Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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.*;
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*

2. 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*

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

Answer

Adds an element to the beginning of the list

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);
```

2116240701322

```
    }
}
```

Answer

[10, 30]

Status : Correct *Marks : 1/1*

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

Answer

push()

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<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*

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

Answer

list.get(0);

2116240701322 2116240701322 2116240701322 2116240701322

Status : Correct *Marks : 1/1*

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

Answer

removeLast()

Status : Correct *Marks : 1/1*

9. 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*

10. What is Collection in Java?

Answer

A group of objects

Status : Correct *Marks : 1/1*

2116240701322

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

Answer

ArrayList&lt;String&gt; list = new ArrayList&lt;&gt;();

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> 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***

13. 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);
```

2116240701322

```
    }
}
```

Answer

[Banana]

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

14. 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***

15. 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());
```

2116240701322

```
    }
}
```

Answer

Size of the list: 3

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

2116240701322

Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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.

2116240701322

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

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

public class Main {

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

      try {
         int N = scanner.nextInt();

         List<Integer> resultList = new ArrayList<>();

         for (int i = 0; i < N; i++) {
            int currentNumber = scanner.nextInt();

            if (resultList.isEmpty()) {
               resultList.add(currentNumber);
            } else {
               int lastElement = resultList.get(resultList.size() - 1);

               if (currentNumber > lastElement) {
                  resultList.add(currentNumber);
```

2116240701322

```
                }
            }
        }

        System.out.println(resultList.toString());

    } catch (Exception e) {

    } finally {
        scanner.close();
    }
  }
}
```

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

2116240701322

Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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

2116240701322

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

2116240701322

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

public class Main {

    private List<String> playlist;
    private int currentSongIndex;

    public Main() {
        this.playlist = new LinkedList<>();
        this.currentSongIndex = -1;
    }

    public void processCommand(String commandLine) {
        String[] parts = commandLine.split(" ", 2);
        String operation = parts[0];

        switch (operation) {
            case "ADD":
                if (parts.length > 1) {
                    String song = parts[1];
                    playlist.add(song);
                    if (playlist.size() == 1) {
                        currentSongIndex = 0;
                    }
                }
                break;

            case "REMOVE":
                if (parts.length > 1) {
                    String songToRemove = parts[1];
                    int initialSize = playlist.size();

                    boolean removed = playlist.remove(songToRemove);

                    if (removed) {
                        if (initialSize > 0) {

                            if (playlist.isEmpty()) {
                                currentSongIndex = -1;
                            } else {
```

2116240701322

```
                                if (currentSongIndex >= playlist.size()) {
                                    currentSongIndex = 0;
                                }
                            }
                        }
                    }
                }
                break;

            case "SHOW":
                if (playlist.isEmpty()) {
                    System.out.println("EMPTY");
                } else {
                    for (int i = 0; i < playlist.size(); i++) {
                        System.out.print(playlist.get(i) + (i < playlist.size() - 1 ? " " : ""));
                    }
                    System.out.println();
                }
                break;

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

            default:
                break;
        }
    }

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

        if (!scanner.hasNextInt()) {
            scanner.close();
            return;
        }
        int N = scanner.nextInt();
```
2116240701322

```
        scanner.nextLine();

        Main manager = new Main();

        for (int i = 0; i < N; i++) {
            if (scanner.hasNextLine()) {
                String commandLine = scanner.nextLine().trim();
                manager.processCommand(commandLine);
            }
        }

        scanner.close();
    }
}
```

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

2116240701322

Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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.

2116240701322

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

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

public class Main {

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

        if (!scanner.hasNextInt()) {
            scanner.close();
            return;
        }

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

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

2116240701322

```
        for (int i = 0; i < N; i++) {
            if (scanner.hasNextLine()) {
                String name = scanner.nextLine().trim();
                names.add(name);
            }
        }

        String searchName = "";
        if (scanner.hasNextLine()) {
            searchName = scanner.nextLine().trim();
        }

        int frequency = 0;

        for (String name : names) {
            if (name.equals(searchName)) {
                frequency++;
            }
        }

        System.out.println(frequency);

        scanner.close();
    }
}
```

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

2116240701322

Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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

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.

2116240701322

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

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

public class Main {

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

      if (!scanner.hasNextInt()) {
         scanner.close();
         return;
      }

      int N = scanner.nextInt();

      List<Double> marks = new ArrayList<>();
      double sum = 0.0;

      for (int i = 0; i < N; i++) {
         if (scanner.hasNextDouble()) {
            double mark = scanner.nextDouble();
            marks.add(mark);
            sum += mark;
         }
```

2116240701322

```
        }

        double average = 0.0;
        if (N > 0) {
            average = sum / N;
        }

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

        scanner.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

2116240701322

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;

public class Main {

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

    if (!scanner.hasNextInt()) {
      scanner.close();
      return;
    }

    int N = scanner.nextInt();
    int[] arr = new int[N];
    int[] result = new int[N];

    for (int i = 0; i < N; i++) {
      if (scanner.hasNextInt()) {
        arr[i] = scanner.nextInt();
```

2116240701322

```
            }
        }

        scanner.close();

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

        for (int i = N - 1; i >= 0; i--) {
            while (!stack.isEmpty() && arr[stack.peek()] <= arr[i]) {
                stack.pop();
            }

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

            stack.push(i);
        }

        for (int i = 0; i < N; i++) {
            System.out.print(result[i] + (i == N - 1 ? "" : " "));
        }
        System.out.println();
    }
}
```

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

3. 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.

2116240701322

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

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

public class Main {

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

      if (!scanner.hasNextInt()) {
```

2116240701322

```
            scanner.close();
            return;
        }

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

        List<String> taskList = new LinkedList<>();

        for (int i = 0; i < N; i++) {
            if (scanner.hasNextLine()) {
                String commandLine = scanner.nextLine().trim();
                String[] parts = commandLine.split(" ", 2);
                String command = parts[0];

                if (command.equalsIgnoreCase("ADD") && parts.length > 1) {
                    String task = parts[1];
                    taskList.add(task);
                } else if (command.equalsIgnoreCase("REMOVE")) {
                    if (!taskList.isEmpty()) {
                        taskList.remove(0);
                    }
                } else if (command.equalsIgnoreCase("SHOW")) {
                    if (taskList.isEmpty()) {
                        System.out.println("EMPTY");
                    } else {
                        System.out.println(String.join(" ", taskList));
                    }
                }
            }
        }

        scanner.close();
    }
}
```

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

2116240701322 2116240701322 2116240701322 2116240701322

Rajalakshmi Engineering College

Name: Mohamed Humaid Zahir Hussain
Email: 240701322@rajalakshmi.edu.in
Roll no: 2116240701322
Phone: 6382261139
Branch: REC
Department: CSE - Section 9
Batch: 2028
Degree: B.E - CSE

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 : 40

Section 1 : Coding

1. 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

2116240701322

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

```
import java.util.Arrays;
import java.util.Scanner;
import java.util.Stack;

public class Main {
```

2116240701322

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

        if (!scanner.hasNextInt()) {
            scanner.close();
            return;
        }
        int N = scanner.nextInt();

        int[] prices = new int[N];
        for (int i = 0; i < N; i++) {
            if (scanner.hasNextInt()) {
                prices[i] = scanner.nextInt();
            }
        }

        scanner.close();

        int[] span = new int[N];
        Stack<Integer> stack = new Stack<>();

        for (int i = 0; i < N; i++) {
            while (!stack.isEmpty() && prices[stack.peek()] <= prices[i]) {
                stack.pop();
            }

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

            stack.push(i);
        }

        for (int i = 0; i < N; i++) {
            System.out.print(span[i] + (i == N - 1 ? "" : " "));
        }
        System.out.println();
    }
}
```

2116240701322

Status : Correct *Marks : 10/10*

2. 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;

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

2116240701322

```
class VowelFilter {
  public static void filterWords(int N, Scanner scanner) {
    List<String> resultWords = new ArrayList<>();

    for (int i = 0; i < N; i++) {
      if (scanner.hasNextLine()) {
        String word = scanner.nextLine().trim();

        if (word.isEmpty()) {
          continue;
        }

        int vowelCount = 0;
        for (char c : word.toCharArray()) {
          if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
            vowelCount++;
          }
        }

        if (vowelCount <= 2) {
          resultWords.add(word);
        }
      }
    }

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

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***

2116240701322

3. Problem Statement

Rahul is working on a list manipulation problem where he needs to reverse a specific subarray using a stack. Given an array and two indices l and r, he wants to reverse only the portion of the array from index l to r (both inclusive) while keeping the rest of the array unchanged.

Since Rahul wants to solve this problem efficiently, he decides to use a stack to reverse the subarray in O(r - l) time.

Your task is to help Rahul by implementing this functionality.

Input Format

The first line contains an integer n, the size of the array.

The second line contains n space-separated integers arr[i].

The third line contains two integers l and r, denoting the start and end indices of the subarray to reverse.

Note: The array follows 0-based indexing.

Output Format

The output prints the modified array after reversing the subarray between indices l and r.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
1 2 3 4 5 6
1 4
Output: 1 5 4 3 2 6

Answer

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

2116240701322

```
public class Main {


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

        int n = 0;
        if (scanner.hasNextInt()) {
            n = scanner.nextInt();
        }

        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            if (scanner.hasNextInt()) {
                arr[i] = scanner.nextInt();
            }
        }

        int l = 0;
        int r = 0;
        if (scanner.hasNextInt()) {
            l = scanner.nextInt();
        }
        if (scanner.hasNextInt()) {
            r = scanner.nextInt();
        }

        scanner.close();

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

        for (int i = l; i <= r; i++) {
            stack.push(arr[i]);
        }

        for (int i = l; i <= r; i++) {
            arr[i] = stack.pop();
        }

        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + (i == n - 1 ? "" : " "));
        }
```

2116240701322

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



}
```

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.

2116240701322

Sample Test Case

Input: 2
5 10
3
1 3 8

Output: 1 3 5 8 10

Answer

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


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

    int m = 0;
    if (scanner.hasNextInt()) {
      m = scanner.nextInt();
    }
    LinkedList<Integer> list1 = readList(scanner, m);

    int n = 0;
    if (scanner.hasNextInt()) {
      n = scanner.nextInt();
    }
    LinkedList<Integer> list2 = readList(scanner, n);

    scanner.close();

    LinkedList<Integer> mergedList = new LinkedList<>();

    int i = 0;
    int j = 0;

    while (i < list1.size() && j < list2.size()) {
      int val1 = list1.get(i);
      int val2 = list2.get(j);

      if (val1 <= val2) {
        mergedList.add(val1);
        i++;
```

2116240701322

```
            } else {
                mergedList.add(val2);
                j++;
            }
        }

        while (i < list1.size()) {
            mergedList.add(list1.get(i));
            i++;
        }

        while (j < list2.size()) {
            mergedList.add(list2.get(j));
            j++;
        }

        for (int k = 0; k < mergedList.size(); k++) {
            System.out.print(mergedList.get(k));
            if (k < mergedList.size() - 1) {
                System.out.print(" ");
            }
        }
        System.out.println();
    }

    private static LinkedList<Integer> readList(Scanner scanner, int size) {
        LinkedList<Integer> list = new LinkedList<>();
        for (int i = 0; i < size; i++) {
            if (scanner.hasNextInt()) {
                list.add(scanner.nextInt());
            }
        }
        return list;
    }

}
```

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

2116240701322