Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

Section 1 : MCQ

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

```
import java.util.*;
public class Main {
   public static void main(String[] args) {
      Stack<Integer> stack = new Stack<>();
      for (int i = 1; i <= 3; i++)
         stack.push(i * 2);
      stack.pop();
      stack.push(10);
      System.out.println(stack.peek());
   }
}
```

Answer

10

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

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

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      ArrayList<Integer> list = new ArrayList<>();
      list.add(1);
      list.add(2);
      list.add(3);
      list.add(4);
      list.add(5);
      System.out.println(list.get(3));
   }
}
```

Answer

4

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

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

```
import java.util.ArrayList;

public class Main {
   public static void main(String[] args) {
      ArrayList<Integer> list = new ArrayList<>();
      list.add(10);
      list.add(20);
      list.add(30);
      System.out.println("Size of the list: " + list.size());
   }
}
```

Answer

Size of the list: 3

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

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

```
import java.util.*;
class Main {
   public static void main(String[] args) {
      ArrayList<Integer> list = new ArrayList<>();
      list.add(10);
      list.add(20);
      list.add(30);
      list.remove(1);
      System.out.println(list);
   }
}
```

Answer

[10, 30]

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

5. How can you access the first element of an ArrayList named as list?

Answer

list.get(0);

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

6. What is Collection in Java?

Answer

A group of objects

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

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

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
        list.add("Java");
        list.add("Python");
        list.add("Java");
        list.add("C++");
        System.out.println(list.indexOf("Java"));
    }
}
```

Answer

0

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

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

```
import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
        list.add("Apple");
        list.add("Banana");
        list.remove("Apple");
        System.out.println(list);

    }
}
```

Answer

[Banana]

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

9. What is the correct way to create an ArrayList in Java?

Answer

ArrayList<String> list = new ArrayList<>();

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

10. Which method is used to add an element to the top of the stack?

Answer

push()

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

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

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
        list.add("apple");
        list.add("banana");
        list.add("cherry");
        list.add("banana");
        System.out.println(list.lastIndexOf("banana"));
    }
}
```

Answer

3

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

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

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Stack<Integer> s = new Stack<>();
```

```
        s.push(10);
        s.push(20);
        s.push(30);
        System.out.println(s.peek());
    }
}
```

Answer

30

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

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

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
        list.add(4);
        list.set(2, 10);
        System.out.println(list);
    }
}
```

Answer

[1, 2, 10, 4]

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

14. What does the addFirst() method of LinkedList do?

Answer

Adds an element to the beginning of the list

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

15. Which of the following methods removes and returns the last element from a LinkedList?

Answer

removeLast()

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

Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
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 9_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bobby is tasked with processing a sequence of numbers from a monitoring system. He needs to extract a strictly increasing subsequence using an ArrayList. The program should dynamically add numbers to the ArrayList only if they are greater than the last number currently stored in the list. Bobby aims to efficiently utilize the dynamic resizing and indexing features of the ArrayList to solve this problem.

Help Bobby implement this solution.

Input Format

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

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

Output Format

The output prints the list of integers in increasing sequence, ignoring out-of-order elements.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 7
3 5 9 1 11 7 13
Output: [3, 5, 9, 11, 13]

Answer

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

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

      int N = scanner.nextInt();
      ArrayList<Integer> list = new ArrayList<>();

      for (int i = 0; i < N; i++)
      {
         int val = scanner.nextInt();
         if (list.isEmpty() || val > list.get(list.size() - 1))
         {
            list.add(val);
         }
      }
      scanner.close();

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

```
    }
}
```

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

Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
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 9_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Vikram loves listening to music and wants to create a simple playlist manager using Java Collections. The playlist supports the following operations:

"ADD <song>" Adds the song to the end of the playlist."REMOVE <song>" Removes the first occurrence of the song from the playlist. If the song is not found, do nothing."SHOW" Displays all songs in the playlist in order. If the playlist is empty, print "EMPTY"."NEXT" Moves to the next song in the playlist and prints its name. If the playlist is empty, print "EMPTY".

The playlist maintains a "current song" position that starts at the first song when it's added. The NEXT command moves to the next song and prints it, wrapping around to the first song after reaching the last song. When removing songs, the current position adjusts accordingly to maintain

proper navigation.

Help Vikram implement this playlist manager.

Input Format

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

The next n lines, each containing a command:

- "ADD <song>"
- "REMOVE <song>"
- "SHOW"
- "NEXT"

Output Format

For each "SHOW" command, print the songs in order, separated by spaces.

For each "NEXT" command, print the next song in the playlist.

If no song exists, print "EMPTY".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
ADD song1
ADD song2
SHOW
NEXT
REMOVE song2
SHOW
NEXT

Output: song1 song2
song2
song1
song1

Answer

```
import java.util.*;

class PlaylistManager {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        LinkedList<String> playlist = new LinkedList<>();
        int currentIndex = -1; // No current song initially
        sc.nextLine(); // Consume newline after reading integer

        for (int i = 0; i < n; i++) {
            String line = sc.nextLine().trim();
            String[] parts = line.split(" ", 2);
            String command = parts[0];

            switch (command) {
                case "ADD":
                    if (parts.length == 2) {
                        String song = parts[1];
                        playlist.add(song);
                        if (currentIndex == -1) currentIndex = 0;
                    }
                    break;

                case "REMOVE":
                    if (parts.length == 2) {
                        String songToRemove = parts[1];
                        int indexToRemove = playlist.indexOf(songToRemove);
                        if (indexToRemove != -1) {
                            playlist.remove(indexToRemove);

                            // Adjust current index
                            if (playlist.isEmpty()) {
                                currentIndex = -1;
                            } else if (indexToRemove < currentIndex) {
                                currentIndex--;
                            } else if (indexToRemove == currentIndex) {
                                if (currentIndex >= playlist.size()) {
                                    currentIndex = 0; // wrap to start
                                }
                            }
                        }
```

```
            }
            break;

        case "SHOW":
            if (playlist.isEmpty()) {
                System.out.println("EMPTY");
            } else {
                System.out.println(String.join(" ", playlist));
            }
            break;

        case "NEXT":
            if (playlist.isEmpty()) {
                System.out.println("EMPTY");
            } else {
                currentIndex = (currentIndex + 1) % playlist.size();
                System.out.println(playlist.get(currentIndex));
            }
            break;

        case "ISEMPTY":
            System.out.println(playlist.isEmpty() ? "TRUE" : "FALSE");
            break;

        default:
            // Ignore unknown commands
            break;
      }
    }

    sc.close();
  }
}
```

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

Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
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 9_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Assist Pranitha in developing a program that takes an integer N as input, representing the number of names to be read. Then read N names and store them in an ArrayList. Finally, input a search string and output the frequency of that string in the list of names.

Note: Some parts of the code are provided as snippets, and you need to complete the remaining sections by writing the necessary code.

Input Format

The first line of input consists of an integer N, representing the number of names to be read.

The following N lines consist of N names, as a string.

The last line consists of a string, representing the name to be searched.

Output Format

The output prints a single integer, representing the frequency of the specified name in the given list.

If the specified name is not found, print 0.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
Alice
Bob
Ankit
Alice
Pranitha
Alice

Output: 2

Answer

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

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

      int N = scanner.nextInt();
      scanner.nextLine();

      ArrayList<String> names = new ArrayList<>();

      for (int i = 0; i < N; i++)
      {
         String name = scanner.nextLine();
```

```
            names.add(name);
        }

        String query = scanner.nextLine();
        scanner.close();

        int count = 0;
        for (String nm : names)
        {
            if (nm.equals(query))
            {
                count++;
            }
        }

        System.out.println(count);
    }
}
```

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

Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

A teacher is filtering a list of words provided by students. Some words contain too many vowels, making them difficult for a spelling competition. The teacher decides to remove all words that contain more than two vowels.

Help the teacher to implement it using ArrayList.

Input Format

The first line contains an integer N, representing the number of words in the list.

The next N lines contain a string representing the words (one per line).

Output Format

The output consists of words that contain two or less than two vowels, printed in the same order they appeared in the input. Each word is printed on a new line.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1
sri

Output: sri

Answer

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

// You are using Java

import java.util.ArrayList;
import java.util.Scanner;

class VowelFilter {
    public static void filterWords(int n, Scanner sc) {
        ArrayList<String> filteredWords = new ArrayList<>();

        for (int i = 0; i < n; i++) {
            String word = sc.nextLine();
            if (countVowels(word) <= 2) {
                filteredWords.add(word);
            }
        }

        for (String word : filteredWords) {
            System.out.println(word);
        }
    }

    private static int countVowels(String word) {
        int count = 0;
        for (char ch : word.toCharArray()) {
            if ("aeiou".indexOf(ch) != -1) {
```

```
                count++;
            }
        }
        return count;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        VowelFilter.filterWords(n, sc);
        sc.close();
    }
}
```

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

2. Problem Statement

Aarav is developing a music playlist application where users can manage their favorite songs. He wants to implement a feature that allows users to reorder the playlist by moving a song from one position to another.

You need to implement a function that performs the following operations using a LinkedList:

Add songs to the playlist in the given order.Move a song from a specified position to another position in the playlist.Print the final playlist after all operations.

Input Format

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

The next n lines, each containing a string representing a song name.

After the songs are given the next line contains an integer m, the number of move operations.

The next m lines, each containing two integers x and y representing the move operation where the song at position x (0-based index) should be moved to position y.

Output Format

The output prints the final playlist, each song on a new line.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
SongA
SongB
SongC
SongD
SongE
2
2 4
0 3
Output: SongB
SongD
SongE
SongA
SongC

Answer

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

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

      // Read number of songs
      int n = sc.nextInt();
      sc.nextLine(); // consume newline
```

```
        LinkedList<String> playlist = new LinkedList<>();

        // Add songs to playlist
        for (int i = 0; i < n; i++) {
            playlist.add(sc.nextLine());
        }

        // Read number of move operations
        int m = sc.nextInt();

        // Perform move operations
        for (int i = 0; i < m; i++) {
            int x = sc.nextInt(); // current position
            int y = sc.nextInt(); // new position

            String song = playlist.remove(x); // remove song from old position
            playlist.add(y, song);          // add it to new position
        }

        // Print final playlist
        for (String song : playlist) {
            System.out.println(song);
        }

        sc.close();
    }
}
```

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

3. Problem Statement

Rahul, a stock trader, wants to analyze the stock prices of a company over several days. For each day, he wants to determine the stock span, which is the number of consecutive days (including the current day) where the stock price is less than or equal to the price on that day.

The stock span helps him understand how long a stock has been continuously increasing or staying the same. You need to help Rahul by

computing the stock span for each day using a Stack data structure efficiently.

Example:

Input:

7

100 80 60 70 60 75 85

Output:

1 1 1 2 1 4 6

Explanation:

For each day:

Day 1: Price = 100 Span = 1 (Only this day)Day 2: Price = 80 Span = 1 (Only this day)Day 3: Price = 60 Span = 1 (Only this day)Day 4: Price = 70 Span = 2 (Includes today and previous day)Day 5: Price = 60 Span = 1 (Only this day)Day 6: Price = 75 Span = 4 (Includes today and previous three days)Day 7: Price = 85 Span = 6 (Includes today and previous five days)

Input Format

The first line contains an integer n, the number of days.

The second line contains n space-separated integers prices[i], where prices[i] represents the stock price on the i-th day.

Output Format

The output prints n space-separated integers representing the stock span for each day.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
100 80 60 70 60 75 85

Output: 1 1 1 2 1 4 6

Answer

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

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

        // Read number of days
        int n = sc.nextInt();

        // Read stock prices
        int[] prices = new int[n];
        for (int i = 0; i < n; i++) {
            prices[i] = sc.nextInt();
        }

        // Array to store spans
        int[] span = new int[n];

        Stack<Integer> stack = new Stack<>();

        // First day span is always 1
        span[0] = 1;
        stack.push(0);

        // Calculate span for remaining days
        for (int i = 1; i < n; i++) {
            // Pop elements from stack while current price >= price at top of stack
            while (!stack.isEmpty() && prices[i] >= prices[stack.peek()]) {
                stack.pop();
            }

            // If stack is empty, all previous prices are smaller
            if (stack.isEmpty()) {
                span[i] = i + 1;
            } else {
                span[i] = i - stack.peek();
            }
```

```
            // Push current index to stack
            stack.push(i);
        }

        // Print the span array
        for (int i = 0; i < n; i++) {
            System.out.print(span[i] + " ");
        }

        sc.close();
    }
}
```

Status : Correct *Marks : 10/10*

4. Problem Statement

Mesa, a store manager, needs a program to manage inventory items. Define a class ItemType with private attributes for name, deposit, and cost per day. Create an ArrayList in the Main class to store ItemType objects, allowing input and display.

Note: Use "%-20s%-20s%-20s" for formatting output in tabular format, display double values with 1 decimal place.

Input Format

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

For each of the n items, there are three lines:

1. The name of the item (a string)
2. The deposit amount (a double value)
3. The cost per day (a double value)

Output Format

The output prints a formatted table with columns for name, deposit and cost per day.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
Laptop
10000.0
250.0
Light
1000.0
50.0
Fan
1000.0
100.0
Output: Name Deposit Cost Per Day
Laptop 10000.0 250.0
Light 1000.0 50.0
Fan 1000.0 100.0

Answer

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

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

class ItemType {
  class ItemType {
   private String name;
   private double deposit;
   private double costPerDay;

   public ItemType(String name, double deposit, double costPerDay) {
      this.name = name;
      this.deposit = deposit;
      this.costPerDay = costPerDay;
   }

   public String getName() {
```

```
        return name;
    }

    public double getDeposit() {
        return deposit;
    }

    public double getCostPerDay() {
        return costPerDay;
    }

    @Override
    public String toString() {
        return String.format("%-20s%-20.1f%-20.1f", name, deposit, costPerDay);
    }
}

}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<ItemType> items = new ArrayList<>();

        int n = Integer.parseInt(sc.nextLine());
        for (int i = 0; i < n; i++) {
            String name = sc.nextLine();
            double deposit = Double.parseDouble(sc.nextLine());
            double costPerDay = Double.parseDouble(sc.nextLine());
            items.add(new ItemType(name, deposit, costPerDay));
        }

        System.out.printf("%-20s%-20s%-20s\n", "Name", "Deposit", "Cost Per Day");
        for (ItemType item : items) {
            System.out.printf("%-20s%-20.1f%-20.1f\n", item.getName(),
item.getDeposit(), item.getCostPerDay());
        }
    }
}
class ArrayListObjectMain {
    public static void main(String args[]) {
        List<ItemType> items = new ArrayList<>();
```

```
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());

        for (int i = 0; i < n; i++) {
            String name = sc.nextLine();
            Double deposit = Double.parseDouble(sc.nextLine());
            Double costPerDay = Double.parseDouble(sc.nextLine());
            items.add(new ItemType(name, deposit, costPerDay));
        }
        System.out.format("%-20s%-20s%-20s", "Name", "Deposit", "Cost Per Day");
        System.out.println();

        for (ItemType item : items) {
            System.out.println(item);
        }
    }
}
```

Status : Wrong **Marks : 0/10**

Rajalakshmi Engineering College

Name: Reshma shri.s
Email: 241001194@rajalakshmi.edu.in
Roll no: 241001194
Phone: 8144945959
Branch: REC
Department: IT - Section 2
Batch: 2028
Degree: B.E - IT

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_PAH

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Arun is building a task manager to keep track of tasks using a LinkedList. The task manager supports the following operations:

"ADD <task>" Adds the given task to the end of the list."REMOVE" Removes the first task from the list."SHOW" Displays all tasks in the list in order. If the list is empty, print "EMPTY".

Help Arun implement this functionality using a LinkedList.

Input Format

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

The next n lines, each containing a command:

- "ADD <task>"
- "REMOVE"
- "SHOW"

Output Format

For each "SHOW" command, the output prints the tasks in order, separated by spaces.

If no tasks exist, print "EMPTY".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
ADD homework
ADD project
SHOW
REMOVE
SHOW
Output: homework project
project

Answer

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

public class Main
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      LinkedList<String> tasks = new LinkedList<>();

      for (int i = 0; i < n; i++)
      {
         String command = sc.next();

         if (command.equals("ADD"))
```

```
        {
            String task = sc.next();
            tasks.add(task);
        }
        else if (command.equals("REMOVE"))
        {
            if (!tasks.isEmpty())
            {
                tasks.removeFirst();
            }
        }
        else if (command.equals("SHOW"))
        {
            if (tasks.isEmpty())
            {
                System.out.println("EMPTY");
            }
            else
            {
                for (String t : tasks)
                {
                    System.out.print(t + " ");
                }
                System.out.println();
            }
        }
    }
    sc.close();
  }
}
```

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

2. Problem Statement

Aditi is analyzing stock market trends and wants to find the Next Greater Element (NGE) for each stock price in a list. The Next Greater Element for an element x in an array is the first element to the right that is greater than x. If no greater element exists, return -1 for that position.

Your task is to help Aditi by efficiently computing the Next Greater Element for each element in the given array using a Stack.

Example:

Input:

6

4 5 2 10 8 6

Output:

5 10 10 -1 -1 -1

Explanation:

For each element:

4 5 (next greater element)5 102 1010 -1 (No greater element)8 -16 -1

Input Format

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

The second line contains n space-separated integers arr[i], where arr[i] is the stock price on the i-th day.

Output Format

The output prints n space-separated integers representing the Next Greater Element for each element in the array.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
4 5 2 10 8 6
Output: 5 10 10 -1 -1 -1

Answer

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

```
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[] result = new int[n];
        Stack<Integer> stack = new Stack<>();


        for (int i = n - 1; i >= 0; i--)
        {

            while (!stack.isEmpty() && stack.peek() <= arr[i])
            {
                stack.pop();
            }

            if (stack.isEmpty())
            {
                result[i] = -1;
            }
            else
            {
                result[i] = stack.peek();
            }

            stack.push(arr[i]);
        }

        for (int i = 0; i < n; i++)
        {
            System.out.print(result[i] + " ");
        }
```

```
        System.out.println();

        sc.close();
    }
}
```

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

3. Problem Statement

Rekha is a teacher who wants to calculate the average of marks scored by her students in a test. She needs to store all the marks dynamically because the number of students may vary each time. Using an ArrayList allows her to easily add any number of marks without worrying about the initial size.

Help her implement the task.

Input Format

The first line of input is an integer n, representing the number of students..

The second line of input consists of n double values, representing the marks of each student, separated by a space.

Output Format

The output prints: "Average of the list: " followed by the average value formatted to two decimal places.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1.0 2.0 3.0 4.0 5.0
Output: Average of the list: 3.00

Answer

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

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

      int n = sc.nextInt();
      ArrayList<Double> marks = new ArrayList<>();

      for (int i = 0; i < n; i++)
      {
         marks.add(sc.nextDouble());
      }

      double sum = 0.0;
      for (double mark : marks)
      {
         sum += mark;
      }

      double average = sum / n;

      System.out.printf("Average of the list: %.2f%n", average);

      sc.close();
   }
}
```

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