Return Array Sum

Send Feedback

Given an array/list(ARR) of length N, you need to find and return the sum of all the elements in the array/list.

Input Format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

The first line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Output Format :

For each test case, print the sum of the numbers in the array/list.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^5

Time Limit: 1sec

Sample Input 1:

```
1
3
9 8 9
```

Sample Output 1:

```
26
```

```
import java.util.Arrays;

import java.util.Scanner;

public class Solution {

                public static int sum(int[] arr) {

                //Your code goes here

          int n=arr.length;

     int i,s=0;

     for(i=0;i<n;i++)

       s=s+arr[i];

     return s;

   }

}
```

Linear Search

Send Feedback

You have been given a random integer array/list(ARR) of size N, and an integer X. You need to search for the integer X in the given array/list using 'Linear Search'.
 You have been required to return the index at which X is present in the array/list. If X has multiple occurrences in the array/list, then you need to return the index at which the first occurrence of X would be encountered. In case X is not present in the array/list, then return -1.
'Linear search' is a method for finding an element within an array/list. It sequentially checks each element of the array/list until a match is found or the whole array/list has been searched.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Third line contains the value of X(integer to be searched in the given array/list)

Output format :

For each test case, print the index at which X is present or -1, otherwise.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^5
-2 ^ 31 <= X <= (2 ^ 31) - 1
Time Limit: 1 sec

Sample Input 1:

```
1
7
2 13 4 1 3 6 28
3
```

Sample Output 1:

```
4
```

```
import java.util.Scanner;

public class Solution {

    public static int[] takeInput(){

        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();

        }

        return arr;

    }

        public static void printArray(int []arr){

        for(int i=0;i<arr.length;i++){

            System.out.println(arr[i]+" ");

        }

    }

    public static int linearSearch(int arr[], int x) {

                            //Your code goes here

               boolean flag = false;

      int i;

      for(i = 0;i < arr.length;i++){

        if(x == arr[i])

        {

          flag = true;
          break;

        }     }

          if(flag == false)

        return -1;

      else

        return i;                        }

    public static void main(String[] args){

        Scanner sc = new Scanner(System.in);

        int [] arr = takeInput();

        int element = sc.nextInt();

        int result = linearSearch(arr,element);

        System.out.println(result);

    }       }
```

Arrange Numbers In Array

Send Feedback

You have been given an empty array(ARR) and its size N. The only input taken from the user will be N and you need not worry about the array.

Your task is to populate the array using the integer values in the range 1 to N(both inclusive) in the order - 1,3,5,.......,6,4,2.

Note:

You need not print the array. You only need to populate it.

Input Format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

The first and the only line of each test case or query contains an integer 'N'.

Output Format :

For each test case, print the elements of the array/list separated by a single space.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^4

Time Limit: 1sec

Sample Input 1 :

```
1
6
```

Sample Output 1 :

```
1 3 5 6 4 2
```

```
import java.util.Scanner;

public class Solution {

     public static int [] takeInput(){

        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();
    }
    return arr;
 }
 public static void printArray(int[] arr){
    for(int i=0;i< arr.length;i++){
        System.out.print(arr[i]+" ");
    }      }
    public static void arrange(int[] arr, int n) {
          //Your code goes here
    int val=1;
 int start = 0 ,end = n-1;
 while(start<=end){
    if(val%2==1){
 arr[start]=val;
 val++;
 start++;
}
    else{
        arr[end]=val;
 val++;
 end--;
    }       }
          }
        public static void main(String []args){
    Scanner sc = new Scanner(System.in);
    int []arr = takeInput();
    // arrange(arr, n);
                    System.out.print(arr+ " ");


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

Swap Alternate

Send Feedback

You have been given an array/list(ARR) of size N. You need to swap every pair of alternate elements in the array/list.

You don't need to print or return anything, just change in the input array itself.

Input Format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Output Format :

For each test case, print the elements of the resulting array in a single row separated by a single space.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^5
Time Limit: 1sec

Sample Input 1:

```
1
6
9 3 6 12 4 32
```

Sample Output 1 :

```
3 9 12 6 32 4
```

```
import java.util.Scanner;

public class Solution {

  public static int [] takeInput(){

    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();

    }

    return arr;

  }

  public static void printArray(int[] arr){

    for(int i=0;i< arr.length;i++){

      System.out.print(arr[i]+" ");

    }

  }

    public static void swapAlternate(int arr[]) {

        //Your code goes here

    for(int i=0;i<arr.length-1;i=i+2){

      int t = arr[i];
```

```
            arr[i] = arr[i+1];
            arr[i+1] = t;
        }
    }
    public static void main(String[] args) {
                        int[] arr = takeInput();
                        swapAlternate(arr);
                        printArray(arr);
          }
}  }
```

Find Unique

Send Feedback

You have been given an integer array/list(ARR) of size N. Where N is equal to [2M + 1].
Now, in the given array/list, 'M' numbers are present twice and one number is present only once.
You need to find and return that number which is unique in the array/list.

Note:

Unique element is always present in the array/list according to the given condition.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Output Format :

For each test case, print the unique element present in the array.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^3
Time Limit: 1 sec

Sample Input 1:

```
1
7
2 3 1 6 3 6 2
```

Sample Output 1:

```
1
```

```
import java.util.Scanner;
public class Solution{
    public static int findUnique(int[] arr){
                        //Your code goes here
        int k = 0;
        boolean [] visited = new boolean[arr.length];
        for (int i = 0; i <arr.length ; i++) {
            int x = arr[i];
            if(visited[i]==false) {
```

```
            boolean isDuplicate = false;

            for (int j = i + 1; j < arr.length; j++) {

               if (x == arr[j]) {

                  isDuplicate = true;

                  visited[j] = true;

               }             }

            if (!isDuplicate)

               k=x;

         }           }

      return k;

   }          }
```

Find Duplicate

Send Feedback

You have been given an integer array/list(ARR) of size N which contains numbers from 0 to (N - 2). Each number is present at least once. That is, if N = 5, the array/list constitutes values ranging from 0 to 3 and among these, there is a single integer value that is present twice. You need to find and return that duplicate number present in the array.

Note :

Duplicate number is always present in the given array/list.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Output Format :

For each test case, print the duplicate element in the array/list.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^3
Time Limit: 1 sec

Sample Input 1:

```
1
9
0 7 2 5 4 7 1 3 6
```

Sample Output 1:

```
7
```

```
import java.util.Scanner;

public class Solution{

      public static int duplicateNumber(int arr[]) {

            //Your code goes here

       int k=0;

   int count=0;

   for(int i=0;i<arr.length;i++)
```

```
  {
      int temp=arr[i];
      if(count==0)
      {
          for(int j=i+1;j<arr.length;j++)
          {
              if (temp==arr[j]){
              count=1;
              k=arr[i];
              break;}
                    }          }
        }
    return k;
  }
}
```

Intersection of Two Arrays II

Send Feedback

You have been given two integer arrays/list(ARR1 and ARR2) of size N and M, respectively. You need to print their intersection; An intersection for this problem can be defined when both the arrays/lists contain a particular value or to put it in other words, when there is a common value that exists in both the arrays/lists.

Note :

Input arrays/lists can contain duplicate elements.

The intersection elements printed would be in the order they appear in the first array/list(ARR1)

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the first array/list.

Second line contains 'N' single space separated integers representing the elements of the first the array/list.

Third line contains an integer 'M' representing the size of the second array/list.

Fourth line contains 'M' single space separated integers representing the elements of the second array/list.

Output format :

For each test case, print the intersection elements in a row, separated by a single space.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^5
0 <= M <= 10^5
Time Limit: 1 sec

Sample Input 1 :

```
2
6
2 6 8 5 4 3
4
2 3 4 7
```

```
2
10 10
1
10
```

Sample Output 1 :

```
2 4 3
10
```

```
import java.util.Scanner;

public class Solution{

    public static void intersections(int arr1[], int arr2[]) {

            //Your code goes here

        boolean visited[] = new boolean [arr2.length];

        for(int i=0;i<arr1.length;i++){

            int temp =arr1[i];

            for(int j=0;j<arr2.length;j++){

                if(visited[j]==false){

                    if(temp==arr2[j]){

                        visited[j]=true;

                        System.out.print(temp+" ");

                        break;

                    }               }           }       }       }           }
```

Pair Sum

Send Feedback

You have been given an integer array/list(ARR) and a number X. Find and return the total number of pairs in the array/list which sum to X.

Note:

Given array/list can contain duplicate elements.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the first array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Third line contains an integer 'X'.

Output format :

For each test case, print the total number of pairs present in the array/list.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^3
0 <= X <= 10^9
Time Limit: 1 sec

Sample Input 1:

```
1
9
1 3 6 2 5 4 3 2 4
7
```

Sample Output 1:

```
7
```

```
import java.util.Scanner;

public class Solution {

    public static int pairSum(int arr[], int x) {

            //Your code goes here

        int count=0;

         int n = arr.length;

        for(int i=0; i<n-1; i++)

        {

            for(int j=i+1; j<n;j++)



                if(arr[i]+arr[j]==x)

                    count++;

        }

              return count;

                                   }      }
```

Triplet Sum

Send Feedback

You have been given a random integer array/list(ARR) and a number X. Find and return the number of triplets in the array/list which sum to X.

Note :

Given array/list can contain duplicate elements.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the first array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Third line contains an integer 'X'.

Output format :

For each test case, print the total number of triplets present in the array/list.

Output for every test case will be printed in a separate line.

Constraints :

```
1 <= t <= 50
0 <= N <= 10^2
0 <= X <= 10^9
Time Limit: 1 sec
```

Sample Input 1:

```
1
7
1 2 3 4 5 6 7
12
```

Sample Output 1:

```
5
```

```
import java.util.Scanner;

import java.util.Arrays;

public class Solution {
    public static int findTriplet(int[] arr, int x) {

        //Your code goes here

     int count=0;

     for (int i = 0; i < arr.length; i++) {


         for (int j = i+1; j < arr.length; j++) {

             for (int k = j+1; k < arr.length; k++) {

                 if (i!=j && ((arr[i]+arr[j]+arr[k])==x)){

                     count++;

                 }           }        }      }

     return (count);

   }

}
```

Sort 0 1

Send Feedback

You have been given an integer array/list(ARR) of size N that contains only integers, 0 and 1. Write a function to sort this array/list. Think of a solution which scans the array/list only once and don't require use of an extra array/list.

Note:

You need to change in the given array/list itself. Hence, no need to return or print anything.

Input format :

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers(all 0s and 1s) representing the elements in the array/list.

Output format :

For each test case, print the sorted array/list elements in a row separated by a single space.

Output for every test case will be printed in a separate line.

Constraints :

1 <= t <= 10^2
0 <= N <= 10^5
Time Limit: 1 sec

Sample Input 1:

```
1
7
0 1 1 0 1 0 1
```

Sample Output 1:

```
0 0 0 1 1 1 1
```

```
import java.util.Scanner;

public class Solution {

    public static void sortZeroesAndOne(int[] arr) {

        //Your code goes here

     int count=0;

              for(int i=0;i<arr.length;i++)
    {
        if(arr[i]==0)
        {
            count++;
        }
    }
    for(int i=0;i<count;i++)
    {
        arr[i]=0;
    }
    for(int i=count;i<arr.length;i++)
    {
        arr[i]=1;
    }
}
  }
```