Due July 20 at midnight to eCampus

First Name Laura Last Name Austin UIN 524006473

User Name laustin254 E-mail address laustin254@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero, read more: Aggie Honor System Office

Type of sources	human source, web source and book source	
People	Aaron Ingram	
Web pages (provide URL)	http://www.cplusplus.com/reference/vector/vector/insert/	
Printed material	*"Data Structures and Algorithms in C++"*	
Other Sources		

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.
"On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work."

Your Name *Laura* *Austin* Date July 20th 2018

Programming Assignment 2 (140 points)

i.

A. Program Description:

The purpose of the assignment was to firstly create a stack ADT, either using My_vec or the STL vector. I chose to use STL vector. The second part of the assignment was to implement a simple spans algorithm (which was given), and it does not use to stack, called span1. The third part was to use the spans algorithm knowledge, and also use the My_stack class that was created to implement a different algorithm that also produces a spans output, called span2.

B. Data Structures Description:

Theoretical definition of Stack: Collection of elements with the operations of push(), pop(), top() and empty(). Push() adds an element to the collection. Pop() removes an element from the collection. Top() returns the element that was most recently added to the top. Empty() checks to see if the collection is empty.

-Real implementation of stack: I did create the class stack, and defined member functions within the class push(), pop(), top() and empty(). The point was to ensure that the last one in is the first one removed, if top or pop is called, assuming stack is nonempty.

-Analysis of best/worse computting spans:

for algorithm spans1, I calculated $f(n) = 5n + 2nlog_2(n)$, so $O(nlogn)$.

for algorithm spans2, I calculated $f(n) = 7n + 2nlog_2(n)$,so $O(nlogn)$.

ii. Instructions for compiling my program:

1. compile g++ -std=c++11 My_stack.h My_stack.cpp Application.cpp
2. ./a.out Application.cpp
3. My current test vectors that will be outputted can only be changed if you open up the Application.cpp, and go to where I comment "Test vector 1 for span1 algorithm" or whichever you wish to change.

iii. Logical Exceptions:

Most of my error checks are checked in the My_stack.cpp file, and printed the error. In Application.cpp, where int main() exists, I have a regular try-catch block.

iv. C++ generic programming features: I used STL vector function, so I called "insert" from <vector>, "empty()" from <vector>, when defining my own functions, [ ] operators from <vector>, erase from <vector>, and iterator from <vector>.

v. Test results:

I just have all the test results printing simultaneously so the rest results of part 2 and 3 contain the previous outputs too.

1. Testing the last in-first out My_stack.cpp output:

```
[laustin254]@linux2 ~/Austin-Laura-PA2> (20:10:23 07/19/17)
:: ./a.out main.cpp
Stack size is 0
Objects inserted in order: 0 1 2 3 4 5
Stack size is 6
Objects are removed as: 5 4 3 2 1 0
```

2. Testing the spans1 algorithm using the given vector on the assignment, and 2 more vectors I chose, which are indicated in my output:

```
[laustin254]@linux2 ~/Austin-Laura-PA2> (16:09:41 07/20/17)
:: ./a.out Application.cpp
My_stack output test:
Stack size is 0
Objects inserted in order: 0 1 2 3 4 5
Stack size is 6
Objects are removed as: 5 4 3 2 1 0

Spans1 output:

The input vector x:     6 3 4 5 2
The span of x, s:       1 1 2 3 1
The input vector x:     1 2 3 5 6
The span of x, s:       1 2 3 4 5
The input vector x:     1 2 3 2 1
The span of x, s:       1 2 3 1 1
```

3. Testing spans2 algorithm using the same vectors as testing spans1:

```
[laustin254]@linux2 ~/Austin-Laura-PA2> (19:33:05 07/20/17)
:: ./a.out Application.cpp
My_stack output test:
Stack size is 0
Objects inserted in order: 0 1 2 3 4 5
Stack size is 6
Objects are removed as: 5 4 3 2 1 0

Spans1 output:

The input vector x:     6 3 4 5 2
The span of x, s:       1 1 2 3 1
The input vector x:     1 2 3 5 6
The span of x, s:       1 2 3 4 5
The input vector x:     1 2 3 2 1
The span of x, s:       1 2 3 1 1

Spans2 output:

The input vector x:     6 3 4 5 2
The span2 of x, s:      1 1 2 3 1
The input vector x:     1 2 3 5 6
The span2 of x, s:      1 2 3 4 5
The input vector x:     1 2 3 2 1
The span2 of x, s:      1 2 3 1 1
```

1. (10 pts) What is the running time function of the algorithm above? The function should define the relation between the input size and the number of comparisons perform on elements of the vector. How can you classify the algorithms using the Big-O asymptotic notation and why? For algorithm spans1, I calculated $f(n) = 5n + 2nlog_2(n)$, so $O(nlogn)$, where n is input size.

2. (10 pts) What is the running time function of the second algorithm? The function should define the relation between the input size and the number of comparisons perform on elements of the vector. How can you classify the algorithms using the Big-O asymptotic notation and why? Compare the performance of both the algorithms. For algorithm spans2, I calculated $f(n) = 7n + 2nlog_2(n)$,so $O(nlogn)$, where n is input size.

Both, I calculated, ended up having the same worst case Big-O notation, $O(nlogn)$. However, spans2 contained more operations than spans1. In my implementation, both algorithms are the same asymptotically, but if I were trying to choose the algorithm with least operations and comparisons, I would choose algorithm 2.