Experiment Number : 13 **Date:**

Implementation of User-defined Exception in Java

Aim:

Write a java program to implement User-defined Exception.

PRE LAB EXERCISE

QUESTIONS

1. What is Java custom exception?

A **custom exception** in Java is a user-defined exception created by extending the Exception or RuntimeException class.

It is used to define application-specific errors that are not covered by built-in Java exceptions.

2. What are the two types of custom exceptions in Java?

A custom exception is a user-defined exception created by extending Exception or RuntimeException to handle application-specific errors.

The two types of custom exceptions are:

1. Checked custom exception
2. Unchecked custom exception

IN LAB EXERCISE

Objective

To understand and implement Checked and Unchecked custom exceptions.

Source Code

Ex 1: // Custom Checked Exception

```
class InvalidAgeException extends Exception {
    public InvalidAgeException(String m) {
        super(m);
    }
}
// Using the Custom Exception
public class AgeCheck{
    public static void validate(int age)
      throws InvalidAgeException {
        if (age < 18) {
            throw new InvalidAgeException("Age must be 18 or above.");
        }
        System.out.println("Valid age: " + age);
    }
public static void main(String[] args) {
  try {
            validate(12);
        } catch (InvalidAgeException e) {
            System.out.println("Caught Exception: " + e.getMessage());
        }
    }
}
```

Output 1

Caught Exception: Age must be 18 or above.

```
Caught Exception: Age must be 18 or above.
```

Ex 2: // Custom Unchecked Exception

```
class DivideByZeroException extends RuntimeException {
    public DivideByZeroException(String m) {
        super(m);
    }
}
// Using the Custom Exception
public class TestSample {
    public static void divide(int a, int b) {
        if (b == 0) {
            throw new DivideByZeroException("Division by zero is not allowed.");
        }
        System.out.println("Result: " + (a / b));
    }
public static void main(String[] args) {
  try {
            divide(10, 0);
        } catch (DivideByZeroException e) {
            System.out.println("Caught Exception: " + e.getMessage());
        }
    }
}
```

Output 2

Caught Exception: Division by zero is not allowed.

```
Caught Exception: Division by zero is not allowed.
```

POST LAB EXERCISE

1. What is required to create a custom exception in Java?
   A. Extend the Object class
   B. Extend the Throwable class
   C. Extend Exception or RuntimeException
   D. Implement the Serializable interface

Extend Exception or Runtime Exception

2. List some use cases for the checked and unchecked exceptions.

To create a custom exception in Java, we must extend Exception or RuntimeException.

Checked exceptions are used for recoverable errors like file handling and database operations, while unchecked exceptions are used for programming errors such as division by zero and null pointer access.

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	
Total	**30**	
Faculty Signature		