CSE 1320: Intermediate Programming
University of Texas at Arlington
Fall 2021
Alex Dillhoff

Assignment 5

This assignment focuses on dynamic memory allocation and linked lists.

Your code must compile without any warnings or errors and run without segmentation faults to receive credit. Points will be taken for inconsistent formatting.

1. A developer has written parking data to a binary file given the data format below. Your task is to create a program which allows the user to filter the data by a search term. Your program should allocate memory for the data ONLY when it is requested. Once loaded, display the data as shown below.

   The string data is separated by a null character.

   Requirements

   - The vehicle `struct` should be declared in a header file with a header guard. Any functions created must have a declaration in the same header file.
   - Accept two command line arguments for the file name and search term.
   - Search through the file and allocate memory for each entry that contains the given keyword. If there are multiple entries, each vehicle should be loaded in an array that is dynamically allocated as well.
   - Once the end of the file is reached, print out the vehicles that match the search term.
   - Allocated memory must be freed when the program terminates.
   - Use the file `vehicles.db` on Canvas to test your program.
   - Save your files as `problem1.h` and `problem1.c`.

   Data Format

   - `long size`
   - `int year`
   - `char *make`
   - `char *model`
   - `char *color`
   - `char *license_plate`

   Example Output

   ```
   $ ./filter_vehicles vehicle.db ford
   2013 Ford F-150 (Black) LIC#412F32Z
   2006 Ford Mustang (Red) LIC#2F2F
   ```

2. Create a program that reads in vehicle data in CSV format and allows the user to filter by the make. The program will take in a filename as a command line argument. When opening the file, the program should load all entries in to a linked list for which each node contains a pointer to a vehicle `struct`. The program will then populate a list of makes based on the data and present it to the user.

   The user should be able to select one of the possible makes via keyboard input (see example output). The program will then create a new linked list and copy the data pointers of the original list into the temporary linked list which only holds entries based on the user's selection.

   Data Format

   - `int year`
   - `char *make`
   - `char *model`
   - `char *color`
   - `char *license_plate`

   Other Requirements

   - The CSV file must be input as a command line argument when running the program.
   - Nodes must be allocated for each vehicle read in the CSV file.
   - Allocated memory must be freed before the program terminates.
   - When building a new linked list of the selected make, use the allocated data that was already created for the original list.
   - Any function declarations, `struct` declarations, and library includes should be in a corresponding header file (with header guard).
   - Save your files as `problem2.h` and `problem2.c`.

   Example Run

```
1. Alfa Romeo
2. Buick
3. Chevrolet
4. Ford
5. Honda
Select Make: 3
2001 Chevrolet Camaro Z28 (White) LIC#123313
2021 Chevrolet Corvette (Black) LIC#453FD3
```

Create a `zip` file using the name template `LASTNAME_ID_A5.zip` which includes the all required code files. Submit the `zip` file through Canvas.