CMPS 260
Spring 2005
Project #3
2005.02.20

Date Assigned: **Monday, February 21, 2005**
Due Date: **10:00 PM, Sunday, March 6, 2005**

The coded solution to the following problem is to be done by you and only you. You may ask help from the class teaching assistants and the instructors, but you may not ask for help on this project from anyone else. You may use your notes, C++ texts, on-line tutorials, etc., but the code must be your own.

1. Project Description:

A line editor is a typical computer program – if this were 1979. A line editor functions in the simplest of environments. There is no mouse, in fact there are no graphical user interface features at all. Within those limits, a line editor allows the user to open or create a new text document, view the whole or parts of a document, edit one line at a time and choose to save or discard an edited document. All commands are issued by key strokes.

For this project, write a C++ program to act as a line editor.

2. Design Requirements:

(a)The Document Class:

The document is to be contained in an object of a version of the class *arrayListType* of Chapter 13 that has been modified to work with strings. This modified version is known as *arrayStringListType* and is available in files arrayStringListType.h and arrayStringListType.cpp. Copies of these files are located on the UCS network at /w1/cs2601, /w1/cs2602, /w1/cs2603, and /w1/cs2604 and on the class web site.

(b) The Line Editor Class:

Design and implement a class that will perform the basic actions of a line editor. These are:

- given a file name, open (i.e. load) a saved document from that file, storing it in the document class; the maximum number of lines the document can contain is read in as the first value in the file
- initiate a new document (i.e. create an empty document), the maximum number of lines are specified by a parameter
- given a file name, save the current document to that file; the first item written to the file must be maximum number of lines that the document is allowed to contain
- display a range of lines within the open text document; the range of lines to be displayed is to be specified by parameters; lines out of range are not to be displayed; line numbers are to be displayed to the left of the lines
- edit a line in the open text document; the line number is to be specified by a parameter

The open document is to be contained in an object of *arrayStringListType*. The variable that refers to the *arrayStringListType* object is to be a private data member of your line editor class.

(c) The File with Main:

The line editor solution is to manged by a program that uses an object of the line editor class type. When this program starts, the user is to be able choose to

- at startup, chose to load a document from a user specified file
- at startup, chose to initiate a new document
- save the existing document to a user specified file
- display a user specified range of lines
- edit a user specified line
- exit the program with a reminder / choice to save or discard the existing document

3. Additional Requirements:

- You are responsible for following the requirements as given in documents reachable from the class web site via the **Minimum Documentation** and **Naming Conventions** links. Methods in your class definitions must has accompanying pre-condition and post-condition documentation. (See the link **Additional Documentation** on the class web site.)

- A makefile is to be created for and submitted with this assignment.

- Each class design is to be placed in a file named for the class and ending in ".h" (a header file). Each class implementation is to be placed in a file named the same as the file of the design, but ending in ".cpp". Each header file is to include wrappers to prevent unnecessary recompilation

- Your program may have global constants but may not have global variables.

- Your program solution must use good programing style. For example, call a function load the value in the data file into the array of account objects at the start of the program, then call a function to at the end of the program to write the values in the array elements back to the file. In between, call another function to handle the logic necessary to maintain communicate with the user and maintain the accounts. To rephrase, try to make your main function an outline of the logic of the program solution, one that uses calls to functions and object methods to direct the order of the operations rather than trying to write all operations in the main.

4. Submitting: You are responsible for submitting your work both electronically and via a hard copy.

- Instructions for submitting an electronic copy of your project can be found on the class web site, via the **Submitting Your Work** link. When prompted, name your project "proj3".

- A hard copy of the code in your project is due the school day after you submit electronically. Place the print out of your code in a manila folder, write your name, clid and section on the folder, then bring the folder to class or your instructor's or TA's office.