Rajalakshmi Engineering College

Name: Kiruthick SM
Email: 240701263@rajalakshmi.edu.in
Roll no: 240701263
Phone: 8122868914
Branch: REC
Department: CSE - Section 4
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 7.5

Section 1 : Coding

1. Problem Statement

Write a program to validate the email address and display suitable exceptions if there is any mistake.

Create 3 custom exception classes as below

DotExceptionAtTheRateExceptionDomainException

A typical email address should have a " . " character, and a "@" character, and also the domain name should be valid. Valid domain names for practice be 'in', 'com', 'net', or 'biz'.

Display Invalid Dot usage, Invalid @ usage, or Invalid Domain message based on email id.

Get the email address from the user, validate the email by checking the above-mentioned criteria, and print the validity status of the input email address.

Input Format

The first line of input contains the email to be validated.

Output Format

The output prints a Valid email address or an Invalid email address along with the suitable exception

If email ends with . or contains not exactly one . after @, it throws:

DotException: Invalid Dot usage

Invalid email address

If @ appears not exactly once, it throws:

AtTheRateException: Invalid @ usage

Invalid email address

If the part after the last dot is not among accepted domains:

DomainException: Invalid Domain

Invalid email address

If all conditions satisfied then print:

Valid email address

Refer to the sample input and output for format specifications.

Sample Test Case

Input: sample@gmail.com
Output: Valid email address

Answer

```
import java.util.Scanner;

class DotException extends Exception {
   DotException(String msg) { super(msg); }
}

class AtTheRateException extends Exception {
   AtTheRateException(String msg) { super(msg); }
}

class DomainException extends Exception {
   DomainException(String msg) { super(msg); }
}

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

      try {
         validateEmail(email);
         System.out.println("Valid email address");
      } catch (DotException | AtTheRateException | DomainException e) {
         System.out.println(e.getClass().getSimpleName() + ": " + e.getMessage());
         System.out.println("Invalid email address");
      }
   }

   static void validateEmail(String email) throws DotException,
AtTheRateException, DomainException {
      // Basic checks
      if (email.length() < 5 || email.length() > 50)
```

```
            throw new DotException("Invalid Dot usage");

        if (email.startsWith(".") || email.endsWith(".") || email.startsWith("@") ||
email.endsWith("@"))
            throw new DotException("Invalid Dot usage");

        if (email.contains("..") || email.contains("@@"))
            throw new DotException("Invalid Dot usage");

        long atCount = email.chars().filter(c -> c == '@').count();
        if (atCount != 1)
            throw new AtTheRateException("Invalid @ usage");

        String[] parts = email.split("@");
        if (parts.length != 2 || parts[1].isEmpty())
            throw new AtTheRateException("Invalid @ usage");

        String domainPart = parts[1];
        int dotIndex = domainPart.indexOf('.');
        int lastDotIndex = domainPart.lastIndexOf('.');

        if (dotIndex == -1 || dotIndex != lastDotIndex || lastDotIndex ==
domainPart.length() - 1)
            throw new DotException("Invalid Dot usage");

        String domain = domainPart.substring(lastDotIndex + 1);
        if (!domain.equals("in") && !domain.equals("com") && !domain.equals("net")
&& !domain.equals("biz"))
            throw new DomainException("Invalid Domain");
    }
}
```

Status : *Partially correct* ***Marks : 7.5/10***