APS105 – Computer Fundamentals

Lab 9: Search and Link

Winter 2021

In this lab you are going to implement and exercise a digital phone book utility which allows you to search contacts with their phone numbers.

You can assume that an entry in your phone book consists in a name and several phone numbers. You can assume that in the name of the contact we store the string of characters that the user enters which may be of the form "Last name First name", as in "Chen Johnny", or “First name Last name” as in "Jane Dunne”. The user may enter a contact two or more times without realizing it and your program will help the user detect and delete duplicates, which are defined as contacts that have a phone number in common.

The command menu for this utility is as follows:

Enter a command by number

1. List all contacts in alphabetical order

2. Print phone(s) for a contact

3. Find duplicate entries

4. Delete contact

5. Exit the program

You will be responsible for implementing these commands. A program with starter code will be provided for you. You should insert your code where indicated in this program, then rename this file to Lab9.c. A populated list of contacts will be provided for you in “`contactList.h`” that you should include in your program with #include “`contactList.h`”.

Programming requirements and notes:

- You should only scan the .h file information once (as if it came from a disk drive file)
- You must use at least one ”struct” in your program to extract the information from the .h file. You may use the following prototype to develop your function to do so:

```
void convertInputToDataStructure();
```

Programming hints:

- Look at what you have to do with the list. You might want to create data structures that are aligned with the manipulations you need to perform.
- Create a simple version of .h file during your development (say "`simpContList.h`". A simple .h file may help in testing the functionality of various sections of your code during testing. After each successful pass, you may want to add to the complexity of your .h file and eventually use the provided "`contactList.h`".
- Write a routine for debugging/development purposes only that will dump out your data structure in a way that allows you to easily tell what is working and what is not working. Remove the calls to this routine for code checks and submission at the end.

In the .h file, each name starts with a capital letter and each phone number with a digit. You can assume that phone numbers are a maximum of 12 digits.

IMPORTANT: When you submit your code for automarking, make sure the included file is contactList.h, spelled exactly like that, including upper- and lower-case letters. Otherwise your code may not compile in the automarker and you will get a zero (0), even though it might work on your laptop.

```
const char* contactList[] = {
"Johnny Chen",
"4164769988",
"6478765679",
"4165463459",
"Jane Dunne",
"4167889900",
"4167886655",
"Stephen Doyle",
"6477889901",
"Chen Johnny",
"6478765679",
"Heidi Klum",
"4169876566",
"6476355533",
```

```
"4167665445",
"Yaya Dyne",
"4167865443",
"John Doe",
"4164164164",
"Xavier Zeda",
"5147654534",
"Johnny Doe",
"4164164164",
};
```

Required Output Format for Auto-Marking

As before, the automarker will be looking for a certain output format. Please follow the format described below.

Enter a command by number

1. List all contacts in alphabetical order

2. Print phone(s) for a contact

3. Find duplicate entries

4. Delete contact

5. Exit the program

Your input: 2
Contact: Johnny Chen

Phone(s): 4165463459 6478765679 4164769988

Enter a command by number

1. List all contacts in alphabetical order

2. Print phone(s) for a contact

3. Find duplicate entries

4. Delete contact

5. Exit the program

Your input: 2
Contact: John Malvern

Contact not found

Enter a command by number

1. List all contacts in alphabetical order

2. Print phone(s) for a contact

3. Find duplicate entries

4. Delete contact

5. Exit the program

Your input: 3

Chen Johnny and Johnny Chen have a phone number in common
John Doe and Johnny Doe have a phone number in common

Enter a command by number

1. List all contacts in alphabetical order

2. Print phone(s) for a contact

3. Find duplicate entries

4. Delete contact

5. Exit the program

Your input: 1

Name: Chen Johnny

Phone(s): 6478765679

Name: Heidi Klum

Phone(s): 4167665445 6476355533 4169876566

Name: Jane Dunne

Phone(s): 4167886655 4167889900

Name: John Doe

Phone(s): 4164164164

Name: Johnny Chen

Phone(s): 4165463459 6478765679 4164769988

Name: Johnny Doe

Phone(s): 4164164164

Name: Stephen Doyle

Phone(s): 6477889901

Name: Xavier Zeda

Phone(s): 5147654534

Name: Yaya Dyne

Phone(s): 4167865443

In case a contact is not found, your program should output “Contact not found”. In case there are no duplicates in a duplicate search, your program should output “No duplicates found”.

Submitting Your Program for Auto-Marking

The total of 6 marks on this lab:

1. By an auto-marking program for 6 marks out of 10. You must submit all of your program files through the ECF computers for marking. Long before you submit your program for marking, you should run the exercise program that compiles and runs your program and gives it sample inputs, and checks that the outputs are correct. Similar to previous labs you should run the following command:

/share/copy/aps105s/lab9/exercise within the directory that contains your program.

This program will look for the file Lab9.c in your directory, compile it, and run it on a some of the test cases that will be used to mark your program automatically later. If there is anything wrong, the exercise program will report this to you, so read its output carefully, and fix the errors that it reports.

2. Once you have determined that your program is as correct as you can make it, then you must submit your program for auto-marking. This must be done by the end of your lab period as that is

the due time. To do so, go into the directory containing your program and type the following command:

/share/copy/aps105s/lab9/submit

This command will re-run the exercise program to check that everything looks fine. If it finds a problem, it will ask you if you are sure that you want to submit. Note that you may submit your work as many times as you want prior to the deadline; only the most recent submission is marked.

The exercise program (and the marker program that you will run after the final deadline) will be looking for the exact letters as described in the output in this handout, including the capitalization. When you test your program using the exercise program, you will see that it is expecting the output to be exactly this, so you will have to use it to see if you have this output correct.

Important Note: You must submit your lab by 11:59 p.m. on the day of your assigned lab period. Late submissions will not be accepted, and you will receive a grade of zero.

You can also check to see if what you think you have submitted is actually there, for peace of mind, using the following command:

/share/copy/aps105s/lab9/viewsubmitted

This command will download into the directory you run it in, a copy of all of the files that have been submitted. If you already have files of that same name in your directory, these files will be renamed with a number added to the end of the filename.

After the Final Deadline — Obtaining Automark

Briefly after all lab sections have finished (after the end of the week), you will be able to run the automarker to determine the automarked fraction of your grade on the code you have submitted. To do so, run the following command:

/share/copy/aps105s/lab9/marker

This command will compile and run your code, and test it with all of the test cases used to determine the automark grade. You will be able to see those test cases output and what went right or wrong.