Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

Experiment No. 10
Roll no :07
Name : Atharva Borse
Implement program on User Defined Exception
Date of Performance:
Date of Submission:

Aim: Implement program on User Defined Exception.

Objective:

Theory:

An exception is an issue (run time error) that occurred during the execution of a program. When an exception occurred the program gets terminated abruptly and, the code past the line that generated the exception never gets executed.

Java provides us the facility to create our own exceptions which are basically derived classes of Exception. Creating our own Exception is known as a custom exception or user-defined exception. Basically, Java custom exceptions are used to customize the exception according to user needs. In simple words, we can say that a User-Defined Exception or custom exception is creating your own exception class and throwing that exception using the 'throw' keyword.

For example, MyException in the below code extends the Exception class.

Why use custom exceptions?

Java exceptions cover almost all the general types of exceptions that may occur in the programming. However, we sometimes need to create custom exceptions.

Following are a few of the reasons to use custom exceptions:

- To catch and provide specific treatment to a subset of existing Java exceptions.

- Business logic exceptions: These are the exceptions related to business logic and workflow.

It is useful for the application users or the developers to understand the exact problem. In order to create a custom exception, we need to extend the Exception class that belongs to **java.lang package.**

Example: We pass the string to the constructor of the superclass- Exception which is obtained using the "getMessage()" function on the object created.

```
// A Class that represents use-defined exception

class MyException extends Exception {

public MyException(String s)

        {

                // Call constructor of parent Exception

                super(s);

        }

}


// A Class that uses above MyException public

class Main {

        // Driver Program       public      static

void main(String args[])

        {

                try {
```

```
                    // Throw an object of user defined exception
        throw new MyException("UserDefined Exception");
                }
                catch (MyException ex) {
                        System.out.println("Caught");

                        // Print the message from MyException object
                        System.out.println(ex.getMessage());
                }
        }
```

} Output:

```
Caught
UserDefined Exception
```

Code:

Conclusion:

Comment on how user defined exceptions used in java.

```
// Custom exception class
class AgeException extends Exception {
    // Constructor to initialize exception message
    public AgeException(String message) {
        super(message);
    }
}

// Main class
public class UserDefinedExceptionDemo {
    // Method to check if age is valid
    static void validateAge(int age) throws AgeException {
        if (age < 18) {
            throw new AgeException("Age is less than 18, access denied.");
        } else {
            System.out.println("Age is valid, access granted.");
        }
    }

    // Main method
    public static void main(String[] args) {
        try {
            // Test with valid and invalid age
            validateAge(16);  // This will throw an exception
            validateAge(21);  // This won't be executed due to the exception above
        } catch (AgeException e) {
            // Handle the custom exception
            System.out.println("Exception caught: " + e.getMessage());
        }
```

```
        System.out.println("Program continues...");
    }
}
```

Output Generated Files

```
Exception caught: Age is less than 18, access denied.
Program continues...
```