Homework 12: MiniDB

Due Date: 11/22/2016

Goals:

- Learn how to use File I/O functions in C standard library to parse a document and print results to another file
- Learn how to implement and use a data structure of your choice

Description:

In this lab, you are going to be implementing a miniature database parser that scans items from a document and places the data into a data structure for use in a program. You will be given complete freedom with how this project is implemented as long as its input and output matches the ones specified below.

The program will take in command line input in the following format:

```
./miniDB database.csv commands.csv outputFile.txt
```

- **database.csv** is a file that contains entries formatted like the following:

  Title, Date, Director, ID

  Title - A string entry of the Title of the Movie
  Date - A string entry for the Date (The date must be valid)
  Director - A string entry for the Name of the Director for the Movie
  ID - A unique key that identifies the Movie.

  *Each entry must have a unique ID, otherwise it must not be added to the data structure.

  An example entry is “Star Wars, 12/23/1970, George Lucas, 67388233”

  We promise that all entries will contain valid data and will be formatted as follows. No string will be greater than 512 characters.

- **commands.csv** is a file that contains a list of commands to be carried out by the program and outputted to the output.csv file.

  commands.csv entries will be formatted as follows:

  Command, Parameter1, Parameter2, Parameter 3, …

- **outputFile.txt** is the file where all of your Program output should be printed too. Your program should not print anything to stdin. The command and it's arguments that were ran must be the first line before it's outputs (only for LOOKUP & DISPLAY). A new output file should be created everytime a new session of the program is run. Output from the last session should not be present.

Program Commands

These are the commands that your program must be able to perform:

ADD	This command takes in the arguments: Title, Date, Director, ID The command adds the following data into the database. This data must be persistent and appear again in the database when the program is run again.
Example: ADD, Star Wars, 12/23/1970, George Lucas, 672312312 ADD, American Graffiti, 03/24/1968, George Lucas, 611232342 ADD, Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231 ADD, Aliens, 04/18/1986, James Cameron, 703211234	

EDIT	This command takes in the arguments: ID, Feature, Data The command edits the data entry associated with the unique ID. The edits are made to Feature, with the new Data. This should be persistent the next time the program starts.
Example: EDIT, 681231231, DIRECTOR, Ken Bone EDIT, 703211234, DATE, 04/18/2016	
REMOVE	This command takes in the arguments: ID The command removes the data entry associated with the unique ID. The data entry should not appear again, the next time the program is run.
Example: REMOVE, 681231231 REMOVE, 703211234	
LOOKUP	This command takes in the arguments: Feature, Data The command outputs all the Data matching the specified Feature into outputFile.txt. Extra Credit: Implement the use of wildcards(*)
Example: LOOKUP, DIRECTOR, George Lucas LOOKUP, DIRECTOR, George Lucas Star Wars, 12/23/1970, George Lucas, 672312312 American Graffiti, 03/24/1968, George Lucas, 611232342	

Extra Credit Example:

LOOKUP, TITLE, A*

LOOKUP, TITLE, A*
American Graffiti, 03/24/1968, George Lucas, 611232342
Aliens, 04/18/1986, James Cameron, 703211234

DISPLAY	This command takes in the arguments: Feature, Order, Max The command outputs all entries into outputFile.txt. The number of listings is limited to Max. Based on the Feature, the listings should be sorted. If the Order flag is 0, then list in ascending order, otherwise descending.

Example:

DISPLAY, DATE, 1, 2

DISPLAY, DATE, 1, 2
Aliens, 04/18/1986, James Cameron, 703211234
Star Wars, 12/23/1970, George Lucas, 672312312

DISPLAY, DIRECTOR, 0, 50

DISPLAY, DIRECTOR, 0, 50
Star Wars, 12/23/1970, George Lucas, 672312312
American Graffiti, 03/24/1968, George Lucas, 611232342
Aliens, 04/18/1986, James Cameron, 703211234
Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231

DISPLAY, TITLE, 1, 3

DISPLAY, TITLE, 1, 3
Star Wars, 12/23/1970, George Lucas, 672312312
Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231
American Graffiti, 03/24/1968, George Lucas, 611232342

Example *database.csv*

Star Wars, 12/23/1970, George Lucas, 672312312
American Graffiti, 03/24/1968, George Lucas, 611232342

Example *commands.csv*

ADD, Star Wars, 12/23/1970, George Lucas, 672312312
ADD, American Graffiti, 03/24/1968, George Lucas, 611232342
ADD, Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231
ADD, Aliens, 04/18/1986, James Cameron, 703211234
LOOKUP, DIRECTOR, George Lucas
LOOKUP, TITLE, A*
DISPLAY, DATE, 1, 2
DISPLAY, DIRECTOR, 0, 50
DISPLAY, TITLE, 1, 3
EDIT, 681231231, DIRECTOR, Ken Bone
EDIT, 703211234, DATE, 04/18/2016
REMOVE, 681231231
REMOVE, 703211234

Example *outputFile.csv*

LOOKUP, DIRECTOR, George Lucas
Star Wars, 12/23/1970, George Lucas, 672312312
American Graffiti, 03/24/1968, George Lucas, 611232342
LOOKUP, TITLE, A*
American Graffiti, 03/24/1968, George Lucas, 611232342
Aliens, 04/18/1986, James Cameron, 703211234
DISPLAY, DATE, 1, 2
Aliens, 04/18/1986, James Cameron, 703211234
Star Wars, 12/23/1970, George Lucas, 672312312
DISPLAY, DIRECTOR, 0, 50
Star Wars, 12/23/1970, George Lucas, 672312312
American Graffiti, 03/24/1968, George Lucas, 611232342
Aliens, 04/18/1986, James Cameron, 703211234
Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231
DISPLAY, TITLE, 1, 3
Star Wars, 12/23/1970, George Lucas, 672312312
Chitty Chitty Bang Bang, 12/18/1968, Ken Hughes, 681231231
American Graffiti, 03/24/1968, George Lucas, 611232342

TURN IN

For this HW you must have a **Makefile** that compiles an executable called **miniDB**, meaning you have free reign over what you do as long as you have files that compile into an executable called minidb. You will also need to create a **README**, describing your work.

Needed in Vocareum:

Makefile (creates an executable called “miniDB”)
README
Your Files needed for the compilation

A basic Test on Vocareum will be made available on **11/16/2016**

Due Date: 11/22/2016