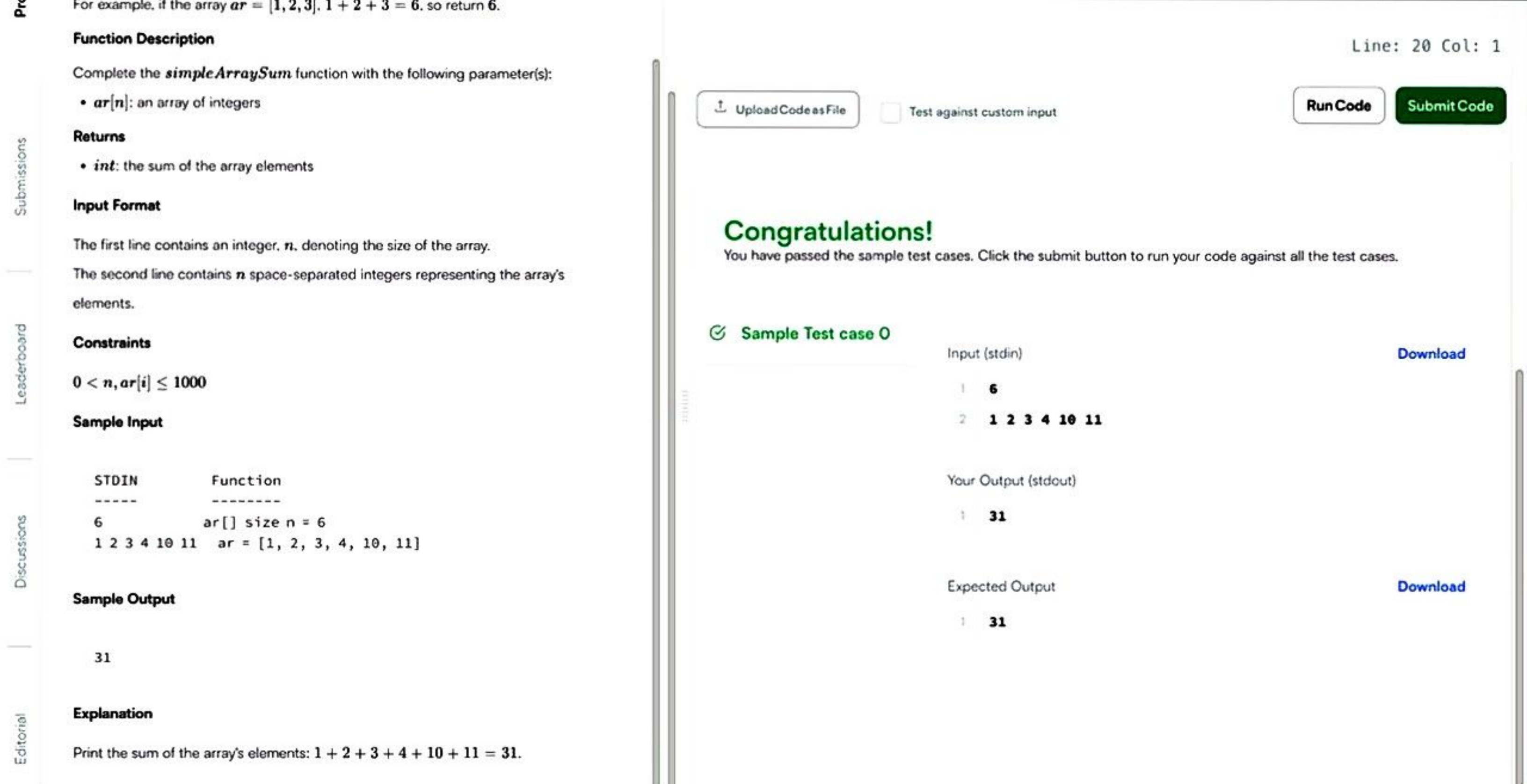


For example, if the array $ar = [1, 2, 3]$, $1 + 2 + 3 = 6$, so return 6.

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
```

Explanation

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

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)

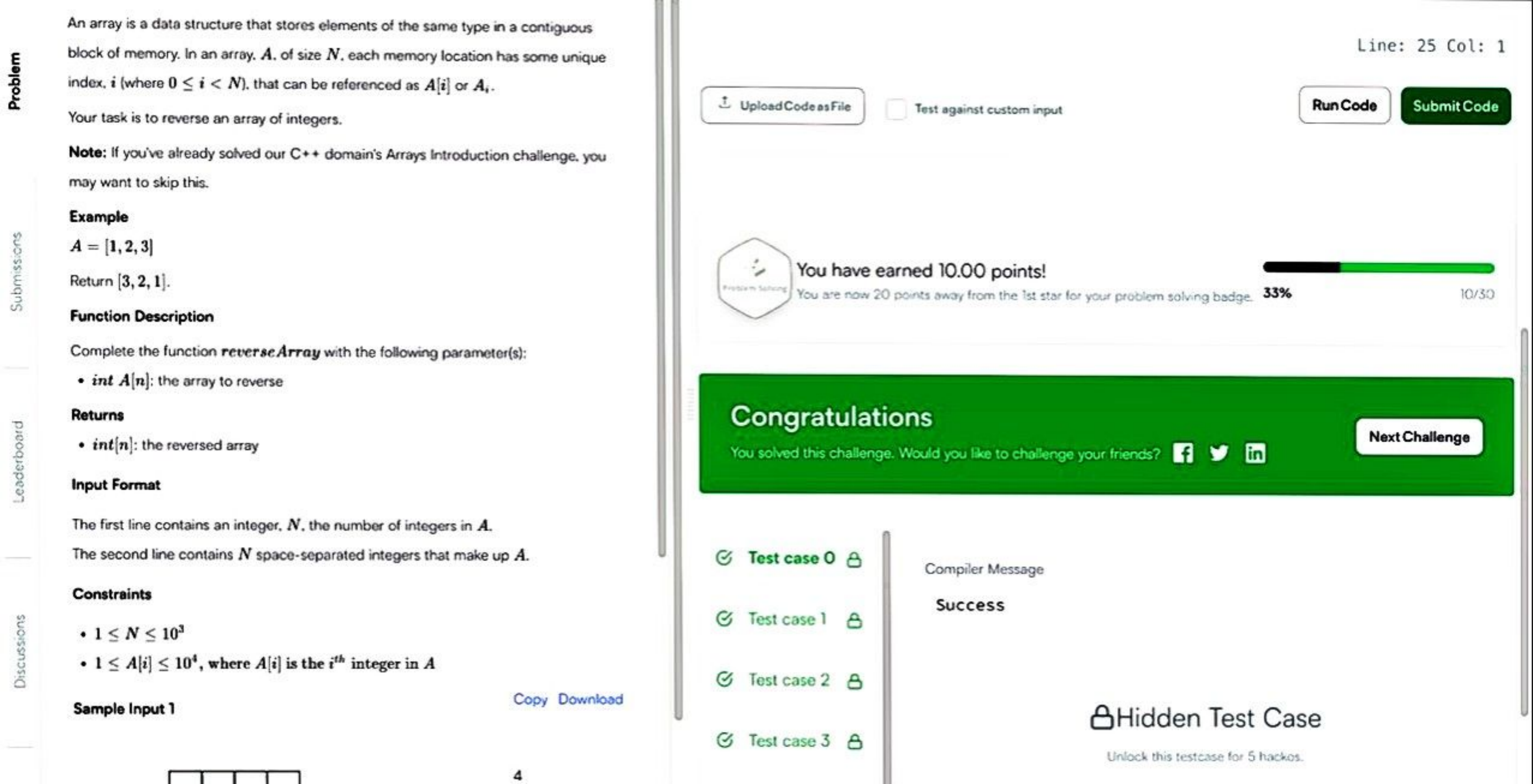
```
6
1 2 3 4 10 11
```

Your Output (stdout)

```
31
```

Expected Output

```
31
```



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 $0 \le i < N$), 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

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

Sample Input 1

You have earned 10.00 points!

You are now 20 points away from the 1st star for your problem solving badge.

Congratulations

You solved this challenge. Would you like to challenge your friends?

✓ Test case 0

✓ Test case 1

✓ Test case 2

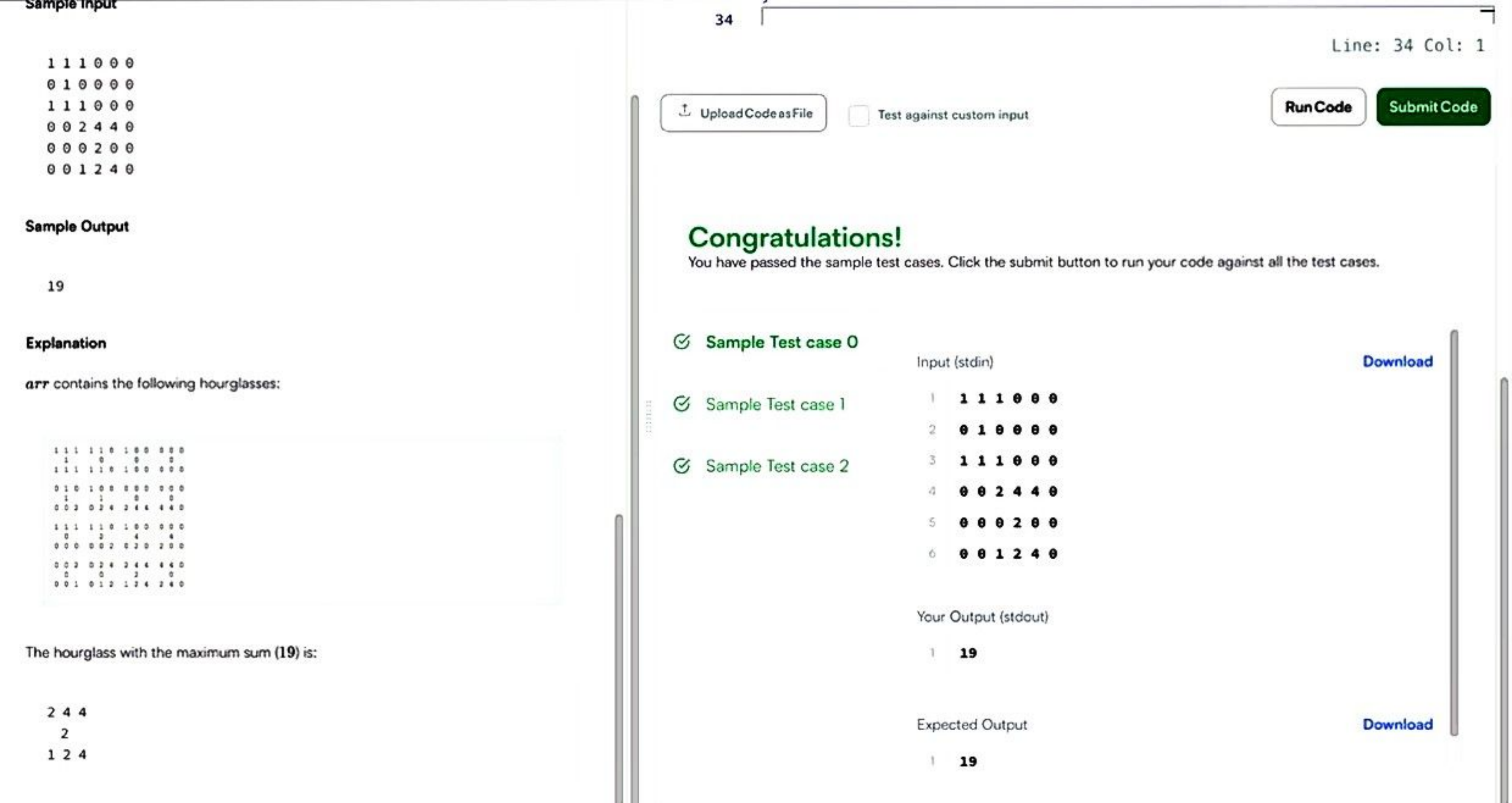
✓ Test case 3

Compiler Message

Success

Hidden Test Case

Unlock this testcase for 5 hackos.



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
```

Sample Output

```
19
```

Explanation

arr contains the following hourglasses:

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

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

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

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

The hourglass with the maximum sum (19) is:

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

```
34
```

Congratulations!

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

Input (stdin)

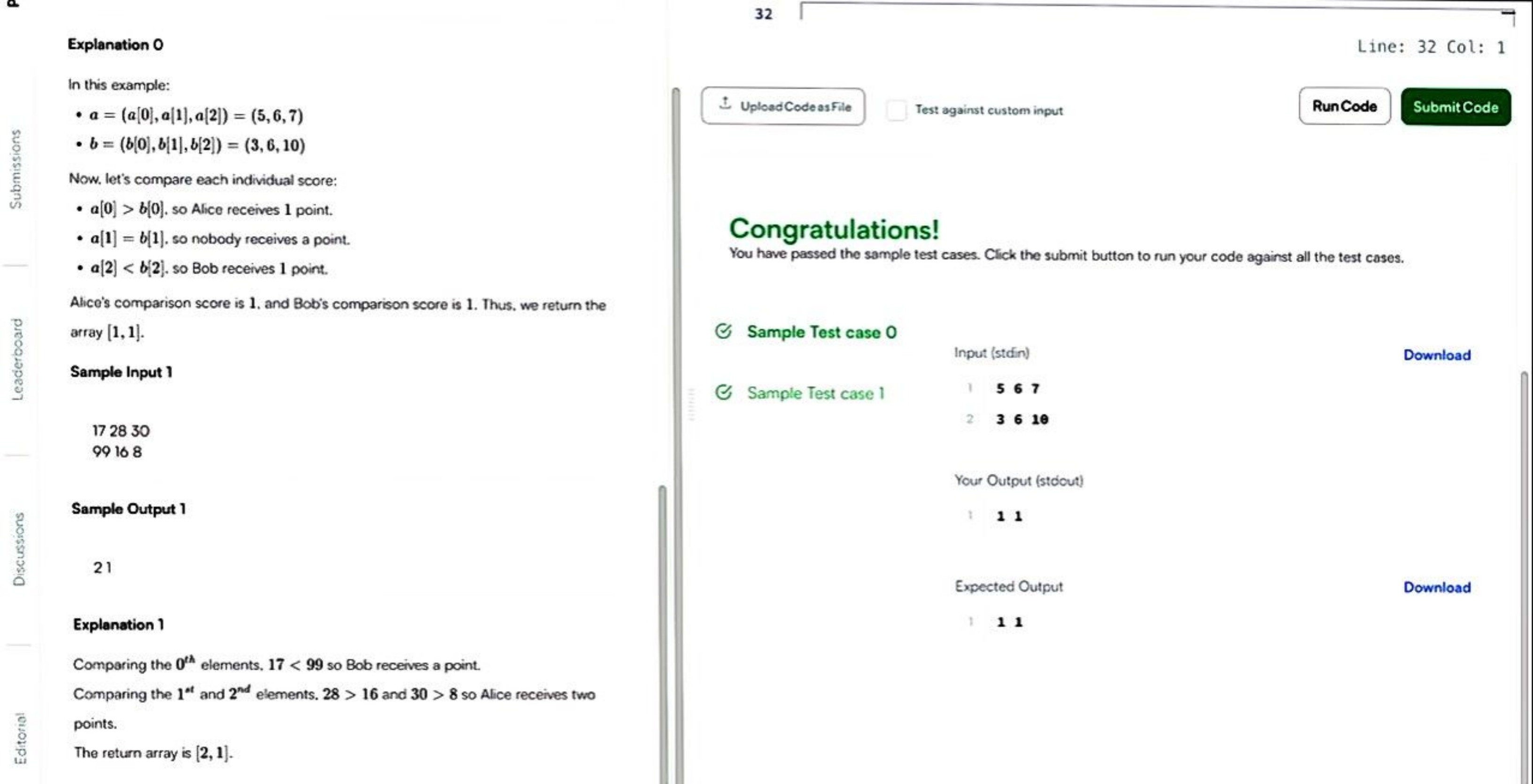
```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 2 4 4 0
0 0 0 2 0 0
0 0 1 2 4 0
```

Your Output (stdout)

```
19
```

Expected Output

```
19
```



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

```
2 1
```

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 $[2, 1]$.

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

✓ Sample Test case 1

Input (stdin)

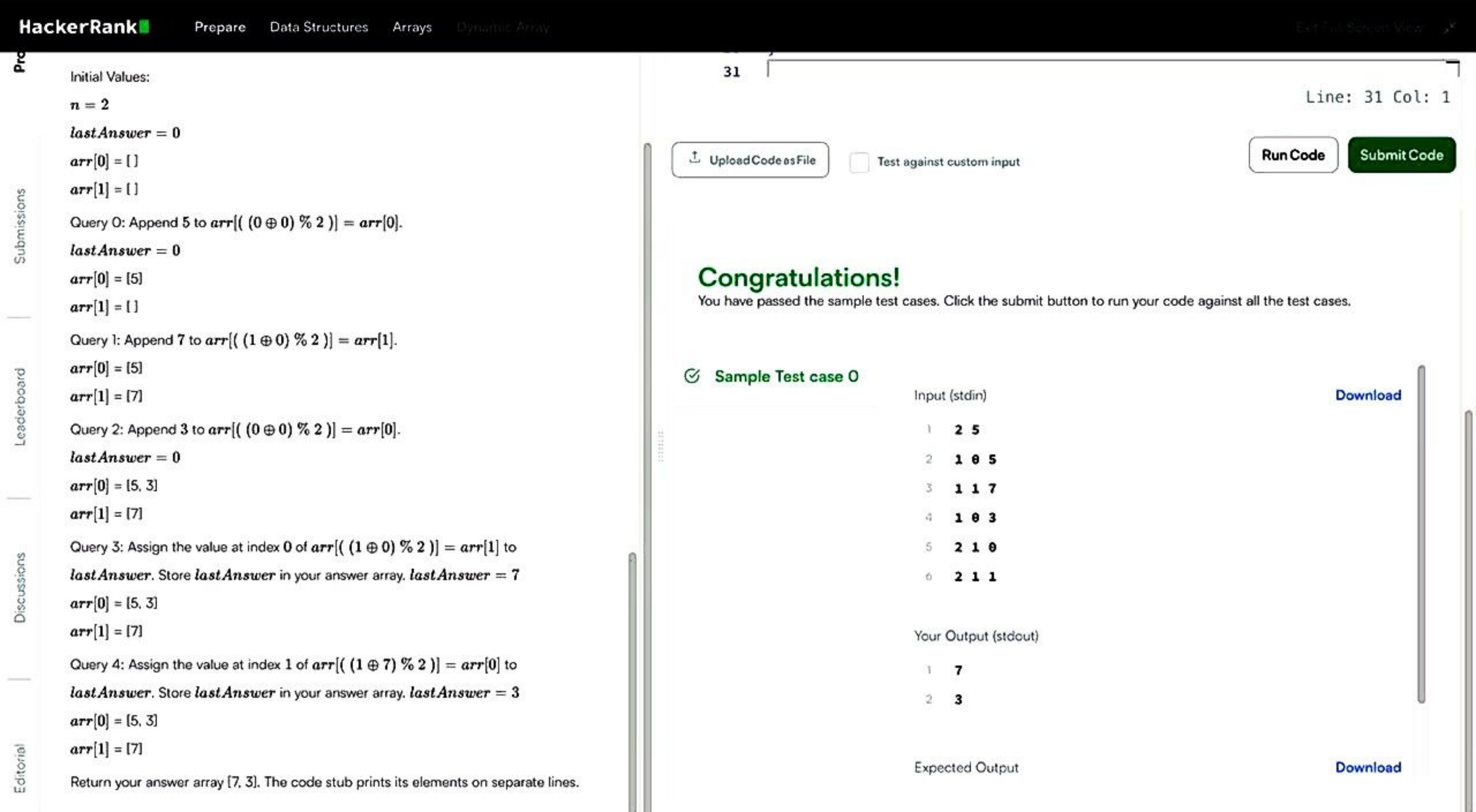
```
5 6 7
3 6 10
```

Your Output (stdout)

```
1 1
```

Expected Output

```
1 1
```



Initial Values:

$n = 2$

$lastAnswer = 0$

$arr[0]$ = [ ]

$arr[1]$ = [ ]

Query 0: Append 5 to $arr[(\ (0 \oplus 0)\ \%\ 2\ )] = arr[0]$.

$lastAnswer = 0$

$arr[0]$ = [5]

$arr[1]$ = [ ]

Query 1: Append 7 to $arr[(\ (1 \oplus 0)\ \%\ 2\ )] = arr[1]$.

$arr[0]$ = [5]

$arr[1]$ = [7]

Query 2: Append 3 to $arr[(\ (0 \oplus 0)\ \%\ 2\ )] = arr[0]$.

$lastAnswer = 0$

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

$arr[1]$ = [7]

Query 3: Assign the value at index 0 of $arr[(\ (1 \oplus 0)\ \%\ 2\ )] = arr[1]$ 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 $arr[(\ (1 \oplus 7)\ \%\ 2\ )] = arr[0]$ 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.

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

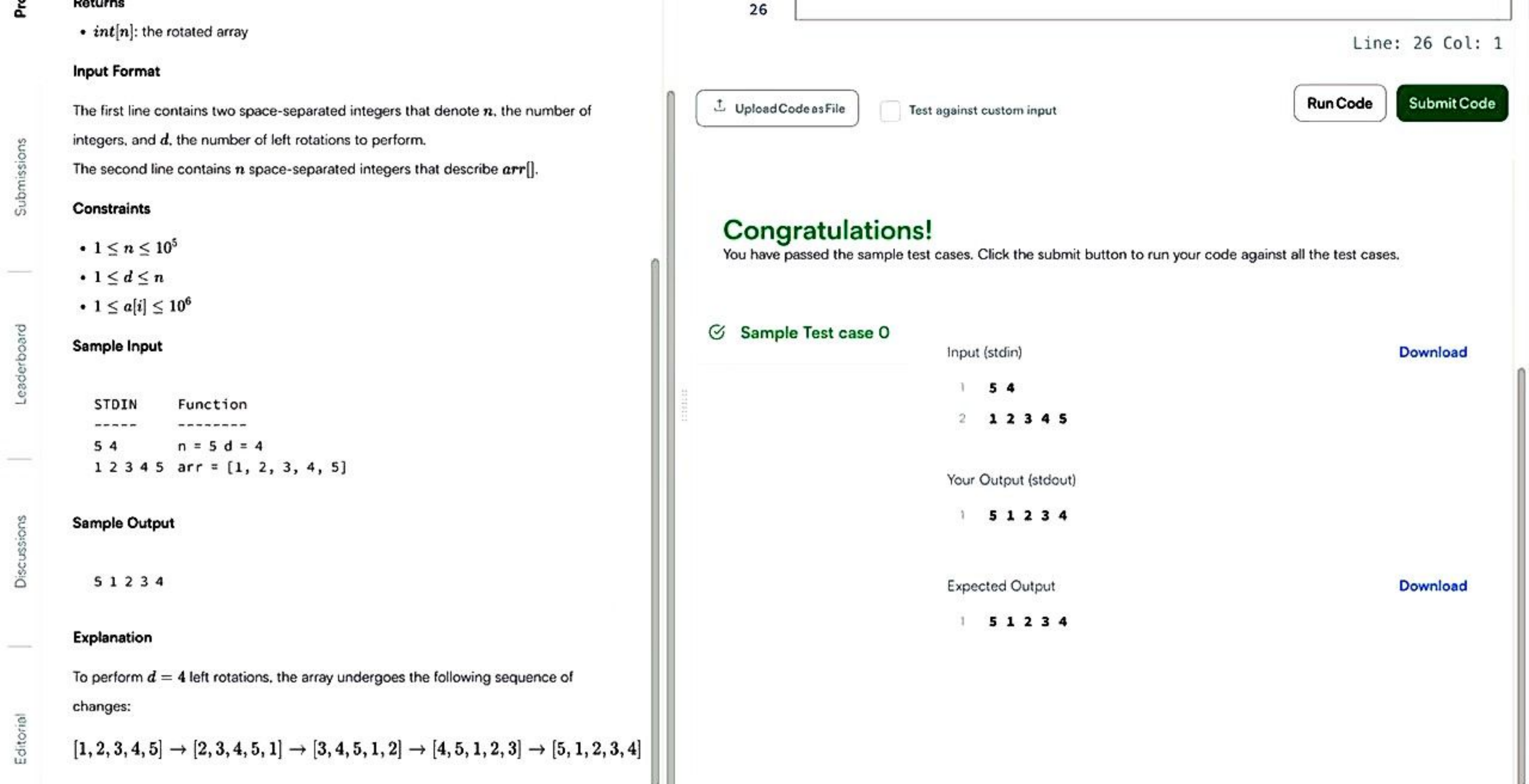
Input (stdin) Download

```
2 5
1 0 5
1 1 7
1 0 3
2 1 0
2 1 1
```

Your Output (stdout)

```
7
3
```

Expected Output Download



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  arr = [1, 2, 3, 4, 5]
```

Sample Output

```
5 1 2 3 4
```

Explanation

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

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

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)

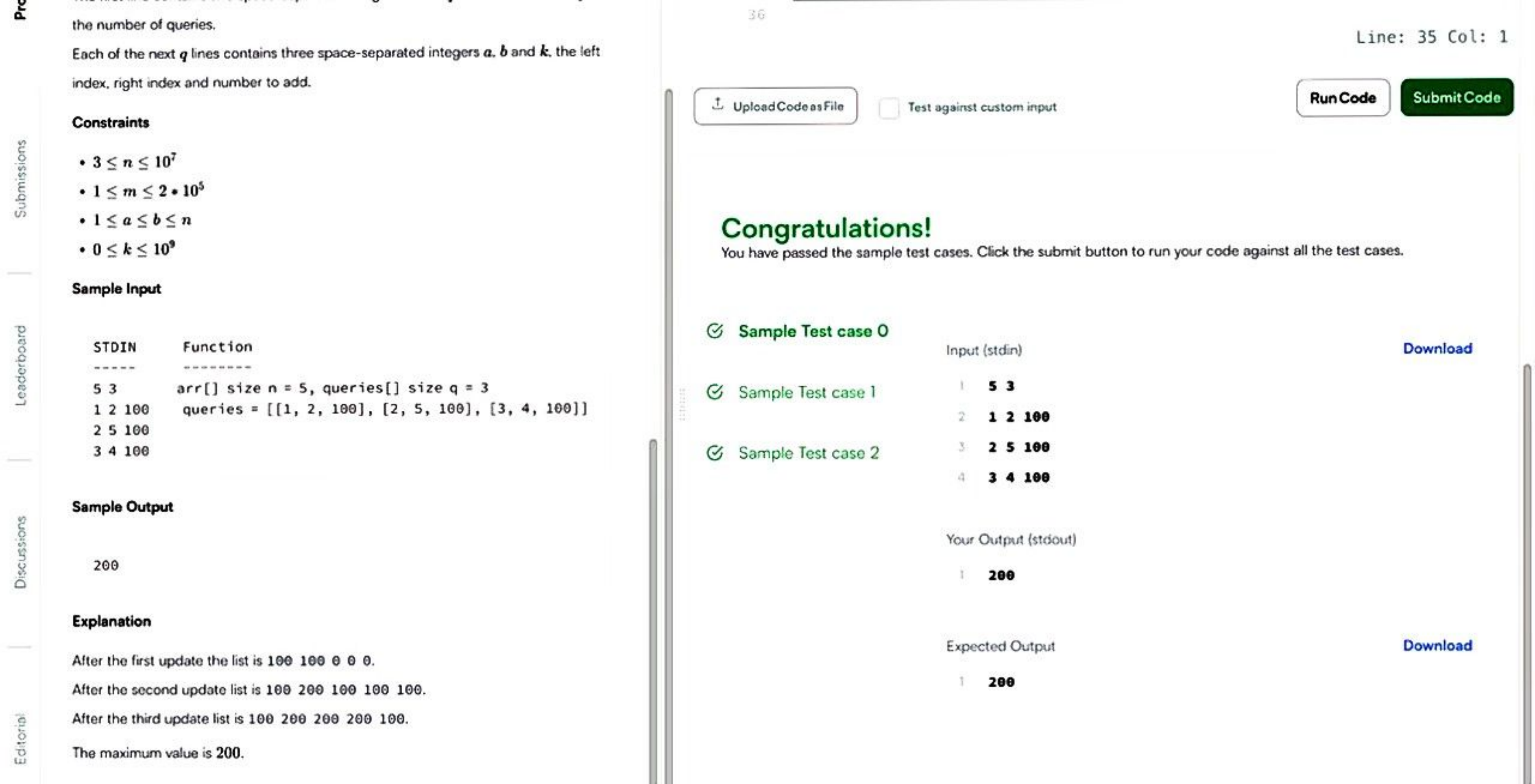
```
5 4
1 2 3 4 5
```

Your Output (stdout)

```
5 1 2 3 4
```

Expected Output

```
5 1 2 3 4
```



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

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

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.

Line: 35 Col: 1

Congratulations!

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

Input (stdin)

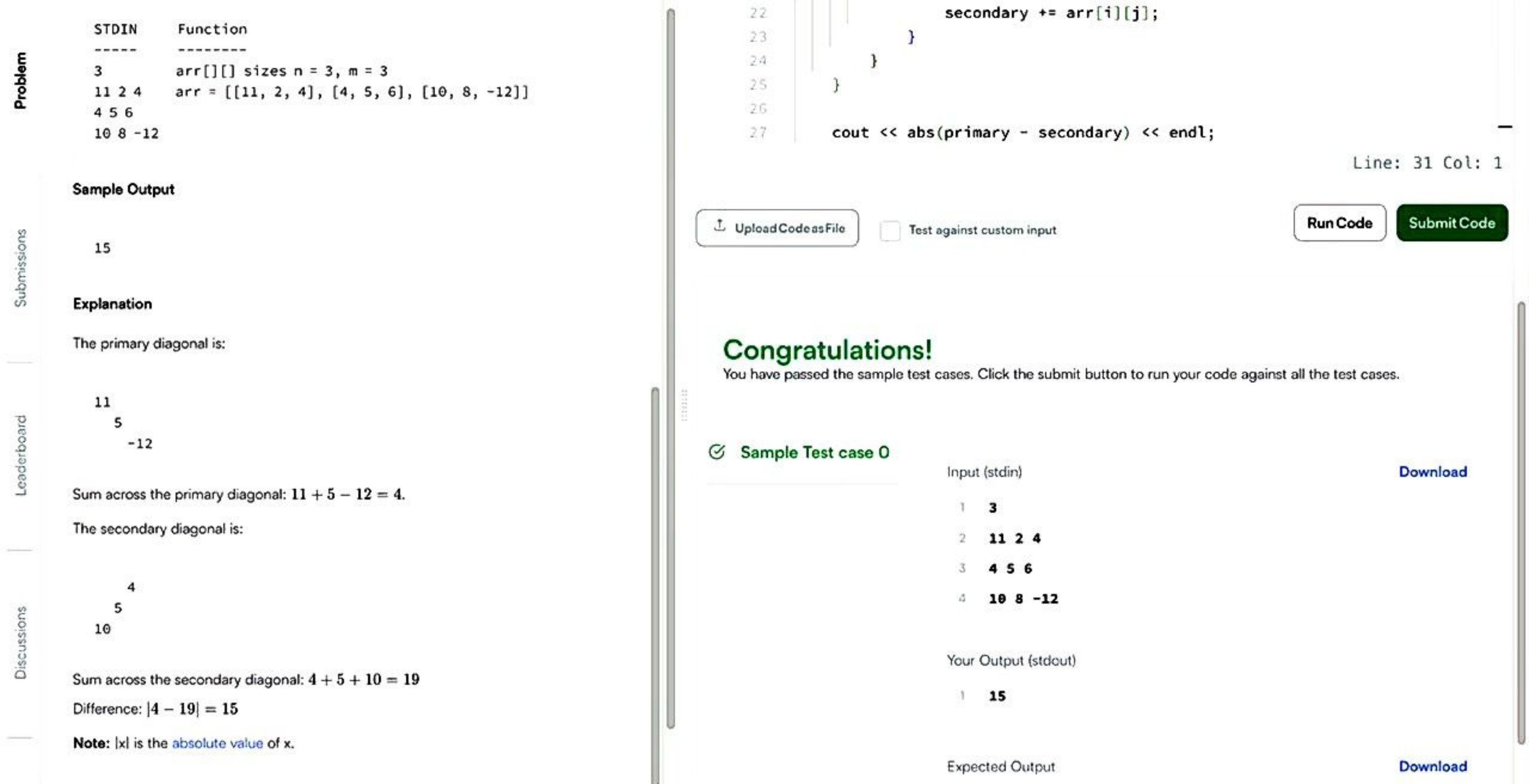
```
5 3
1 2 100
2 5 100
3 4 100
```

Your Output (stdout)

```
200
```

Expected Output

```
200
```



```
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
```

Sample Output

```
15
```

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.

```
                secondary += arr[i][j];
            }
        }
    }

    cout << abs(primary - secondary) << endl;
```

Congratulations!

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

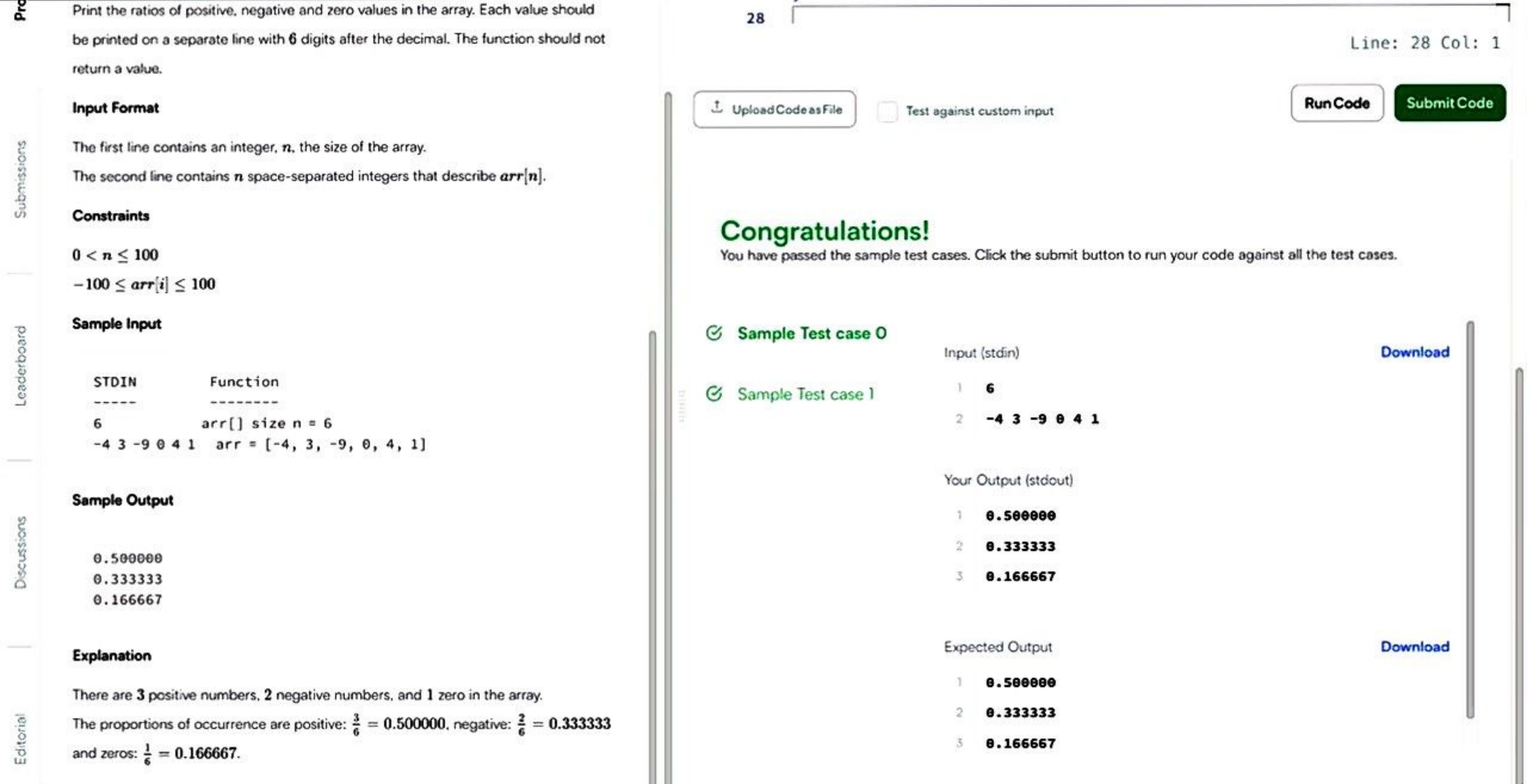
Input (stdin)

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

Your Output (stdout)

```
15
```

Expected Output



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: $\frac{1}{6} = 0.166667$.

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

✓ Sample Test case 1

Input (stdin) Download

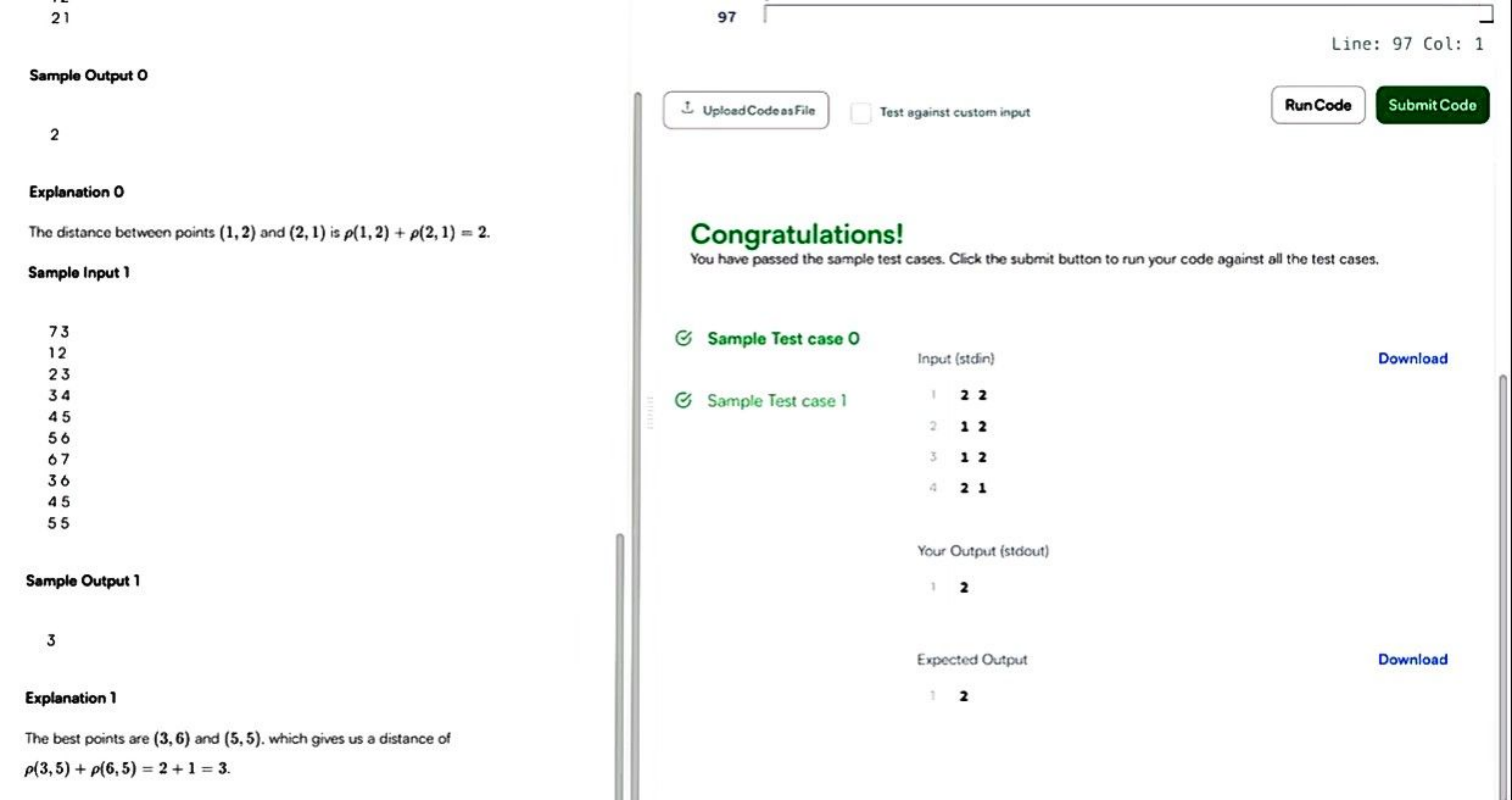
```
6
-4 3 -9 0 4 1
```

Your Output (stdout)

```
0.500000
0.333333
0.166667
```

Expected Output Download

```
0.500000
0.333333
0.166667
```



```
21
```

Sample Output 0

```
2
```

Explanation 0

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

Sample Input 1

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

Sample Output 1

```
3
```

Explanation 1

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

```
97
```

Line: 97 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

Sample Test case 1

Input (stdin)

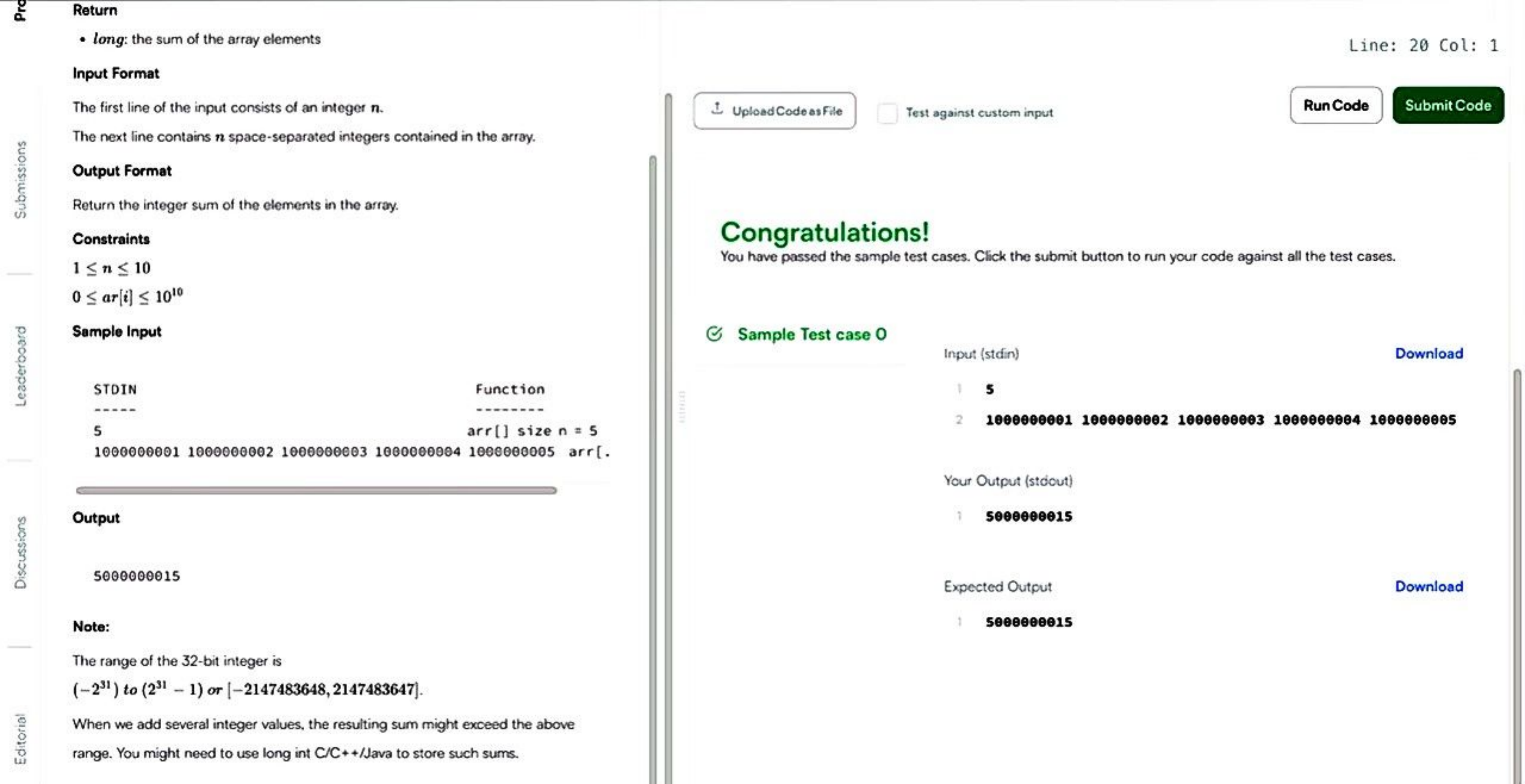
```
2 2
1 2
1 2
2 1
```

Your Output (stdout)

```
2
```

Expected Output

```
2
```



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.

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)

```
5
1000000001 1000000002 1000000003 1000000004 1000000005
```

Your Output (stdout)

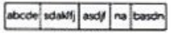
```
5000000015
```

Expected Output

```
5000000015
```

Sample Input 3

Copy Download



Array: queries

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

Sample Output 3

```
1
3
4
3
2
```

```
30
```

Line: 30 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**

✓ Sample Test case 1

✓ Sample Test case 2

Input (stdin) Download

```
4
aba
baba
aba
xzxb
3
aba
xzxb
ab
```

Your Output (stdout)

```
2
```