Name; Sumit Patel
Roll no. 40
Experiment No. 10
Implement program on User Defined Exception
Date of Performance: 09-10-2024
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:

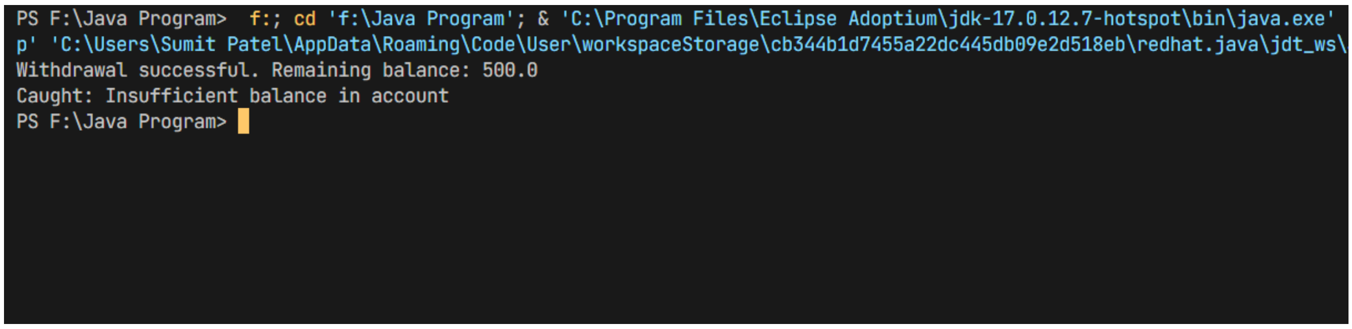
```
// A Class that represents user-defined exception
class InsufficientBalanceException extends Exception {
    public InsufficientBalanceException(String message) {
        // Call constructor of parent Exception
        super(message);
    }
}

// A Class that uses above InsufficientBalanceException
public class BankAccount {
    private double balance;

    public BankAccount(double balance) {
        this.balance = balance;
    }

    public void withdraw(double amount) throws
InsufficientBalanceException {
        if (amount > balance) {
            throw new InsufficientBalanceException("Insufficient
balance in account");
        } else {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining
balance: " + balance);
        }
    }

    public static void main(String[] args) {
        BankAccount account = new BankAccount(1000.0);

        try {
            account.withdraw(500.0);
            account.withdraw(600.0); // This will throw
InsufficientBalanceException
        } catch (InsufficientBalanceException e) {
            System.out.println("Caught: " + e.getMessage());
        }
    }
}
```

Output:

Withdrawal successful. Remaining balance: 500.0

Caught: Insufficient balance in account

Screenshot:



```
PS F:\Java Program>  f:; cd 'f:\Java Program'; & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.12.7-hotspot\bin\java.exe'
p' 'C:\Users\Sumit Patel\AppData\Roaming\Code\User\workspaceStorage\cb344b1d7455a22dc445db09e2d518eb\redhat.java\jdt_ws\
Withdrawal successful. Remaining balance: 500.0
Caught: Insufficient balance in account
PS F:\Java Program>
```

Conclusion:

Comment on how user defined exceptions used in java.

In Java, user-defined exceptions are used to handle specific error scenarios that are not covered by the built-in exceptions. By creating custom exceptions, developers can provide more informative and meaningful error messages, making it easier to diagnose and fix problems. User-defined exceptions are typically used to represent business logic errors, such as invalid data or unexpected conditions, and can be thrown and caught just like built-in exceptions. This allows developers to write more robust and maintainable code, with better error handling and recovery mechanisms, ultimately leading to more reliable and user-friendly applications.