

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sabrina is working on a project that involves analyzing a set of numbers. In her exploration, she encounters scenarios where extracting even numbers and finding their sum is essential.

Create a program that calculates the sum of even numbers from a given array of integers using a lambda expression.

Input Format

The first line of input consists of an integer N, representing the size of the array.

The second line consists of N space-separated integers, representing the elements of the array.

Output Format

The output prints the sum of the even integers from the array.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3

29 37 45

Output: 0

Answer

```
import java.util.*;
import java.util.stream.*;

public class Main

{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) arr[i] = sc.nextInt();
        int sum = Arrays.stream(arr).filter(x -> x % 2 == 0).sum();
        System.out.println(sum);

    }
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Alex is learning about Java's functional interfaces and lambda expressions.

He wants to write a simple program that prints the square of each number in an array using a predefined functional interface.

Help Alex complete this task using the Consumer functional interface.

Input Format

- The first line contains an integer N, the number of elements in the array.
- The second line contains N space-separated integers.

Output Format

- Print the squares of all elements in the array, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4

1 2 3 4

Output: 1 4 9 16

Answer

```
import java.util.*;
import java.util.function.*;

public class Main

{

    public static void main(String[] args)

    {

        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) arr[i] = sc.nextInt();
        Consumer<Integer> square = x -> System.out.print((x * x) + " ");
        for (int num : arr) square.accept(num);

    }
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In the mystical realm of programming, there exists a magical incantation to reveal hidden words.

Elara, the skilled enchantress, wishes to summon a word using her spell and then reverse its characters to uncover its enchanted reflection.

Write a program that uses the predefined functional interface Supplier<String> and a lambda expression to:

Supply (generate) a string, and

Display its reversed form.

Input Format

No input is required from the user.

The string must be supplied internally using a Supplier<String>.

Output Format

Print the reversed version of the supplied string.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Wizard!!

Output: !!draziW

Answer

```
import java.util.*;  
import java.util.function.*;
```

```
public class Main
```

```
{
```

```
    public static void main(String[] args)
```

```
{
```

```
    Scanner sc = new Scanner(System.in);  
    String input = sc.nextLine();  
    Supplier<String> supplier = () -> new StringBuilder(input).reverse().toString();  
    System.out.println(supplier.get());
```

```
}
```

```
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 12_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Abi is working on a text analysis project where she needs to categorize words based on their length.

Words that have three or fewer characters are considered “Short”, while words with more than three characters are classified as “Long.”

Write a Java program that takes a sentence as input, analyzes each word, and prints a list showing whether each word is “Short” or “Long.”

Use the predefined functional interface Function<String, String> along with a lambda expression for categorization.

Input Format

A single line containing a sentence (words separated by spaces).

Output Format

- A single line with each word categorized as "Short" or "Long", separated by spaces.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: I love my cat

Output: Short Long Short Short

Answer

```
import java.util.*;
import java.util.function.*;

public class Main

{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        String sentence = sc.nextLine();
        String[] words = sentence.split(" ");
        Function<String, String> categorize = w -> w.length() <= 3 ? "Short" : "Long";
        for (String word : words)

        {
            System.out.print(categorize.apply(word) + " ");
        }
    }
}
```

```
    }  
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

REC_Week 12_Java_Lambda Expressions_PAH

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : COD

1. Problem Statement

Sneha is developing a feature for an e-commerce application that helps display product details after applying a seasonal discount.

She decides to use lambda expressions with the Consumer functional interface to print each product's name, original price, and discounted price neatly.

The program should:

Accept a list of product names and their prices. Apply a 15% discount on all products. Use a Consumer lambda expression to display the details in a formatted manner.

Input Format

The first line of input consists of an integer n , representing the number of products.

The next n lines each contain a String (product name) and a double (price) separated by a space.

Output Format

For each product, print the details in the format:

Product: <name>, Original Price: <price>, Discounted Price: <discounted price>

If there are no products, print:

No products available

Sample Test Case

Input: 1

Phone 60000

Output: Product: Phone, Original Price: 60000.0, Discounted Price: 51000.0

Answer

```
// You are using Java
import java.util.*;
import java.util.function.*;

class Product
{
    String name;
    double price;
    Product(String name, double price)

    {
        this.name = name;
        this.price = price;
    }
}
```

```
}

public class Main

{

    public static void main(String[] args)

    {

        Scanner sc = new Scanner(System.in);
        if (!sc.hasNextInt())

        {

            System.out.println("No products available");
            return;

        }

        int n = sc.nextInt();
        sc.nextLine();
        if (n <= 0)

        {

            System.out.println("No products available");
            return;

        }

        List<Product> products = new ArrayList<>();
        for (int i = 0; i < n; i++)

        {
```

```
String line = sc.nextLine().trim();
while (line.isEmpty() && sc.hasNextLine()) line = sc.nextLine().trim();
int lastSpace = line.lastIndexOf(' ');
if (lastSpace == -1)

{
    products.add(new Product(line, 0.0));

} else
{
    String name = line.substring(0, lastSpace);
    double price = Double.parseDouble(line.substring(lastSpace + 1));
    products.add(new Product(name, price));
}

}
Consumer<Product> display = p ->
{
    double discounted = p.price * 0.85;
    System.out.println("Product: " + p.name + ", Original Price: " +
String.format("%.1f", p.price) + ", Discounted Price: " + String.format("%.1f",
discounted));
};

products.forEach(display);
}
```

}

Status : Correct

Marks : 10/10

2. Problem Statement

Emily, an analyst at a data processing firm, is tasked with cleaning up datasets to remove duplicate values from lists of integers.

Create a Java program that allows Emily to input a series of integers, with the program then utilizing a lambda expression to efficiently remove any duplicates.

Input Format

The first line of input consists of an integer N, representing the size of the array.

The second line consists of N space-separated integers, each denoting an array element.

Output Format

The output prints the array elements after removing the duplicates inside the square bracket separated by a comma and space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 15

1 2 3 4 3 2 1 2 3 4 4 4 5 5 6

Output: [1, 2, 3, 4, 5, 6]

Answer

```
// You are using Java
import java.util.*;
import java.util.function.*;
import java.util.stream.*;
```

```
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        List<Integer> list = new ArrayList<>();
        for (int i = 0; i < n; i++) list.add(sc.nextInt());
        Function<List<Integer>, List<Integer>> removeDuplicates = l ->
        l.stream().distinct().collect(Collectors.toList());
        List<Integer> result = removeDuplicates.apply(list);
        System.out.println(result);

    }
}
```

Status : Correct

Marks : 10/10

3. Problem Statement

Rishi is working as an HR analyst in a software company. He wants to filter a list of employees based on their salary using modern Java techniques. He has a list of employee names and salaries and wants to use lambda expressions to filter those who earn more than a specific threshold.

Implement a program using lambda expressions and functional interfaces to print the names of employees whose salary is greater than or equal to 50,000.

Input Format

The first line of input consists of an integer n, representing the number of employees.

The next n lines. Each line contains a String (employee name) and an int (salary).

Output Format

The output prints the names of employees whose salary is greater than or equal to 50000, each on a new line.

If no employee found with salary greater than 50000, print: No employee found with salary >= 50000

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
Amit 45000
Sneha 50000
Ravi 60000
Priya 30000

Output: Sneha
Ravi

Answer

```
// You are using Java
import java.util.*;
import java.util.function.*;
import java.util.stream.*;

class Employee

{
```

```
    String name;
    int salary;
    Employee(String name, int salary)
```

```
{
```

```
this.name = name;
this.salary = salary;

}

}

public class Main

{

    public static void main(String[] args)

    {

        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        List<Employee> employees = new ArrayList<>();
        for (int i = 0; i < n; i++)

        {

            String name = sc.next();
            int salary = sc.nextInt();
            employees.add(new Employee(name, salary));

        }

        Predicate<Employee> highSalary = e -> e.salary >= 50000;
        List<Employee> filtered =
        employees.stream().filter(highSalary).collect(Collectors.toList());
        if (filtered.isEmpty())

        {

            System.out.println("No employee found with salary >= 50000");

        }

    }

}
```

```
        } else
    {
        filtered.forEach(e -> System.out.println(e.name));
    }
}
```

Status : Correct

Marks : 10/10

4. Problem Statement

Aditya is developing a reading app that recommends books to users based on a predefined list.

Each time a user opens the app, it should supply the next book title in the list, one at a time, using a lambda expression and the Supplier functional interface.

When all books have been recommended, the list should start again from the beginning.

Input Format

The first line contains an integer n – the total number of available book titles.

The next n lines each contain a book title (a string).

The next line contains an integer m – the number of times users open the app (i.e., the number of recommendations to be made).

Output Format

Print the supplied book title for each recommendation, one per line.

If $m > n$, repeat the list from the start.

Sample Test Case

Input: 3

The Alchemist

Atomic Habits

Ikigai

5

Output: The Alchemist

Atomic Habits

Ikigai

The Alchemist

Atomic Habits

Answer

```
import java.util.*;
import java.util.function.*;

public class Main

{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        List<String> books = new ArrayList<>();
        for (int i = 0; i < n; i++) books.add(sc.nextLine());
        int m = sc.nextInt();
        final int[] index =
        {
            0
        };
    }
}
```

```
Supplier<String> supplier = () ->
{
    String book = books.get(index[0]);
    index[0] = (index[0] + 1) % books.size();
    return book;
};

for (int i = 0; i < m; i++) System.out.println(supplier.get());
}
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Dharshini A
Email: 241001049@rajalakshmi.edu.in
Roll no: 241001049
Phone: 8056618801
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results



2024_28_III_OOPS Using Java Lab

REC_Week 12_Java_Lambda Expressions_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Nethra is a researcher working on a project that involves analyzing experimental data. As part of her analysis, she needs to determine whether a given word is a palindrome or not.

Create a Java program that allows Nethra to input a word, and then check and display whether the entered word is a palindrome. Use lambda expressions to perform the palindrome check.

Input Format

The first line of input consists of a word.

Output Format

The output prints whether the given word is a palindrome or not in the following format:

"<input> is palindrome" or "<input> is not palindrome".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: malayalam

Output: malayalam is palindrome

Answer

```
import java.util.*;
import java.util.function.*;

public class Main

{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        String word = sc.nextLine();
        Predicate<String> isPalindrome = s -> s.equalsIgnoreCase(new
StringBuilder(s).reverse().toString());
        if (isPalindrome.test(word))
            System.out.println(word + " is palindrome");
        else
            System.out.println(word + " is not palindrome");
    }
}
```

Status : Correct

Marks : 10/10

2. Problem Statement

Problem Statement

Sophia, a data analyst, is studying experimental results collected from various lab sensors. Each sensor provides a list of numeric readings, and Sophia wants to calculate the average of these readings to analyze consistency.

She decides to use lambda expressions and the Function functional interface to compute the average of all the recorded values efficiently.

Your Task

Write a Java program that:

Reads the total number of measurements. Reads all the measurement values as doubles. Uses a `Function<double[], Double>` lambda expression to calculate the average value. Displays the final average, formatted to two decimal places.

Input Format

The first line of input consists of an integer N, representing the number of measurements.

The second line contains N space-separated double values.

Output Format

Print the average of the entered values, rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6

2.2 1.2 5.4 4.6 2.9 55.7

Output: 12.00

Answer

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

```
import java.util.function.*;
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        double[] readings = new double[n];
        for (int i = 0; i < n; i++) readings[i] = sc.nextDouble();
        Function<double[], Double> average = arr ->
        {
            double sum = 0;
            for (double val : arr) sum += val;
            return sum / arr.length;
        };
        System.out.printf("%.2f", average.apply(readings));
    }
}
```

Status : Correct

Marks : 10/10

3. Problem Statement

Riya is developing a college admission system that assigns unique roll numbers to each newly admitted student.

Each roll number should follow this fixed format:

<DEPT>-<YEAR>-<4-digit-sequence>

where:

<DEPT> is the department code (in uppercase, e.g., CSE, ECE, MECH).<YEAR> is the admission year (e.g., 2025).<4-digit-sequence> starts from a given number and increases sequentially for each student. Write a Java program using a Supplier<String> lambda to generate and print the roll numbers for n students.

Input Format

First line: integer n – number of roll numbers to generate

Second line: string DEPT – department code (uppercase letters only)

Third line: integer YEAR – admission year

Fourth line: integer start – starting sequence number ($0 \leq \text{start} \leq 9999$)

Output Format

Print n roll numbers, one per line, in the required format

Sequence must be zero-padded to 4 digits

If sequence exceeds 9999, wrap around to 0000

Sample Test Case

Input: 5

CSE

2025

98

Output: CSE-2025-0098

CSE-2025-0099

CSE-2025-0100

CSE-2025-0101

CSE-2025-0102

Answer

```
import java.util.*;  
import java.util.function.*;
```

```
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String dept = sc.next();
        int year = sc.nextInt();
        int start = sc.nextInt();
        final int[] seq =
        {
            start
        };
        Supplier<String> rollSupplier = () ->
        {
            String roll = String.format("%s-%d-%04d", dept, year, seq[0]);
            seq[0] = (seq[0] + 1) % 10000;
            return roll;
        };
        for (int i = 0; i < n; i++)
        {
            System.out.println(rollSupplier.get());
        }
    }
}
```

```
}
```

```
}
```

```
}
```

Status : Correct

Marks : 10/10

4. Problem Statement

A company named TechNova is collecting feedback from its customers. Each customer gives a feedback score (an integer between 1 and 10) along with their name.

The company wants to:

Display each customer's name along with their feedback in a formatted way using a lambda expression and a Consumer functional interface. After displaying all feedbacks, calculate and display the average feedback score. You need to implement this functionality using Java lambda expressions and streams, emphasizing the Consumer interface for displaying formatted output.

Input Format

The first line of input contains an integer n, representing the number of customers.

The next n lines each contain a String (customer name) followed by an int (feedback score).

Output Format

- Each line prints a customer's name and feedback in the format:
- Customer: <name>, Feedback Score: <score>

- After all customers are displayed, print the average feedback as:

- Average Feedback: <average_value>

(Average should be displayed up to two decimal places.)

Sample Test Case

Input: 3

Ravi 7

Ananya 9

Kiran 8

Output: Customer: Ravi, Feedback Score: 7

Customer: Ananya, Feedback Score: 9

Customer: Kiran, Feedback Score: 8

Average Feedback: 8.00

Answer

```
import java.util.*;
import java.util.function.*;
import java.util.stream.*;

class Customer

{
    String name;
    int feedback;
    Customer(String name, int feedback)

    {
        this.name = name;
        this.feedback = feedback;
    }

    public class Main
```

```
{  
    public static void main(String[] args)  
  
    {  
  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        List<Customer> customers = new ArrayList<>();  
        for (int i = 0; i < n; i++)  
  
        {  
  
            String name = sc.next();  
            int feedback = sc.nextInt();  
            customers.add(new Customer(name, feedback));  
  
        }  
        Consumer<Customer> display = c -> System.out.println("Customer: " +  
c.name + ", Feedback Score: " + c.feedback);  
        customers.forEach(display);  
        double average = customers.stream().mapToInt(c ->  
c.feedback).average().orElse(0);  
        System.out.printf("Average Feedback: %.2f", average);  
  
    }  
}
```

Status : Correct

Marks : 10/10