

## KISHORE SIVAKUMAR...

@11249a163

### Complete your profile

Add your missing details →

This data will be helpful to auto-fill your job applications

0%

## Personal Information

6

✉ 11249a163@kanchiuniv.ac.in

 Add your mobile number

② India

## My Badges

You have not unlocked any badges yet. [Get Badges](#)

## My Certifications

You have not earned any certificates yet. [Get Certified](#)



**Returns**

- `int[n]`: the rotated array

**Input Format**

The first line contains two space-separated integers that denote  $n$ , the number of integers, and  $d$ , the number of left rotations to perform.

The second line contains  $n$  space-separated integers that describe `arr[]`.

**Constraints**

- $1 \leq n \leq 10^5$
- $1 \leq d \leq n$
- $1 \leq a[i] \leq 10^6$

**Sample Input**

STDIN	Function
-----	-----
5 4	$n = 5$ $d = 4$
1 2 3 4 5	<code>arr = [1, 2, 3, 4, 5]</code>

**Sample Output**

5 1 2 3 4

**Explanation**

To perform  $d = 4$  left rotations, the array undergoes the following sequence of changes:

$[1, 2, 3, 4, 5] \rightarrow [2, 3, 4, 5, 1] \rightarrow [3, 4, 5, 1, 2] \rightarrow [4, 5, 1, 2, 3] \rightarrow [5, 1, 2, 3, 4]$

26

Line: 26 Col: 1

 Upload Code as File Test against custom input**Run Code****Submit Code**

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

**Sample Test case 0**

Input (stdin)

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

[Download](#)

Your Output (stdout)

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

Expected Output

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

[Download](#)

HackerRank | Prepare > Data Structures > Arrays 2D Array - DS

Sample Input

```
1 1 1 0 0 0  
0 1 0 0 0 0  
1 1 1 0 0 0  
0 0 2 4 4 0  
0 0 0 2 0 0  
0 0 1 2 4 0
```

34

Line: 34 Col: 1

Upload Code as File  Test against custom input

Run Code Submit Code

Sample Output

```
19
```

Explanation

`arr` contains the following hourglasses:

```
1 1 1 1 1 0 1 0 0 0 0 0 0  
1 0 0 0 0 0 0 0 0 0 0 0 0  
1 1 1 1 1 0 1 0 0 0 0 0 0  
0 1 0 0 0 0 0 0 0 0 0 0 0  
0 1 0 0 0 0 0 0 0 0 0 0 0  
0 0 2 4 4 4 4 4 4 4 4 4 4  
1 1 1 1 1 0 1 0 0 0 0 0 0  
0 1 0 0 0 0 0 0 0 0 0 0 0  
0 0 2 4 4 4 4 4 4 4 4 4 4  
0 0 0 2 0 0 0 0 0 0 0 0 0  
0 0 0 2 0 0 0 0 0 0 0 0 0  
0 0 1 2 4 4 4 4 4 4 4 4 4  
0 0 1 2 4 4 4 4 4 4 4 4 4
```

The hourglass with the maximum sum (19) is:

```
2 4 4  
2  
1 2 4
```

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

Download

Sample Test case 1

1 1 1 0 0 0  
0 1 0 0 0 0

Sample Test case 2

1 1 1 0 0 0  
0 0 2 4 4 0  
0 0 0 2 0 0  
0 0 1 2 4 0

Your Output (stdout)

19

Expected Output

19

Download

Pro Submissions Leaderboard Discussions Editorial

HackerRank | Prepare Algorithms > Warmup Simple Array Sum

For example, if the array  $ar = [1, 2, 3]$ ,  $1 + 2 + 3 = 6$ , so return 6.

**Function Description**

Complete the *simpleArraySum* function with the following parameter(s):

- $ar[n]$ : an array of integers

**Returns**

- *int*: the sum of the array elements

**Input Format**

The first line contains an integer,  $n$ , denoting the size of the array.  
The second line contains  $n$  space-separated integers representing the array's elements.

**Constraints**

$0 < n, ar[i] \leq 1000$

**Sample Input**

STDIN	Function
-----	-----
6	ar[] size n = 6
1 2 3 4 10 11	ar = [1, 2, 3, 4, 10, 11]

**Sample Output**

31

Line: 20 Col: 1

Upload Code as File  Test against custom input [Run Code](#) [Submit Code](#)

**Congratulations!**

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

[Sample Test case 0](#) [Download](#)

Input (stdin)		Your Output (stdout)		Expected Output	
1	6	1	31	1	31
2	1 2 3 4 10 11				

[Download](#)

Print the sum of the array's elements:  $1 + 2 + 3 + 4 + 10 + 11 = 31$ .

HackerRank | Prepare Algorithms > Warmup Diagonal Difference

Exit Full Screen View

Problem

STDIN Function

```
-----  
3 arr[][] sizes n = 3, m = 3  
11 2 4 arr = [[11, 2, 4], [4, 5, 6], [10, 8, -12]]  
4 5 6  
10 8 -12
```

22 secondary += arr[i][j];  
23 }  
24 }  
25 }  
26 cout << abs(primary - secondary) << endl;

Line: 31 Col: 1

Sample Output

```
15
```

Upload Code as File  Test against custom input Run Code Submit Code

Explanation

The primary diagonal is:

```
11  
5  
-12
```

Sum across the primary diagonal:  $11 + 5 - 12 = 4$ .

The secondary diagonal is:

```
4  
5  
10
```

Sum across the secondary diagonal:  $4 + 5 + 10 = 19$

Difference:  $|4 - 19| = 15$

Note:  $|x|$  is the absolute value of  $x$ .

Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0 (selected) Download

Input (stdin)

```
1 3  
2 11 2 4  
3 4 5 6  
4 10 8 -12
```

Your Output (stdout)

```
1 15
```

Expected Output Download

12  
21

97

Line: 97 Col: 1

**Sample Output 0**

2

 Upload Code as File Test against custom input**Run Code****Submit Code****Explanation 0**

The distance between points  $(1, 2)$  and  $(2, 1)$  is  $\rho(1, 2) + \rho(2, 1) = 2$ .

**Sample Input 1**73  
12  
23  
34  
45  
56  
67  
36  
45  
55**Sample Output 1**

3

**Explanation 1**

The best points are  $(3, 6)$  and  $(5, 5)$ , which gives us a distance of  $\rho(3, 5) + \rho(6, 5) = 2 + 1 = 3$ .

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

 **Sample Test case 0**

Input (stdin)

[Download](#) **Sample Test case 1**

1	<b>2</b>	<b>2</b>
2	<b>1</b>	<b>2</b>
3	<b>1</b>	<b>2</b>
4	<b>2</b>	<b>1</b>

Your Output (stdout)

1 **2**

Expected Output

1 **2**[Download](#)

Pro

HackerRank | Prepare Algorithms Warmup A Very Big Sum

Exit Full Screen View

Return  
• *long*: the sum of the array elements

Input Format  
The first line of the input consists of an integer *n*.  
The next line contains *n* space-separated integers contained in the array.

Output Format  
Return the integer sum of the elements in the array.

Constraints  
 $1 \leq n \leq 10$   
 $0 \leq ar[i] \leq 10^{10}$

Sample Input

STDIN	Function
-----	-----
5	arr[] size n = 5
1000000001 1000000002 1000000003 1000000004 1000000005	arr[.

Output  
5000000015

Note:  
The range of the 32-bit integer is  
 $(-2^{31})$  to  $(2^{31} - 1)$  or  $[-2147483648, 2147483647]$ .

When we add several integer values, the resulting sum might exceed the above range. You might need to use long int C/C++/Java to store such sums.

Upload Code as File  Test against custom input

Run Code Submit Code

Congratulations!  
You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin) Download

1	5
2	1000000001 1000000002 1000000003 1000000004 1000000005

Your Output (stdout)

1	5000000015
---	------------

Expected Output Download

1	5000000015
---	------------

**HackerRank** | Prepare Data Structures Arrays Arrays - DS Exit Full Screen View ↗

**Problem**

An array is a data structure that stores elements of the same type in a contiguous block of memory. In an array,  $A$ , of size  $N$ , each memory location has some unique index,  $i$  (where  $0 \leq i < N$ ), that can be referenced as  $A[i]$  or  $A_i$ .

Your task is to reverse an array of integers.

**Note:** If you've already solved our C++ domain's Arrays Introduction challenge, you may want to skip this.

**Example**

$A = [1, 2, 3]$

Return  $[3, 2, 1]$ .

**Function Description**

Complete the function `reverseArray` with the following parameter(s):

- `int A[n]`: the array to reverse

**Returns**

- `int[n]`: the reversed array

**Input Format**

The first line contains an integer,  $N$ , the number of integers in  $A$ .  
The second line contains  $N$  space-separated integers that make up  $A$ .

**Constraints**

- $1 \leq N \leq 10^3$
- $1 \leq A[i] \leq 10^4$ , where  $A[i]$  is the  $i^{th}$  integer in  $A$

**Sample Input 1**

3  
1 2 3

[Copy](#) [Download](#)

**Code Editor**

Line: 25 Col: 1

Test against custom input [Run Code](#) [Submit Code](#)

 You have earned 10.00 points!  
You are now 20 points away from the 1st star for your problem solving badge. 33% 10/30

**Congratulations**

You solved this challenge. Would you like to challenge your friends? [f](#) [t](#) [in](#) [Next Challenge](#)

**Test cases**

<input checked="" type="radio"/> Test case 0 	Compiler Message
<input checked="" type="radio"/> Test case 1 	Success
<input checked="" type="radio"/> Test case 2 	
<input checked="" type="radio"/> Test case 3 	

**Hidden Test Case**

Unlock this testcase for 5 hackos.

the number of queries.

Each of the next  $q$  lines contains three space-separated integers  $a$ ,  $b$  and  $k$ , the left index, right index and number to add.

#### Constraints

- $3 \leq n \leq 10^7$
- $1 \leq m \leq 2 * 10^5$
- $1 \leq a \leq b \leq n$
- $0 \leq k \leq 10^9$

#### Sample Input

STDIN	Function
-----	-----
5 3	arr[] size n = 5, queries[] size q = 3
1 2 100	queries = [[1, 2, 100], [2, 5, 100], [3, 4, 100]]
2 5 100	
3 4 100	

#### Sample Output

200

#### Explanation

After the first update the list is 100 100 0 0 0.

After the second update list is 100 200 100 100 100.

After the third update list is 100 200 200 200 100.

The maximum value is 200.

36

Line: 35 Col: 1

Upload Code as File

Test against custom input

Run Code

Submit Code

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

#### Sample Test case 0

Input (stdin)

Download

1 5 3

2 1 2 100

3 2 5 100

4 3 4 100

Your Output (stdout)

1 200

Expected Output

1 200

## Sample Input 3

[Copy](#) [Download](#)

30

Line: 30 Col: 1

```
13  
abcde  
sdaklfj  
asdjf  
na  
basdn  
sdaklfj  
asdjf  
na  
asdjf  
na  
basdn  
sdaklfj  
asdjf  
5  
abcde  
sdaklfj  
asdjf  
na  
basdn
```

abcde	sdaklfj	asdjf	na	basdn
-------	---------	-------	----	-------

Array: queries

[Upload Code as File](#) Test against custom input[Run Code](#)[Submit Code](#)

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

### Sample Test case 0

Input (stdin)

[Download](#)

```
1 4  
2 aba  
3 baba  
4 aba  
5 zxzb  
6 3  
7 aba  
8 zxzb  
9 ab
```

### Sample Test case 1

### Sample Test case 2

Your Output (stdout)

```
1 2
```

## Sample Output 3

```
1  
3  
4  
3  
2
```

Print the ratios of positive, negative and zero values in the array. Each value should be printed on a separate line with 6 digits after the decimal. The function should not return a value.

#### Input Format

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

The second line contains  $n$  space-separated integers that describe  $arr[n]$ .

#### Constraints

$0 < n \leq 100$

$-100 \leq arr[i] \leq 100$

#### Sample Input

STDIN	Function
-----	-----
6	arr[] size n = 6
-4 3 -9 0 4 1	arr = [-4, 3, -9, 0, 4, 1]

#### Sample Output

```
0.500000
0.333333
0.166667
```

#### Explanation

There are 3 positive numbers, 2 negative numbers, and 1 zero in the array.

The proportions of occurrence are positive:  $\frac{3}{6} = 0.500000$ , negative:  $\frac{2}{6} = 0.333333$  and zeros:  $\frac{1}{6} = 0.166667$ .

28

 Upload Code as File

 Test against custom input

 Run Code

 Submit Code

Line: 28 Col: 1

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

#### Sample Test case 0

Input (stdin)

Download

```
1 6
2 -4 3 -9 0 4 1
```

Your Output (stdout)

```
1 0.500000
2 0.333333
3 0.166667
```

Expected Output

```
1 0.500000
2 0.333333
3 0.166667
```

Download

### Explanation 0

In this example:

- $a = (a[0], a[1], a[2]) = (5, 6, 7)$
- $b = (b[0], b[1], b[2]) = (3, 6, 10)$

Now, let's compare each individual score:

- $a[0] > b[0]$ , so Alice receives 1 point.
- $a[1] = b[1]$ , so nobody receives a point.
- $a[2] < b[2]$ , so Bob receives 1 point.

Alice's comparison score is 1, and Bob's comparison score is 1. Thus, we return the array  $[1, 1]$ .

### Sample Input 1

```
17 28 30  
99 16 8
```

### Sample Output 1

```
21
```

### Explanation 1

Comparing the  $0^{th}$  elements,  $17 < 99$  so Bob receives a point.

Comparing the  $1^{st}$  and  $2^{nd}$  elements,  $28 > 16$  and  $30 > 8$  so Alice receives two points.

The return array is  $[2, 1]$ .

32

Line: 32 Col: 1

Upload Code as File

Test against custom input

Run Code

Submit Code

## Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

### Sample Test case 0

Input (stdin)

Download

```
1 5 6 7  
2 3 6 10
```

### Sample Test case 1

Your Output (stdout)

```
1 1
```

Expected Output

```
1 1
```

Download

HackerRank | Prepare Data Structures Arrays Dynamic Array

Exit Full Screen View

Initial Values:

`n = 2`

`lastAnswer = 0`

`arr[0] = []`

`arr[1] = []`

Query 0: Append 5 to `arr[( (0 ⊕ 0) % 2 )] = arr[0]`.

`lastAnswer = 0`

`arr[0] = [5]`

`arr[1] = []`

Query 1: Append 7 to `arr[( (1 ⊕ 0) % 2 )] = arr[1]`.

`arr[0] = [5]`

`arr[1] = [7]`

Query 2: Append 3 to `arr[( (0 ⊕ 0) % 2 )] = arr[0]`.

`lastAnswer = 0`

`arr[0] = [5, 3]`

`arr[1] = [7]`

Query 3: Assign the value at index 0 of `arr[( (1 ⊕ 0) % 2 )] = arr[1]` to `lastAnswer`. Store `lastAnswer` in your answer array. `lastAnswer = 7`

`arr[0] = [5, 3]`

`arr[1] = [7]`

Query 4: Assign the value at index 1 of `arr[( (1 ⊕ 7) % 2 )] = arr[0]` to `lastAnswer`. Store `lastAnswer` in your answer array. `lastAnswer = 3`

`arr[0] = [5, 3]`

`arr[1] = [7]`

Return your answer array [7, 3]. The code stub prints its elements on separate lines.

31

Line: 31 Col: 1

Upload Code as File  Test against custom input

Run Code Submit Code

**Congratulations!**

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

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

Download

Your Output (stdout)

1 7  
2 3

Expected Output

Download