************* ATTENTION: Attendance in lab is mandatory for all students. NO EXCEPTION!!!! *************

Lebanese University
Faculty of Science
Section I

BS - Computer Science
2018-2019

I2206
Data Structures
LS 3 : Stack

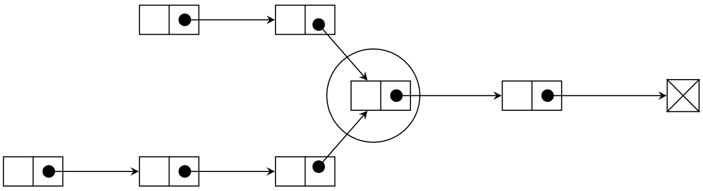
ATTENTION !

You should upload your solution at the latest by Saturday, February 23, 2019 11 p.m. via the web page of the course on www.antoun.me.

Exercise 1 (*2017-2018 Partial Exam*)
Given the following list structure :

```
typedef ... element;
typedef struct cell {
        element data;
        struct cell * next;
} * list;
```

Suppose we have two simply linked lists that can intersect at a given node to become a single linked list. The position of the intersection node between the two lists is unknown. In addition, the number of nodes in each list before the intersection node may differ. For example, in the figure below, the first list contains two nodes before the intersection, while the second one contains three nodes.



The goal of this question is to find the node address at the intersection as well as the number of nodes before the intersection in each list.

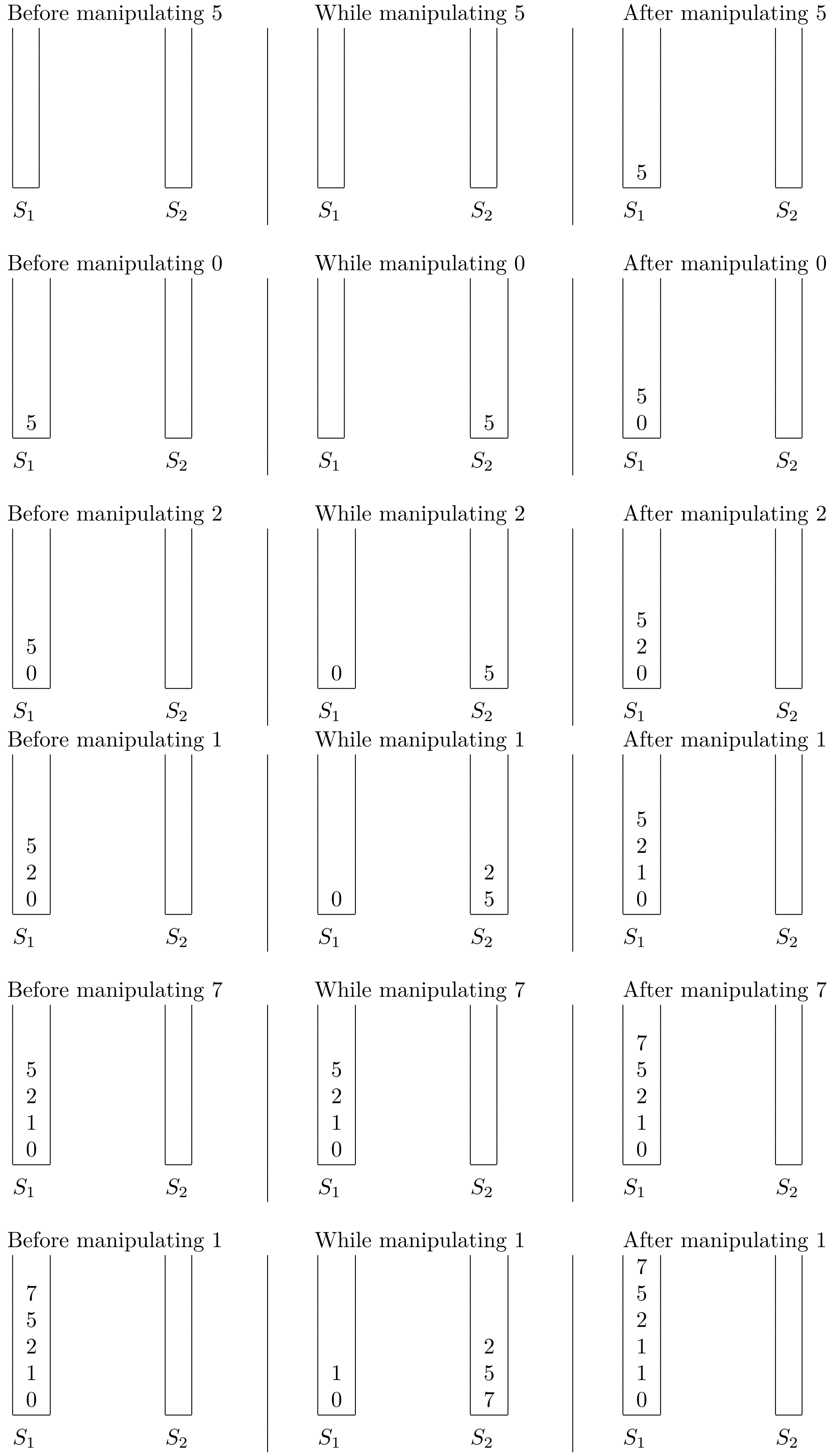
1. Using the stack ADT, write a function that takes two lists as parameters, and returns, in the case where the two lists intersect, the address of the intersection node and the number of nodes before it in each of the two lists.
2. Without performing any calculations, give the worst case time complexity of your function.
   Justify your answer.

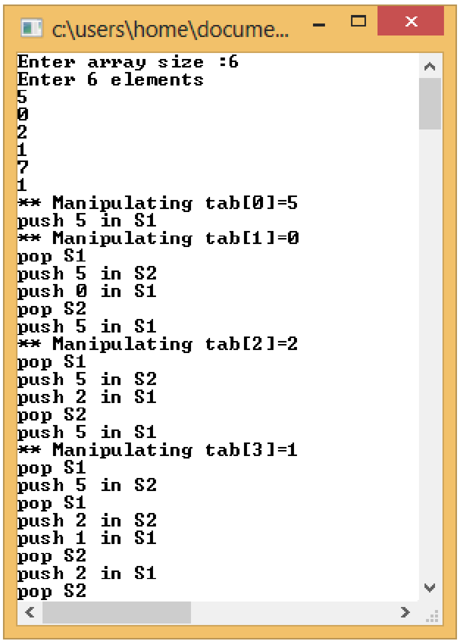
Exercise 2 We want to sort an array of N integers in decreasing order using a stack. To do this, you should use two stacks : a current stack (S_1) and an auxiliary stack (S_2) that will be used temporarily for eventual pops.
Write a C program that resolves this problem.

Running Example : Let *tab* be the following array :

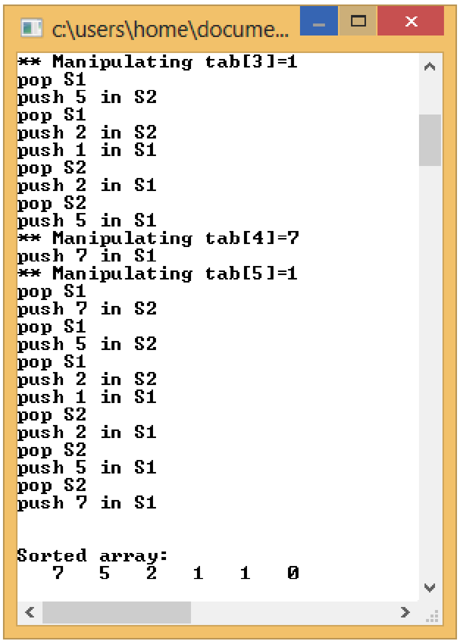
5	0	2	1	7	1

Execution trace :


Before manipulating 5
S1
S2
While manipulating 5
S1
S2
After manipulating 5
5
S1
S2
Before manipulating 0
5
S1
S2
While manipulating 0
S1
5
S2
After manipulating 0
5
0
S1
S2
Before manipulating 2
5
0
S1
S2
While manipulating 2
0
S1
5
S2
After manipulating 2
5
2
0
S1
S2
Before manipulating 1
5
2
0
S1
S2
While manipulating 1
0
S1
2
5
S2
After manipulating 1
5
2
1
0
S1
S2
Before manipulating 7
5
2
1
0
S1
S2
While manipulating 7
5
2
1
0
S1
S2
After manipulating 7
7
5
2
1
0
S1
S2
Before manipulating 1
7
5
2
1
0
S1
S2
While manipulating 1
1
0
S1
2
5
7
S2
After manipulating 1
7
5
2
1
1
0
S1
S2



```
Enter array size :6
Enter 6 elements
5
0
2
1
7
1
** Manipulating tab[0]=5
push 5 in S1
** Manipulating tab[1]=0
pop S1
push 5 in S2
push 0 in S1
pop S2
push 5 in S1
** Manipulating tab[2]=2
pop S1
push 5 in S2
push 2 in S1
pop S2
push 5 in S1
** Manipulating tab[3]=1
pop S1
push 5 in S2
pop S1
push 2 in S2
push 1 in S1
pop S2
push 2 in S1
pop S2
```



```
** Manipulating tab[3]=1
pop S1
push 5 in S2
pop S1
push 2 in S2
push 1 in S1
pop S2
push 2 in S1
pop S2
push 5 in S1
** Manipulating tab[4]=7
push 7 in S1
** Manipulating tab[5]=1
pop S1
push 7 in S2
pop S1
push 5 in S2
pop S1
push 2 in S2
push 1 in S1
pop S2
push 2 in S1
pop S2
push 5 in S1
pop S2
push 7 in S1


Sorted array:
   7   5   2   1   1   0
```