Rajalakshmi Engineering College

Name: naveenraj s
Email: 240701350@rajalakshmi.edu.in
Roll no: 240701350
Phone: 6379711376
Branch: REC
Department: CSE - Section 3
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

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

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

// You are using Java

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        LinkedList<Integer> list1 = new LinkedList<>();
        LinkedList<Integer> list2 = new LinkedList<>();
        int n = sc.nextInt();
        for(int i=0;i<n;i++){
            int ele = sc.nextInt();
            list1.add(ele);
        }
        int m = sc.nextInt();
        for(int i=0;i<m;i++){
            int ele = sc.nextInt();
            list2.add(ele);
        }
        list1.addAll(list2);
        Collections.sort(list1);
```

```
        for(int i : list1){
            System.out.print(i+" ");
        }
    }
}
```

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

2. 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
class ItemType {
    String name;
    double deposit;
    double costPerDay;

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

}

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 : Correct **Marks : 10/10**

3. 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.*;
class list{
   public static void main(String[] args){
      Scanner sc = new Scanner(System.in);
      LinkedList<String> list = new LinkedList<>();
      int n = sc.nextInt();
      for(int i=0;i<n;i++){
         String song = sc.next();
         list.add(song);
      }
      int m = sc.nextInt();
      for(int i=0;i<m;i++){
         int change1 = sc.nextInt();
         int change2 = sc.nextInt();
```

```
            String temp = list.remove(change1);
             list.add(change2,temp);
        }
        for(String i : list)
         System.out.print(i+" ");
    }
}
```

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

4. 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.*;

public class Main {

// You are using Java

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      Stack<Integer>stack = new Stack<>();
      int n =sc.nextInt();
      int count=0;
      for(int i=0;i<n;i++){
         int num1=sc.nextInt();
         stack.push(num1);
      }
      int k=sc.nextInt();
      int l=sc.nextInt();
      for(int i=count;i<n;i++){
         if(i==k){
            do{
               System.out.print(stack.get(l)+" ");
               l--;
               i++;
               if(l==k){
                  System.out.print(stack.get(k)+" ");
                  break;
               }
            }while(i<stack.size());
         }
         else
            System.out.print(stack.get(i)+" ");
      }
   }
}
```

Status : Correct *Marks : 10/10*