CGS 3460 - Programming Using C
Summer 2009

Assignment #10
20 Points
Assigned: 07/31/2009, Friday
Due: 08/06/2009, Thu, 11:59 PM

For this assignment, there is a single project as well. We will be implementing an abstract data type for a database. You will be given a format to load and save files of an existing data base, and define functions to modify the data base.

The database being define is a database of people.
The fields(columns) in your data base, as well as there respective types are the following:
First Name – string (char *)
Last Name – string (char *)
Salary - int
Accounts - int
Division – string (char *)
You may assume all string variables will be no longer than 50 characters.

For terminology, we will refer to an individual person in the database as a record and the characteristics of the people as fields.

You may use any type of data structure you want to represent the database. I would suggest a structure to represent a field. You could either use a linked list or an array to represent the collection of records. Each will have advantages and disadvantages for the particular functions you will be defining.

File Format
The files used to load and save data bases will contain one record per line and fields will be separated by pipe characters ('|'). After the last record in the database file, there will be a new line character ('\n'), with nothing on the last line of the file. An example is provided. The ordering of the fields is the same ordering as above, so it goes:

<first name>|<last name>|<salary>|<accounts>|<division>

All string types are surrounded by " ".

Functions:
The functions you will be defining along with a description follow:

void loadFile(const char* fileName); - This will take the given database file which is of the format specified and make the contents of that file your current database. If a database already exists, then it will be overwritten and replaced with the records contained in this file. If the given file is blank, or does not exist, a database containing no records is produced. Note that the quotes are not part of the field values for strings, they are simply their for delimiting purposes. So, if "Peter" is found in the file, only ***Peter*** should be the value of the field in the database. It should not be ***"Peter"***.

void saveFile(const char* fileName); - This will save the current database to the file specified using the format specified. If no records exist in the database, it will produce a blank file. If fileName has the value NULL than the database should be printed to stdout.

void findRecord(int field, const char* data); - This will find and print all records which have a value of "data" for the specified field. The field to use is specified by a number in the following manner.

1. First Name
2. Last Name
3. Salary
4. Accounts
5. Division

"data" is the string version of the field value. So, if the search is to be done on salary or accounts, it must first be converted to an integer before comparing.

Any record found that has a match for the specified field and value will be printed to the screen. The entire record will be printed in the same format as the database files, so one record per line in the format:
<first name>|<last name>|<salary>|<accounts>|<division>
where string types will be surrounded by " ".

void removeRecord(int field, const char* data); - This will find and remove all records which have a value of "data" for the specified field. The field to use is specified by a number in the following manner.

1. First Name
2. Last Name
3. Salary
4. Accounts
5. Division

"data" is the string version of the field value. So, if the search is to be done on salary or accounts, it must first be converted to an integer before comparing.

Any record found that has a match for the specified field and value will be removed from the screen.

void insertRecord(const char* fn, const char* ln, int salary, int accounts, const char* division); - This will create a new record with the given values and insert it into your database at "the end" of the database. By "the end", we mean a call to save, right after inserting a record would create a save file where the newly inserted record was the last one in the file.

void sortDataBase(int field); - This will sort the current database, using the given field as the key(the values considered when sorting). You may sort it however you want, so there will be no testing of whether a stable sort or unstable sort was used. I.e., if you sort on a field which has duplicate values for the keys the relative order of the records with duplicate keys does not matter.

Instructions for submitting Assignment 10

1. First compress your program files into a tar file. For this, change to the directory where you have stored both the files. At the terminal command prompt (putty, Linux console, MAC terminal, etc.) type:

tar cvf assign10.tar dataBase.h dataBase.c

Let's say you have stored your files in assign10 directory on your "rain" machine.

> cd assign10
> ls
main.c
dataBase.h
dataBase.c
..
>tar cvf assign10.tar dataBase.h dataBase.c
>ls
main.c
dataBase.h
dataBase.c
assign10.tar
....

2. If you are working from Windows platform using Putty and WinSCP, you can create a directory assign10 in your home directory on rain machine using putty. Create the directory using this command on putty

> cd //to change to the home directory
>mkdir assign10 //to create the directory under home directory

3. Then transfer dataBase.h and dataBase.c to assign10 directory on rain using WinSCP.
4. Go to step 1 to tar the file using putty.
5. Use WinSCP to copy assign10.tar from rain machine back to your local computer, say desktop.
6. Upload the assign10.tar to WebCT.
7. Please check the discussion page for assignment 10 on WebCT for common questions and their answers.