Etude 13 - Bug Squashing
Levi Schimanski
Student ID 6923634

To whom this may concern, this code was a good start but had a few things in need of changing and there are a few things you should keep in mind for future coding projects.

Memory leakage
When you allocate memory for things you should either be using that memory or free it up so it doesn't clog up the computer's memory with unused data that is no longer pointed to. Val was being allocated new memory everytime the while(1) loop iterated and each time it was used once and never used again, old information wasn't even accessible because val was being pointed to it's newly allocated data.

Sanity Checking
When users are expected to input data you can never be sure what they might input, it could be crazy, it could be malicious. Either way it is always good to put failsafes in place so that when they enter input it can only be a certain length and won't cause errors in the program or have other negative effects. I added a max count of 10 to this fgets function because the buffer can only fit 10 characters. However, I recommend also adding code to check that there are not too many arguments, that the incoming data is correct and that anything the user enters is restricted to what they should be allowed to input.

Insufficient User Interface
User interface is an important part of any program, in your original program, users would not know what they were doing, or what the program was even for. I recommend adding a README and giving more descriptive prompts to the user so they can know what to do without going through the code. I added a bit to the user interface and so now, if you input peoples data, the program will ask you to type a number from 0 to 4, 0 = exit program, 1 = email, 2 = firstname, 3 = last name, 4 = phone. Once you have done this you will be asked to type in a string and it will return the email, first name, last name and phone number of the person/email/phone you typed in. But, this is still very rudimentary and can be added to a lot.

Syntax Errors
A small problem as it was consistent throughout the code, but correct spelling of emailAddress is nice for other users. Also you are using camel case (emailAddress) when snake case is most often used in c (email_address), once again it was consistent however so not too bad.

Lack of Comments
Comments throughout your code are very useful for other programmers and yourself to understand what your code is meant to be doing at any given area. Your code had no comments making deciphering what was happening more difficult.

Overwriting data

This program is about taking in a database, keeping it, ordering it and finding things within it. If you lose pieces of data from the database unintentionally the program is doing something wrong. For a start, when you allocated space for s, you did it outside the first while loop, this meant that everytime the parts of the struct were being added, you were overwriting the old parts. Then when ss was being pointed to the struct with all the new data in it, it was the same struct that all the other elements of ss were pointing to. In addition to this, I also changed the sort functions to sort the arrays in ascending order. You had the general idea there, but they weren't working properly because you needed to put the struct holding the bigger/smaller value into a temporary place. This would mean that when you go to make that struct equal to the other struct, the first struct is not lost and the index of the other struct can then be made to point to the first struct. I am not convinced a sort function is needed as the program currently is however.

Miscellaneous Changes Made

Instead of a 1 for yes matching data has been found and a 0 for no it has not been found, now if matched data is found it will be printed along with its associated data. However, if more than one matched data piece is found, only the index of one struct will be kept and only the data in that struct will be printed.
TLDR: matching data is printed

I removed the & (address modifier) from the 3 strings in fscanf because s is a pointer to the struct S and we don't need the addresses of the char pointers, we need the actual pointers so that we can find the start of the strings.
TLDR: removed & from fscanf (important)

I changed the find functions to return i which is the index of the struct holding the firstname/lastname/email/phone that match what you type in for each respective case. This is important because the function is now meant to find data, returning 1 just told you that data has been found, not where it is.
TLDR: find functions return location of matching data

I also changed the for loop to a while loop so that we get all people's details not just 50, the loop only stops once we hit the end of the file or when we run out of space for more. If you wanted only 50 structs at a time you might want to change this back however. Or if you wanted more then 100 you may want to allocate more space and also change the amount of space and all references to it, to a global variable, so allocating more space is just a matter of increasing this variable.
TLDR: increased allowed database length to maximum space of program

I removed the gets functions and replaced them with fgets, as gets has been removed from c for a while.
TLDR: changed out old code for new code

I separated the while loop that is initialising data from the while loop that is sorting and finding data, so that the searching while loop isn't searching an incomplete list.
TLDR: changed program so it doesn't search database before database is completed

I changed fgets(buffer….) to fgets(val….) and removed strcpy(val, buffer) as this was making you overwrite buffer, which can only be length 10 and then changing val to equal buffer, limiting the amount of characters that val could be to 10.
TLDR: changed code so val can hold more than 10 characters (important!)

I added an if statement that would break from the while loop and therefore program if command == 0.
TLDR: added a way to instantly leave program

Conclusion

With all these changes your code works fine now and even has a shiny new feature. However, sanity checking and an improved user interface still need to be added. Also, because the phone numbers are being stored as an int, the leading zeroes are being lost, defeating the purpose of this program as a database, I recommend changing it to string format like the others.

Signed
Levi Schimanski