Rajalakshmi Engineering College

Name: Mithran VT
Email: 240701312@rajalakshmi.edu.in
Roll no:
Phone: 9952919350
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 : 30

Section 1 : Coding

1. Problem Statement

Alice is designing a program that requires users to enter positive numbers. She wants to implement a solution that validates whether the entered number is positive. In case the input is not a positive number, she wants to throw a custom exception.

The number should be a positive integer.If this condition is violated, the program should throw a custom exception:InvalidPositiveNumberException with the message "Invalid input. Please enter a positive integer."

Implement a custom exception, InvalidPositiveNumberException , to handle cases where the entered number does not meet the specified criteria.

Input Format

The input consists of an integer value 'n', representing the entered number.

Output Format

The output is displayed in the following format:

If the validation passes, print

"Number {number} is positive."

The {number} represents the entered positive integer.

If the entered number is negative then it displays

"Error: Invalid input. Please enter a positive integer."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 100
Output: Number 100 is positive.

Answer

```
// You are using Java
// Custom exception class
class InvalidPositiveNumberException extends Exception {
   public InvalidPositiveNumberException(String message) {
      super(message);
   }
}

 class PositiveNumberValidator {
   public static void validate(int number) throws InvalidPositiveNumberException
{
      if (number <= 0) {
         throw new InvalidPositiveNumberException("Invalid input. Please enter a
positive integer.");
      } else {
            System.out.println("Number " + number + " is positive.");
        }
    }

    public static void main(String[] args) {
        java.util.Scanner scanner = new java.util.Scanner(System.in);
        int n = scanner.nextInt();

        try {
            validate(n);
        } catch (InvalidPositiveNumberException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

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

2. Problem Statement

Hemanth is designing a banking system for XYZ Bank. The system should allow customers to perform deposit, withdrawal, and balance inquiry operations. Implement exception handling for scenarios involving invalid transaction amounts or insufficient funds.

Create two custom exception classes, InvalidAmountException and InsufficientFundsException, both extending the Exception class.Throw an InvalidAmountException with a message if the deposit amount is less than or equal to zero.Throw an InsufficientFundsException if the withdrawal amount is greater than the available balance.Deduct the withdrawal amount from the balance if the withdrawal is successful.

Assist Hemanth in designing the program.

Input Format

The first line of input consists of a double value B, representing the initial balance.

The second line consists of a double value D, representing the deposit amount.

The third line consists of a double value W, representing the withdrawal amount.

Output Format

If the withdrawal is successful, print the amount withdrawn and the current balance, rounded off to one decimal place.

If an InvalidAmountException occurs, print "Error: [D] is not valid".

If an InsufficientFundsException occurs, print "Error: Insufficient funds".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1050.1
270.2
150.3

Output: Amount Withdrawn: 150.3
Current Balance: 1170.0

Answer

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

// Custom exception for invalid deposit amount
class InvalidAmountException extends Exception {
   public InvalidAmountException(String message) {
      super(message);
   }
}

// Custom exception for insufficient funds
class InsufficientFundsException extends Exception {
   public InsufficientFundsException(String message) {
      super(message);
   }
}

 class XYZBank {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);

      double balance = scanner.nextDouble();
      double deposit = scanner.nextDouble();
      double withdrawal = scanner.nextDouble();

      try {
          // Deposit validation
          if (deposit <= 0) {
              throw new InvalidAmountException("Error: " + deposit + " is not valid");
          }
          balance += deposit;

          // Withdrawal validation
          if (withdrawal > balance) {
              throw new InsufficientFundsException("Error: Insufficient funds");
          }

          balance -= withdrawal;
          System.out.printf("Amount Withdrawn: %.1f%n", withdrawal);
          System.out.printf("Current Balance: %.1f%n", balance);

      } catch (InvalidAmountException | InsufficientFundsException e) {
          System.out.println(e.getMessage());
      }
   }
}
```

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

3. 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;

// Custom exception for invalid email format
class InvalidEmailException extends Exception {
   public InvalidEmailException(String message) {
      super(message);
   }
}

 class EmailValidator {
    public static void validateEmail(String email) throws InvalidEmailException {
        // Check for leading/trailing spaces
        if (!email.equals(email.trim())) {
            throw new InvalidEmailException("Error: Invalid email format.");
        }

        // Check for exactly one '@' symbol
        int atCount = email.length() - email.replace("@", "").length();
        if (atCount != 1) {
            throw new InvalidEmailException("Error: Invalid email format.");
        }

        String[] parts = email.split("@");
        if (parts.length != 2 || parts[0].isEmpty() || parts[1].isEmpty()) {
            throw new InvalidEmailException("Error: Invalid email format.");
        }

        // Check if domain contains at least one '.'
        if (!parts[1].contains(".")) {
            throw new InvalidEmailException("Error: Invalid email format.");
        }

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

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String email = scanner.nextLine();

        try {
            validateEmail(email);
        } catch (InvalidEmailException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

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

4. 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

-

Status : Skipped ***Marks : 0/10***