**SSN College of Engineering, Kalavakkam**

**Department of Computer Science and Engineering**

**III Semester - CSE**

# UCS 1312 Data Structures Lab Laboratory

|  |  |
| --- | --- |
| **Academic Year: 2021-2022** | **Batch: 2020-2024** |

**Exercise 4: StackADT and its Applications**

Create StackADT with the members integer data array, size and top. It contains the following operations

* + 1. Initialize the stack

void initStack(struct stack \*S)

* + 1. Push an integer element into the stack

void push(struct stack \*S,int c)

* + 1. Pop to remove the top element from the stack by adjusting top

void pop(struct stack \*S)

* + 1. Returns the top element from the stack

int top(struct stack \*S)

* + 1. Check whether stack is Full

int isFull(struct stack \*S)

* + 1. Check whether stack is Empty

int isEmpty(struct stack \*S)

* + 1. Display the elements of the stack

void display(struct stack \*S)

1. Create StackADTImpl.h with the implementations of the above-mentioned operations
2. Create StackADTAppl.c is menu driven program which utilizes StackADT and StackADTImpl to perform the operations.

Write an application to play the following game of two stacks

Alexa has two stacks of non-negative integers, Stack A and Stack B

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

* 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 and B, maxSum, find the maximum possible score Nick can achieve.

**Example**

A = [1, 2, 3, 4, 5]

B = [6, 7, 8, 9]

maxSum =10

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:

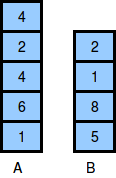
- *int maxSum:* the maximum allowed sum   
- *A:* the first stack   
- *B:* the second stack

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

**Testcase**

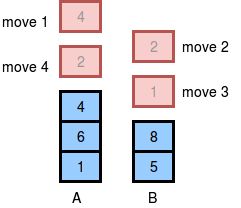
**Input**

The two stacks initially look like this:



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

.



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

**Output**

4