# Triplet Sum

## Description

Given an array of integers, find if there exists two numbers in the array whose sum is equal to a third number c, which is also present in the array.

i.e. a + b = c

If there exist any such triplet in the array output 1 else output 0.

Note: a,b,c all need to be at different indices, i.e. you cannot use any element twice

## Input

The first line of input will contain an integer T, denoting the number of test cases

The first line of each test case will contain an integer N, denoting the length of the array

The next line will contain N array elements.

Constraints:
1<= T <= 10

1<= N <= 10^3

1<= array[i] <= 10^6

## Sample Input 1

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

## Sample Output 1

```
1
```

1 ✓3 2 4 5

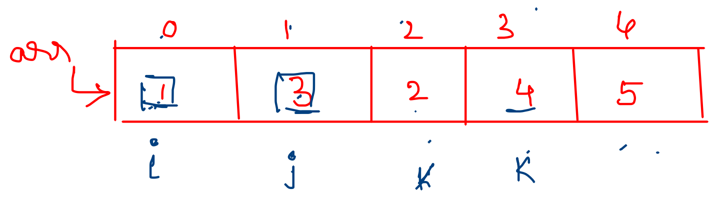
## Hint

In Sample 1:

T = 1, N = 5

1 + 3 = 4 which is also present in the array , So output is 1

AP1 :- a, b, c



n = 5

3 - Nested loops $\Rightarrow O(n^3)$

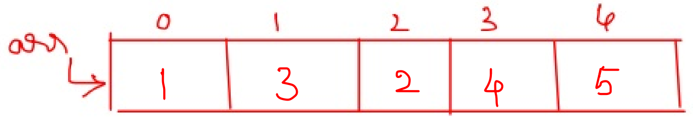
TLE

Ap2 :- 2-ptr Tech. $a+b=c$

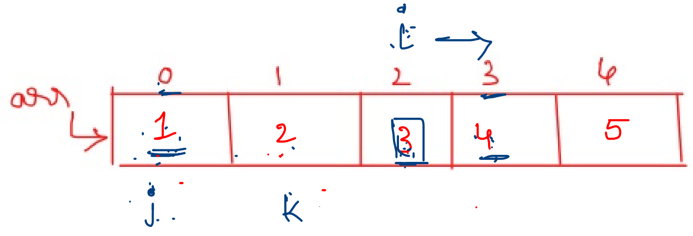
```
function fun(arr[],n)
{
        1. sort the array        ⇒ i : loop-var
n ↳     for(i=0;i<n;i++)
        {
                j=0
                k=i+1
                while(j<k)
                {
                        if(arr[i]==arr[j]+arr[k] && i!=j && i!=k)
                        {
                                return 1
                        }
                        else if(arr[i]>arr[j]+arr[k] && i!=j && i!=k)
                        {
                                j++
                        }
                        else
                                k--
                }
        }
        return 0
}
```

$i=0, j=0, k=i+1$

↳ 0 to n

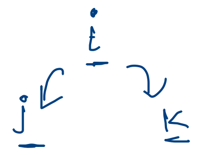


⤸ in-built sort() ✓



⇒ Sorted

3 3



$j != k$ ✗

$n^2 + \underbrace{n \cdot \log_2 n}$ ⇒ $O(n^2)$ ✓

↳ Merge sort

✓ arr[ ] = { 1, 2, 3, 4, 5 }

$\overrightarrow{\text{arr.includes( 3 )}}$ ⇒ True ⇒ O(n)

o/p

Console.log(arr) ⇒ 1, 2, 3, 4, 5

↳ loop

↳ ~~O(1)~~ ✓

↳ O(n) ✓

a = 10

console.log(a) ⇒ O(1)

# Encrypted Sequence

-37:10:48 Edit

## Description

Jack gave Jill a secret code, a sequence of numbers. Jill decided to encrypt the code. The encryption used by Jill is as follows. ✓

Lets say the sequence of N numbers is a1, a2, a3, a4, ...... an-1.Jillwill encrypt it to [a1,a3,a5.... a6,a4,a2].

Jill mistakenly shared encrypted code with Romeo, but she is confident that Romeo won't be able to crack it. Can you help Romeo to get the secret code from the encrypted sequ ence?

$n = 10$ ✓

$a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad a_7 \cdots a_{10}$

$\downarrow$

$a_1 \quad a_3 \quad a_5 \quad a_7 \cdots a_6 \quad a_4 \quad a_2]$

$\rightarrow \qquad \leftarrow$

## Input

First-line contains T, no of test cases.

First-line contains N, no of numbers in sequence.

Second-line contains N spaced numbers, Encrypted code (sequence of numbers) by Jill.

**Contraints**

1 <= T <= 10

1 <= N <= 10^6

1 <= A[i] <= 10^6

## Output

For each test output the original secret code, (sequence of numbers) on new line.

## Sample Input 1

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

## Sample Output 1

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

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ |
|---|---|---|---|---|
| 1 | 3 | 5 | 4 | 2 |
| $\cancel{1}$ | $\cancel{1}$ | 1 | $\cancel{0}$ | $\cancel{0}$ |
|  |  | 0 |  |  |
| 1 | 2 | 3 | 4 | 5 |

| | | | | | |
|---|---|---|---|---|---|
| 1 | 3 | 5 | 6 | 4 | 2 |
| 1 | 2 | 3 | 4 | 5 | 6 |

```
function fun(arr[],n)
{
        l=0,r=n-1,res=[]
        while(l<=r)
        {
                if(res.length<n) ✓
                {
                        res.push(arr[l]) ✓

                }                        ⇒ ✗
                if(res.length<n)
                {                         ✓ ✓
                        res.push(arr[r])
                }
                l++
                r--
        }
        console.log(res)
}
```

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 4 | 9 | 3 | 6 | 8 | 7 |
| ~~l~~ | ~~l~~ | ~~l~~ | ~~l~~ ~~r~~ | ~~r~~ | ~~r~~ | ~~r~~ |

$n=7$

✓ res: [1 7 4 8 9 6 3] ⇒ 7

✓ Sum of Subarray

● -36:58:49 Edit

## Description

You are given an array of size N. You will given Q queries. Each query has L and R such that 0<=L<=R<=N-1. You need find sum from L to R for each Query.

## Input

First line contains N, size of array.

Second line contains N space separated integers, which elements of array.

Third line contains Q, number of queries.

Next Q lines contains 2 space integers L and R.

**Constraints**

1 <= N,Q <= 10^6

1 <= A[i] <= 1000

① 1 to 3 : $\underbrace{3+2}+1 = 6$ ✓

② 2 to 4 : $2+1+5 = 8$

③ 1 to 4 : $3+2+1+5 = 11$ ✓

④ 3 to 3 : 3 ✓

**Sample Input 1**

```
4
3 2 1 5
4
1 3 ✓
2 4 ✓
1 4
3 3
```

$n = 4$

**Sample Output 1**

```
6 ✓
8 ✓
11
1
```

$$a \hookrightarrow \begin{array}{|c|c|c|c|} 0 & 1 & 2 & 3 \\ \hline 3 & 2 & 1 & 5 \\ \hline \end{array}$$

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 5 \end{array}$$

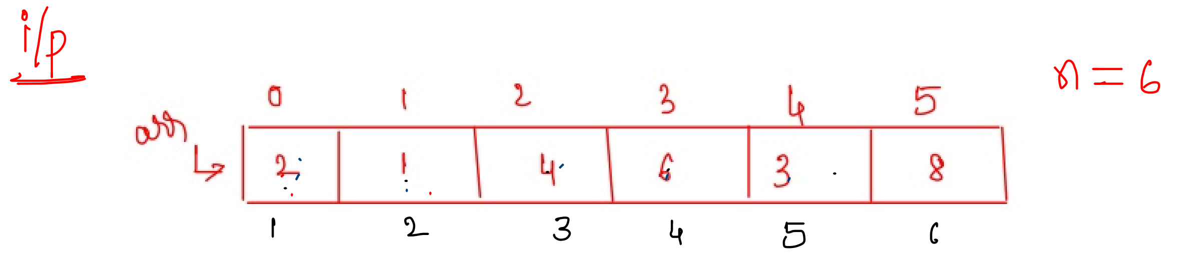
$\Rightarrow \quad b[0] = 0$

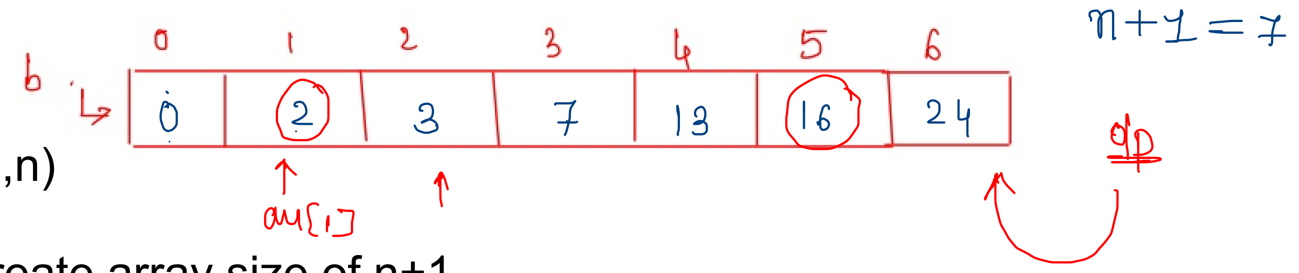
$$\underline{b} \hookrightarrow \begin{array}{|c|c|c|c|c|} 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 3 & 5 & 6 & 11 \\ \hline \end{array}$$

$\underline{n+1} = 5$

$$\begin{array}{cc} \underline{L} & \underline{R} \\ 1 & 3 \end{array} \Rightarrow$$

```
for(i=1 ; i≤n ; i++)
{
    b[i] = b[i-1] + a[i-1]
}
```





```
function fun(arr[],n)
{
        b[n+1] // create array size of n+1
        b[0]=0
        for(i=1;i<=n;i++)
        {
                b[i]=arr[i-1]+b[i-1]
        }
        for(i=1;i<=q;i++)
        {
                read l,r
                print(b[r]-b[l-1])
        }
}
```

Question

1) L R
2 5 $\Rightarrow 1+4+6+3=14$

2) L R
3 6 $\Rightarrow 4+6+3+8$
$= 21$

3) L R
1 3 $\Rightarrow 2+1+4$
$= 7$

① $b[5] - b[1] = 16-2 = 14$

② $b[6] - b[2] = 24 - 3 = 21$

$b[r] - b[l-1]$ ✓

# First Negative Integer

-36:38:6 Edit

## Description

Given an array A containing N space-separated integers. Find the first negative integer for each and every window(contiguous subarray) of sizeK.

## Input

**Input Format**

First-line contains T, no of test cases.

First-line of each test case contains N, the size of the array, and an integer K.

Second-line of each test case contains N spaced integers, elements of an array A.

**Constraints**

1 <= T <= 10

1 <= N <= 10^5

0 <= abs(A[i]) <= 10^5

1 <= K <= N

## Output

For each test case, print N-K+1 space-separated integers in a new line.

## Output

$n = 8$

For each test case, print N-K+1 space-separated integers in a new line.

```
for(i=0; i< n-k+1; i++)
{
```

## Sample Input 1

$\underline{K=3}$ (fixed)

```
2
5 2
-8 2 3 -6 10
8 3
12 -1 -7 8 -15 30 16 28 ✓
```

$i=0$ : 0 to $\underline{2}$

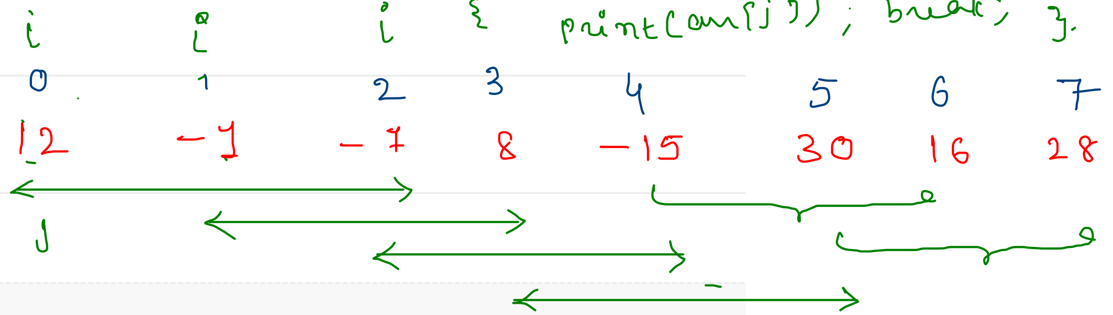
$i=1$ : 1 to $\underline{3}$

$i=2$ : 2 to $\underline{4}$

## Sample Output 1

```
-8 0 -6 -6
-1 -1 -7 -15 -15 0
```

```
for(j = i ; j ≤ i+k-1; j++)
{
    if(arr[j] < 0)
    {  print(arr[j]) ; break; }.
```



## Hint

For the first test case,

```
First negative integer for each window of size 2
{-8, 2} = -8
{2, 3} = 0 (does not contain a negative integer)
{3, -6} = -6
{-6, 10} = -6
```

$\underline{o/p}$ -1 -1 -1 -15, -15, $\underline{0}$

```
for( j=i; j ≤ i+k-1 ; j++ )
{
```

# Subarrays Having Sum Less Than M

$n \Rightarrow \frac{n(n+1)}{2} \Rightarrow 5 : \frac{5 \times \cancel{6}^{3}}{\cancel{2}} = 15$

## Description

Given an array A of size n with positive numbers, find the total number of subarrays that have sum < m.

(sum) $M=5$

$n=5$

## Input

The first line of the input contains one integer t (1 ≤ t ≤ 10) — the number of test cases. Then t test cases follow.

The first line of each test case contains a single integer n (1 ≤ n ≤ 100000) and M as mentioned in the question.

The second line of the test case contains n integers (1 ≤ Ai ≤ 100).

$$\text{arr} \rightarrow \begin{array}{|c|c|c|c|c|} \multicolumn{1}{c}{0} & \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{3} & \multicolumn{1}{c}{4} \\ \hline 1 & 5 & 1 & 3 & 2 \\ \hline \end{array}$$

## Output

For each test case, print the answer: The number of subarrays.

## Sample Input 1

```
1
5 5
1 5 1 3 2
```

## Sample Output 1

```
5
```
✓

```
function fun(arr,n,m)
{
        i=0,j=0,sum=0,count=0
        while(j<n)
        {
                sum=sum+arr[j]
                while(sum>=m)
                {
                        sum=sum-arr[i]
                        i++
                }
                count=count+j-i+1
                j++
        }
        return count
}
```

$m=5$

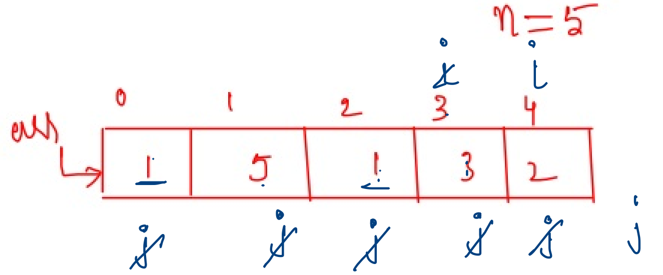
$i=0$

$j=0$

$<\underline{5}$ we want

(sum) $M=\underline{5}$ (given sum)

$n=5$



sum $= \cancel{0}\ \cancel{1}\ \cancel{6}\ \cancel{5}\ \cancel{0}\ \cancel{1}\ \cancel{4}\ \cancel{6}\ \cancel{5}\ 2$

count $= \cancel{0}\ \cancel{1}\ \cancel{2}\ \cancel{4}\ \underline{5}$

$j=1$

$i=2$

$-\cancel{1}+\cancel{1}=0$