**C++ Programming**

**Homework 1**

* Read [this](https://docs.google.com/a/uci.edu/document/d/1rp3-oBsKPFDFC3IWcL3zYHRIJYS_aTgfG1hq0O3PXD0/edit?usp=sharing) to see how to prepare and submit your homework. **Use the file names given in this document (not in the “how to submit” document).**
* Read about Test Driven Development (TDD) and follow it every time you write a program. <https://en.wikipedia.org/wiki/Test-driven_development>
* Here is a draft Makefile you could use for this homework, but look at HW0 for ideas for how to make it do more work for you

all: hello convert\_knots test\_stack

hello: hello.cpp

g++ hello.cpp -o hello

convert\_knots: convert\_knots.cpp

g++ convert\_knots.cpp -o convert\_knots

test\_stack: test\_stack.cpp stack.h

g++ test\_stack.cpp -o test\_stack

* (10 points) Enter (in a file called hello.cpp), compile, and execute the following C++ program.   
   #include <iostream>  
  using namespace std;  
  int main( int argc, char \*argv[] )  
  {  
   cout << "Hello World!\n";  
   return 0;  
  }
* (10 points) Write a function (in a file called convert\_knots.cpp) that converts knots into miles per minute. The function needs to take in an int (knot) as the argument and return a double value (miles per minute). Write a main function (at the bottom of convert\_knots.cpp) that reads an integer from the terminal, and prints out the number converted to miles per minute (as a double floating point) by calling your function.  
  Use:   
  1 knot = 1 nautical mile per hour = 6076 feet per hour  
  1 mph = 1 mile per hour = 5280 feet per hour
* (40 points) Write a class, called  *Stack*, in a file called stack.h, that implements a stack of up to 1000 characters (use an array of 1000 char to implement your stack. You should include the following member functions in your class. (Replace the semicolns with the body of each of these methods. So the class declaration given below, you will convert to a class definition.) Remember, never copy and paste any code - even the class Stack start below - type it!  
   #define STACK\_CAPACITY 1000  
   class Stack  
   {  
   public:  
   Stack(); // constructor for a stack  
   void push( char c ); // adds c to the top of the stack  
   char pop(); // removes top element, returns it  
   char top(); // returns the top element, w/o removing  
   bool isEmpty(); // returns true iff the stack is empty  
   bool isFull(); // returns true iff the stack is full  
   ~Stack(); // destructor for a stack  
   };
* (30 points) Write a main function (in a file called test\_stack.cpp which includes stack.h (note you must use double quotes around stack.h instead of angle brackets in the #include since it is your file not the standard library file) that repeatedly reads a line (using getline(cin, str)) into a std:: string character string variable, str, from  *cin* and then outputs the reverse of that line by pushing the characters onto an instance of your stack class, then printing them as they are removed from the stack. Do this for each line in the input.
* (10 points) Modify your program above to exit on end of file (when the user types a ^D, on Linux, that is pressing a D while holding down the control key). This should cause getline(cin, str) to return false. Just added (sunday morning) in response to a Piazza question: For your report, test your final version of the stack test program with several lines of input each containing multiple words. You can make up your own test cases or you may use the following three lines. Feel free to copy/paste these as input for your program.

abcdefghijklmnopqrstuvwxyz

1234567890

abcdefghijklmnopqrstuvwxyz 1234567890 abcdefghijklmnopqrstuvwxyz

* Notes about submission: Create two files: 1) a zip file (called hw1.zip) containing your three programs above (hello.cpp, convert\_knots.cpp, stack.h, and test\_stack.cpp; and an optional Makefile which builds all your programs. 2) one write-up (called “report” with an extension, like .pdf, showing its file type) showing screen shots of your three program compiling and running called report.pdf. Submit the two files (zip and report) in the folder HW1 under the Dropbox in EEE for ICS 45C. If you don’t yet have Dropbox access, email your two files to the TA or instructor before the deadline. Your report may be in any standard format like pdf, doc, or html.
* SPOILER: HInts for how to write the string reversal program for folks who are lost. It is in white so you can’t see it until you change the color or just select it.

START of SPOILER.

Type in the stack class start from the homework (remember no copy/paste). Add two data members: one an integer, count, which will be the index of the top slot of the stack that is available, and another, stk, which is an array of STACK\_CAPACITY char. Write a private member function, error, that takes a **const char star** argument called msg. It should print "ERROR: " followed by msg followed by newline. You will call error whenever you detect an error while executing one of the stack member functions. Now write main (following TDD). You can write the simplest main you can imagine. C++ also has assert(). You must include assert.h then it works just like in Python. Declare a Stack. Check if it isEmpty(). Check if it isFull(). push a character onto the stack. Check if it isEmpty(), check if it isFull(). Print out the top character. pop off the top character. Now check if it isFull() and isEmpty() again. After that is written, go implement the methods of class stack. Write the constructor first, then write isEmpty() and isFull() then write push() and pop().

To put something, c, onto the stack, you can do this

stk[count] = c;

then you must increment count. To remove it, decrement count. However, for both, you must first do error checking. If there is an error, call the error method we wrote above. If not, then do the push or pop instructions respectively. Here is an example

void push(char c)

{

if (isFull())

{

error(“Push on a full stack you idiot!!!”);

return;

}

stk[count++] = c;

}

Revise your main to do what the problem askes you to do. Declare a std::string. Write a while loop that reads a string and prints that string back to the terminal. The while loop expression should be the call to getline(). Before you try reversal of the string, test your program to see if it will read strings and print them until you type control D to indicate end of file. Now declare an object of type Stack. It should be straight forward how to finish up from here.

END of SPOILER.