Started on	Friday, 26 January 2024, 4:56 PM
State	Finished
Completed on	Friday, 26 January 2024, 6:04 PM
Time taken	1 hour 7 mins
Marks	30.00/30.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1

Correct

Mark 10.00 out of 10.00

An *array* is a type of 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.

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* in the editor below.

reverseArray has 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

For example:

Input	Result
4 1 4 3 2	2 3 4 1
3 1 2 3	3 2 1

Answer: (penalty regime: 0 %)

Reset answer

```
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

/*
 * Complete the 'reverseArray' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
```

```cpp
 * The function accepts INTEGER_ARRAY a as parameter.
 */

vector<int> reverseArray(vector<int> a) {
    int length= a.size();                 //length of a
    vector<int> reversedArray(length) ;    // new array to s
    for (int i=length-1;i>=0;i--){
        reversedArray[length-i-1] = a[i];   //asssign values
    }
    for (int j=0;j<length;j++){           //print the array
        cout<<reversedArray[j]<<" ";
    }
    return reversedArray;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string arr_count_temp;
    getline(cin, arr_count_temp);

    int arr_count = stoi(ltrim(rtrim(arr_count_temp)));

    string arr_temp_temp;
    getline(cin, arr_temp_temp);

    vector<string> arr_temp = split(rtrim(arr_temp_temp));

    vector<int> arr(arr_count);

    for (int i = 0; i < arr_count; i++) {
        int arr_item = stoi(arr_temp[i]);

        arr[i] = arr_item;
    }

    vector<int> res = reverseArray(arr);

    for (size_t i = 0; i < res.size(); i++) {
```

	Input	Expected	Got	
✓	4 1 4 3 2	2 3 4 1	2 3 4 1	✓
✓	3 1 2 3	3 2 1	3 2 1	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

Question 2

Correct

Mark 10.00 out of 10.00

Given a 6×6 *2D Array*, arr:

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

An hourglass in A is a subset of values with indices falling in this pattern in arr 's graphical representation:

```
a b c
  d
e f g
```

There are 16 hourglasses in arr. An *hourglass sum* is the sum of an hourglass' values. Calculate the hourglass sum for every hourglass in arr, then print the *maximum* hourglass sum. The array will always be 6×6.

Example

$arr =$

```
-9 -9 -9  1 1 1
 0 -9  0  4 3 2
-9 -9 -9  1 2 3
 0  0  8  6 6 0
 0  0  0 -2 0 0
 0  0  1  2 4 0
```

The 16 hourglass sums are:

```
-63, -34, -9, 12,
-10,   0, 28, 23,
-27, -11, -2, 10,
  9,  17, 25, 18
```

The highest hourglass sum is 28 from the hourglass beginning at row 1, column 2:

```
0 4 3
  1
8 6 6
```

Note: If you have already solved the Java domain's *Java 2D Array* challenge, you may wish to skip this challenge.

Function Description

Complete the function *hourglassSum* in the editor below.

hourglassSum has the following parameter(s):

- *int arr[6][6]*: an array of integers

Returns

- *int:* the maximum hourglass sum

Input Format

Each of the 6 lines of inputs $arr[i]$ contains 6 space-separated integers $arr[i][j]$.

Constraints

- $-9 \leq arr[i][j] \leq 9$
- $0 \leq i, j \leq 5$

Output Format

Print the largest (maximum) hourglass sum found in arr.

Sample Input

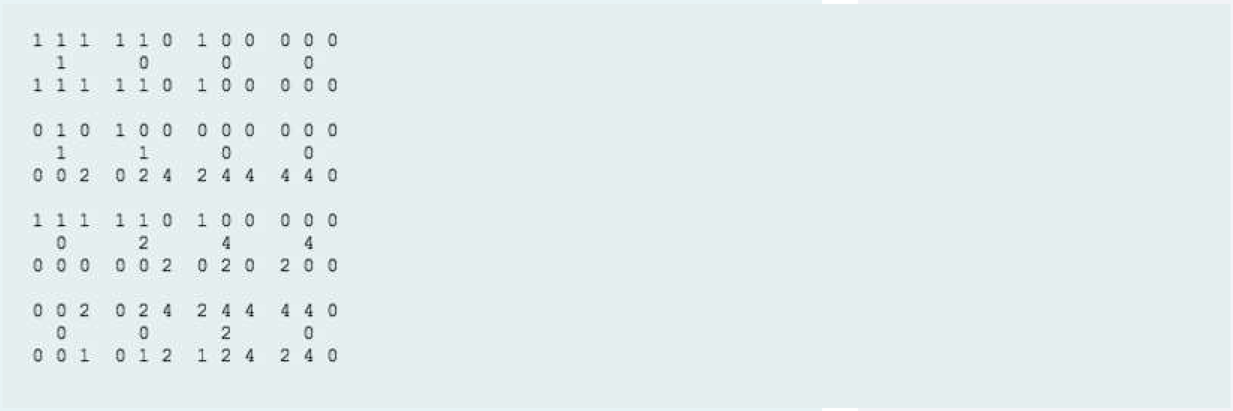
```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 2 4 4 0
0 0 0 2 0 0
0 0 1 2 4 0
```

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
```

For example:

Input	Result

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
	19

Answer: (penalty regime: 0 %)

Reset answer

```
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

/*
 * Complete the 'hourglassSum' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts 2D_INTEGER_ARRAY arr as parameter.
 */

int hourglassSum(vector<vector<int>> arr) {
    int sum = arr[0][0]+arr[0][1]+arr[0][2]+arr[1][1]+arr[2][(
    int tempSum=0;
    for (int i=0;i<4;i++){
        for (int j=0; j<4;j++){
            tempSum= arr[i][j]+arr[i][j+1]+arr[i][j+2]+arr[i+1
            if (sum<tempSum){                    // update only
                sum = tempSum;
            }
        }
    }
    cout << sum;                                 //print the fi
    return sum;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    vector<vector<int>> arr(6);

    for (int i = 0; i < 6; i++) {
        arr[i].resize(6);

        string arr_row_temp_temp;
        getline(cin, arr_row_temp_temp);

        vector<string> arr_row_temp = split(rtrim(arr_row_temp

        for (int j = 0; j < 6; j++) {
            int arr_row_item = stoi(arr_row_temp[j]);

            arr[i][j] = arr_row_item;
        }
    }

```

	Input	Expected	Got	
✓	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	19	19	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

Question 3

Correct

Mark 10.00 out of 10.00

A *left rotation* operation on an array of size n shifts each of the array's elements 1 unit to the left. Given an integer, d, rotate the array that many steps left and return the result.

Example

$d = 2$
$arr = [1, 2, 3, 4, 5]$

After 2 rotations, $arr' = [3, 4, 5, 1, 2]$.

Function Description

Complete the *rotateLeft* function in the editor below.

rotateLeft has the following parameters:

- *int d:* the amount to rotate by
- *int arr[n]:* the array to rotate

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

```
5 4
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:

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

For example:

Input	Result
5 4 1 2 3 4 5	5 1 2 3 4

Answer: (penalty regime: 0 %)

Reset answer

```
#include <bits/stdc++.h>
```

```cpp

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

/*
 * Complete the 'rotateLeft' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
 * The function accepts following parameters:
 *  1. INTEGER d
 *  2. INTEGER_ARRAY arr
 */

vector<int> rotateLeft(int d, vector<int> arr) {
    int length = arr.size();                // get the size o
    vector<int> rotatedArray(length);
    for (int i=0;i<length;i++){
        if(i>=d){                           //change elements
            rotatedArray[i-d]=arr[i];
        }
        else{                               //change elements
            rotatedArray[length+i-d]=arr[i];
        }
    }
    for (int j=0;j<length;j++){             //print the array
        cout<<rotatedArray[j]<<" ";
    }
    return rotatedArray;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string first_multiple_input_temp;
    getline(cin, first_multiple_input_temp);

    vector<string> first_multiple_input = split(rtrim(first_mu

    int n = stoi(first_multiple_input[0]);

    int d = stoi(first_multiple_input[1]);

    string arr_temp_temp;
    getline(cin, arr_temp_temp);

    vector<string> arr_temp = split(rtrim(arr_temp_temp));

```

	Input	Expected	Got	
✓	5 4 1 2 3 4 5	5 1 2 3 4	5 1 2 3 4	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.