Rajalakshmi Engineering College

Name: MRIDHULA DEVI M
Email: 240701337@rajalakshmi.edu.in
Roll no:
Phone: 9840329629
Branch: REC
Department: CSE - Section 8
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 8_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Tim was tasked with creating a user profile system that validates the user's date of birth input. The system should throw a custom exception, InvalidDateOfBirthException, if the date is not in the specified format "dd-mm-yyyy" or if it represents an invalid calendar date.

The main method takes user input, validates the date of birth, and prints whether it is valid or not.

Input Format

The input consists of a string, representing the date of birth of the user.

Output Format

The output displays one of the following results:

If the entered date of birth is valid according to the specified format, the program prints:

"[Date] is a valid date of birth"

If the entered date of birth is not valid according to the specified format, the program prints:

"Invalid date: [Date]"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 01-01-2000

Output: 01-01-2000 is a valid date of birth

Answer

```
// You are using Java
import java.util.Scanner;
import java.text.SimpleDateFormat;
import java.text.ParseException;

class InvalidDateOfBirthException extends Exception {
   InvalidDateOfBirthException(String date) {
      super("Invalid date: " + date);
   }
}

public class Main {
   static void validateDOB(String dob) throws InvalidDateOfBirthException {
      SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
      sdf.setLenient(false);
      try {
         sdf.parse(dob);
      } catch (ParseException e) {
         throw new InvalidDateOfBirthException(dob);
      }
      System.out.print(dob + " is a valid date of birth");
```

```
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String dob = sc.nextLine();
        try {
            validateDOB(dob);
        } catch (InvalidDateOfBirthException e) {
            System.out.print(e.getMessage());
        }
    }
}
```

Status : Correct **_Marks : 10/10_**

2. Problem Statement

Camila, a user of a social media platform, is looking to change her password to enhance account security. The platform enforces specific rules for password strength to ensure the safety of user accounts. Camila needs a program that prompts her to enter a new password and throws custom exceptions based on the strength of the password.

Password Strength Criteria:

Weak Password:

Length less than 8 characters.Medium Password:

Length 8 or more characters.Missing a mix of uppercase letters, lowercase letters, and digits.

Implement a custom exception, to assist Camila in changing her password securely. The program should interactively take user input for a new password, categorize its strength, and handle custom exceptions (WeakPasswordException and MediumPasswordException) if the password fails to meet the specified criteria.

Input Format

The input consists of a string s, representing the new password.

Output Format

The output is displayed in the following format:

If the entered password meets the strength criteria, the program outputs

"Password changed successfully!"

If the entered password is weak, the program outputs

"Error: Weak password. It must be at least 8 characters long."

If the entered password is of medium strength, the program outputs

"Error: Medium password. It must include a mix of uppercase letters, lowercase letters, and digits."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: ComplexP@ss1

Output: Password changed successfully!

Answer

```
// You are using Java
import java.util.Scanner;

class WeakPasswordException extends Exception {
   WeakPasswordException() {
      super("Error: Weak password. It must be at least 8 characters long.");
   }
}

class MediumPasswordException extends Exception {
   MediumPasswordException() {
      super("Error: Medium password. It must include a mix of uppercase letters,
lowercase letters, and digits.");
   }
}
```

```
public class Main {
   static void checkPassword(String s) throws WeakPasswordException,
MediumPasswordException {
      if (s.length() < 8) throw new WeakPasswordException();

      boolean hasUpper = false, hasLower = false, hasDigit = false;
      for (char c : s.toCharArray()) {
         if (Character.isUpperCase(c)) hasUpper = true;
         else if (Character.isLowerCase(c)) hasLower = true;
         else if (Character.isDigit(c)) hasDigit = true;
      }
      if (!(hasUpper && hasLower && hasDigit)) throw new
MediumPasswordException();

      System.out.print("Password changed successfully!");
   }

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      String s = sc.nextLine();
      try {
         checkPassword(s);
      } catch (WeakPasswordException | MediumPasswordException e) {
         System.out.print(e.getMessage());
      }
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Theo is trying to update his payment information on a subscription-based streaming service. To proceed, the system requires Theo to provide a valid credit card number consisting of 16 digits. However, Theo wants to make sure that the credit card number he enters meets the specified criteria with proper exception handling.

The credit card number must consist of exactly 16 digits.If the entered credit card number does not meet the specified criteria, the program should throw a custom exception, InvalidCreditCardException, and provide Theo with specific error messages:If the length of the credit card number is not 16 digits, the exception message should be: "Invalid credit card number length."If the credit card number contains non-numeric characters, the exception message should be: "Invalid credit card number format."

Implement a custom exception, InvalidCreditCardException, to fulfill Theo's requirements and keep his payment information secure.

Input Format

The input consists of a string value 's', consisting of the 16-digit credit card number.

Output Format

The output is displayed in the following format:

If the entered credit card number is valid, the program should output a success message:

"Payment information updated successfully!"

If the entered credit card has more than 16 digits or less than 16 digits it displays

"Error: Invalid credit card number length."

If the entered 16-digit credit card has non-integers it displays

"Error: Invalid credit card number format."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1234567890123456
Output: Payment information updated successfully!

Answer

```
// You are using Java
import java.util.Scanner;

class InvalidCreditCardException extends Exception {
   InvalidCreditCardException(String message) {
      super(message);
   }
}

public class Main {
   static void validateCard(String s) throws InvalidCreditCardException {
      if (s.length() != 16) throw new InvalidCreditCardException("Error: Invalid 
credit card number length.");
      for (char c : s.toCharArray()) {
         if (!Character.isDigit(c)) {
            throw new InvalidCreditCardException("Error: Invalid credit card number 
format.");
         }
      }
      System.out.print("Payment information updated successfully!");
   }

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      String s = sc.nextLine();
      try {
         validateCard(s);
      } catch (InvalidCreditCardException e) {
         System.out.print(e.getMessage());
      }
   }
}
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

A company is developing a user registration system that requires users to provide valid email addresses. The development team is implementing an EmailValidator program to ensure that the entered email addresses meet certain criteria using exception handling.

The email address must contain the "@" symbol.The email address must consist of a non-empty username(before "@" symbol) and a non-empty domain(after '"@" symbol).The domain part of the email address must contain at least one period (".").The email address must not contain leading or trailing spaces.

Implement a custom exception, InvalidEmailException, to fulfill the company's requirements and validate it according to the specified rules.

Input Format

The input consists of a string value 's', which represents the email address.

Output Format

The output is displayed in the following format:

If the entered email address is valid according to the specified rules, the program prints:

"Email address is valid!"

If the entered email address misses the username or domain part or misses "@" symbol or has two or more "@" symbols or misses '.' in the domain part it outputs:

"Error: Invalid email format."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: johndoe@example.com
Output: Email address is valid!

Answer

```
// You are using Java
import java.util.Scanner;

class InvalidEmailException extends Exception {
    InvalidEmailException() {
        super("Error: Invalid email format.");
    }
}

public class Main {
    static void validateEmail(String s) throws InvalidEmailException {
        if (!s.equals(s.trim())) throw new InvalidEmailException();
        int atCount = 0;
        for (char c : s.toCharArray()) {
            if (c == '@') atCount++;
        }
        if (atCount != 1) throw new InvalidEmailException();

        int atIndex = s.indexOf('@');
        if (atIndex == 0 || atIndex == s.length() - 1) throw new
InvalidEmailException();

        String domain = s.substring(atIndex + 1);
        if (!domain.contains(".")) throw new InvalidEmailException();

        System.out.print("Email address is valid!");
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        try {
            validateEmail(s);
        } catch (InvalidEmailException e) {
            System.out.print(e.getMessage());
        }
    }
}
```

Status : Correct ***Marks : 10/10***