Lab 10

Introduction to C
CSCI 112, Spring 2015

Objectives

- Practice open-ended program design
- Practice using a linked list
- More practice with `struct`
- More practice with designing user menus

Overview

For this lab, you will make a database, store it as a linked list, and provide a menu allows users to: print out the records, add a new record, delete a record, and search for an existing record.

You will design a C structure as your database record. You will create a linked list data structure to hold your records. You will create linked list operations: insert, delete, and print. Lastly, you will create an text-based menu to allow user to interact with your database, including searching records, adding a new record, removing a record, searching your database for a record, and printing out your database.

Struct

Using a `struct`, design a database record. This record could represent any information of your choosing. It could represent a student record, similar to your Lab 7. Or you could choose to represent a book, a movie, an employee, or a merchandise item from a store, just to name a few examples. You are encouraged to choose a domain of relevant interest to yourself. For example, do you like video games? Create a database of video game titles. Or perhaps your uncle owns a ranch and you wish to make a database of his agricultural equipment. Choose something of meaningful interest to you and be creative!

Your record should contain at least five data fields, although providing more than five is encouraged. For example, if I design a structure to represent a Car record, my fields might be the make (string), model (string), year (int), color (custom enum), and mileage (float). This data can be anything you deem relevant to your database. You may use any mixture of types you like, including arrays, pointers, and other structures, as relevant to your domain.

In addition to your data fields, make sure to add a pointer to the next record. Finally, wrap this structure with a `typedef` type definition.

Linked List Database

Using your `struct` template of a record, you will now create a new linked list. To do so, first create (at least) five new records and populate with values. You will hard-code this data in your code so that your program will begin with a populated database. Link this records together to form a linked list. Make sure your initial list is in sorted order. You will access your data structure by storing a link to your head (the first record).

Linked List Operations

The following functions describe the linked list operations you must complete. You should refer to the textbook, lecture slides or other internet resources to learn how to accomplish these operations. Make sure you adapt the operations for your specific `struct`.

Searching for an Element

Create a function to search for an element in your database. You should assume your list is properly sorted. This function will walk your list trying to find your element. It should return the element contains your search criteria. If the element is not in the list, return the element "closest" to your desired element without going past its location is the list (this will be useful when designing your insert element function). You will use this function as a helper function in the subsequent three functions.

Inserting an Element

This method will insert a single element into your linked list database. You must insert the element into list while maintaining a proper sorted order. Therefore, first search your list to find the location in which to insert (i.e., after the "closest" element mentioned in the search function).

Deleting an Element

Create a function that deletes an element from a linked list. Given a piece of information (such as an id number for a Car), search your database for the record and delete it. Make sure to update your pointers correctly or you may potentially break your linked listed.

Printing your Database

This method allows the user to print out the records from your database. If the user chooses this option, you will iterate over your linked list and print each record. You may choose how to organize and display this output. You should produce clean readable output.

User Menu(s)

Finally you will create a hierarchy of user menus. You have freedom in your design of your menus and the organization of your code. You will have a main menu which should give the users the following abilities:

- Print out the database
- Search for an element
- Insert an element
- Delete an element

You should customize the menu text to suit your database application. For each option the user chooses, you will then prompt for relevant information.

Protip: Realize the grader will be seeing your program for the very first time when they run your program. Make sure your menus are organized, neat, descriptive, and intuitive. You will be graded on the quality of your menu.

For example, if the user chooses the “Insert an Element” option, you should then prompt them to enter in information for each and every field in your structure record. You will then create a new structure record, populate it with the data, and call your insert element function, which should add the record to your database.

Protip: Make sure you validate the user’s input and repeat the menu if the user makes an invalid choice. Be sure to test your user menu with both valid and invalid choices.

As another example, if the user chooses the delete operation, you first need to prompt the user for a “search field” (any field from your record that you choose). Then call your search function to find a node, verify the found node is your target node, and lastly, delete the target node and update your pointers. If the node is not found, you should print a message informing the user.

Your search and print procedures should follow in similar manner.

Protip: This lab gives you freedom in your design choices. Remember to practice good program organization. Strive to make a separate function for each task you need to accomplish.

Try to avoid combining unrelated tasks together. For instance, make sure to separate out your menu interactions from your linked list operations. You will likely have a function that helps a user to insert a new record and proceeds to input their data with a series of prints and scans. This menu-related function should be separate from your linked-list-function that will insert the element into the linked list.

Grading Criteria

Given the freedom in design, there is no Example Execution to show. Use the Checklist below to ensure you remembered all requirements of the program. Use the Rubric to understand how your program will be graded.

Checklist

1. `struct` record contains at least five data fields
2. `struct` record contains a pointer to next element
3. custom type (`typdef`) of your `struct` record
4. Initial new (sorted) linked list with at least five populated data records
5. Linked list operations
   - search for an element
   - insert an element
   - delete an element
   - print out database
6. User menu
   - enter a new record
   - search for an existing record
   - delete an existing record
   - print out the contents of the database

Rubric

The grading criteria for Lab 10 will differ from the previous labs. For this lab, you will be graded on the organization of your code. For example, if you write all of your code in your main function, you will receive a low score in this category. You will also be graded on the functionality, ease-of-use, and organization of your user menu. The category "Linked List Functionality" grades you on your insert, delete, print, and search functionality of your database.

Submission	10 points
Compilation	10 points
Comments	20 points
Code Organization	10 points
Menu Functionality	20 points
Linked List Functionality	30 points

Compile & Test

Compile your program using this `gcc` command. `c99` is a shortcut for running `gcc -std=c99`, which uses the C99 standard instead of the default C89 standard.

```
$ c99 lab10.c -o program10
```

Resources

- You should attend the Friday Lab session to seek assistance from the TAs and CAs.
- For general questions, check the D2L `Lab 10 Help Forum` to see if another student has already asked your question. Otherwise, post your question on the D2L forum. This forum is intended for general questions that will benefit all students. Do not paste your code nor give away any specific answers to the lab on this forum.
- For specific questions, attend the weekly lab session or attend the TA's office hours.
- You are encouraged to use resources or tutorials on the internet to learn unix or C. Check the class resource list on D2L for some links to useful resources.

Submission

- Assigned: April 24st
- Lab Day: May 1st
- Due Date: Tuesday May 5, 8:00am **(* note the day of the week *)**

1. Make sure you included ample and informative comments – it is 20% of your grade!
2. Rename your C file to `last_first_lab10.c` and substitute your last and first name.

   Protip: If you fail to follow the above file naming conventions, your program may not be graded automatically and you will lose points.

3. Submit your `.c` file to the D2L dropbox Lab10

Each student will complete and submit this assignment individually. Submit your work before the deadline as late work. Given the tight time constraints of final's week, late submissions will not be accepted for Lab 10.