C++ Strings

Lab Sections

Strings

1. ## Objectives

   After you complete this experiment you will be able to use the standard string class

2. ## Introduction

   Strings in C++ are objects of the string class. This means that the programmer does not need to manage the memory that a string uses as in c-style strings.

3. ## Definitions & Important Terms

   Following is some preliminary information you need to know to understand the string class:

   a. The string library includes all the functions necessary to use the string class.
   b. The string class has a default, explicit-value and copy constructor.
   c. The [ ] operator is used to access the members of a string
   d. The index/subscript of a cell in a string must be between 0 and the length of the string minus 1.
   e. The capacity of a string is the number of elements it can hold..

4. ## Declaration Syntax

   Consider the following syntax when declaring strings in C:

   ```
   a. string s1; //creates a empty string
   b. string s2 = "hello"; //creates a string initialized to "hello".
   c. string s3("hello"); //creates a string initialized to "hello";
                          //an explicit-valued constructor called.
   d. string s4 = s2; //declaration and initialization; the copy constructor
                      //called.
   ```

   More information on the string class can be found in your course textbook and on the web.

5. <u>Experiments</u>

Step 1: In this experiment you will investigate the declaration, initialization and implementation of C++ styles.   Enter, save, compile and execute the following program in MSVS. Call the new directory "cPlusPlusStringsExp1" and the program "cPlusPlusStrings1.cpp".   Answer the questions below:

```cpp
#include <iostream>
#include <string>
using namespace std;

int main()
{
      string my_name = "Thomas Jefferson";
      string her_name("Michelle Obama");
      string his_name = my_name;

      cout<<"my_name = "<<my_name<<endl;
      cout<<"The length of my_name is "<<my_name.length()<<endl;
      cout<<"The size of my_name is "<<sizeof(my_name)<<endl<<endl;

      cout<<"her_name = "<<her_name<<endl;
      cout<<"The length of her_name is "<<her_name.length()<<endl;
      cout<<"The size of her_name is "<<sizeof(her_name)<<endl<<endl;

      his_name = "Barack Obama";
      cout<<"his_name = "<<his_name<<endl;
      cout<<"The length of his_name is "<<his_name.length()<<endl;
      cout<<"The size of his_name is "<<sizeof(his_name)<<endl<<endl;

      string their_name;

      their_name = my_name + " " + her_name
                   + " " + his_name + " " + my_name + " ";
      cout<<"their_name = "<<their_name<<endl;
      cout<<"The length of their_name is "<<their_name.length()<<endl;
      cout<<"The size of their_name is "<<sizeof(their_name)<<endl;

      return 0;

}
```

Question 1:   Please explain why the size and the length values for each string are different (or the same) in the output produced by the program in Step 1 above?

Question 2:   Why is the size the same for each string in the program in Step 1?

Question 3: Please write the statements from the program in Step 1 in which a constructor is used? Name each constructor.

Question 4: Please name the string operations that were preformed in the program?

Step 2: In this experiment you will investigate performance of the subscript and the relational operators on strings.   Enter, save, compile and execute the following program in MSVS. Call the new directory "cPlusPlusStringsExp2" and the program "cPlusPlusStrings2.cpp".   Answer the questions below:

```cpp
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string my_name = "Thomas Jefferson";
    string her_name("Michelle Obama");
    string his_name = "Joe Biden";

    for (int i=0; i < her_name.length(); i++)
    {
        cout<<her_name[i];
    }
    cout<<endl<<endl;

    if (my_name == "Thomas Jefferson")
    {
        cout << "my_name is equal to Thomas Jefferson"
            << endl << endl;
    }

    if (my_name > her_name)
    {
        cout << my_name <<" is lexicographically greater than "
            << her_name << endl << endl;
    }
```

```cpp
    if (my_name < his_name)
    {
        cout << my_name <<" is lexicographically less than "
            << his_name <<endl <<endl;
    }

    if (my_name >= her_name)
    {
        cout << my_name <<" is lexicographically greater than or equal to "
            << her_name << endl << endl;
    }

    if (my_name <= his_name)
    {
        cout << my_name <<" is lexicographically less than or equal to "
            << his_name <<endl <<endl;
    }

    return 0;

}
```

Question 5:   What does the for loop do in the program in Step 2?

Question 6:   Please explain the operation of each relational operator used in the program in Step 2?

Step 3: **In this experiment you will investigate the performance of the find and substring functions. Enter, save, compile and execute the following program in MSVS. Call the new directory "cPlusPlusStringsExp3" and the program "cPlusPlusStrings3.cpp". Answer the questions below:**

```cpp
#include <iostream>
#include <string>
using namespace std;

int main()
{
        string s1 = "123abc456de111ef789eehij111ee";
        string s2;
        int loc = 0;

        for (int i =0; i<s1.length(); i++)
        {
                loc = s1.find("1",loc);
                if (loc != -1)
                {
                        cout<<"1 appears at location "<<loc<<endl;
                loc++;
                }
        }
        cout << endl << endl;

        s2 = s1.substr(0,3);
        cout << "s2 equals " << s2<<endl;

        s2 = s1.substr(3,3);
        cout << "s2 equals " << s2<<endl;

        s2 = s1.substr(0,500);
        cout << "s2 equals " << s2 << endl << endl;


        loc = 0;
        s2 = "";

        for (int i =0; i<s1.length(); i++)
        {
                loc = s1.find("e",loc);
                if (loc != -1)
                {
                        s2+= s1.substr(loc,1);
                        loc++;
                }
        }
        cout << "s2 equals " << s2 << endl <<endl;

        return 0;

}
```

Question 7: What compiler warnings were given?

Question 8:   What does the first **for** loop perform in the program in Step 3? Explain by observing the output of the program.

Question 9:   Please explain **the first three statements** that use the substr function?

Question 10:   Please explain the last **for** loop in the program?