Experiment No 5

Name: Divyesh Khunt **Sapid:60009210116** **Batch :D12**

Aim: - Implement Exception Handling and Multithreading in Java
Theory: -

1. **Exception in Java**

An exception is an "unwanted or unexpected event", which occurs during the execution of the program i.e., at run-time, that disrupts the normal flow of the program's instructions. When an exception occurs, the execution of the program gets terminated.

Why does an Exception occur?
An exception can occur due to several reasons like a Network connection problem, Bad input provided by a user, Opening a non-existing file in your program, etc.

Try, Catch, Throw, Throws and Finally
try: The try block contains a set of statements where an exception can occur.

```
try
{
   // statement(s) that might cause exception
}
```

catch: The catch block is used to handle the uncertain condition of a try block. A try block is always followed by a catch block, which handles the exception that occurs in the associated try block.

```
catch
{
  // statement(s) that handle an exception
  // examples, closing a connection, closing
  // file, exiting the process after writing
  // details to a log file.
}
```

throw: The throw keyword is used to transfer control from the try block to the catch block.

throws: The throws keyword is used for exception handling without try & catch block. It specifies the exceptions that a method can throw to the caller and does not handle itself.

finally:

The finally block performs the clean-up operations such closing the database connections, closing the files and resources which is opened.
The finally clause would always be executed irrespective of whether the exceptions happened or not and whether exception is handled or not.
This block will not get executed in a certain situation such as when the system got hung or

the program crashed due to some fatal error or exiting the JVM using System. Exit ()

The finally block also cannot exist separately, it has to be associated with a try block. Valid scenarios would be try – finally and try – catch – finally.

throw and throws in Java

If a method does not handle a checked exception, the method must declare it using the throws keyword. The throws keyword appears at the end of a method's signature.

You can throw an exception, either a newly instantiated one or an exception that you just caught, by using the throw keyword.

The throws are used to postpone the handling of a checked exception and throw is used to invoke an exception explicitly.

Catching Multiple Exceptions

There can be a situation where a particular code has the tendency to throw multiple exceptions. You want to catch each of the exceptions that may occur within a try block. There are two ways to handle this scenario

2. Multithreading in Java

Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread. So, threads are light-weight processes within a process.

Threads can be created by using two mechanisms:

1. Extending the Thread class
2. Implementing the Runnable Interface

Thread creation by extending the Thread class

We create a class that extends the **java.lang.Thread** class. This class overrides the run () method available in the Thread class. A thread begins its life inside run () method. We create an object of our new class and call start () method to start the execution of a thread. Start () invokes the run () method on the Thread object.

Thread Class vs Runnable Interface

1. If we extend the Thread class, our class cannot extend any other class because Java doesn't support multiple inheritance. But, if we implement the Runnable interface, our class can still extend other base classes.
2. We can achieve basic functionality of a thread by extending Thread class because it provides some inbuilt methods like yield (), interrupt () etc. that are not available in Runnable interface.
3. Using runnable will give you an object that can be shared amongst multiple threads.

3. Lab Assignments to complete in this session

i. Write A Program for producer consumer problem in java

ii. Write a Java Program to input the data through command Line and Find out total valid and in-valid integers. (Hint: use exception handling)

q2.java > ...

```java
import java.util.Scanner;

public class q2 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        System.out.print(s:"Enter the number of integers: ");
        n = sc.nextInt();
        int[] numbers = new int[n];
        int validCount = 0;
        int invalidCount = 0;
        System.out.println("Enter " + n + " integers:");
        for (int i = 0; i < n; i++) {
            try {
                numbers[i] = sc.nextInt();
                validCount++;
            } catch (Exception e) {
                sc.nextLine();
                System.out.println(x:"Invalid input");
                invalidCount++;
            }
        }
        System.out.println("Valid integers:" + validCount);
        System.out.println("Invalid integers:" + invalidCount);
        sc.close();
    }
}
```

```
Enter the number of integers: 5
Enter 5 integers:
16
-54
h
Invalid input
```

iii. Write a Java Program to calculate the Result. Result should consist of name, seatno, date, center number and marks of semester three exam. Create a User Defined Exception class MarksOutOfBoundsException, If Entered marks of any subject is greater than 100 or less than 0, and then program should create a user defined Exception of type MarksOutOfBoundsException and must have a provision to handle it

q3.java > ...

```java
import java.util.Scanner;

class myexception extends Exception {
    public myexception(String message) {
        super(message);
    }
}

public class q3 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int seatno, centerno, sem1, sem2, sem3;
        String name, date;

        System.out.print(s:"Enter name: ");
        name = sc.nextLine();
        System.out.print(s:"Enter seat number: ");
        seatno = sc.nextInt();
        sc.nextLine();
        System.out.print(s:"Enter date: ");
        date = sc.nextLine();
        System.out.print(s:"Enter center number: ");
        centerno = sc.nextInt();
        try {
            System.out.print(s:"Enter marks for semester 1: ");
            sem1 = sc.nextInt();
            if (sem1 < 0 || sem1 > 100) {
                throw new myexception(message:"Marks out of bound");
            }
            System.out.print(s:"Enter marks for semester 2: ");
            sem2 = sc.nextInt();
            if (sem2 < 0 || sem2 > 100) {
                throw new myexception(message:"Marks out of bound");
            }
            System.out.print(s:"Enter marks for semester 3: ");
            sem3 = sc.nextInt();
            if (sem3 < 0 || sem3 > 100) {
                throw new myexception(message:"Marks out of bound");
            }
        } catch (myexception e) {
            System.out.println(e.getMessage());
        } finally {
        }
    }
}
```

```
e60d7541\redhat.java\jdt_ws\exp5_f0a629a7\bin' 'q3'
Enter name: Divyesh khunt
Enter seat number: 116
Enter date: 15-11-2023
Enter center number: 04
Enter marks for semester 1: 98
Enter marks for semester 2: 120
Marks out of bound
PS D:\dk\exp5>
```

iv. Write java program to print Table of Five, Seven and Thirteen using Multithreading (Use Thread class for the implementation). Also print the total time taken by each thread for the execution.

q4.java > ...

```java
class seven extends Thread {
    public void run() {
        for (int i = 1; i < 11; i++) {
            System.out.println(7 +"*"+ i+ "=" + 7*i);
        }
    }
}
class five extends Thread {
    public void run() {
        for (int i = 1; i < 11; i++) {
            System.out.println(5 +"*"+ i+ "=" + 5*i);
        }

    }
}
class thirteen extends Thread {
    public void run() {
        for (int i = 1; i < 11; i++) {
            System.out.println(13 +"*"+ i+ "=" + 13*i);
        }
    }
}

class q4{
    Run | Debug
    public static void main(String args[]) {
      seven t1=new seven();
      five t2=new five();
      thirteen t3=new thirteen();
      t1.start();
      t2.start();
      t3.start();
    }
}
```

```
13*1=13
13*2=26
5*1=5
7*1=7
13*3=39
5*2=10
7*2=14
7*3=21
7*4=28
13*4=52
5*3=15
7*5=35
13*5=65
5*4=20
7*6=42
13*6=78
5*5=25
7*7=49
13*7=91
5*6=30
7*8=56
13*8=104
5*7=35
7*9=63
13*9=117
5*8=40
5*9=45
7*10=70
13*10=130
5*10=50
PS D:\dk\exp5>
```

v. Write a program Create User-Defined Exception in Java for Validating Login Credentials

vi. Write java program to implement the concept of Thread Synchronization