

## Competitive Coding lab

### Experiment – 2(stacks & queues)

Name -Bhanu Pundir

Uid- 20BCS1439

SECTION/GROUP- 620 B

Branch- CSE 5th

#### Question 1:

### Game of Two Stacks ★

Problem | Submissions | Leaderboard | Discussions | Editorial

Alexa has two stacks of non-negative integers, stack  $a[n]$  and stack  $b[m]$  where index  $0$  denotes the top of the stack. Alexa challenges Nick to play the following game:

- In each move, Nick can remove one integer from the top of either stack  $a$  or stack  $b$ .
- Nick keeps a running sum of the integers he removes from the two stacks.
- Nick is disqualified from the game if, at any point, his running sum becomes greater than some integer  $maxSum$  given at the beginning of the game.
- Nick's final score is the total number of integers he has removed from the two stacks.

Given  $a$ ,  $b$ , and  $maxSum$  for  $g$  games, find the maximum possible score Nick can achieve.

**Example**

$a = [1, 2, 3, 4, 5]$   
 $b = [6, 7, 8, 9]$

The maximum number of values Nick can remove is **4**. There are two sets of choices with this result.

1. Remove **1, 2, 3, 4** from  $a$  with a sum of **10**.
2. Remove **1, 2, 3** from  $a$  and **6** from  $b$  with a sum of **12**.

**Function Description**

Complete the `twoStacks` function in the editor below.

`twoStacks` has the following parameters:

- `int maxSum`: the maximum allowed sum
- `int a[n]`: the first stack
- `int b[m]`: the second stack

#### Code:

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

```
#include <iostream> using namespace std;
```

```
int main(){

    int g;

    cin >> g;

    for (int uu = 0; uu < g; uu++){ int n;

        int m; int x;

        cin >> n >> m >> x;
        vector<int> a(n);

        for (int u = 0; u < n; u++)

        {

            cin >> a[u];

        }

        vector<int> b(m);

        for (int k = 0; k < m; k++)

        {

            cin >> b[k];

        }

        int sum = 0, count = 0, temp = 0, i = 0, j = 0; while (i < n &&
sum + a[i] <= x)

        {

            sum += a[i];
            i++;

        }


        count = i;

        while (j < m && i >= 0)

        {
```

```
sum += b[j];  
j++;  
  
while (sum > x && i > 0)  
{  
i--;  
  
sum -= a[i];  
  
}  
  
if (sum <= x && i + j > count)  
count = i + j;  
  
}  
  
cout << count << endl;  
  
}  
  
return 0;  
  
}
```

**Output:**



You have earned 30.00 points!  
You are now 4 points away from the 2nd star for your problem solving badge.

94% 96/100

### Congratulations

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

[Next Challenge](#)

✓ Test case 0

✓ Test case 1

✓ Test case 2

✓ Test case 3

✓ Test case 4

✓ Test case 5

✓ Test case 6

Compiler Message

Success

Input (stdin) [Download](#)

```

1 1
2 5 4 10
3 4 2 4 6 1
4 2 1 8 5

```

Expected Output [Download](#)

```

1 4

```

Question 2:

NEW

PREPARE

CERTIFY

COMPETE

Search

Prepare

Data Structures

Queues

Down to Zero II

264

Rank

## Down to Zero II ★

Problem

Submissions

Leaderboard

Discussions

Editorial

You are given  $Q$  queries. Each query consists of a single number  $N$ . You can perform any of the 2 operations on  $N$  in each move:

- 1: If we take 2 integers  $a$  and  $b$  where  $N = a \times b (a \neq 1, b \neq 1)$ , then we can change  $N = \max(a, b)$
- 2: Decrease the value of  $N$  by 1.

Determine the minimum number of moves required to reduce the value of  $N$  to 0.

**Input Format**

The first line contains the integer  $Q$ .

The next  $Q$  lines each contain an integer,  $N$ .

**Constraints**

$$1 \leq Q \leq 10^3$$

$$0 \leq N \leq 10^6$$

**Output Format**

Output  $Q$  lines. Each line containing the minimum number of moves required to reduce the value of  $N$  to 0.

**Sample Input**

**Code:**

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

using namespace std;

int main()
{
    int test;

    cin >> test;

    while (test--)
    {
        int n;

        cin >> n;

        int steps = 0;

        if (n == 0)
        {
            cout << 0 << endl;

            continue;
        }

        if (n == 1)
        {
            cout << 1 << endl;

            continue;
        }

        vector<int> dist(n + 1, 0);

        queue<int> q;
```

```
q.push(n);

dist[n] = 1;

while (1)

{

int element = q.front();

q.pop();

if (element == 2)

{

cout << dist[2] + 1 << endl;

break;

}

if (dist[element - 1] == 0)

{

dist[element - 1] = dist[element] + 1;

q.push(element - 1);

}

for (int i = 2; i * i <= element; i++)

{

if (element % i == 0)

{

int maxfrac = element / i;

if (dist[maxfrac] == 0)

dist[maxfrac] = dist[element] + 1, q.push(maxfrac);

}

}

}
```

```
}  
  
}  
  
return 0;  
  
}
```

**Output:**

The screenshot displays the HackerRank interface after a challenge is solved. At the top, a green banner reads "Congratulations" with social media icons and a "Next Challenge" button. To the right, a white box encourages earning a certificate with a "Get Certified" button. The main area shows a list of test cases on the left, all marked as passed. On the right, the "Compiler Message" shows "Success". Below that, the "Input (stdin)" is displayed as three lines: "2", "3", and "4". The "Expected Output" is shown as two lines: "3" and "3". Each section has a "Download" link.

**Congratulations**  
You solved this challenge. Would you like to challenge your friends?  
[Next Challenge](#)

**Earn a certificate in Problem Solving**  
Kudos on your progress! Take the HackerRank Skills Certification test and enrich your profile.  
[Get Certified](#)

**Test case 0** ✓  
Test case 1 ✓  
Test case 2 ✓  
Test case 3 ✓  
Test case 4 ✓  
Test case 5 ✓  
Test case 6 ✓

**Compiler Message**  
Success

**Input (stdin)** [Download](#)  
1 2  
2 3  
3 4

**Expected Output** [Download](#)  
1 3  
2 3