Lab 12: Extending the String class

In the last lab, you developed your own version of the C++ string class, with a very small set of member functions. In this lab you will get more practice designing and implementing a class and working with C strings and character data. This project is again a pair programming project. You will be working in pairs following the guidelines and procedures described in http://ecee.colorado.edu/∼ecen2120/Manual/pair.html. I have taken excerpts from this University of Colorado webpage as the basis for a short tutorial on pair programming, which is included in the lab's directory on the server. Before you begin this assignment, make sure you know who is the driver and who is the navigator, and know what your respective roles are.

Exercise

Write a program that extends the `String` class that was developed in Lab 11. The class should have a data member that is an array of 80 characters to store the characters of the string including the `NULL` character at the end. A proper empty object of type `String` should have a single character it in (this character is `'\0'`, and is located in the first array location). The `String` class specified in the previous lab was defined to have the following functions:

- A default constructor that creates an empty string;
- A constructor that initializes the object to the C string passed as its argument;
- `copy()`, which copies its C string argument into the current object, overwriting whatever was in the current object before;
- `concat()`, which appends its C string argument at the end of the string in the current object, provided that there is storage available;
- `size()`, which returns the current size of the string in the object;
- `str()`, which returns a constant pointer to the current C string in the object, exactly as the `c_str()` member function of the C++ `string` class does.

This extension to the `String` class must also have the additional functions described in plain English below. You must determine their signatures and create their prototypes.

- A constructor that initializes the object to the `String` object passed as its argument;
- `find()`, which returns the index of the first occurrence of its string argument in the string in the `String` object, or -1 if it is not found;
- `find()`, which returns the index of the first occurrence of its string argument in the `String` object after a given position in the string in the `String` object specified as its second argument, or -1 if it is not found; e.g. `find(str,6)` returns the position of the first occurrence of `str` in the `String` object starting at or after position 6;
- `toupper()`, which changes the string in the object by changing all lowercase letters in it to uppercase letters;
- `tolower()`, which changes the string in the object by changing all uppercase letters in it to lowercase letters;
- `replace()`, which replaces every occurrence of its second character argument by its first character argument in the string in the object. In other words, `replace('a', '1')` would replace all `'1'`s by `'a'`s. As a special case, if `replace()` has just a single argument, it replaces every character in the string by that character, so `replace(' ')` replaces all characters by the blank.

You must write a `main()` function that demonstrates the correctness of the new features of the `String` class. The `main()` function must ensure that every new function of the class is called, and it must demonstrate that the function works properly.

You are free to use the C string library and C character handling functions if you like, or you may implement the functionality without using any library functions. It is your choice. You are free to use my solution to Lab 11 as a starting point.

For full credit, make sure all documentation meets the standard rules. Think of ways to make your program efficient in its use of time and its use of storage.

What to Submit

Because this is a pair programming project, the submission requirement is slightly different. As usual, you will submit your program, however complete it is, by the end of today's lab, i.e., before the end of the class at 2:00 P.M. But what you submit is differrent. The instructions for submitting are:

1. Create a directory in
   `/data/biocs/b/student.accounts/cs135_sw/cs136labs/lab12/submissions`
   whose name is *username*1_*username2*, where these are the usernames of the two people working on the project.
2. Copy all files, which should be named `String.h`, `String.cpp`, and `lab12_main.cpp`, to that directory. You will lose 5% of the grade if you misname the files! Make sure that each file has a preamble with *both names* and other appropriate information in it. Make sure both partners have a copy of these files in their home directories.
3. Change the permission on the submitted directory that you created so that no one else can read or modify it. It does not matter who the owner of that directory is as long as both names are part of its name.

Do not submit executable files. Your work will be graded based on the rubric outlined in the Programming Rules document.