CSCI 330 Assignment 7 (Spring 2021)

Automobile Database Shell Program (100 pts)

Goal

Your task for this assignment is to write a *shell script* that will allow its user to create, view, and modify a simple text-based database for automobiles. The script must be implemented with the Bourne shell syntax (use `bash`). You may use any features available in the version of `bash` found on `turing/hopper`, as well as any Unix command available there. Name your script file `z1234567sh`. Use your zid instead of `z1234567`, obviously.

Specification

Your script should support two ways of being run:

① When run without any direct command-line options – your program is in *interactive* mode and should prompt the user to find out what to do. Once you finish one task from the user, you should return to that initial prompt and allow the user to request to do more, until they indicate that they would like to quit. You have some leeway in how you accomplish this, but it must be possible to have it perform any of the functions from below.

② When run with direct command line options (as below) – do the task requested and exit immediately when done.

```
# dbname  - filename of database file to use
# command - which of the functions to call: "new", "insert", "display", "delete"
# param1  - first non-dbname parameter to whichever command function chosen
# ...     - placeholder for parameters between 1 and N
# paramN  - Nth parameter to whichever command function chosen
% ./z1234567sh dbname command param1 param2 ... paramN
```

Implementation

You must implement the following as `bash` functions:

- `new()` - This function is used to create a new database file. It takes up to two parameters:

  ① The filename to use for the new database file. This *must* be specified.
  ② The label to put on the first line of the new database file. If not specified, use `"Untitled database"` instead.

- `insert()` - This function is used to add a record to an existing database file. It will always take five parameters. If any of them are missing, it is an error.

  ① The filename of the database to add the record to.
  ② The make of the car to be stored in this new record. It is a string that must be longer than zero characters.
  ③ The model of the car to be stored in this new record. It is a string that must be longer than zero characters.
  ④ The year of the car to be stored in this new record. It must be a four-digit number, greater than 1921 and smaller than 2029.
  ⑤ The color of the car to be stored in this new record. It is a string that must be longer than zero characters.

- `display()` - This function is used to show record(s) found in an existing database. This will take up to four parameters, depending on the value of the second.

  ① The filename of the database to show the record(s) from. Must be the filename of a readable file.
  ② how many to show (one of `all`, `single`, or `range`)

  - `all` - Show all of the records, example follows:

```
Automobile Database
Ford, Mustang, 2008, blue with white stripes
Mitsubishi, Lancer, 2009, white
Toyota, Camry LE, 2004, black
Porsche, Cayenne S, 2007, red
```

  - `single` - Shows the single record in the position indicated by the third parameter. Notice that record #1 is on the second line, after the label.
  - `range` - Show the records in the range starting at the position indicated by the third parameter, up to and including the record indicated by the fourth.

- `delete()` - This function is used to delete records from an existing database.

  ① The filename of the database to delete record(s) from. Must be the filename of file readable and writable by the current user..
  ② how many to show (one of `all`, `single`, or `range`)

  - `all` - Delete all of the records, but not the label for the database.
  - `single` - Delete the single record in the position indicated by the third parameter. Notice that record #1 is on the second line, after the label. It is an error to try to delete a record number that does not exist.
  - `range` - Delete the records in the range starting at the position indicated by the third parameter, up to and including the record indicated by the fourth.

- `count()` - This function is used to count and print the number of rows in an existing database. It has one parameter, which is *not* optional.

  ① The filename of the database to count the records in, which must be readable by the current user.

Database Format

The format you must use for this database will be a text file, with a label for the database on the first line. This label is what is written by the `new()` function. After the label, there is line per record that is added to the database. The individual fields in the database records should be separated by commas. (“,”).

For example, the database file that produced the example for `display()` with `all` as its parameter would have contained the following:

```
Automobile Database
Ford, Mustang, 2008, blue with white stripes
Mitsubishi, Lancer, 2009, white
Toyota, Camry LE, 2004, black
Porsche, Cayenne S, 2007, red
```

Error Checking

If an error occurs, print an error message. If the script was run in interactive mode, return to the top of the loop and allow them to try again. If it was used with directly specified command line arguments, it should exit with a non-successful status code.

- Ensure that the command is spelled correctly.
- Ensure that all the required parameters to the appropriate command are present.
- Ensure that record numbers actually fit within lines present in the database file.
- Ensure that the database file exists and is readable (except for `new()`, where it shouldn't exist) and, in the case of `insert()` and `delete()`, also writable.
- If the file is empty (no records present), your script should print out a message that no records are found.

Example runs

```
% ./z1234567sh DB new "Example for Assignment"
New database created

% ./z1234567sh DB insert Ford Mustang 2008 "blue with white stripes"
Successfully added a record to the database

% ./z1234567sh DB add Mitsubishi Lancer 2009 white
Successfully added a record to the database

% ./z1234567sh DB add Toyota "Camry LE" 2004 black
Successfully added a record to the database

% ./z1234567sh DB add Porsche "Cayenne S" 2007 red
Successfully added a record to the database

% ./z1234567sh DB display all
Example for Assignment
Ford, Mustang, 2008, blue with white stripes
Mitsubishi, Lancer, 2009, white
Toyota, Camry LE, 2004, black
Porsche, Cayenne S, 2007, red

% ./z1234567sh DB delete single 2
1 record deleted

% ./z1234567sh DB display all
Example for Assignment
Ford, Mustang, 2008, blue with white stripes
Toyota, Camry LE, 2004, black
Porsche, Cayenne S, 2007, red

% ./z1234567sh DB count
3
```

```
% cat DB
Example for Assignment
Ford, Mustang, 2008, blue with white stripes
Toyota, Camry LE, 2004, black
Porsche, Cayenne S, 2007, red
```

Additional notes

- Be sure to test your script thoroughly.
- Your script file must use `/bin/bash` as the path in its shebang line.
- Make sure your script does not create any temporary files that are not removed before it ends.
- Make sure that your shell script is a regular Unix text file. (not a compiled program or C++ source code).
- This is to be done as a *shell script*. C++ is not acceptable for this assignment.
- All testing during grading will be done on `turing` and `hopper`. If the script you submit does not work there, it is a *you* problem.
- Blackboard won't allow files that end in `.sh` to be submitted, so make sure you call it `z1234567sh` (no file extension) and not `z1234567.sh`.

What to turn in?

Turn in, via Blackboard, the script file you wrote. Make sure its name is as specified above.