**CSCI 1411 – Lab 10 – CStrings and Strings**

**Goals:**

* Understand how to build a CString
* Understand how to use some of the CString functions
* Understand how to use some of the string member functions

**Development Environment:** (all students must use Visual Studios)

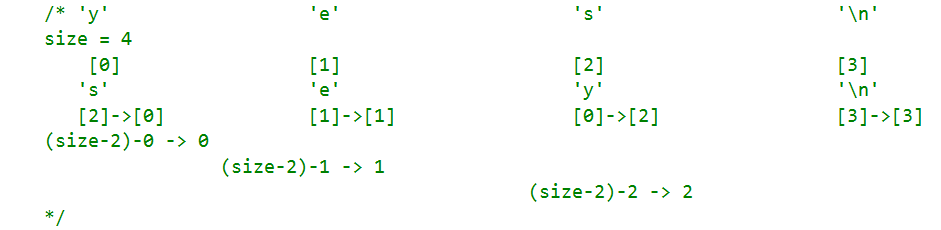
**Skills:** use of strings, cstrings, arrays, functions

**Reading**: Chap 10

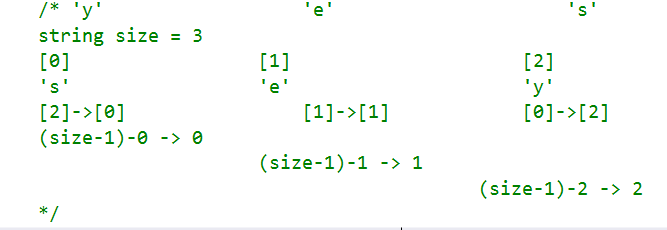
**Deliverables:** 1) This lab with 2 screen shots 2) lastnameFirstLab10.cpp

**Part I – Skills Practice (10 points)**

* Open a new project in Visual Studios. Call the project Lab10. Don’t forget to uncheck the Pre-compiled headers and Security Development Lifecycle checks. Then check the box for an empty project
* For this project we are reversing a CString and a string. A C String isn’t a data type itself, but just defines an array of characters with a ‘/n’ single character at the end. So if a CString has 5 characters, we must have an array of at least 6 characters. 5 characters plus a ‘n’. Then when printing in C, you would print until you get to a ‘\n’ which is a non-printable character.
* Remember that a char \* is a pointer to a character that can point to the start of a char array.
* In order to determine the algorithm needed to reverse characters in a CString we must first try to map out how we would manually move the characters within the array for a specific word, then try to generalize.
* Write down the math formula you would use for the 4th letter in a 4 letter word (based on the chart below)



* Similarly, we have to figure the algorithm for a string (which is a C++ object, not an array of characters). A string has its own “member functions” or functions associated with the object. When it gives size, it does not count the trailing ‘\n’. So if there are 3 characters, the size would be 3.
* Again, write the formula for the 4th letter in a 4 letter word (based on the chart below for strings.)



You will be creating 3 files called lab10a.cpp, functions.h and functions.cpp

* In functions.h.

**#ifndef FUNCTIONS\_H**

**#define FUNCTIONS\_H**

**using namespace std;**

**#include <string>**

**char \*backwardCString(char \*cstring, int size);**

**string backwardString(string stringObject);**

**void printCString(char\* cString, int size);**

**#endif**

* In functions.cpp

**#include <iostream>**

**#include "functions.h"**

**using namespace std;**

**char\* backwardCString(char \*cstring, int size)**

**{**

**char\* backCString = new char[size];**

**//remember that the backCString[size-1] needs to contain a '\n'.**

**//'\n' is just one character**

**backCString[size - 1] = '\n';**

**//now we want to reverse all of the letters before the '\n'**

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

**{**

**backCString[size-2 - i] = cstring[i];**

**}**

**return backCString;**

**}**

**string backwardString(string stringObject)**

**{**

**string backString = stringObject;**

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

**backString[stringObject.size()-1 -i] = stringObject[i];**

**return backString;**

**}**

**void printCString(char\* cString, int size)**

**{**

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

**cout << cString[i];**

**cout << endl;**

**}**

* In Lab10a.cpp

**#include <iostream>**

**#include "functions.h"**

**using namespace std;**

**int main()**

**{**

**char\* cstring = "C string";**

**char\* reverse;**

**const int size = 9;**

**string mystring = "string", reverseString;**

**cout << "cstring:" << endl;**

**printCString(cstring, size);**

**reverse = backwardCString(cstring, size);**

**printCString(reverse, size);**

**cout << endl << "string:" << endl;**

**cout << mystring << endl;**

**reverseString = backwardString(mystring);**

**cout << reverseString << endl;**

**return 0;**

**}**

* **Run** it. Take a **screen shot** of the successful output and place it below. Replace this output with your output (and 2 different cars) For a Windows 10 screen shot: Alt key + PrtSc key. Then Ctrl + V to paste. For Mac: Shift + Command + 4. You will not credit unless you have a successful screen shot with Your name in the output.

**Part II - outline/pseudo-code/algorithms (5 pts)**

* Work with your lab partner to write an outline in comments and psuedocode to complete the following program. Use plain English for your outline.
* You should have one file broken into sections with comments. Each .h and .cpp should be in a separate section. Every program you write should have the following block at the top in comments. Make sure to fill in the Name, Class, Description and Lab Partner at the top of the file. Ensure your status is accurate.

/\* Name:  
Class: CSCI 1411-00X  
Description: [fill in description]

Lab Partner:  
Status: successfully compiled and run on csegrid [if it doesn’t run or meet all of the requirements, list the actual status!/\*

**Part III -Average Scores. (10 pts)**

* Write a function that accepts a pointer to a C-string (char \*, not a string!) as an argument and displays its contents backward. For instance, if the string argument is “Gravity” the function should display “ytivarG”. Demonstrate the function in a program that asks the user to input a string then passes it to the function as a C-string. Remember to convert a string class to a C-string you would use the

Example code segment:

**#include <string>**

**using namespace std;**

**string input;**

**char \* cstring;**

**cin >> input;**

**cstring = input.c\_str();**

* Run it. Take a **screenshot** of the output and put it below:
* Turn in yourlastnamefirstLab10.cpp to canvas along with this lab with 2 screenshots .