OOPM Lab
Lab Assingment number 08

Name: Aamir Ansari **Batch:** A **Roll no.** 01

Aim: Write a JAVA program to create user defined exception.

Problem statement:

Write a java program where the user will enter login id and password as input. Program will check whether the entered password is valid or invalid based on following password rules:

1. Passwords should contain at least one digit(0-9).
2. Password length should be between 8 to 15 characters.
3. Password should contain at least one special character (@, #, %, &, !, $, etc….)

If a user enters a valid password satisfying above criteria then show “Login Successful Message”. Otherwise create InvalidPasswordException stating "Please enter valid password of length 8 to 15 characters containing at least one digit and one special symbol"

Theory:

1. Exception handling in java

   An exception (or exceptional event) is a problem that arises during the execution of a program. When an Exception occurs the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommended, therefore, these exceptions are to be handled.

   1. An exception can occur for many different reasons. Following are some scenarios where an exception occurs.
   2. A user has entered an invalid data.
   3. A file that needs to be opened cannot be found.
   4. A network connection has been lost in the middle of communications or the JVM has run out of memory.

2. Create and throw user defined or custom exceptions

   To display the message override the toString() method or, call the superclass parameterized constructor by passing the message in String format.

```
AgeDoesnotMatchException(String msg){
        super(msg);
}
```

   Or,

```
public String toString(){
        return "CustomException[Age is not between 17 and 24]";
}
```

Then, in other classes wherever you need this exception to be raised, create an object of the created custom exception class and, throw the exception using the throw keyword.

```
MyException ex = new MyException ();
If(condition…….){
  throw ex;
}
```

3. Advantages of exception handling

1. Separating Error-Handling Code from "Regular" Code
2. Propagating Errors Up the Call Stack
3. Grouping and Differentiating Error Types
4. Meaningful Error Reporting

// **code**

```
import java.util.*;
import java.lang.*;


class InvalidPasswordException extends Exception {
   // data member
   int conditionViolated = 0;
   // constructor
   public InvalidPasswordException(int condition) {
      super("Invalic Password: ");
      conditionViolated = condition;
   }

   public String printMessage() {
      switch (conditionViolated) {
         // length
         case 1:
```

```
            return ("Password length should be between 8 to 15 characters");
        // spaces
        case 2:
            return ("Password should not contain SPACES");
        // digit
        case 3:
            return ("Password should contain at least one DIGIT");
        // special characters
        case 4:
            return ("Password should contain at least one SPECIAL CHARACTER");
        // capital alphabet
        case 5:
            return ("Password must contain at least one CAPTIAL alphabet");
        // small alphabet
        case 6:
            return ("Password must contain at least one SMALL alphabet");
      }
      return("");
   }
}

public class PasswordChecker {
   public static void isValid(String password) throws InvalidPasswordException  {
      // password length between 8-15
      if (!(password.length()>=8 && password.length()<=15)) {
         throw new InvalidPasswordException(1);
      }

      // check for presence of spaces
      if (password.contains(" ")) {
         throw new InvalidPasswordException(2);
      }

      // check for presence of digit
      if (true) {
```

```
    boolean flag = false;
    for (int i=0 ; i<=10 ; i++) {
        String number = Integer.toString(i);
        if (password.contains(number)) {
            flag = true;
        }
    }
    if (flag == false) {
        throw new InvalidPasswordException(3);
    }
}

// checks for special characters
if (!(password.contains("@") || password.contains("#")
    || password.contains("!") || password.contains("~")
    || password.contains("$") || password.contains("%")
    || password.contains("^") || password.contains("&")
    || password.contains("*") || password.contains("(")
    || password.contains(")") || password.contains("-")
    || password.contains("+") || password.contains("/")
    || password.contains(":") || password.contains(".")
    || password.contains(",") || password.contains("<")
    || password.contains(">") || password.contains("?")
    || password.contains("|"))) {
    throw new InvalidPasswordException(4);
}

// check for capital letters
if (true) {
    boolean flag = false;
    for (int i=65 ; i<=90 ; i++) {
        char alphabet = (char)i;
        String str = Character.toString(alphabet);
        if (password.contains(str)) {
            flag = true;
```

```
                break;
            }
        }
        if (flag == false) {
            throw new InvalidPasswordException(5);
        }
    }

    // check for small letter
    if (true) {
        boolean flag = false;
        for (int i=97 ; i<=122 ; i++) {
            char alphabet = (char)i;
            String str = Character.toString(alphabet);
            if (password.contains(str)) {
                flag = true;
                break;
            }
        }
        if (flag == false) {
            throw new InvalidPasswordException(6);
        }
    }
}

public static void main(String args[]) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter Password :  ");
    String password = sc.nextLine();
    try {
        System.out.println("Is password "+password+" valid?");
        isValid(password);
        System.out.println("Password is Valid!!");
    }
    catch (InvalidPasswordException e) {
```

```
            System.out.println(e.getMessage());
            System.out.println(e.printMessage());
        }
    }
}
```

// output

```
E:\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 08\Code>javac PasswordChecker.java

E:\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 08\Code>java PasswordChecker
Enter Password :  !@Wifi34
Is password !@Wifi34 valid?
Password is Valid!!

E:\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 08\Code>java PasswordChecker
Enter Password :  hello,there
Is password hello,there valid?
Invalic Password:
Password should contain at least one DIGIT

E:\Sem-3\LabWork - Assignments\OOPM\Lab Assignment 08\Code>java PasswordChecker
Enter Password :  Wifi34as
Is password Wifi34as valid?
Invalic Password:
Password should contain at least one SPECIAL CHARACTER
```