Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 8_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 14

Section 1 : MCQ

1. What will be the output for the following code?

```
class InvalidUsernameException extends Exception {
    public InvalidUsernameException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            String username = "abc";
            if (username.length() < 5) {
                throw new InvalidUsernameException("Username must be at
least 5 characters long");
            }
```

```
        } catch (InvalidUsernameException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Username must be at least 5 characters long

Status : Correct ***Marks : 1/1***

2. What will be the output for the following code?

```
import java.io.*;

class UnderageException extends Exception {
    public UnderageException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            int age = 17;
            if (age < 18) {
                throw new UnderageException("Underage, cannot proceed");
            }
        } catch (UnderageException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Underage, cannot proceed

Status : Correct ***Marks : 1/1***

3. Which keyword is used to explicitly throw a custom exception?

Answer

throw

Status : Correct ***Marks : 1/1***

4. what is the output of the following code?

```
class MyException extends Exception {
    public MyException(String message) {
        super(message);
    }
}

class Test {
    static void check() throws MyException {
        throw new MyException("Custom Exception Occurred");
    }

    public static void main(String[] args) {
        try {
            check();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Custom Exception Occurred

Status : Correct ***Marks : 1/1***

5. What will be the output of the following code?

```
class MyException extends Exception {
    public MyException() {
```

```
        super("Default Exception Message");
    }
}

class Test {
    public static void main(String[] args) {
        try {
            throw new MyException();
        } catch (MyException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Default Exception Message

Status : Correct ***Marks : 1/1***

6. What will be the output for the following code?

```
class NegativeBalanceException extends Exception {
    public NegativeBalanceException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            double balance = -500;
            if (balance < 0) {
                throw new NegativeBalanceException("Balance cannot be
negative");
            }
        } catch (NegativeBalanceException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
```

```
}
```

Answer

Error: Balance cannot be negative

Status : Correct ***Marks : 1/1***

7. what is the output of the following code?

```
class MyException extends Exception {
    public MyException(String message) {
        super(message);
    }
}

class Test {
    public static void main(String[] args) {
        try {
            throw new MyException("Error occurred");
        } catch (MyException e) {
            System.out.println(e);
        }
    }
}
```

Answer

MyException: Error occurred

Status : Correct ***Marks : 1/1***

8. What is the purpose of a custom exception in Java?

Answer

To create user-defined exceptions for specific scenarios

Status : Correct ***Marks : 1/1***

9. What will be the output for the following code?

```
class InvalidVotingAgeException extends Exception {
   public InvalidVotingAgeException(String message) {
      super(message);
   }
}

class Test {
   public static void main(String[] args) {
      try {
         int age = 15;
         if (age < 18) {
            throw new InvalidVotingAgeException("You are not eligible to
vote");
         }
         System.out.println("Eligible to vote");
      } catch (InvalidVotingAgeException e) {
         System.out.println(e.getMessage());
      }
   }
}
```

Answer

You are not eligible to vote

Status : Correct ***Marks : 1/1***

10. What will be the output for the following code?

```
import java.io.*;

class OutOfStockException extends Exception {
   public OutOfStockException(String message) {
      super(message);
   }
}

class Test {
   public static void main(String[] args) {
```

```
        try {
            int stock = 0;
            if (stock == 0) {
                throw new OutOfStockException("Item is out of stock");
            }
        } catch (OutOfStockException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Answer

Item is out of stock

Status : Correct ***Marks : 1/1***

11. How do you create an unchecked custom exception?

Answer

By extending RuntimeException

Status : Correct ***Marks : 1/1***

12. Which of the following is true about custom exceptions?

Answer

Custom exceptions must extend either Exception or RuntimeException

Status : Correct ***Marks : 1/1***

13. What will be the output for the following code?

```
import java.io.*;

class NegativeAgeException extends Exception {
    public NegativeAgeException(String message) {
        super(message);
    }
```

```
}

class Test {
   public static void main(String[] args) {
      try {
         int age = -5;
         if (age < 0) {
            throw new NegativeAgeException("Age cannot be negative");
         }
      } catch (NegativeAgeException e) {
         System.out.println(e.getMessage());
      }
   }
}
```

Answer

Age cannot be negative

Status : Correct ***Marks : 1/1***

14. What will happen if a checked custom exception is thrown inside a method without being caught or declared?

Answer

Compilation Error

Status : Correct ***Marks : 1/1***

15. What will be the output for the following code?

```
import java.io.*;

class TemperatureTooHighException extends Exception {
   public TemperatureTooHighException(String message) {
      super(message);
   }
}
```

```
class Test {
   public static void main(String[] args) {
      try {
         int temperature = 110;
         if (temperature > 100) {
            throw new TemperatureTooHighException("Temperature too
high");
         }
      } catch (TemperatureTooHighException e) {
         System.out.println(e.getMessage());
      }
   }
}
```

Answer

110

Status : Wrong ***Marks : 0/1***

Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

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 : 10

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;

// Custom Exception Classes
class DotException extends Exception {
    public DotException(String message) {
        super(message);
    }
}

class AtTheRateException extends Exception {
    public AtTheRateException(String message) {
        super(message);
    }
}

class DomainException extends Exception {
    public DomainException(String message) {
        super(message);
    }
}

class EmailValidator {

    public static void validateEmail(String email) throws DotException,
AtTheRateException, DomainException {
        // Check if email starts or ends with '.' or '@'
        if (email.startsWith(".") || email.startsWith("@") || email.endsWith(".") ||
email.endsWith("@")) {
            throw new DotException("Invalid Dot usage");
        }

        // Check for exactly one '@'
```

```
        int atCount = email.length() - email.replace("@", "").length();
        if (atCount != 1) {
            throw new AtTheRateException("Invalid @ usage");
        }

        // Split email into local part and domain part
        String[] parts = email.split("@");
        String localPart = parts[0];
        String domainPart = parts[1];

        // Check for consecutive '.' or '@'
        if (localPart.contains("..") || domainPart.contains("..") ||
domainPart.contains("@")) {
            throw new DotException("Invalid Dot usage");
        }

        // Check for at least one '.' in the domain part
        int dotCount = domainPart.length() - domainPart.replace(".", "").length();
        if (dotCount < 1 || domainPart.indexOf('.') < domainPart.indexOf('@')) {
            throw new DotException("Invalid Dot usage");
        }

        // Check the domain extension
        String[] domainParts = domainPart.split("\\.");
        String domainExtension = domainParts[domainParts.length - 1];
        if (!domainExtension.equals("in") && !domainExtension.equals("com") &&
            !domainExtension.equals("net") && !domainExtension.equals("biz")) {
            throw new DomainException("Invalid Domain");
        }
    }

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

        String email = scanner.nextLine();

        try {
            validateEmail(email);
            System.out.println("Valid email address");
        } catch (DotException e) {
            System.out.println("DotException: " + e.getMessage());
            System.out.println("Invalid email address");
```

```
        } catch (AtTheRateException e) {
            System.out.println("AtTheRateException: " + e.getMessage());
            System.out.println("Invalid email address");
        } catch (DomainException e) {
            System.out.println("DomainException: " + e.getMessage());
            System.out.println("Invalid email address");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Elsa, a busy professional, is using a scheduling application to plan her meetings efficiently. The application requires users to input meeting durations in minutes, ensuring that the duration is a positive integer and does not exceed 240 minutes (4 hours). Elsa needs a program to assist her in scheduling meetings securely with proper exception handling.

Create a Java class named ElsaMeetingScheduler.Implement a custom exception:InvalidDurationException for invalid meeting duration entries.Implement the main method to interactively take user input for a meeting duration.Implement the validateMeetingDuration method to validate the meeting duration based on the specified rules and throw a custom exception if the validation fails.Print appropriate success or error messages based on the meeting duration.

Implement a custom exception, InvalidDurationException, to handle cases where the entered meeting duration does not meet the specified criteria.

Input Format

The input consists of an integer value 'n', representing the meeting duration.

Output Format

The output is displayed in the following format:

If the entered meeting duration meets the specified criteria, the program outputs

"Meeting scheduled successfully!"

If the entered meeting duration is invalid, the program outputs an error message indicating the issue.

"Error: Invalid meeting duration. Please enter a positive integer not exceeding 240 minutes (4 hours)."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 120
Output: Meeting scheduled successfully!

Answer

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

// Custom Exception Class
class InvalidDurationException extends Exception {
    public InvalidDurationException(String message) {
        super(message);
    }
}

 class ElsaMeetingScheduler {
```

```
    // Method to validate meeting duration
    public static void validateMeetingDuration(int duration) throws
InvalidDurationException {
        if (duration <= 0 || duration > 240) {
            throw new InvalidDurationException("Invalid meeting duration. Please
enter a positive integer not exceeding 240 minutes (4 hours).");
        }
    }

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


        try {
            int duration = scanner.nextInt();
            // Validate the meeting duration
            validateMeetingDuration(duration);
            System.out.println("Meeting scheduled successfully!");
        } catch (InvalidDurationException e) {
            System.out.println("Error: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("Error: Invalid input. Please enter a valid integer.");
        } finally {
            scanner.close();
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a user registration system, there is a requirement to implement a username validation module. Users attempting to register must adhere to specific criteria for their usernames to be considered valid.

Your task is to develop a program that takes user input for a desired username and validates it according to the following rules:

The username must not contain any spaces.The username must be at least 5 characters long.

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

Input Format

The input consists of a string S, representing the desired username.

Output Format

If the username is valid, print "Username is valid: [S]".

If the username is invalid:

1. If the username is short, print "Invalid Username: Username must be at least 5 characters long"
2. If the username contains spaces, print "Invalid Username: Username cannot contain spaces"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: John
Output: Invalid Username: Username must be at least 5 characters long

Answer

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

// Custom Exception Class
class InvalidUsernameException extends Exception {
    public InvalidUsernameException(String message) {
        super(message);
    }
}

class UsernameValidator {

    // Method to validate the username
    public static void validateUsername(String username) throws
InvalidUsernameException {
        // Check for spaces
        if (username.contains(" ")) {
            throw new InvalidUsernameException("Invalid Username: Username
cannot contain spaces");
        }
```

```
        // Check for minimum length
        if (username.length() < 5) {
            throw new InvalidUsernameException("Invalid Username: Username must
be at least 5 characters long");
        }
    }

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


        String username = scanner.nextLine();

        try {
            // Validate the username
            validateUsername(username);
            System.out.println("Username is valid: " + username);
        } catch (InvalidUsernameException e) {
            System.out.println(e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 7.5

Section 1 : Coding

1. Problem Statement

A local municipality is implementing an online voting system for a community event and wants to ensure that only eligible voters (those aged 18 or older) can participate.

Your task is to develop a program that validates the age of individuals attempting to vote online. If the user's age is below 18, the program should throw a custom exception, InvalidAgeException, preventing them from casting their vote. If the input is invalid, catch the appropriate InputMismatchException and print the in-built exception message.

Input Format

The input consists of an integer representing the age.

Output Format

If the age is 18 or older, print "Eligible to vote"

If the age is below 18, print "Exception occurred: InvalidAgeException: Age is not valid to vote"

If there is any other type of exception, print "An error occurred: " followed by the in-built exception message.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 20

Output: Eligible to vote

Answer

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

// Custom Exception Class
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}

class VotingEligibilityValidator {

    // Method to validate the age
    public static void validateAge(int age) throws InvalidAgeException {
        if (age < 18) {
            throw new InvalidAgeException("Age is not valid to vote");
        }
    }

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

```
        try {
            int age = scanner.nextInt();
            // Validate the age
            validateAge(age);
            System.out.println("Eligible to vote");
        } catch (InvalidAgeException e) {
            System.out.println("Exception occurred: InvalidAgeException: " +
e.getMessage());
        } catch (InputMismatchException e) {
            System.out.println("An error occurred: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("An error occurred: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Mukesh Raghavendhar N
Email: 241501120@rajalakshmi.edu.in
Roll no:
Phone: 9342823404
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 8_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a file management system, users are required to provide a valid file name when creating new files. The system enforces specific rules for file names to maintain consistency and avoid potential issues. Your task is to implement a Java program named FileNameValidator that takes user input for a file name and validates it according to the specified rules.

Rules for Valid File Name:

The file name must consist of alphanumeric characters (letters and digits) only.The file name must have a minimum length of 3 characters.

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

Input Format

The input consists of a string S, representing the desired filename.

Output Format

The output is displayed in the following format:

If the entered file name meets the specified criteria, the program outputs

"Valid file name"

If the entered file name does not meet the criteria and triggers the InvalidFileNameException, the program outputs

"Error: Invalid file name. It must be alphanumeric and have a minimum length of 3 characters."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: myfile123
Output: Valid file name

Answer

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

// Custom Exception Class
class InvalidFileNameException extends Exception {
    public InvalidFileNameException(String message) {
        super(message);
    }
}

class FileNameValidator {

    // Method to validate the file name
    public static void validateFileName(String fileName) throws
InvalidFileNameException {
        // Check for minimum length
        if (fileName.length() < 3) {
```

```
            throw new InvalidFileNameException("Invalid file name. It must be
alphanumeric and have a minimum length of 3 characters.");
        }
        // Check for alphanumeric characters only
        if (!fileName.matches("[a-zA-Z0-9]+")) {
            throw new InvalidFileNameException("Invalid file name. It must be
alphanumeric and have a minimum length of 3 characters.");
        }
    }

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


        String fileName = scanner.nextLine();

        try {
            // Validate the file name
            validateFileName(fileName);
            System.out.println("Valid file name");
        } catch (InvalidFileNameException e) {
            System.out.println("Error: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("An unexpected error occurred: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

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