Lab 11: Write your own string class

In this lab you will get practice designing and implementing a class with constructors, and working with C strings. This project is 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 defines a class called `String`. The class should have a data member that is an array of 80 characters to store the characters of the string (yes, the largest number of characters in an object of type `String`, including the newline character at the end, is 80). 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. You should have the following functions for this class. The functions are described in plain sentences. You must determine their signatures and create their prototypes.

- 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.

If the C string passed to either `copy()` or `concat()` would cause the internal `String` array to be overflowed, then the function should copy whatever can be copied and make sure that the last character is set to the `NULL` character.

You must write a `main()` function that demonstrates the correctness of your `String` class. The `main()` function must ensure that every function of the class is called, and it must demonstrate that the function works properly. It should purposely try to overflow the array and show what happens as a result.

You are free to use the C string library functions if you like, or you may implement the functionality without using any library functions. It is your choice.

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.

Extra Credit

If you complete all the above, try to add a few other features to the class. You can add the following functions and modify the `main()` function to demonstrate their correctness. You will not get credit for any one of them unless it is correct and shown to be so in `main()`, so try them one at a time.

- An overload of `copy()` that has a parameter of type `String` and has the same action as the original `copy()`; (10% extra)

- An overload of `concat()` that has a parameter of type `String` and has the same action as the original `concat()`; (10% extra)
- A `comparedto()` function that has a parameter that is either a C string or an object of type `String` and performs comparison, returning the same result as the C library `strcmp()` would. I.e., if `mystring` and `yourstring` are `String` objects, then `mystring.comparedto(yourstring)` would return a negative value if `mystring` precedes `yourstring`, zero if they are identical, and positive if `mystring` follows `yourstring`. (10% extra)

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/lab10/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 `lab11_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.