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

First Name Jialu Last Name Zhao UIN 426000712

User Name zhaojialu123 E-mail address zhaojialu123@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			
Web pages (provide URL)			
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 Jialu Zhao Date 01/29/2017

Report: Assignment 1

(i)Program description and purpose of the assignment

Program description: Set up our own string class with a lot of member functions. There is a significant overlap between implementation of the STL vector and STL classes.

Purpose: 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 us 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.

(ii)Data structures description

1.Theoretical definition Abstract Data Type(ADT) that specifies the type tof the data stored operations that support the data. our purpose is to provide specific functions to support these data.

2.Real implementation This my_string structure provides functions to know its size, capacity, whether it is empty or not and use [] or at to get the element in that index position. Also, there are functions += to concatenate two strings or a string and a character. In the end my_string class can do initialization and assignment.

(iii)Instructions to compile and run

1.Compile: After put my_string.cpp, my_string.h, main.cpp and Makefile in one folder, type in "make" in command line.

Note: There may be some warnings, but don't worry about that. Those warnings don't matter for the whole program.

2.Run: After compile successfully, type in "./_my.string".

Remark: After successfully run program, there is one instruction that prompt users to enter some string, just type in whatever string you like.

If you want to get the result in a txt file, use this command:

./my_string>>output

(iv)Logical exceptions

range errors: when users want to use at(i) function to find the element in that index position, they should make sure i is between 0 and the size of this string, or it will cause an out of range error. Same as that, when users try to insert some string in another string, they should make sure the index was greater than 0.

(v)C++ object oriented or generic programming features, including C++11 features.

This my_string class is object oriented programming.

C++11 features:

(1)Control of defaults: move and copy

(2)Copying and rethrowing exceptions

(vi) Testing results

```
cout << "Testing my_string class: \n";
```

Output: Testing my_string class which inplies the testing begins.

```
my_string v1;
```

Using default constructor to construct my_string v1.

```
my_string v2("first");
my_string v3("second");
```

Using cstring constuctor to construct v2 and v3.

```
v1 += v2;
v1 += ' ';
v1 += v3;
```

Concatenate v2, space and v3 using operator +=.

```
cout << "v1 = " << v1 << endl;
cout << "v1 size = " << v1.size() << endl;
cout << "v1 capacity = " << v1.capacity() << endl;
```

Output v1 itself and v1's size and its capacity.
result:
v1 = first second
v1 size = 12
v1 capacity = 20

```
cout << "v1 as [] characters:\n";
for (int i = 0; i < v1.size(); i++) {
cout << v1[i] << ' ' ;
}
```

Using operator [] to get the element in i position without range checking
result:
v1 as [] characters:
f i r s t s e c o n d v1 as at() characters:
f i r s t s e c o n d

```
cout << "v1 as at() characters:\n";
for (int i = 0; i < v1.size(); i++) {
cout << v1.at(i) << ' ' ;
}
```

Using at() function to get the element that in i position with range checking, if it is out of range, it will throw an error.

```
catch(const out_of_range& e) {
cerr << "Out of range: " << e.what() << endl;   }
```

Catch out of range error.

```
{
my_string v4(4);
v4 += 'a';
v4 += 'b';
v4 += 'c';
v4 += 'd';
cout << "v4 = " << v4 << endl;
cout << "v4 size = " << v4.size() << endl;
cout << "v4 capacity = " << v4.capacity() << endl;
const char* empty = v4.empty() ? "true" : "false";
cout << "is v4 empty: " << empty << endl;
cout << endl;
}
```

Using default constructor to construct my_string v4, and then using operator+= for character to concatenate a,b,c,d, and then output v4's size,capacity and whether it is empty or not.

result:
v4 = abcd
v4 size = 4
v4 capacity = 4
is v4 empty: false

```
my_string v5(v1);
cout << "v5 = " << v5 << endl;
my_string v5a("ly");
cout << "v5.insert(5, \"ly\") and v5.insert(14, \"ly\"): \n";
v5.insert(5, v5a);
v5.insert(14, v5a);
cout << "v5 = " << v5 << endl;
cout << endl;
```

Using copy constructor copy v1 to v5 and then output its value. Using insert function to insert v5a to v5 and output new v5.

result:
v5 = first second
v5.insert(5, "ly") and v5.insert(14, "ly"):
v5 = firstly secondly

```
my_string v6;
cout << "Enter a string:\n";
cin >> v6;
cout << "v6 = " << v6 << endl;
```

Using overload operator >> to read some string from key board and assign it to v6.

```
my_string v7(v6);
v7 += v2;
v7.insert(v6.size(), my_string(" "));
cout << "v6 + \" \" + v2 = " << v7 << endl;
my_string v8(v6);
v8 += v6[v6.size()-1];
cout << "v6 + last char of v6 = " << v8 << endl;
cout << endl;
```

Copy v6 to v7 and then using += to concatenate and insert function to insert whitespace to v7. Using [] to get the last operator of v8.

result:
Enter a string:

csce 221
v6 = csce v6 + " " + v2 = csce first
v6 + last char of v6 = cscee
note: it will stop reading at white space and new line