PROGRAMMING ASSIGNMENT 5

CS1410 - 100 points

OUTCOMES

After you finish this assignment, you will be able to do the following:

- Define a class
- Overload its constructors
- Overload operators using member functions
- Overload operators using friend functions
- Split class code between **`.h`** and **`.cpp`** files

DESCRIPTION

Create a class named **`StringBuilder`** and split its code into two files: a header file named **`stringbuilder.h`** and an implementation file named **`stringbuilder.cpp`**. This class can be used to piece together larger strings from smaller ones. It should have a single private **`string`** data member called **`content`**. It must also define the following public members:

- **`StringBuilder()`**: a public no-argument constructor that initializes **`content`** to an empty string.
- **`StringBuilder(const string& str)`**: a public constructor that takes a string argument and adds it to its **`content`**.
- **`const string& str() const`**: returns its **`content`**.

This class must also overload the following three operators using member functions:

- **`void operator*(unsigned int n)`**: takes the value of **`content`** and add it to itself **`n`** times if n is > 1. It should have no impact If **`n <= 1`**. This allows you to write something like:

```
StringBuilder sb("foo"); // sb's content = "foo"
sb * 5; // Now sb's content = "foofoofoofoofoo"
```

- **`bool operator==(StringBuilder& sb)`**: used to compare two **`StringBuilder`** objects. It returns **`true`** if the contents of both builders are the same. This allows you to write something like:

```
StringBuilder sb1("One"), sb2("One");
if(sb1 == sb2){
   cout << "The same";
}
```

- **`bool operator!=(StringBuilder& sb)`**: the opposite of **`operator==`** returning **`true`** of the contents of both builders are different.

Finally, this class must overload the following two operators using **friend** functions:

- **void operator>>(string str, StringBuilder& sb)**: used for adding a string to the content of a **StringBuilder** object by writing something like this:

  ```
  StringBuilder sb;
  "Hello" >> sb; // Now sb contains Hello
  ```

- **ostream& operator<<(ostream& out, StringBuilder& sb)**: used for printing the content of the **StringBuilder** object to the console by writing something like this:

  ```
  StringBuilder sb;
  cout << sb;
  ```

In a second **.cpp** file, write a **main()** function that tests this class and makes sure it runs as expected. This function must call every constructor/function/overloaded operator in this class.

HINT: Check out **worksheet 8** to learn about splitting a class into a header file and an implementation file and **worksheet 9** to learn about overload operators.

INSTRUCTIONS

For this assignment, you need to have a GitHub account. If you don't have one already, please sign up for one at https://github.com/.

Getting the assignment starter code from GitHub:

- Sign in to GitHub.
- Go to the assignment link https://classroom.github.com/a/mtmnxzNn and accept the assignment. This should create a private repository under your GitHub username for this assignment. Click on the given link to open this repository and see the starter code.
- Click on the **Clone or Download** button dropdown and copy the given URL.
- Navigate to your assignments folder (or any folder you want this assignment to be placed in) and open it using Visual Studio code.
- In Visual Studio Code, open a new terminal and then run:

  ```
  wsl                                (for Windows 10 only)
  git clone THE_URL_YOU_COPIED
  ```

  This will download the starter code of this assignment from GitHub and create a folder for it with a name like **cs1410-assignment-XX-github_username**. This is the folder where your program file(s) (**.cpp** and/or **.h**) should reside.
- Open the assignment folder (whose name looks like **cs1410-assignment-XX-github_username)** in Visual Studio Code and start writing your program.

Compiling your C++ program:

- From inside the assignment folder in Visual Studio Code, open a new terminal and run:

  ```
  wsl                                (for Windows 10 only)
  ```

- To compile your program run:

  `make`

  This command will call the C++ compiler on your program, compile it, and, if no compilation errors are found, create an executable program named "**`run`**" for it. If there are compilation errors, read the console error messages and then go back to your source files (.cpp and/or .h) and fix them. Save your changes and run the "**`make`**" command to compile the program again.
- To run your program, run:

  `./run`
- To clean (remove) old compilation files and start over, run the command:

  `make clean`

  You can now run the "**`make`**" command to compile your program again and the "**`./run`**" command to run it.

Submitting your program to GitHub:

- Make sure to save your changes and commit them to GitHub when you are done. You can do that by running the following commands from inside your assignment folder:

  `wsl` (for Windows 10 only)

  `git add .`

  `git commit -m 'short commit message goes here'`

  `git push`

  Make sure to do this at least once by the deadline. For your final submission, I recommend using "**Final submission**" for the commit message. **Note that committing changes is not enough; you have to push them to GitHub; otherwise, your changes will stay on your local machine and I will not be able to see your submission.**
- Go to your assignment repository in github.com and make sure your changes are there.
- Click on the **Clone or Download** button dropdown and copy the given URL. Go to Canvas and submit the copied URL. **This URL must be submitted in Canvas after you make your "Final submission" to GitHub.**

RUBRIC

CRITERIA	POINTS
`.h` file: Class definition	20
First **`.cpp`** file: Constructors	10
First **`.cpp`** file: Member function str()	5
First **`.cpp`** file: Overloading operators using member functions	21
First **`.cpp`** file: Overloading operators using friend functions	14

Second **`.cpp`** file: **`main()`**	20
Readable, commented, and properly indented code	10
TOTAL	100