CSE3032 - Competitive Programming
WIN SEM (2022-2023) AMR
Class Number: AP2022236001007
Slot: L11+L12+L19+L20
ASSIGNMENT - 11
Last Date for Submission: Thursday (27-04-2023) @ 12.40PM

Name: Chilamakuru Sai Venkata Yaswanth

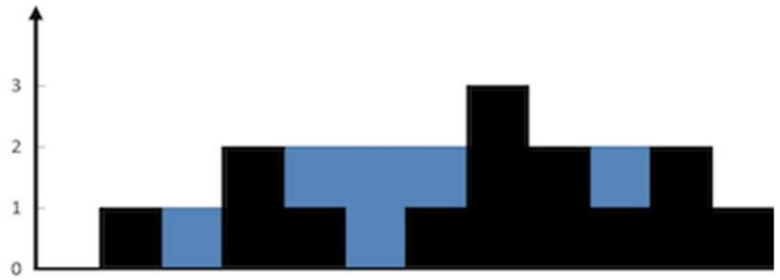
Reg No: 20BCI7076

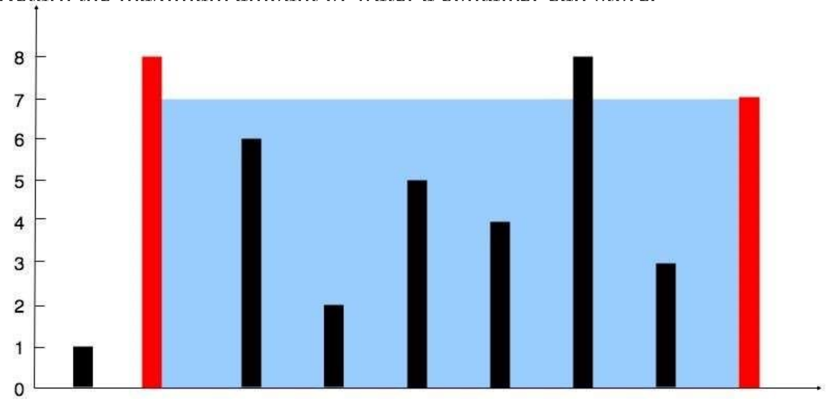
Write the program using (C / C++ / Java / Python) to solve the following problems.

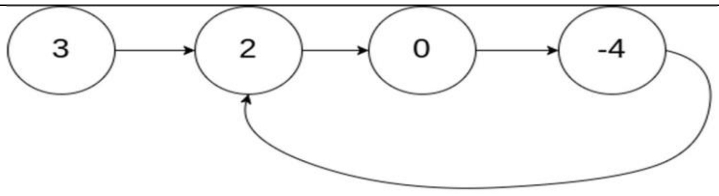
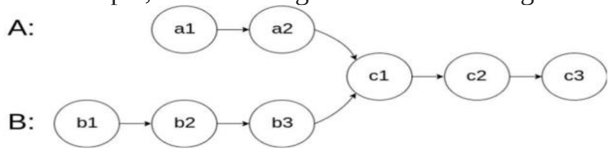
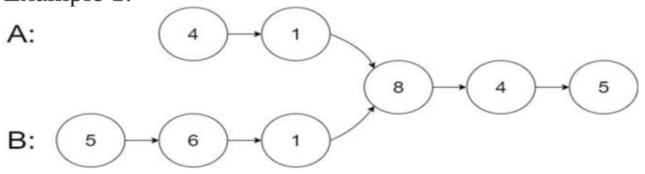
Concept: Two pointer & Fast and Slow

S.No	Problem Name	Statement
1	Pair With Given Difference	Given a one-dimensional unsorted array A containing N integers. You are also given an integer B, find if there exists a pair of elements in the array whose difference is B. Return 1 if any such pair exists else return 0. Input 1: A = [5, 10, 3, 2, 50, 80] B = 78 Output 1: 1 Explanation 1: Pair (80, 2) gives a difference of 78. Input 2: A = [-10, 20] B = 30 Output 2: 1 Explanation 2: Pair (20, -10) gives a difference of 30 i.e., 20 - (-10) => 20 + 10 => 30
2	Two Sum	Given an array, find the indexes of two elements whose sum is equal to the given sum. If there exist multiple solutions, print the minimum indices. Example 1: Input: arr[]={1,3,4,5,7,10,11,19,20}, sum=7 Output: {1,2} Example 2: Input: arr[]={2,9,13,21,54}, sum=63 Output: {1,4}

3	Remove Duplicates in-place from Sorted Array	Given an integer array sorted in non-decreasing order, remove the duplicates in place such that each unique element appears only once. The relative order of the elements should be kept the same. If there are k elements after removing the duplicates, then the first k elements of the array should hold the final result. It does not matter what you leave beyond the first k elements. Note: Return k after placing the final result in the first k slots of the array. Example 1: Input: arr[1,1,2,2,2,3,3]

		Output: arr[1,2,3,_,_,_,_] Explanation: Total number of unique elements are 3, i.e[1,2,3] and therefore return 3 after assigning [1,2,3] in the beginning of the array. Example 2: Input: arr[1,1,1,2,2,3,3,3,3,4,4] Output: arr[1,2,3,4,_,_,_,_,_,_,_] Explanation: Total number of unique elements are 4, i.e[1,2,3,4] and therefore return 4 after assigning [1,2,3,4] in the beginning of the array.
4	Trapping Rain Water	Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.  **Input:** height = [0,1,0,2,1,0,1,3,2,1,2,1] **Output:** 6 **Explanation:** The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.
5	3Sum	Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0. Notice that the solution set must not contain duplicate triplets. Input: nums = [-1,0,1,2,-1,-4] Output: [[-1,-1,2],[-1,0,1]]
6	3Sum Closest	Given an integer array nums of length n and an integer target, find three integers in nums such that the sum is closest to target. Return the sum of the three integers. You may assume that each input would have exactly one solution. Input: nums = [-1,2,1,-4], target = 1 Output: 2 Explanation: The sum that is closest to the target is 2. (-1 + 2 + 1 = 2). Input: nums = [0,0,0], target = 1 Output: 0 Explanation: The sum that is closest to the target is 0. (0 + 0 + 0 = 0).

7	4Sum	Given an array nums of n integers, return an array of all the unique quadruplets [nums[a], nums[b], nums[c], nums[d]] such that: 0 <= a, b, c, d < n a, b, c, and d are distinct. nums[a] + nums[b] + nums[c] + nums[d] == target You may return the answer in any order. Input: nums = [1,0,-1,0,-2,2], target = 0 Output: [[-2,-1,1,2],[-2,0,0,2],[-1,0,0,1]] Input: nums = [2,2,2,2,2], target = 8 Output: [[2,2,2,2]]
8	Container with Most Water	You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]). Find two lines that together with the x-axis form a container, such that the container contains the most water. Return the maximum amount of water a container can store.  **Input:** height = [1,8,6,2,5,4,8,3,7] **Output:** 49 **Explanation:** The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.
9	Next Permutation	Implement next permutation, which rearranges numbers into the lexicographically next greater permutation of numbers. If such an arrangement is not possible, it must rearrange it as the lowest possible order (i.e., sorted in ascending order). Input: nums = [1,2,3] Output: [1,3,2]Input: nums = [3,2,1] Output: [1,2,3]
10	Linked List Cycle II	Given the head of a linked list, return the node where the cycle begins. If there is no cycle, return null. There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Internally, pos is used to denote the index of the node that tail's next pointer is connected to (0-indexed). It is -1 if there is no cycle. Note that pos is not passed as a parameter. Do not modify the linked list.

		 **Input:** head = [3,2,0,-4], pos = 1 **Output:** tail connects to node index 1 **Explanation:** There is a cycle in the linked list, where tail connects to the second node.
11	Intersection of Two Linked Lists	Given the heads of two singly linked-lists headA and headB, return the node at which the two lists intersect. If the two linked lists have no intersection at all, return null. For example, the following two linked lists begin to intersect at node c1:  The test cases are generated such that there are no cycles anywhere in the entire linked structure. Note that the linked lists must retain their original structure after the function returns. Example 1:  Input: intersectVal = 8, listA = [4,1,8,4,5], listB = [5,6,1,8,4,5], skipA = 2, skipB = 3 Output: Intersected at '8' Explanation: The intersected node's value is 8 (note that this must not be 0 if the two lists intersect). From the head of A, it reads as [4,1,8,4,5]. From the head of B, it reads as [5,6,1,8,4,5]. There are 2 nodes before the intersected node in A; There are 3 nodes before the intersected node in B.

Note:

- If Code similarity is found, assignment will not be considered and Zero (0) Marks will be awarded.
- You have to upload a single document consisting of all the above programs and corresponding Output.

Reference:

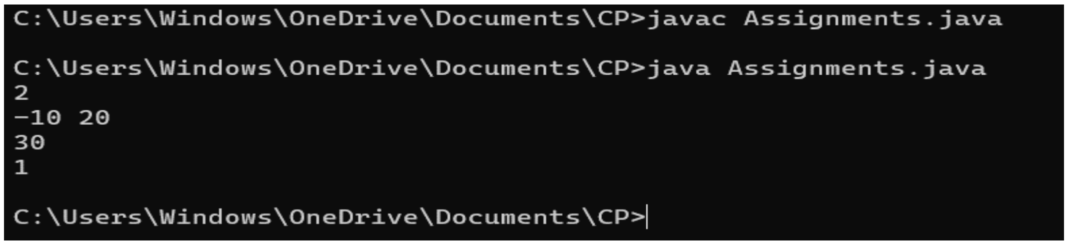
- https://www.interviewbit.com/courses/programming/two-pointers/two-pointers/
- https://leetcode.com/articles/two-pointer-technique/

1) Pair with given difference:

Code:

```
import java.util.*;
public class Assignments{
	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 b=sc.nextInt();
		Arrays.sort(arr);
//2,3,5,10,50,80
		int i=0,j=1;
		while(i<n && j<n){
			int diff=arr[j]-arr[i];
			if(diff==b){
				System.out.println("1");
				return;
			}
			else if(diff<b){
			j++;
			}
			else{
				i++;
			}
		}
		System.out.println("0");
	}
}
```

Output:



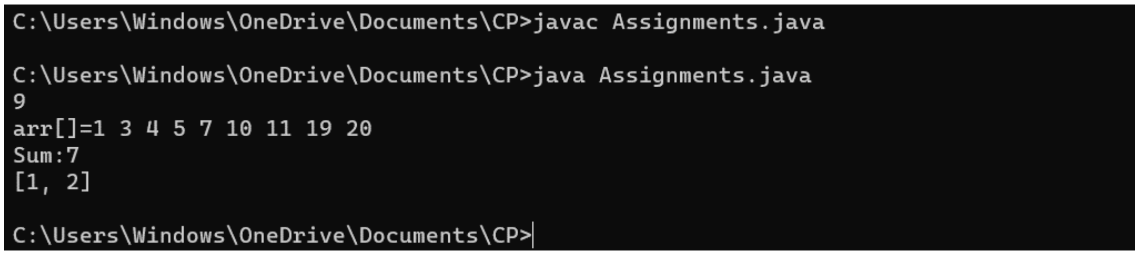
```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
2
-10 20
30
1

C:\Users\Windows\OneDrive\Documents\CP>
```

2) Two Sum:

Code:

```
import java.util.*;
public class Assignments{
         public static int[] twosum(int[] arr,int sum){
       int left=0,right=arr.length-1;
       while(left<right){
          int cur=arr[left]+arr[right];
          if (cur==sum) {
             return new int[]{left,right};
          }
                        else if(cur<sum){
             left++;
          }
                        else{
             right--;
          }
       }
       return new int[]{-1,-1};
    }
         public static void main(String [] args){
                 Scanner sc=new Scanner(System.in);
                 System.out.print("");
                 int n=sc.nextInt();
                 int[] arr=new int[n];
                 System.out.print("arr[]=");
                 for(int i=0;i<n;i++){
                         arr[i]=sc.nextInt();
                 }
                 System.out.print("Sum:");
                 int sum=sc.nextInt();
                 int[] indices=twosum(arr,sum);
       System.out.println(Arrays.toString(indices));
         }
}
```

Output:



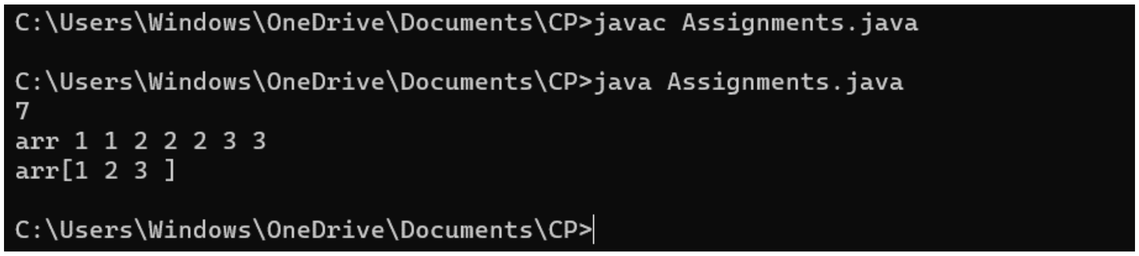
```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
9
arr[]=1 3 4 5 7 10 11 19 20
Sum:7
[1, 2]

C:\Users\Windows\OneDrive\Documents\CP>
```

3) Remove Duplicates in-place from Sorted Array

Code:

```
import java.util.*;
public class Assignments{
        static int duplicate(int[] arr,int n){
                if(n==0 || n==1)
                        return n;
                int j=0;
                for(int i=0;i<n-1;i++)
                        if(arr[i]!=arr[i+1])
                        arr[j++]=arr[i];
                arr[j++]=arr[n-1];
                return j;
        }
        public static void main(String [] args){
                Scanner sc=new Scanner(System.in);
                int n = sc.nextInt();
                int[] arr = new int[n];
                System.out.print("arr ");
                for(int i=0;i<n;i++){
                        arr[i]=sc.nextInt();
                }
                n=duplicate(arr,n);
                System.out.print("arr[");
                for(int i=0;i<n;i++)
                 System.out.print(arr[i]+" ");
                System.out.println("]");
        }
}
```

Output:



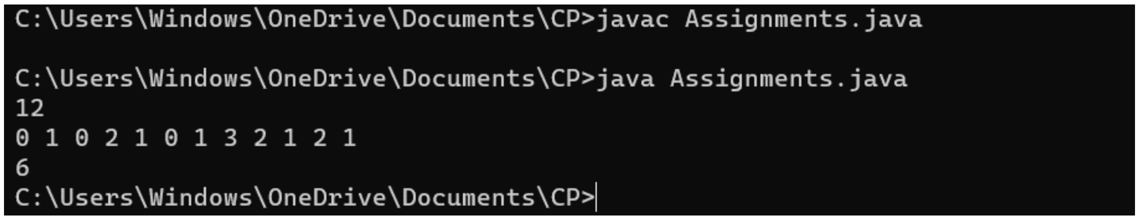
```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
7
arr 1 1 2 2 2 3 3
arr[1 2 3 ]

C:\Users\Windows\OneDrive\Documents\CP>
```

4) Trapping Rain Water:

Code:

```
import java.util.*;
public class Assignments{
        public static int rain(int[] arr,int n){
                int res=0;
                for(int i=1;i<n-1;i++){
                        int left=arr[i];
                        for(int j=0;j<i;j++){
                                left=Math.max(left,arr[j]);
                        }
                        int right=arr[i];
                        for(int j=i+1;j<n;j++){
                                right=Math.max(right,arr[j]);
                        }
                        res+=Math.min(left,right)-arr[i];
                }
                return res;
        }
        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();
                }
                System.out.print(rain(arr, n));
        }
}
```

Output:



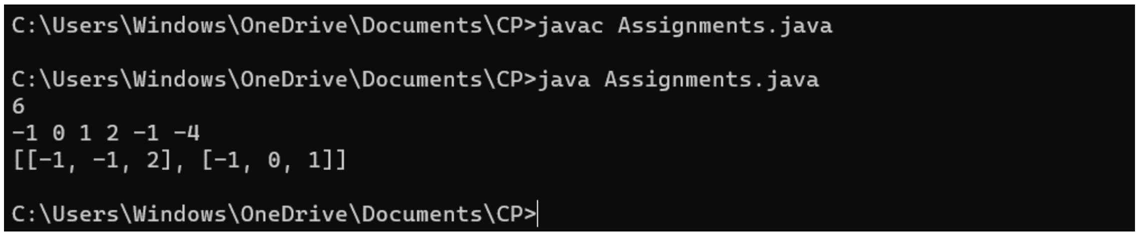
```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
12
0 1 0 2 1 0 1 3 2 1 2 1
6
C:\Users\Windows\OneDrive\Documents\CP>
```

5) Three Sum:

Code:

```
import java.util.Arrays;
import java.util.*;
public class Assignments{
        public static List<List<Integer> > threesum(int[] arr,int sum){
                Arrays.sort(arr);
                List<List<Integer> > list=new ArrayList<>();
                TreeSet<String> set=new TreeSet<String>();
                List<Integer> l=new ArrayList<>();
                for(int i=0;i<arr.length-2;i++){
                        int j=i+1;
                        int k=arr.length-1;
                        while(j<k){
                                if(arr[i]+arr[j]+arr[k]==sum){
                                        String str=arr[i]+":"+arr[j]+":"+arr[k];
                                        if(!set.contains(str)){
                                                l.add(arr[i]);
                                                l.add(arr[j]);
                                                l.add(arr[k]);
                                                list.add(l);

                                                l=new ArrayList<>();
                                                set.add(str);
                                        }
                                        j++;
                                        k--;
                                }
                                else if(arr[i]+arr[j]+arr[k]<sum)
                                        j++;
                                else
                        k--;
                    }
                }
                return list;
        }
        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 sum=0;
                List<List<Integer> > l=threesum(arr, sum);
                if (!l.isEmpty()){
                        System.out.println(l);
                }
                else {
                        System.out.println("No triplets can be formed");
                }
        }
}
```

Output:

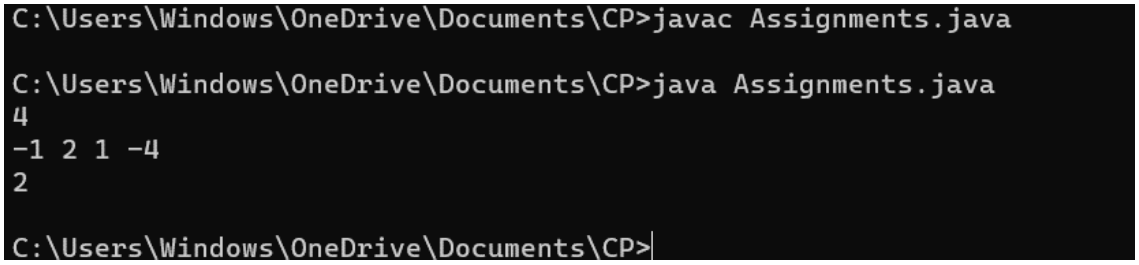


```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
6
-1 0 1 2 -1 -4
[[-1, -1, 2], [-1, 0, 1]]

C:\Users\Windows\OneDrive\Documents\CP>
```

6) Three Sum Closest:

Code:

```
import java.util.*;
public class Assignments{
	public static void closest(int[] arr,int n,int x,int count,int sum,int
ind,int[] ans,int[] min){
		if(ind==n){
			if(count==3){
				if(Math.abs(x-sum)<min[0]){
					min[0]=Math.abs(x-sum);
					ans[0]=sum;
				}
			}
			return;
		}
		closest(arr,n,x,count+1,sum+arr[ind],ind+1,ans,min);
		closest(arr,n,x,count,sum,ind+1,ans,min);
	}
	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 x=1;
		int[] min={Integer.MAX_VALUE};
		int[] ans=new int[1];
		closest(arr,n,x,0,0,0,ans,min);
		System.out.println(ans[0]);
	}
}
```



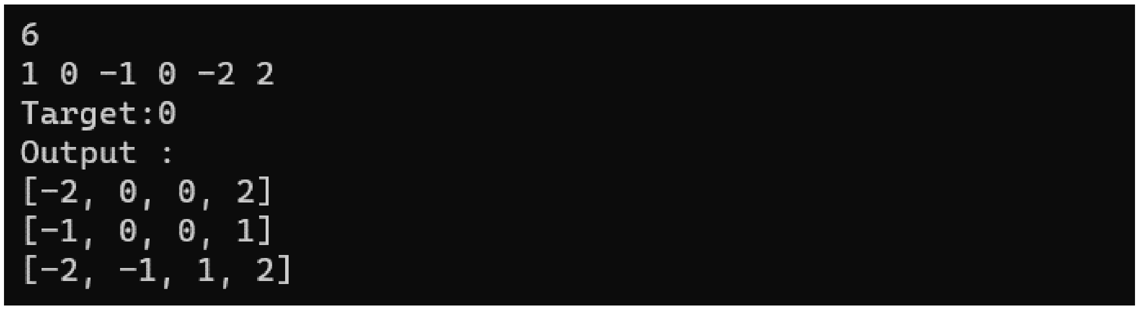
```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
4
-1 2 1 -4
2

C:\Users\Windows\OneDrive\Documents\CP>
```

7) Four Sum:

Code:

```
import java.util.*;
public class Assignments{
        static class Pair{
         int index1;
         int index2;
         Pair(int x,int y){
                 index1=x;
                 index2=y;
         }
        }
        public static void four(ArrayList<Integer> list,int target){
                HashMap<Integer, ArrayList<Pair> > map=new
HashMap<>();
                for(int i=0;i<list.size()-1;i++){
                        for(int j=i+1;j<list.size();j++){
                                int sum=list.get(i)+list.get(j);
                                if(!map.containsKey(sum)){
                                        ArrayList<Pair> temp=new ArrayList<>();
                                        Pair p=new Pair(i,j);
                                        temp.add(p);
                                        map.put(sum,temp);
                                }
                                else{
                                        ArrayList<Pair> temp=map.get(sum);
                                        Pair p=new Pair(i,j);
                                        temp.add(p);
                                        map.put(sum,temp);
                                }
                        }
                }
                HashSet<ArrayList<Integer> > ans=new
HashSet<ArrayList<Integer>>();
                 for(int i=0;i<list.size()-1;i++){
                        for(int j=i+1;j<list.size();j++){
                                int lookUp=target-(list.get(i)+list.get(j));
                                if(map.containsKey(lookUp)){
                                        ArrayList<Pair> temp=map.get(lookUp);
                                        for(Pair pair:temp){
                                             if(pair.index1!=i && pair.index1!=j
&& pair.index2!=i && pair.index2!=j){
                                              ArrayList<Integer> values=new
ArrayList<>();
                                              values.add(list.get(pair.index1));
                                              values.add(list.get(pair.index2));
                                              values.add(list.get(i));
                                              values.add(list.get(j));
                                              Collections.sort(values);
                                              ans.add(values);
                                             }
                                        }
                                   }
                              }
                         }
                         System.out.println("Output : ");
                         for (ArrayList<Integer> arr : ans) {
                              System.out.println(arr);
                         }
               }
               public static void main(String [] args){
                         Scanner sc=new Scanner(System.in);
                         int m=sc.nextInt();
                         for(int f=0;f<m;f++){
                              System.out.println("");
                              int n=sc.nextInt();
                              ArrayList<Integer> arr=new ArrayList<>();
                              for(int i=0;i<n;i++) {
                                   int x=sc.nextInt();
                                   arr.add(x);
                              }
                              System.out.print("Target:");
                              int K=sc.nextInt();
                              four(arr,K);
                         }
               }
}
```

Output:



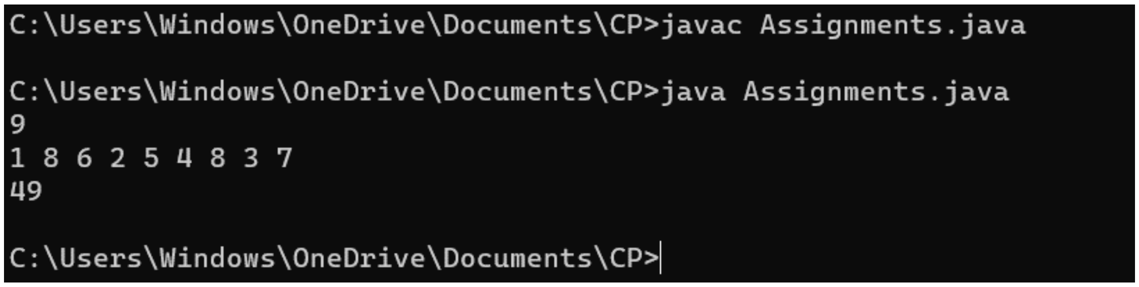
```
6
1 0 -1 0 -2 2
Target:0
Output :
[-2, 0, 0, 2]
[-1, 0, 0, 1]
[-2, -1, 1, 2]
```

8) Container with Most Water:

Code:

```
import java.util.*;
public class Assignments{
        public static int max(int a[],int len){
         int l=0;
         int r=len-1;
         int area=0;
         while(l<r){
                area=Math.max(area,Math.min(a[l],a[r])*(r-l));
                if(a[l]<a[r])
                        l+=1;
                else
                        r-=1;
         }
         return area;
         }
        public static void main(String [] args){
         Scanner sc=new Scanner(System.in);
         System.out.print("");
         int n=sc.nextInt();
         int arr[]=new int[n];
         for(int i=0;i<n;i++){
                arr[i]=sc.nextInt();
         }
         System.out.print(""+ max(arr,n)+"\n" );
         }
}
```

Output:

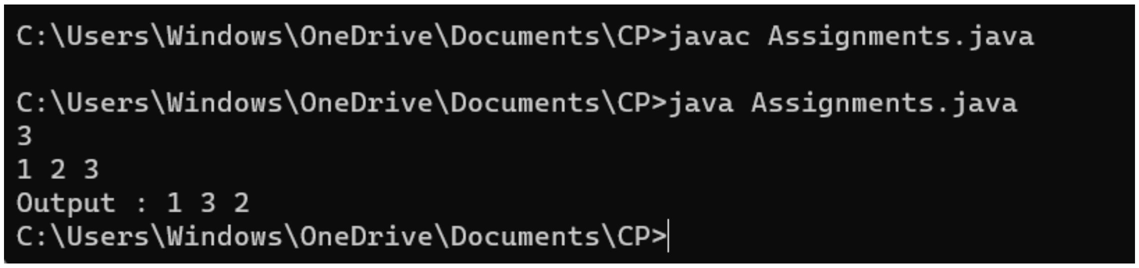


```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
9
1 8 6 2 5 4 8 3 7
49

C:\Users\Windows\OneDrive\Documents\CP>
```

9) Next Permutation:

Code:

```
import java.util.*;
public class Assignments{
        static void perm(int[] arr){
         int n=arr.length,i,j;
         for(i=n-2;i>=0;i--){
                if(arr[i]<arr[i+1]){
                        break;
                }
         }
         if(i<0){
                reverse(arr,0,arr.length-1);
         }
         else{
                for(j=n-1;j>i;j--){
                        if(arr[j]>arr[i]){
                                break;
                        }
                }
                swap(arr,i,j);
                reverse(arr,i+1,arr.length-1);
         }
        }
        static void reverse(int[] arr,int start,int end){
                while(start<end){
                 swap(arr,start,end);
                 start++;
                 end--;
                }
        }
         static void swap(int[] arr,int i,int j){
                  int temp=arr[i];
                  arr[i]=arr[j];
                  arr[j]=temp;
         }
        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();
                 }
                 perm(arr);
                 System.out.print("Output : ");
                 for (int i : arr) {
                         System.out.print(i + " ");
                 }
        }
}
```



```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
3
1 2 3
Output : 1 3 2
C:\Users\Windows\OneDrive\Documents\CP>
```

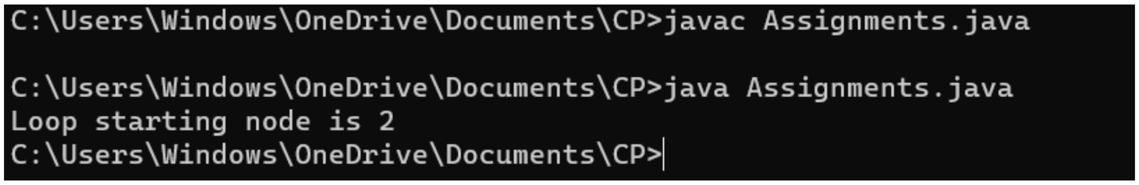
10) Linked List Cycle II:

Code:

```
public class Assignments {
        static class Node {
                int key;
                Node next;
        };

        static Node newNode(int key) {
                Node temp = new Node();
                temp.key = key;
                temp.next = null;

                return temp;
        }

        static void printList(Node head) {
                while (head != null) {
                    System.out.print(head.key + " ");
                    head = head.next;
                }
                System.out.println();
        }

        public static Node detectCycle(Node head) {
         Node slow = head, fast = head;
         while (fast != null && fast.next != null) {
             slow = slow.next;
             fast = fast.next.next;
             if (slow == fast) break;
         }

         if (fast == null || fast.next == null) return null;

         while (head != slow) {
             head = head.next;
             slow = slow.next;
         }
         return head;
        }

        public static void main(String[] args) {
                Node head = newNode(3);
                head.next = newNode(2);
                head.next.next = newNode(0);
                head.next.next.next = newNode(-4);

                head.next.next.next.next = head.next;
                Node res = detectCycle(head);
```

```
                if (res == null)
                        System.out.print("Loop does not exist");
                else
                        System.out.print("Loop starting node is " + res.key);
                }
}
```

Output:



```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
Loop starting node is 2
C:\Users\Windows\OneDrive\Documents\CP>
```

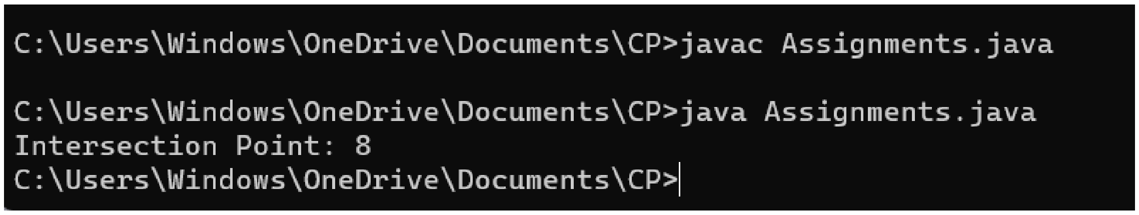
11) Intersection of Two Linked Lists:

Code:

```
public class Assignments {
        static class Node {
                int data;
                Node next;
                Node(int d){
                        data = d;
                        next = null;
                }
        }
        public Node getIntersectionNode(Node head1, Node head2) {
                while (head2 != null) {
                        Node temp = head1;
                        while (temp != null) {
                                if (temp == head2) {
                                        return head2;
                                }
                                temp = temp.next;
                        }
                        head2 = head2.next;
                }
                return null;
        }
```

```
	public static void main(String[] args){
		Assignments list = new Assignments();
		Node head1, head2;

		head1 = new Node(4);
		head2 = new Node(5);
		Node newNode = new Node(6);
		head2.next = newNode;
		newNode = new Node(1);
		head1.next = newNode;
		newNode = new Node(1);
		head2.next.next = newNode;
		newNode = new Node(8);
		head1.next.next = newNode;

		head2.next.next.next = newNode;
		newNode = new Node(4);

		head1.next.next.next = newNode;
		head1.next.next.next.next = null;

		Node intersectionPoint = list.getIntersectionNode(head1,
head2);
		if (intersectionPoint == null) {
			System.out.print(" No Intersection Point \n");
		}
		else {
			System.out.print("Intersection Point: " +
intersectionPoint.data);
		}
	}
}
```

Output:



```
C:\Users\Windows\OneDrive\Documents\CP>javac Assignments.java

C:\Users\Windows\OneDrive\Documents\CP>java Assignments.java
Intersection Point: 8
C:\Users\Windows\OneDrive\Documents\CP>
```