CSCE 221 Cover Page
Programming Assignment #1
Due **February 1** by midnight to CSNet

First Name Cody Last Name Williams UIN 924008283

User Name Will77868 E-mail address will77868@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			
People	Bailey Bauman Ellie Miller	Abhishek Joshi Christopher Ridley	
Web pages (provide URL)	http://www.cplusplus.com /reference/istream/istream/istream/	http://courses.cs.tamu.edu/ teresa/csce221/csce221-index.html	
Printed material			
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 Cody Williams Date 02/01/2017

Programming Assignment 1 (100 points)

- A string type is not part of the C++ programming language. However C++ supports two string types:
  - A *C-style* string is an array of zero or more elements of type `char` terminated by a null character `'\0'`. A literal string is enclosed in double quotation marks, e.g. `"Howdy"`. This style string by itself does not support more complex string operations, but they exist in the C-string library (string.h).
  - An *STL* string is a class with all required private and public members used for creating and manipulating string type objects. There is a significant overlap between implementation of the STL vector and STL string classes.
- The purpose of the first programming assignment is to provide an elementary design, implementation, and testing of a simple C++ string class called `my_string`. The class implementation allows you to understand how the STL string class is implemented and it provides the overview of the basic C++ concepts about, among others, pointers, dynamic arrays, copy constructors, copy assignments, and destructors.
- Similarly to the STL string, the class `my_string` should be implemented based on a private C++ array of characters (`char` type) that from the users point of view is not fixed in size. Internally, this C++ array should have the functionality for reallocating to a larger array if required by the user.
- The *private* members of the class `my_string` should contain at least these elements:
  - `ptr` is a pointer to the dynamic array of type `char`
  - `sz` is the number of characters in the string (`sz >= 0`)
  - `cap` is the length in bytes of the allocated memory pointed to by `ptr`. Also, `sz <= cap`.
- The *public interface* of the class `my_string` should contain at least the operations listed below. Let `s` and `q` denote objects of the class `my_string`.
  - `size()` returns the number of characters (length) of `s.`
  - `capacity()` returns the length in bytes of the allocated memory. It cannot be smaller than `size()` for the same string.
  - `empty()` returns true if the string `s` is empty and false otherwise.
  - `operator[](i)` returns the character at index `i` of `s`, without performing arrays bounds checking
  - `at(i)` returns the character at index `i` of `s`, with performing arrays bounds checking. An `out_of_range` exception is thrown if `i` is not in range of the string size (between 0 and `size()-1`).
  - `operator+=(q)` appends the string `q` to `s`.
  - `operator+=(c)` appends the character `c` to `s`.
  - `insert(i, s)` inserts the string `s` before the position `i` in s and returns a reference to the resulting string. *This function is optional for extra credit.*
  - default constructor creates an empty string without any memory allocation.
  - constructor with an `int` argument `n` creates an empty string with allocated memory of size `n` bytes.
  - constructor with a C-string creates a string with the content taken from the C-string.
  - copy constructor makes a copy of the argument string.
  - destructor deallocates allocated memory and makes an empty string.
  - copy assignment assigns a string to another string (`s = q`).
- Non-member functions or the overloaded operators:
  - `operator>>` (input operator) reads input to `s.` It should stop when a white space is detected.
  - `operator<<` (output operator) prints the content of a string.
- Be sure to increase the length of the allocated memory when you insert a character or a string into the string or append a character or a string. You do not need to decrease the allocated memory. You need to check the string size against the array capacity. If the string size is greater than the allocated capacity, you need to allocate more memory by *doubling* the current capacity of the array. You can use a special private function for doing this.

- **Important:** You *cannot* use the STL class `string`. Therefore you *cannot* include the header files `<string>` or `<cstring>` in your code.

Implementation Instructions and Point Distributions

1. Download the supplementary file(s) from the class webpage.
2. Your files should be arranged as below:
   (a) (10 points) Declaration of the `my_string` class should be in the file `my_string.h` – due on the second lab.
   (b) (48 points) Implementation of the `my_string` class should be in the file `my_string.cpp` – due on Feb. 1st.
   (c) (16 points) Passing the test cases included in the file `main.cpp` and providing the results in your report – due on Feb 1st.
   (d) (5 pt) Programming style: naming, indentation, whitespace, comments, declaration, variables and constants, expressions and operators, line length, error handling and reporting. Please refer to this document: http://www.stroustrup.com/Programming/PPP-style.pdf
   (e) (21 points) Prepare a report (PDF and LyX) and cover page (PDF and LyX) in the electronic version.
      i. (1 point) Program description and purpose of the assignment
      ii. (1 point) Data structures description
      iii. (1 point) Instructions to compile and run your program including input and output specifications (if any).
      iv. (1 point) Logical exceptions (and possible bug descriptions)
      v. (1 point) C++ object oriented or generic programming features, including C++11 features.
      vi. (16 points) Testing results
3. Compile and run your program.
   (a) Use this Linux machine command line or provided `Makefile`:

```
c++ -std=c++11 *.cpp -o my_string
```
   or
```
make all
```
   It should generate an executable file called `my_string`.

   (b) Run your program by executing
```
./my_string
```

Submission

1. Please create a tar file following the instructions at:
   http://courses.cs.tamu.edu/teresa/csce221/html/tar-file.html.
   - Tar together all related files along with the report files in LyX and PDF formats.
   - "`turnin`" your tar file to the CSNet.