Started on	Sunday, 25 February 2024, 10:11 PM
State	Finished
Completed on	Sunday, 25 February 2024, 10:49 PM
Time taken	38 mins 6 secs
Marks	20.00/20.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1

Correct

Mark 10.00 out of 10.00

This challenge is part of a tutorial track by MyCodeSchool and is accompanied by a video lesson.

Given a pointer to the head of a singly-linked list, print each $data$ value from the reversed list. If the given list is empty, do not print anything.

Example

$head*$ refers to the linked list with $data$ values $1 \rightarrow 2 \rightarrow 3 \rightarrow NULL$

Print the following:

```
3
2
1
```

Function Description

Complete the *reversePrint* function in the editor below.

reversePrint has the following parameters:

- *SinglyLinkedListNode pointer head:* a reference to the head of the list

Prints

The $data$ values of each node in the reversed list.

Input Format

The first line of input contains t, the number of test cases.

The input of each test case is as follows:

- The first line contains an integer n, the number of elements in the list.
- Each of the next *n* lines contains a data element for a list node.

Constraints

- $1 \leq n \leq 1000$
- $1 \leq list[i] \leq 1000$, where $list[i]$ is the i^{th} element in the list.

Sample Input

```
3
5
16
12
4
2
5
3
7
3
9
5
5
1
18
3
13
```

Sample Output

```
5
2
4
12
16
9
3
7
13
3
18
1
5
```

Explanation

There are three test cases. There are no blank lines between test case output.

The first linked list has 5 elements: $16 \to 12 \to 4 \to 2 \to 5$. Printing this in reverse order produces:

```
5
2
4
12
16
```

The second linked list has 3 elements: $7 \to 3 \to 9 \to NULL$. Printing this in reverse order produces:

```
9
3
7
```

The third linked list has 5 elements: $5 \to 1 \to 18 \to 3 \to 13 \to NULL$. Printing this in reverse order produces:

```
13
3
18
1
5
```

For example:

Input	Result
3 5 16 12 4 2 5 3 7 3 9 5 5 1 18 3 13	5 2 4 12 16 9 3 7 13 3 18 1 5

Input	Result
3 3 11 1 17 3 12 11 15 4 5 7 15 14	17 1 11 15 11 12 14 15 7 5

Answer: (penalty regime: 0 %)

Reset answer

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

using namespace std;

class SinglyLinkedListNode {
    public:
        int data;
        SinglyLinkedListNode *next;

        SinglyLinkedListNode(int node_data) {
            this->data = node_data;
            this->next = nullptr;
        }
};

class SinglyLinkedList {
    public:
        SinglyLinkedListNode *head;
        SinglyLinkedListNode *tail;

        SinglyLinkedList() {
            this->head = nullptr;
            this->tail = nullptr;
        }

        void insert_node(int node_data) {
            SinglyLinkedListNode* node = new SinglyLinkedListN

            if (!this->head) {
                this->head = node;
            } else {
                this->tail->next = node;
            }

            this->tail = node;
        }
};

void print_singly_linked_list(SinglyLinkedListNode* node, stri
    while (node) {
        cout << node->data;

        node = node->next;

        if (node) {
            cout << sep;
        }
    }
}
```

```

void free_singly_linked_list(SinglyLinkedListNode* node) {
    while (node) {
```

	Input	Expected	Got	
✓	3 5 16 12 4 2 5 3 7 3 9 5 5 1 18 3 13	5 2 4 12 16 9 3 7 13 3 18 1 5	5 2 4 12 16 9 3 7 13 3 18 1 5	✓
✓	3 3 11 1 17 3 12 11 15 4 5 7 15 14	17 1 11 15 11 12 14 15 7 5	17 1 11 15 11 12 14 15 7 5	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

Question **2**

Correct

Mark 10.00 out of 10.00

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

Returns

- *int:* the maximum number of selections Nick can make

Input Format

The first line contains an integer, g (the number of games). The $3 \cdot g$ subsequent lines describe each game in the following format:

1. The first line contains three space-separated integers describing the respective values of n (the number of integers in stack a), m (the number of integers in stack b), and $maxSum$ (the number that the sum of the integers removed from the two stacks cannot exceed).
2. The second line contains n space-separated integers, the respective values of $a[i]$.
3. The third line contains m space-separated integers, the respective values of $b[i]$.

Constraints

- $1 \leq g \leq 50$
- $1 \leq n, m \leq 10^5$
- $0 \leq a[i], b[i] \leq 10^6$
- $1 \leq maxSum \leq 10^9$

Subtasks

- $1 \leq n, m, \leq 100$ for 50% of the maximum score.

Sample Input 0

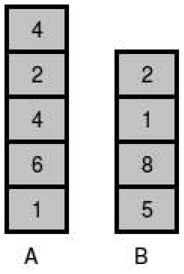
```
1
5 4 10
4 2 4 6 1
2 1 8 5
```

Sample Output 0

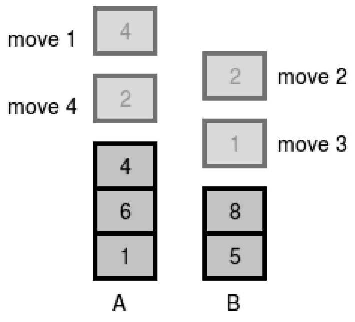
```
4
```

Explanation 0

The two stacks initially look like this:



The image below depicts the integers Nick should choose to remove from the stacks. We print 4 as our answer, because that is the maximum number of integers that can be removed from the two stacks without the sum exceeding $x = 10$.



(There can be multiple ways to remove the integers from the stack, the image shows just one of them.)

For example:

Input	Result

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

3
7 2 668
12 54 75 66 99 22 66
93 32
3 10 541
34 60 55
47 68 67 23 18 99 24 39 56 12
5 7 580
29 21 75 81 73
42 32 49 22 48 91 67
	9
	11
	11

Answer: (penalty regime: 0 %)

Reset answer

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

using namespace std;

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

/*
 * Complete the 'twoStacks' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts following parameters:
 *  1. INTEGER maxSum
 *  2. INTEGER_ARRAY a
 *  3. INTEGER_ARRAY b
 */

int twoStacks(int maxSum, vector<int> a, vector<int> b) {
    int sizeA = a.size();
    int sizeB = b.size();
    int indexA = 0, indexB = 0;
    int currentSum = 0;
    int maxElements = 0;

    // Calculate the maximum number of elements that can be pi
    while (indexA < sizeA && currentSum + a[indexA] <= maxSum)
        currentSum += a[indexA];
        indexA++;
        maxElements++;
    }

    // Pick elements from stackB and update the maxElements co
    while (indexB < sizeB) {
        // If the currentSum exceeds maxSum, remove elements f
        while (currentSum + b[indexB] > maxSum && indexA > 0)
            indexA--;
            currentSum -= a[indexA];
        }

        // If the currentSum is less than or equal to maxSum,
        if (currentSum + b[indexB] <= maxSum) {
            currentSum += b[indexB];
            indexB++;
            maxElements = max(maxElements, indexA + indexB);
        } else {
            break;
        }
    }

    return maxElements;
```

	Input	Expected	Got	
✓	1 5 4 10 4 2 4 6 1 2 1 8 5	4	4	✓

	Input	Expected	Got	
✓	3	9	9	✓
	7 2 668	11	11	
	12 54 75 66 99 22 66	11	11	
	93 32			
	3 10 541			
	34 60 55			
	47 68 67 23 18 99 24 39 56 12			
	5 7 580			
	29 21 75 81 73			
	42 32 49 22 48 91 67			

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.