CS163 Lab Session #3 –Stacks and Queues

Please complete this to become familiar with Stacks and Queues.
Submit code to the CS199 D2L dropbox. Limit the time invested to 1 hour and 50 minutes maximum.

Coding: *With this lab, we will be working with an existing class implementing a list ADT for a journal. The data structure is a* ***linear linked list of arrays****. Each element of the array will be a journal entry (each array has at most 5 entries). You have access to the .h class interface in D2L's online "locker" to see what data members and member functions are available. Your job will be to implement functions assigned in unix.*

Develop an ADT: Create a stack for your journal entries. Pushing, popping, and peeking journal entries

_____Step 1. Examine the stack.h file and answer these questions. If you have questions with any of these, post them on D2L, the scribblar link for Lab #2 or contact karlaf@cs.pdx.edu::

a. Since pop does not have any arguments, what should it do?

   Simply remove the head item, and not return the information

b. Modify the peek prototype to retrieve the information at the top of stack and supply it back to the client program ______________________________
c. Modify the push prototype to add the journal information at the top of stack that is supplied BY the client program to the function

   ____

d. What should the destructor for the list call do?

   Deallocate all of the dynamic memory

_____Step 2. Begin implementing the member functions, in a .cpp file and upload these to D2L's CS199 dropbox:

a. Constructor stack();
b. Destructor ~stack ();
c. The push function
d. The peek function
e. The pop function

_____Step 3. Compile and run:

a. Modify main to call your functions (or double check main is correct for your implementation)
b. Download the .h and .o files from D2L's online "locker"
c. Compile: g++ *.cpp *.o
d. Run: ./a.out

_____Step 4. **Develop the test plan:** *For each member function that you plan to write, think about how to test it – what flow of control exists in the member function and how would you test out all conditions:*

Test Case(s)	**Expected Result**	**Verified? (yes/no)**
Enter no items, try to pop	Return a 0 from the function	Yes
Enter no items, try to peek	Return a 0 from the function	Yes
Enter 1 entry, try to pop	Remove the entry, return a 1	Yes
Enter 1 entry, try to peek	Return the entry, return a 1	Yes
Enter 6 entries, then display_all	Display all entrued	No, seg fault

Verify correctness: Using the above test plan, create a test program that tests the interactions of all functions together.

_____Step 5. **Challenge (*Optional*):** Examine how this would change if you were implementing a Queue instead of a Stack? Discuss.

Self-Assessment: *What could you do to improve for next time?*