**CS311 Yoshii - Homework 1 Part 2 Vector Stack (based on Week2)**

**===============================================================**

**Due: Week 4 Tuesday at the beginning of class**

**Total: 23 points Your score is:**

**Your name: Thomas Griffin**

**Date submitted: 9/22/2015**

**===========================================================**

**Purpose: To appreciate the fact that we can use a different implementation of a stack.**

===========================================================

**Make sure it works with g++**

**Problem Analysis [5pts]<separate sheet> Your score:**

Do the problem (Problem Analysis 1 or 2) you did not do in class. This time, do it by yourself.

**Questions: [3pts]<answer here> Your score:**

Q1. Name 3 types of information found in an activation record of a function.

Local variables, parameters, and return address

Q2. What is the name of the stack that holds activation records?

Call stack

Q3. Where are dynamically allocated data structures found?

They are found in the heap

**Programming - Vector-based Stack: [15pts] Your score:**

Re-write the HW1P1 stack class (**copy and then rename as vstack.h and vstack.cpp**) so that

* It now uses a **vector** instead of an array.
* There is no maximum size now. The stack **starts out having no slots** and it will grow as elements are added.
* N o need for top. Take it out. You can call size()-1 to get top.
* Destructor has to do some work to make sure it leaves no cells behind.
* Constructor has no work to do.
* isFull always returns false.
* Pop\_back() and front() do not return the top item which is at size()-1.
* Client: in case of error, destructor will not be called but OK.

Now, use this stack class with the **HW1P1 client program** to evaluate post-fix expressions. **Test with the same cases as before.**

**HW1P1 client should not be changed except to include the new header file. Your new implementation should be invisible to the client.**

**Q1) Did you notice any different results from HW1P1 test results?[1pt] <answer here>**

Everything seemed fairly similar except there aren’t any overflow errors

**Q2) The state of the program statement : [2pts] <answer here>**

* **Does your program compile without errors?**

Program compiles with no errors

* **List any bugs you are aware of, or state “No bugs”:**

No bugs

**SUBMIT THESE FILES TO COUGAR COURSES:**

**Always make sure the files you submit can be opened on a PC**

**Always include the word Client in the client file name.**

**Always start the name of the script/screen dump with Test.**

1. **This assignment sheet with your inserted answers.**
2. **Problem Analysis sheet with your answers.**
3. **The source file(s) (header, implementation, and client) with good comments. Always check against the How to Comment file.**
4. **The script or screen dump of compiling and executing your program.**

**Whether working or not, test result must include the lines for compiling your files or we will not grade our program i.e. 0 points for the program.**

**Did you check your comments and style against CS311 How To Comment.doc??**