\_\_\_\_\_ /100

**CS 515 Exercise D02: Stack ADT Application**

**Names: Ryan J. Skelly\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Wildcats ID’s : 988977861\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Lecture Section: 02**

**Part I:**

Trace the following algorithms on the inputs provided to fill the result array S.

**Algorithm *spans1*(*X, n*)**

0 1 2 3 4

**Input:** array ***X*** of ***n*** integers

6 3 4 5 2

***n***

***X***

**Output:** array ***S*** of spans of ***X 5***

0 1 2 3 4

***S*** ← new array of ***n*** integers

***s***

***i***

***S***

for ***i*** ← 0 to ***n*** − 1 do 1 1 2 3 1

***s*** ← 1 0 1

while ***s***≤ *i*  and ***X***[***i***- ***s***] ≤ ***X***[***i***] 1

***s*** ← ***s*** + 1 2 2

***S***[***i***] ← ***s*** 3 1

return ***S***  *2*

3

4 1

0 1 2 3 4 5 6 7

**Algorithm** ***spans2***(***X, n***)

***S***

***X***

**Input:** array ***X*** of ***n*** integers 8

6 3 4 1 2 3 5 4

***n***

***i***

**Output:** array ***S*** of spans of ***X***

0 1 2 3 4 5 6 7

***S*** ← new array of ***n*** integers

***A*** ← new empty stack 1 1 1 2 1 2 3 6

for ***i*** ← 0 to ***n*** − 1 do 0

while ( ! ***A***.*empty*() and ***X***[***A***.*top*()] ≤ ***X***[***i***] ) do 1

***A****.pop*() 2

if ***A***.*empty*() then 3

***S***[***i***] ← ***i***+1 4

Else 3 4 5 7 5

***S***[***i***] ← ***i***- ***A****.top*() 1 2 6

***A***

***A***.*push*(***i***) 0 7

return ***S***

**Part II:**

Analyze the following algorithms to determine their overall time complexity (“Big-Oh”).

**Algorithm *spans1*(*X, n*)**

**Input:** array ***X*** of ***n*** integers

**Output:** array ***S*** of spans of ***X***

***S*** ← new array of ***n*** integers

for ***i*** ← 0 to ***n*** − 1 do O(N)

***s*** ← 1 O(1)

while ***s***≤ *i*  and ***X***[***i***- ***s***] ≤ ***X***[***i***] O(N)

***s*** ← ***s*** + 1

***S***[***i***] ← ***s*** O(1)

return ***S***

**SIZE (O(x)) == O(N^2)**

**Algorithm** ***spans2***(***X, n***)

**Input:** array ***X*** of ***n*** integers

**Output:** array ***S*** of spans of ***X***

***S*** ← new array of ***n*** integers

***A*** ← new empty stack

for ***i*** ← 0 to ***n*** − 1 do O(N)

while ( ! ***A***.*empty*() and ***X***[***A***.*top*()] ≤ ***X***[***i***] ) do O(1)

***A****.pop*()

if ***A***.*empty*() then O(1)

***S***[***i***] ← ***i***+1

else

***S***[***i***] ← ***i***- ***A****.top*()

***A***.*push*(***i***) O(1)

return ***S***

**SIZE (O(x)) == O(N)**