Assignment 4

100 points

Purpose

This assignment should give more experience with streams, command line arguments, and lots of decisions.

Assignment

Write a program that changes line commenting to block based commenting.

Program

Occasionally, one runs across sytems (both software and hardware) that can understand block-style comments but cannot not deal with line-style comments. Therefore there is a need for utilities that can take a file with line-style comments and turn them into block-style comments.

Block-style comments are any text beginning with `/*` and ending with `*/` possibly many lines later.

Line-style comments are any text beginning with `//` and going through the end of the line.

Write a C++ program that copies characters from the input stream to the output stream except that line-style comments are turned into block style comments. This is as simple as recognizing the existence of `//`, replacing it with `/*` and then placing `*/` at the end of the line.

The problem is complicated by the fact that `//` appearing inside a block comment is OK, and does not represent the start of a line comment. Likewise, `//`, `/*`, and `*/` inside double quoted strings are OK, and do not represent a comment. In addition, the double quote character can be escaped (`\"`) inside a quoted string and not represent the end of the string.

Do not worry about making the layout of the output of your program look nice, and do not worry about incorrect input files, but beware of `//`, `/*`, and `*/` in comments, strings and character constants.

Main

Your program should take two command line arguments, the first being the name of the input file, and the second being the name of the output file. If the program does not receive the proper number of arguments, it should print out an informative message about how to use the program and exit.

The program should open the files, one for input and one for output and test for proper opening. If an error occurs, the program should print a message and exit.

The program should use a loop to read in a piece of information from the input file, and process it using a function described below, send the result to the output. The definition of "piece of information" can be taken as a single character, a string, or a line. It is possible to implement the assignment using any of the three definitions, but the implementations will be very different.

You are not allowed to read the entire file into memory before processing. The data must be input, processed, and output, piece by piece.

Functions

You should write at least one function which will assist you in writing your main program.

- `process_data()`: This function takes an output stream and a piece of input data and returns nothing. The type of the input data is up to you depending on your implementation approach. You may add additional arguments to the function as needed as long as you are not passing in the input stream.

  This is the principal function of this assignment. This routine takes the data passed in, processes it, and places it on the output stream.

  This function may become rather large. It should be placed in a separate source code file.

- You may design and implement other functions as needed to support `process_data()`.

Header File

Create a header file to share `process_data()`, in one file, with the rest of the program.

Data and Execution

There are two data files provided with this program: `a4-in.txt` and `a4-out.txt`. Both can be obtained from the course web site or on turing/hopper. The first file contains a sample file with many comments. The second file gives the corresponding output.

Implementation Hints

- Streams have methods for placing a character onto an output stream (`.put()`) and for retrieving a character from an input stream (`.get()`).
- There is a function (`getline()`) that allows you to retrieve an entire line of input from a file into a C++ `string`.
- If you are familiar with state machines, that is probably one of the best ways to implement this assignment. However, it is not required to use a state machine.
- Opening a stream changes it. Closing a stream changes it. Placing a charactre on or retrieving a character from a stream changes it.

Other Points

(In future assignments, some things mentioned here will no longer be listed. You will still be expected to do them, however.)

- Place this assignment in an "`assign4`" directory.
- Be sure to use `const` and referencing (`&`) in your function arguments when it is appropriate to do so. References on function arguments are used to prevent large amounts of copying and/or when the argument is to be changed.
- As always, don't forget your documentation.

Submission

Place your source code files (including header file) and `Makefile` in a directory name `assign4`. Use the `mailprog` script on this directory as described on the course web site to submit the assignment. Please do not submit the data files.