            } else {
                throw new ArithmeticException("Cannot divide by zero. Please enter a non-zero
value.");
            }
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please enter a valid number.");
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        }
    }
    double result = firstNumber / secondNumber;
    System.out.println("Result: " + firstNumber + " / " + secondNumber + " = " + result);
    inputScanner.close();
  }
}
```

Output:

```
d:\Coding\Java\Ex7\4>cd "d:\Coding\Java\Ex7\4\" && javac DivCalc.java && java DivCalc
Enter the first number: 10
Enter the second number: 0
Cannot divide by zero. Please enter a non-zero value.
Enter the second number: vinayak
Invalid input. Please enter a valid number.
Enter the second number: 5
Result: 10.0 / 5.0 = 2.0
```