Lab 3 Report

Name: Hau Tao

Lab number: 24193 - CSE 330- Data Structures

Winter 2016

1. Status

I completed 100% of the lab implementing the String class without any issues

2. Complexity analysis

	Time complexity	Storage complexity
String()	O(1)	O(1)
String(const String &)	O(n)	O(1)
String(const char *)	O(n)	O(1)
~String()	O(n)	O(1)
int length()	O(n)	O(1)
char &operator[](const unsigned)	O(1)	O(1)
String operator=(const String&)	O(n)	O(1)
String &operator+= (const String&)	O(n)	O(1)
friend bool operator==(const String &, const String &);	O(n)	O(1)
friend bool operator<=(const String &, const String &);	O(n)	O(1)
friend bool operator<(const String &, const String &);	O(n)	O(1)
friend ostream & operator<<(ostream &, const String &);	O(n)	O(1)

3. Source Code

1. String.h

```
/*******************************************************************************************
**********
 * Hau Tao
 * String.h
 * 02/02/2016
 * This program implements String class declaration
 * Including one default constructor, and 2 String constructors, the destructor, and the
length function
 * and overloading operators ([];= ;+=;==;<=;<<)

********************************************************************************************
*********/

#ifndef STRING_H
#define STRING_H

// String.h

#include <iostream>

using namespace std;

class String {
       int size;
       char * buffer;
public:
       String();
       String(const String &);
       String(const char *);
       ~String();

    // other methods
       int length();
       char &operator[](const unsigned );
       String operator=(const String&);
       String &operator+= (const String&);


       friend bool operator==(const String &, const String &);
       friend bool operator<=(const String &, const String &);
       friend bool operator<(const String &, const String &);
       friend ostream & operator<<(ostream &, const String &);
};

#endif
```

2. String.cpp

```
/********************************************************************************************
**********
 * Hau Tao
 * String.h
 * 02/02/2016
 * This program implements String class declaration
 * Including one default constructor, and 2 String constructors, the destructor, and the
length function
 * and overloading operators ([];= ;+=;==;<=;<<)

*********************************************************************************************
*********/
#include <cstring>
#include "String.h"

String:: String(){
    size = 0;
    buffer = 0;

}
int String::length(){
    return size;
}
char &String::operator[](const unsigned i){
    return buffer[i];
}

String String::operator=(const String &s){
    delete[] buffer;
    size = s.size;
    buffer = new char[size];
    for( int i =0; i<size; i++)
        buffer[i] = s.buffer[i];
    return *this;

}

String &String:: operator+=(const String &s){
    size +=s.size; // new size = old size + size of new string
    char *newbuffer = new char[size ];
    strcpy(newbuffer, buffer); // copy buffer to another one
    int j = 0;
    delete[] buffer;
    buffer = new char[size];
    for(int i = strlen(newbuffer); i< size ; i++){
        newbuffer[i] = s.buffer[j++];
    }
    // move back to buffer
    for ( int i =0; i< size; ++i)
        buffer[i] = newbuffer[i];
    delete[] newbuffer;
    return *this;
}

String:: String (const String &source){
    size = source.size;
    buffer = new char[size];
    for( int i =0; i< size;++i){
        buffer[i] = source.buffer[i];
    }
}
String:: String(const char *p){
    size = strlen(p);
    buffer = new char[size];
    for( int i = 0 ; i< size; i++)
        buffer[i] = p[i];

}
String:: ~ String(){
    delete[] buffer;
    size = 0;
}
bool operator==(const String &left, const String &right ){
    int i, j;
    for(i=0, j=0; i<left.size && j<right.size; i++,j++)
        if(left.buffer[i] != right.buffer[i])
                return false;
    if (i == left.size && j == right.size)
        return true;

    return false;

}
bool operator <= (const String &left, const String &right){
    int i ,j ;
    for(i=0, j=0; i<left.size && j<right.size; i++,j++)
        if(left.buffer[i] > right.buffer[i])
                return false;
    return true;
}
bool operator < (const String &left, const String &right){
    int i ,j ;
    for(i=0, j=0; i<left.size && j<right.size; i++,j++)
        if(left.buffer[i] >= right.buffer[i])
                return false
```

```
;    return true;
}
ostream & operator<< (ostream &out , const String &s){
    for (int i =0; i< s.size; i++)
        out << s.buffer[i];
    return out;

}
```

4. Sample Run

```
Script started on Tue 02 Feb 2016 09:40:45 PM PST
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ g++ -c String.cpp
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ g++ -c String_test.cpp
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ g++ -c test.cpp
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ g++ String.o String_test.o
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ ./a.out
SUCCESS
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ g++ String.o test.o
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ ./a.out
abcd <= abe
hau@hau-Lenovo-Y50-70:~/Desktop/CSE 330/Lab3$ exit

Script done on Tue 02 Feb 2016 09:41:42 PM PST
```