**CSE 330 Lab 3 Report**

Daniel Meyer

Data Structures

Fall 2017

**Status:** 100%

**Time Complexity:** O(n)

**Storage Complexity:** O(1)

**Source Code:**

Page 2: String.h

Page 3-5: String.cpp

**Sample Run:** Page 6

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Daniel Meyer

\*String.h

\*10/9/17

\*Lab 3: String

\*Header file for a cstring style string class with c++ string

\*functionality without a null or return terminated end instead

\*using the size to remain within the buffer. Contains the

\*necessary fuinctions and constructors such as returning the

\*size of the buffer and accessing indiviual elements in the char

\*array.

\*CSE 330

\*Fall 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef STRING\_H

#define STRING\_H

// String.h

#include <iostream>

#include <cstring>

using namespace std;

class String {

private:

int size;

char \*buffer;

public:

String();

String(const String &s);

String(const char \*c);

~String();

char\* begin();

char\* end();

int length();

char& operator[](const unsigned int i);

void operator=(const String &s);

friend bool operator==(const String &s1, const String &s2);

String& operator+=(const String &s);

friend bool operator<=(const String &s1, const String &s2);

friend bool operator<(const String &s1, const String &s2);

friend bool operator>=(const String &s1, const String &s2);

friend bool operator>(const String &s1, const String &s2);

friend ostream &operator<<(ostream &os, const String &s2);

};

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Daniel Meyer

\*String.cpp

\*10/9/17

\*Lab 3: String

\*Implementation of String class in String.h

\*This file implements the functions needed to use a non-null

\*terminated cstring style class with c++ string functionality

\*using a size value and char array.

\*CSE 330

\*Fall 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "String.h"

using namespace std;

//Default constructor for String class

String::String()

{

size = 0;

buffer = '\0';

}

//Constructor for creating a String using an existing String class

String::String(const String &s)

{

buffer = s.buffer;

size = s.size;

}

//Constructor for creating a String from a char array

String::String(const char \*c)

{

int temp\_size = strlen(c);

buffer = new char[temp\_size];

for (int i = 0; i < temp\_size; i++) {

buffer[i] = c[i];

}

size = temp\_size;

}

//String deconstructor

String::~String()

{

delete[] buffer;

}

//Used to return a pointer to the beginning of the array

char\* String::begin()

{

return &this->buffer[0];

}

//Used to return a pointer to the end of the array

char\* String::end()

{

return &this->buffer[size - 1];

}

//Return the size of the char buffer

int String::length()

{

return size;

}

//Overloaded operator for accessing indivual chars in String class

char& String::operator[](const unsigned int i)

{

return buffer[i];

}

//Overloaded operator for assigning existing String class to existing one

void String::operator=(const String &s)

{

size = s.size;

buffer = new char[size];

for (int i = 0; i < size; i++) {

buffer[i] = s.buffer[i];

}

}

//Overloaded operator to check equality of two String class's lengths & contents

bool operator==(const String &s1, const String &s2)

{

if (s1.size == s2.size) {

for (int i = 0; i < s1.size; i++) {

if (s1.buffer[i] != s2.buffer[i])

return false;

}

return true;

}

return false;

}

//Overloaded operator for added a String class to existing one

String& String::operator+=(const String &s)

{

int temp\_size = size + s.size;

char \*temp = new char[temp\_size];

int i = 0;

for (; i < size; i++) {

temp[i] = buffer[i];

}

for (int j = 0; j < s.size; j++, i++) {

temp[i] = s.buffer[j];

}

delete[] buffer;

size = temp\_size;

buffer = temp;

return \*this;

}

//Overlaoded operator to check if two Strings are equal or less than

bool operator<=(const String &s1, const String &s2)

{

return (s1 < s2 || s1 == s2);

}

//Overlaoded operator to check if two Strings are less than

bool operator<(const String &s1, const String &s2)

{

return (s1.size < s2.size);

}

//Overlaoded operator to check if two Strings are equal or greater than

bool operator>=(const String &s1, const String &s2)

{

return (s1 < s2 || s1 == s2);

}

//Overlaoded operator to check if two Strings are greater than

bool operator>(const String &s1, const String &s2)

{

return (s1.size < s2.size);

}

//Overlaoded output operator to handle new String class

ostream &operator<<(ostream &os, const String &s)

{

for (int i = 0; i < s.size; i++)

{

os << s.buffer[i];

}

return os;

}

**Sample Run:**

Script started on Sun 15 Oct 2017 11:40:16 PM UTC

To run a command as administrator (user "root"), use "sudo <command>".

See "man sudo\_root" for details.

]0;ubuntu@ubuntu: ~/Desktop/Lab3[01;32mubuntu@ubuntu[00m:[01;34m~/Desktop/Lab3[00m$ g++ -c String.cpp

]0;ubuntu@ubuntu: ~/Desktop/Lab3[01;32mubuntu@ubuntu[00m:[01;34m~/Desktop/Lab3[00m$ g++ -c String\_test.cpp

]0;ubuntu@ubuntu: ~/Desktop/Lab3[01;32mubuntu@ubuntu[00m:[01;34m~/Desktop/Lab3[00m$ g++ String.o String\_test.o

]0;ubuntu@ubuntu: ~/Desktop/Lab3[01;32mubuntu@ubuntu[00m:[01;34m~/Desktop/Lab3[00m$ ./a.out

SUCCESS

]0;ubuntu@ubuntu: ~/Desktop/Lab3[01;32mubuntu@ubuntu[00m:[01;34m~/Desktop/Lab3[00m$ exit

Script done on Sun 15 Oct 2017 11:40:53 PM UTC