Jeremy Prater

CS-162 – Lab F

Testing Results

You must also write a driver program that uses your stack and queue to demonstrate they work correctly. You should prompt the user to enter a series of integers, with some way to indicate they have finished entering values. Just enter the each number into your stack and queue.

Then you print out the values as you remove them from your structure. Make sure you label the output as to which is coming from the stack or the queue.

How can you tell they are working correctly?

The results coming from the stack::peek and stack::pop function are in the correct order as a stack would operate FILO (First in, last out). The results coming from queue::getFront and queue::removeFront are in the correct order as a queue would operate FIFO (First in, first out)

Your code should also generate an error if a user attempts to remove more items than were entered into the structure. Add to your driver program a test to fill the queue and stack with N values and attempt to print N+1.

What happens?

The program prints 0-N values correctly, and displays an error when N+1 is printed. This is because isEmpty() evaluates to true, causing the warning to print and the return from the function is null.

Why do you not need to test for putting too many items into either the stack or queue?

This check is required in the event that the client of the Stack or Queue may improperly call peek/pop or getFront/removeFront before any items are added. This is a case that should be handled by the stack/queue and prevent a segment violation, or bad pointer access.

Program output

```
Enter stack contents (int). 0 entered. Enter 'q' to finish. : 1
Enter stack contents (int). 1 entered. Enter 'q' to finish. : 2
Enter stack contents (int). 2 entered. Enter 'q' to finish. : 3
Enter stack contents (int). 3 entered. Enter 'q' to finish. : 4
```

```
Enter stack contents (int). 4 entered. Enter 'q' to finish. : 5
Enter stack contents (int). 5 entered. Enter 'q' to finish. : q
Printing stack contents.

stack pop: 5
stack pop: 4
stack pop: 3
stack pop: 2
stack pop: 1
Printing stack contents (n+1).


Attempted peek, but Stack is empty.
stack pop: 0
Enter queue contents (int). 0 entered. Enter 'q' to finish. : 1
Enter queue contents (int). 1 entered. Enter 'q' to finish. : 2
Enter queue contents (int). 2 entered. Enter 'q' to finish. : 3
Enter queue contents (int). 3 entered. Enter 'q' to finish. : 4
Enter queue contents (int). 4 entered. Enter 'q' to finish. : 5
Enter queue contents (int). 5 entered. Enter 'q' to finish. : q
Printing queue contents.

queue getFront: 1
queue getFront: 2
queue getFront: 3
queue getFront: 4
queue getFront: 5
Printing queue contents (n+1).
Attempted getFront, but Queue is empty.
queue getFront: 0
```