

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

Scan to verify results



## 2024\_28\_III\_OOPS Using Java Lab

### REC\_Week 12\_Java\_Lamba Expressions\_MCQ

Attempt : 1  
Total Mark : 10  
Marks Obtained : 9

#### Section 1 : MCQ

1. What is a lambda expression in Java?

**Answer**

A way to define anonymous methods

**Status : Correct**

**Marks : 1/1**

2. Which of the following interfaces is NOT a functional interface in Java?

**Answer**

Iterable

**Status : Correct**

**Marks : 1/1**

3. Which of the following is a valid lambda expression in Java?

**Answer**

(x) -> {return x \* 2;}

**Status : Wrong**

**Marks : 0/1**

4. Which functional interface in Java takes two arguments and returns a result?

**Answer**

BiFunction

**Status : Correct**

**Marks : 1/1**

5. What is the syntax for a basic lambda expression in Java?

**Answer**

(parameters) -> expression

**Status : Correct**

**Marks : 1/1**

6. Can a lambda expression in Java have a body with multiple statements?

**Answer**

Yes, if the statements are enclosed in curly braces

**Status : Correct**

**Marks : 1/1**

7. Can a lambda expression in Java have a body with multiple statements?

**Answer**

Yes, if the statements are enclosed in curly braces

**Status : Correct**

**Marks : 1/1**

8. What is the return type of a lambda expression in Java?

**Answer**

The return type is inferred from the context

**Status : Correct**

**Marks : 1/1**

9. Can a lambda expression have more than one parameter?

**Answer**

Yes, it can have multiple parameters

**Status : Correct**

**Marks : 1/1**

10. Which functional interface is commonly used with lambda expressions in Java?

**Answer**

Runnable

**Status : Correct**

**Marks : 1/1**

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

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.Arrays;
import java.util.Scanner;
import java.util.function.Predicate;
import java.util.stream.IntStream;

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

        for (int i = 0; i < n; i++) {
            numbers[i] = scanner.nextInt();
        }

        Predicate<Integer> isEven = num -> num % 2 == 0;
        int evenSum = calculateEvenSum(numbers, isEven);

        System.out.println(evenSum);
    }

    private static int calculateEvenSum(int[] numbers, Predicate<Integer>
predicate) {
        return Arrays.stream(numbers)
            .filter(num -> predicate.test(num))
            .sum();
    }
}
```

Status : Correct

Marks : 10/10

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

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.Arrays;
import java.util.Scanner;
import java.util.function.Consumer;
class SquareArray {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        int[] numbers = new int[n];

        for (int i = 0; i < n; i++) {
            numbers[i] = scanner.nextInt();
        }

        Consumer<Integer> squareConsumer = num -> System.out.print(num * num
+ " ");
        printSquares(numbers, squareConsumer);
    }

    private static void printSquares(int[] numbers, Consumer<Integer> consumer) {
        Arrays.stream(numbers)
            .mapToObj(Integer::valueOf)
            .forEach(consumer);
    }
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

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 : 0

#### 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**

// You are using Java

```
import java.util.function.Supplier;
```

```
class StringReverse {  
    public static void main(String[] args) {  
        Supplier<String> stringSupplier = () -> "Enchanted";  
        reverseString(stringSupplier);  
    }  
  
    private static void reverseString(Supplier<String> supplier) {  
        String originalString = supplier.get();  
        String reversedString = new StringBuilder(originalString).reverse().toString();  
        System.out.println(reversedString);  
    }  
}
```

**Status : Wrong**

**Marks : 0/10**

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

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**

```
// You are using Java
import java.util.Arrays;
import java.util.Scanner;
import java.util.function.Function;
import java.util.stream.Collectors;

class WordCategorizer {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String sentence = scanner.nextLine();
        categorizeWords(sentence);
    }

    private static void categorizeWords(String sentence) {
        Function<String, String> wordCategorizer = word -> word.length() <= 3 ?
        "Short" : "Long";
        String result = Arrays.stream(sentence.split(" "))
            .map(wordCategorizer)
            .collect(Collectors.joining(" "));
        System.out.println(result);
    }
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

Scan to verify results



## 2024\_28\_III\_OOPS Using Java Lab

### REC\_Week 12\_Java\_Lamba Expressions\_PAH

Attempt : 1  
Total Mark : 40  
Marks Obtained : 30

#### Section 1 : COD

##### 1. 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  $\geq$  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.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.function.Predicate;
import java.util.stream.Collectors;
class EmployeeFilter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.nextLine(); // consume newline character

        List<Employee> employees = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            String[] input = scanner.nextLine().split(" ");
            String name = input[0];
            int salary = Integer.parseInt(input[1]);
            employees.add(new Employee(name, salary));
        }
    }
}
```

```

    }
    filterEmployees(employees);
}

private static void filterEmployees(List<Employee> employees) {
    Predicate<Employee> salaryThreshold = emp -> emp.getSalary() >= 50000;
    List<String> employeeNames = employees.stream()
        .filter(salaryThreshold)
        .map(Employee::getName)
        .collect(Collectors.toList());

    if (employeeNames.isEmpty()) {
        System.out.println("No employee found with salary >= 50000");
    } else {
        employeeNames.forEach(System.out::println);
    }
}

private static class Employee {
    private final String name;
    private final int salary;

    public Employee(String name, int salary) {
        this.name = name;
        this.salary = salary;
    }

    public String getName() {
        return name;
    }

    public int getSalary() {
        return salary;
    }
}
}

```

**Status :** Correct

**Marks :** 10/10

## 2. 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.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.function.Consumer;
```

```

class ProductDiscounter {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.nextLine(); // consume newline character

        List<Product> products = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            String[] input = scanner.nextLine().split(" ");
            String name = input[0];
            double price = Double.parseDouble(input[1]);
            products.add(new Product(name, price));
        }

        displayProductDetails(products);
    }

    private static void displayProductDetails(List<Product> products) {
        if (products.isEmpty()) {
            System.out.println("No products available");
            return;
        }

        Consumer<Product> productPrinter = product -> {
            double discountedPrice = product.getPrice() * 0.85;
            System.out.printf("Product: %s, Original Price: %.1f, Discounted Price: %.1f\n",
                               product.getName(), product.getPrice(), discountedPrice);
        };

        products.forEach(productPrinter);
    }

    private static class Product {
        private final String name;
        private final double price;

        public Product(String name, double price) {
            this.name = name;
            this.price = price;
        }
    }
}

```

```
public String getName() {  
    return name;  
}  
  
public double getPrice() {  
    return price;  
}  
}  
}
```

**Status :** Correct

**Marks :** 10/10

### 3. 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**

-

**Status :** Skipped

**Marks :** 0/10

#### 4. 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.ArrayList;
```

```
import java.util.List;
```

```
import java.util.Scanner;
```

```
import java.util.stream.Collectors;
```

```
class DuplicateRemover {
```

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

```
        Scanner scanner = new Scanner(System.in);
```

```
        int n = scanner.nextInt();
```

```
        List<Integer> numbers = new ArrayList<>();
```

```
        for (int i = 0; i < n; i++) {
```

```
            numbers.add(scanner.nextInt());
```

```
        }
```

```
        List<Integer> uniqueNumbers = removeDuplicates(numbers);
```

```
        System.out.println(uniqueNumbers);
```

```
    }
```

```
    private static List<Integer> removeDuplicates(List<Integer> numbers) {
```

```
        return numbers.stream()
```

```
            .distinct()
```

```
            .collect(Collectors.toList());
```

```
    }
```

```
}
```

Status : Correct

Marks : 10/10

# Rajalakshmi Engineering College

Name: Nivedithaa S  
Email: 241901076@rajalakshmi.edu.in  
Roll no: 241901076  
Phone: 6374185608  
Branch: REC  
Department: CSE (CS) - Section 2  
Batch: 2028  
Degree: B.E - CSE (CS)

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 : 30

#### Section 1 : Coding

##### 1. 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**

```
// You are using Java
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.function.Consumer;
class FeedbackProcessor {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        scanner.nextLine(); // consume newline character
```

```

List<Feedback> feedbacks = new ArrayList<>();
for (int i = 0; i < n; i++) {
    String[] input = scanner.nextLine().split(" ");
    String name = input[0];
    int score = Integer.parseInt(input[1]);
    feedbacks.add(new Feedback(name, score));
}

displayFeedbacks(feedbacks);
double averageFeedback = calculateAverageFeedback(feedbacks);
System.out.printf("Average Feedback: %.2f%n", averageFeedback);
}

private static void displayFeedbacks(List<Feedback> feedbacks) {
    Consumer<Feedback> feedbackConsumer = feedback -> {
        System.out.printf("Customer: %s, Feedback Score: %d%n",
feedback.getName(), feedback.getScore());
    };
    feedbacks.forEach(feedbackConsumer);
}

private static double calculateAverageFeedback(List<Feedback> feedbacks) {
    return feedbacks.stream()
        .mapToInt(Feedback::getScore)
        .average()
        .orElse(0.0);
}

private static class Feedback {
    private final String name;
    private final int score;

    public Feedback(String name, int score) {
        this.name = name;
        this.score = score;
    }

    public String getName() {
        return name;
    }
}

```

```
public int getScore() {  
    return score;  
}  
}  
}
```

**Status :** Correct

**Marks :** 10/10

## 2. 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***

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

```

import java.util.function.Predicate;

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

        Predicate<String> isPalindrome = str -> {
            int left = 0;
            int right = str.length() - 1;
            while (left < right) {
                if (str.charAt(left) != str.charAt(right)) {
                    return false;
                }
                left++;
                right--;
            }
            return true;
        };

        if (isPalindrome.test(word)) {
            System.out.printf("%s is palindrome%n", word);
        } else {
            System.out.printf("%s is not palindrome%n", word);
        }
    }
}

```

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

```
class RollNumberGenerator {
```

```
    public static void main(String[] args) {
        // Input values (for demonstration purposes)
        int n = 5; // Number of roll numbers to generate
```

```

String DEPT = "CSE"; // Department code
int YEAR = 2025; // Admission year
int start = 98; // Starting sequence number

// Supplier lambda to generate roll numbers
Supplier<String> rollNumberSupplier = new Supplier<String>() {
    private int sequence = start;

    @Override
    public String get() {
        String formattedSequence = String.format("%04d", sequence %
10000); // Wrap around and zero-pad
        sequence++; // Increment sequence
        return DEPT + "-" + YEAR + "-" + formattedSequence;
    }
};

// Generate and print roll numbers
IntStream.range(0, n).forEach(i ->
System.out.println(rollNumberSupplier.get()));
}
}

```

**Status : Wrong**

**Marks : 0/10**

#### 4. 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***

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

class SensorAverageCalculator {

    public static void main(String[] args) {
        // Create a Scanner object to read input
        Scanner scanner = new Scanner(System.in);

        // Read the number of measurements
        int N = scanner.nextInt();

        // Read the measurements into a double array
        double[] measurements = new double[N];
```

```
for (int i = 0; i < N; i++) {  
    measurements[i] = scanner.nextDouble();  
}  
  
// Define a Function lambda to calculate the average  
Function<double[], Double> averageCalculator = array -> {  
    double sum = Arrays.stream(array).sum();  
    return sum / array.length;  
};  
  
// Calculate the average using the lambda  
double average = averageCalculator.apply(measurements);  
  
// Print the average rounded to two decimal places  
System.out.printf("%.2f%n", average);  
  
// Close the scanner  
scanner.close();  
}  
}
```

**Status :** Correct

**Marks :** 10/10