Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
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_2028_OOPS using Java_Week 9_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 14

Section 1 : MCQ

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

Answer

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

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(10);
        list.add(20);
```

```
        list.add(30);
        list.remove(1);
        System.out.println(list);
    }
}
```

Answer

[10, 30]

Status : Correct   *Marks : 1/1*

3. 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*

4. 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*

5. 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*

6. 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*

7.   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*

8.   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*

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

Answer

Adds an element to the beginning of the list

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("Java");
    list.add("Python");
    list.add("Java");
    list.add("C++");
    System.out.println(list.indexOf("Java"));
  }
}
```

Answer

0

Status : Correct   *Marks : 1/1*

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

Answer

list.get(0);

Status : Correct   *Marks : 1/1*

13. 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*

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

Answer

pop()

Status : Wrong   *Marks : 0/1*

15. What is Collection in Java?

Answer

A group of objects

Status : Correct   *Marks : 1/1*

Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
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 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

```java
import java.util.ArrayList;
import java.util.Scanner;
class IncreasingSubsequence{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int N=sc.nextInt();
        ArrayList<Integer>increasingSubsequence=new ArrayList<>();
        for(int i=0;i<N;i++){
            int number=sc.nextInt();
            if(increasingSubsequence.isEmpty()||
number>increasingSubsequence.get(increasingSubsequence.size()-1)){
                increasingSubsequence.add(number);
            }
        }
        System.out.println(increasingSubsequence);
        sc.close();
    }
}
```

Status : Correct   Marks : 10/10

Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

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

```java
import java.util.LinkedList;
import java.util.Scanner;
class PlaylistManager {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        LinkedList<String> playlist = new LinkedList<>();
        int currentIndex = -1;
        int n = scanner.nextInt();
        scanner.nextLine();

        for (int i = 0; i < n; i++) {
            String command = scanner.nextLine();
            String[] parts = command.split(" ");

            if (parts[0].equals("ADD")) {
                String song = parts[1];
                playlist.add(song);
                if (currentIndex == -1) {
                    currentIndex = 0;
                }
            }
            else if (parts[0].equals("REMOVE")) {
                String song = parts[1];
                int indexToRemove = playlist.indexOf(song);
                if (indexToRemove != -1) {
                    playlist.remove(indexToRemove);
                    if (playlist.isEmpty()) {
                        currentIndex = -1;
                    } else if (indexToRemove < currentIndex) {
                        currentIndex--;
                    } else if (indexToRemove == currentIndex) {
                        if (currentIndex >= playlist.size()) {
                            currentIndex = 0;
                        }
                    }
                }
            }
            else if (parts[0].equals("SHOW")) {
                if (playlist.isEmpty()) {
                    System.out.println("EMPTY");
                } else {
                    for (int j = 0; j < playlist.size(); j++) {
```

```java
                    System.out.print(playlist.get(j));
                    if (j < playlist.size() - 1) {
                        System.out.print(" ");
                    }
                }
                System.out.println();
            }
        }
        else if (parts[0].equals("NEXT")) {
            if (playlist.isEmpty()) {
                System.out.println("EMPTY");
            } else {
                currentIndex = (currentIndex + 1) % playlist.size();
                System.out.println(playlist.get(currentIndex));
            }
        }
    }

    scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

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

```java
import java.util.ArrayList;
import java.util.Scanner;
class NameFrequenceCounter{
public static void main(String[] args){
    Scanner sc=new Scanner(System.in);
    int N=sc.nextInt();
    sc.nextLine();
    ArrayList<String> names=new ArrayList<>();

    for(int i=0;i<N;i++){
      String name=sc.nextLine();
      names.add(name);
    }
    String searchName=sc.nextLine();
    int frequency=0;
    for(String name:names){
      if(name.equals(searchName)){
         frequency++;
```

```
        }
    }
    System.out.println(frequency);
    sc.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
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_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

```java
import java.util.LinkedList;
import java.util.Scanner;
class TaskManager{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        sc.nextLine();
        LinkedList<String> tasks=new LinkedList<>();
        for(int i=00;i<n;i++){
            String command=sc.nextLine();
            if(command.startsWith("ADD ")){
                String task=command.substring(4);
                tasks.add(task);
            }
            else if(command.equals("REMOVE")){
                if(!tasks.isEmpty()){
```

```java
                tasks.removeFirst();
            }
        }
        else if(command.equals("SHOW")){
            if(tasks.isEmpty()){
                System.out.println("EMPTY");
            }
            else{
                System.out.println(String.join(" ",tasks));
            }
        }
    }
    sc.close();
    }
}
```

Status : Correct   *Marks : 10/10*

2.   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

```java
import java.util.ArrayList;
import java.util.Scanner;
class AverageMarksCalculator{
    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++){
            double mark=sc.nextDouble();
            marks.add(mark);
        }
        double sum=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***

3. 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

```
import java.util.Scanner;
import java.util.Stack;
class NextGreaterElement{
```

```java
public static void main(String[] args){
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    int[] arr=new int[n];
    int[] nge=new int[n];
    for(int i=0;i<n;i++){
        arr[i]=sc.nextInt();
    }
    Stack<Integer> stack=new Stack<>();
    for(int i=0;i<n;i++){
        while(!stack.isEmpty()&&arr[i]>arr[stack.peek()]){
            nge[stack.pop()]=arr[i];
        }
        stack.push(i);
    }
    while(!stack.isEmpty()){
        nge[stack.pop()]=-1;
    }
    for(int i=0;i<n;i++){
        System.out.print(nge[i]+" ");
    }
    sc.close();
}
}
```

Status : Correct   *Marks : 10/10*

Rajalakshmi Engineering College

Name: akshaya d
Email: 241901002@rajalakshmi.edu.in
Roll no: 241901002
Phone: 9363744696
Branch: REC
Department: CSE (CS) - Section 2
Batch: 2028
Degree: B.E - CSE (CS)

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 9_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Sarah, a warehouse manager, is managing a list of product names in her store's inventory system. She needs to perform basic operations like adding (inserting) new products, removing products that are sold out or discontinued, displaying all the products in stock, and searching for a specific product in the inventory list.

Sarah's goal is to manage the inventory using a list of product names (strings). The system allows her to perform the following operations using ArrayList:

Insert a Product: Sarah adds a new product to the inventory.Delete a Product: Sarah removes a product from the inventory when it's sold or discontinued.Display the Inventory: Sarah checks all the products currently available in the inventory.Search for a Product: Sarah searches for a

specific product in the inventory to check if it's available.

Input Format

The input consists of multiple space-separated values representing different operations on a product list. Each operation follows a specific format:

1 <product_name> - Adds <product_name> to the product list.

2 <product_name> - Removes <product_name> from the product list if it exists.

3 - Print all products currently on the list.

4 <product_name> - Checks if <product_name> exists in the list.

Output Format

The output displays,

For (choice 1) prints, " <item> has been added to the list."

For (choice 2) prints, " <item> has been removed from the list."

For (choice 3) prints, "Items in the list:" followed by each item in the list on a new line, or "The list is empty." if the list is empty.

For (choice 4) prints," <item> is found in the list." or " <item> not found in the list."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 apple 1 banana 2 apple 3 4 apple

Output: apple has been added to the list.
banana has been added to the list.
apple has been removed from the list.
Items in the list:
banana
apple not found in the list.

Answer

import java.util.ArrayList;

```java
import java.util.Scanner;
// You are using Java
class StringListOperations {
    public static void insertItem(ArrayList<String> list,String item){
        list.add(item);
        System.out.println(item+" has been added to the list.");
    }
    public static void deleteItem(ArrayList<String> list,String item){
        if(list.remove(item)){
            System.out.println(item+" has been removed from the list.");
        }
        else{
            System.out.println(item+" not found in the list.");

        }
    }
    public static void displayList(ArrayList<String> list){
        if(list.isEmpty()){
            System.out.println("The list is empty.");
        }
        else{
            System.out.println("Items in the list: ");
            for(String product:list){
                System.out.println(product);
            }
        }
    }
    public static void searchItem(ArrayList<String> list,String item){
        if(list.contains(item)){
            System.out.println(item+" is found in the list.");
        }
        else{
            System.out.println(item+" not found in the list.");

        }
    }

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

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

String input = sc.nextLine();
String[] commands = input.split(" ");
int i = 0;
while (i < commands.length) {
    int choice = Integer.parseInt(commands[i]);
    switch (choice) {
        case 1:
            if (i + 1 < commands.length) {
                StringListOperations.insertItem(list, commands[i + 1]);
                i += 2;
            } else {
                System.out.println("No string provided for insertion.");
                i++;
            }
            break;
        case 2:
            if (i + 1 < commands.length) {
                StringListOperations.deleteItem(list, commands[i + 1]);
                i += 2;
            } else {
                System.out.println("No string provided for deletion.");
                i++;
            }
            break;
        case 3:
            StringListOperations.displayList(list);
            i += 1;
            break;
        case 4:
            if (i + 1 < commands.length) {
                StringListOperations.searchItem(list, commands[i + 1]);
                i += 2;
            } else {
                System.out.println("No string provided for searching.");
                i++;
            }
            break;
    }
}
}
```

}

2.   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

```java
import java.util.Scanner;
import java.util.Stack;
class StockSpan{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int[] prices=new int[n];
        for(int i=0;i<n;i++){
            prices[i]=sc.nextInt();
        }
        int[] spans=new int[n];
        Stack<Integer> stack=new Stack<>();
        for(int i=0;i<n;i++){
            while(!stack.isEmpty()&&prices[i]>=prices[stack.peek()]){
                stack.pop();
            }
            if(stack.isEmpty()){
                spans[i]=i+1;
            }
            else{
                spans[i]=i-stack.peek();
            }
            stack.push(i);
        }
        for(int span:spans){
```

```
        System.out.print(span+" ");
        }
        sc.close();
    }
}
```

Status : Correct   *Marks : 10/10*

3.   Problem Statement

Raman, a computer science teacher, is responsible for registering students
for his programming class. To streamline the registration process, he
wants to develop a program that stores students' names and allows him to
retrieve a student's name based on their index in the list.

Raman has decided to use an ArrayList to store the names of students, as
it provides efficient dynamic resizing and indexing.

Write a program that enables Raman to input the names of students and
fetch a student's name using the specified index. If the entered index is
invalid, the program should return an appropriate message.

Input Format

The first line of input consists of an integer n, representing the number of
students to register.

The next n lines of input consist of the names of each student, one by one.

The last line of input is an integer, representing the index (0-indexed) of the
element to retrieve.

Output Format

If the index is valid (within the bounds of the ArrayList), print "Element at index
[index]: " followed by the element (student name as string).

If the index is invalid, print "Invalid index".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
Alice
Bob
Ankit
Alice
Prajit
2

Output: Element at index 2: Ankit

Answer

```java
import java.util.ArrayList;
import java.util.Scanner;
class StudentRegistration{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        sc.nextLine();
        ArrayList<String> students=new ArrayList<>();
        for(int i=0;i<n;i++){
            String name=sc.nextLine();
            students.add(name);
        }
        int index=sc.nextInt();
        if(index>=0&&index<students.size()){
            System.out.println("Element at index "+index+": "+students.get(index));
        }
        else{
            System.out.println("Invalid index");
        }
        sc.close();
    }
}
```

Status : Correct   *Marks : 10/10*

4.   Problem Statement

Sanjay is working on a program to merge two sorted linked lists into a single sorted list using Java's LinkedList class from the Collections framework. Given two sorted linked lists, he wants to merge them while maintaining the sorted order.

Write a Java program that:

Reads two sorted linked lists.Merges them into a single sorted linked list.Prints the merged list in ascending order.

Input Format

The first line contains an integer m (the size of the first linked list).

The second line contains m space-separated integers (sorted).

The third line contains an integer n (the size of the second linked list).

The fourth line contains n space-separated integers (sorted).

Output Format

The output prints the merged linked list as space-separated integers.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
5 10
3
1 3 8
Output: 1 3 5 8 10

Answer

```java
import java.util.*;
class MergeSortedLinkedLists {

// You are using Java

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

```java
        Scanner sc=new Scanner(System.in);
        int m=sc.nextInt();
        LinkedList<Integer> list1=new LinkedList<>();
        for(int i=0;i<m;i++){
            list1.add(sc.nextInt());
        }
        int n=sc.nextInt();
        LinkedList<Integer> list2=new LinkedList<>();
        for(int i=0;i<n;i++){
            list2.add(sc.nextInt());
        }
        LinkedList<Integer> mergedList=mergeSortedLists(list1,list2);
        for(int num:mergedList){
          System.out.print(num+" ");

        }
        sc.close();


    }
    private static LinkedList<Integer> mergeSortedLists(LinkedList<Integer> list1,LinkedList<Integer> list2){
        LinkedList<Integer> mergedList=new LinkedList<>();
        ListIterator<Integer> iterator1=list1.listIterator();
        ListIterator<Integer> iterator2=list2.listIterator();
        int val1=iterator1.hasNext()?iterator1.next():Integer.MAX_VALUE;
        int val2=iterator2.hasNext()?iterator2.next():Integer.MAX_VALUE;
        while(val1!=Integer.MAX_VALUE||val2!=Integer.MAX_VALUE){
            if(val1<=val2){
                mergedList.add(val1);
                val1=iterator1.hasNext()?iterator1.next():Integer.MAX_VALUE;
            }
            else{
                mergedList.add(val2);
                val2=iterator2.hasNext()?iterator2.next():Integer.MAX_VALUE;
            }
        }
        return mergedList;

    }
}
```