Computer Science 310

Lab #1

Back In the Saddle Again!

Due Date : Thursday, January 14th, 4 PM

25 Points

Objective

The purpose of this lab is to gain experience implementing two simple *Linux* system utilities for text files along with writing a program that uses classes, inheritance, and polymorphism in C++.

Part I : Simple File Utilities

1. Implement the *Linux* command `tail -`n, where n is the number of lines from the end of a text file to be copied to the screen.

   For example, `tail -2 myfile` would display the last two lines of file `myfile`. You must use command line arguments, and may assume the command is always typed in the format above.

   You could easily solve this problem with a stack data structure. However, that might mean your program would have more than n lines from the file in RAM at once. Think of another, more efficient solution. That is, try to not read in all the lines from the file first. Hint : Use `seekg` with negative values.

2. Implement the *Linux* command `wc` which displays the number of lines, words, and bytes in a text file to the screen.

   For example, `wc myfile` might print something like the following for a five line file named `myfile`:

   ```
   5      25     122 myfile
   ```

   Use similar format with a tab prior to each integer, and the filename included at the end. The number of lines should always be followed by the number of words and the number of bytes.

   Your program should also handle zero or more of the following options in any order prior to the filename :

   - `-c`
   - `-l`
   - `-w`

Type `man wc` to learn more.

Be sure to use multiple functions and commenting in both of your programs.

Part II : Classes, Inheritance, and Polymorphism in C++

Given the following specification and implementation for class `Base`.

```
class Base
{
        public :
                virtual void iam();
};

void Base::iam()
{
          cout << "base" << endl;
}
```

Derive two classes from `Base`, and for each define `iam()` to write out the name of the class. Also, for each class define `iam2()` to call the `iam()` that is a member function of `Base`. Create objects of these two classes and call `iam()` and `iam2()` for them.

Assign the address of objects of the derived classes to `Base *` pointers and call `iam()` through these pointers.

Since these classes are all so short, feel free to code this problem all in one source file.

Submit

Create a typescript file using the `script` command. Perform each of the following steps using `cat`:

1. display your source file for the `tail` command
2. display your source file for the `wc` command
3. display all source code used for the Part II

You should now have a file named `typescript` in your current directory. Copy over your files for grading on /scratch. Remember to replace `last_fm` with your cobra login name below.

```
cp *.cpp /scratch/csc310/last_fm/Labs
cp typescript /scratch/csc310/last_fm/Labs
```