

Assignment 3 (10% of Course Total)

Due date: 11:59pm, Nov 8, 2024

This is an individual assignment. This assignment might be graded automatically. Make sure that your code compiles without warnings/errors and produces the required output. Also use the file names and structures indicated as requested. Deviation from that might result in 0 mark.

Your code **MUST** compile and run reasonably efficiently in the CSIL machines (see the **Before You Submit** section for details). It is possible that even with warnings your code would compile. But this still indicates there is something wrong with your code and you have to fix them, or marks will be deducted.

Your code **MUST** be readable and have reasonable documentation (comments) explaining what it does. Use your own judgement, for example, no need to explain `i += 2` is increasing `i` by 2, but explain how variables are used to achieve something, what a certain loop is doing, and the purpose of each `#include`.

Description

There is only 1 question in this assignment. However, there are several files you need to submit. Write your answer in the corresponding file as instructed along with your student information. You can include any libraries that are covered in class (i.e., `stdio`, `stdlib`, `string`, `math`, `stdbool`). You can also write your own helper functions. **Only one of these files should contain the main function.**

Question 1 [16 marks]

In this question you are going to extend the program you made in Assignment 2 by providing more functionalities. You are again going to write **your own main function** so the program runs.

Create a program that reads all the superhero information from a file and provide an interface for the user to query superhero entries. Your program must support the following functionalities as options:

- **Option 1: to load a data file** – ask the user to input the name of the file containing superhero information and load the entries from that file (report # of entries). Note that the file can be non-existent (if so the program should print the error and keep running), but if it exists you can assume the format inside that file is valid and consistent, with each line having at most 300 characters. It is possible for the user to load a different file by choosing this option again, which will replace the previously loaded entries. You can assume the filename has at most 50 characters and no space.
- **Option 2: to list entries sorted by height** – list the superheroes from shortest to tallest. Within entries with the same height (if exist), they should be ordered by name (as determined by `strcmp`).
- **Option 3: to list entries sorted by name** – list the superheroes sorted by name (as determined by `strcmp`). Within entries with the same name (if exist), they should be ordered by height.
- **Option 4: to lookup a superhero** – ask the user for a superpower for at most 50 characters (space included), then report all entries with superpower having the input as a substring. There can be more than one entry. The lookup is case-sensitive (i.e., “fly” and “Fly” are not the same).
 - If one or more entries are found, print all the information. Any order is acceptable.
 - If no entry is found, print “No such superhero on record.”You can assume the user will always enter at most 50 characters.
- **Option 5: to terminate the program** – thank the user and end the program.

Your program must meet these requirements:

- Include the provided header file (.h file) and use the functions defined in the corresponding implementation file (.c file) in your driver file (where main is), do not redefine those functions.
- Use a dynamic array of Superhero struct pointers to store the superhero information. This is the recommended approach from A2 (instead of a static array with a fixed size) as we might change the number of entries in the provided file, and dynamic arrays can accommodate that variation.
- There must be no memory leaks (e.g., your program requests some memory but does not release them all before it terminates). In CSIL you can use the Valgrind tool as described to check for that.
- Start with a fancy banner. There is no specific requirement besides it must include your name, 9-digit SFU ID, and your SFU email address. Let your creativity shine, just nothing offensive.
- If a functionality (e.g., list, lookup) is requested by the user but no file has been loaded, the program should print an error message telling the user that no file have been loaded and prompt the user to do so (e.g., “No superheroes file loaded. Load one first.”).
- The sorting for Options 2&3 can be done using any sorting algorithms covered in class, including the built-in `qsort`, which requires to you to write your own compare functions. See **a3_superherolib.h** for details.

And here are some hints for you to write your code:

- The program interface is essentially a do-while loop where in each iteration it asks for an option and determines what to do, with a repeating conditional of the terminating option has not been inputted. It is a common approach for menu-based interfaces.
- Since each item in the dynamic array for the superhero entries are pointers, be careful with how you are casting the pointer variables in the compare functions for `qsort` (they are pointers to the items) – suppose the dynamic array has the type `A**`, then the variables should be cast to `A**`.
- The `scanf()` function works slightly differently between different formats. One difference is how it handles leading and trailing newlines (look up “dangling newline character” if you are curious). One option to get around this is put a space in front of the format string (e.g., “ %d”, “ %s”). You are encouraged to explore different ways to get around this issue.

*The entries are randomly generated using ChatGPT with a specific format and then lightly modified. Hence, the content might not make sense and have very similar wording patterns – no need to worry.

Include the function definitions corresponding to `a3_superherolib.h` (and your helper functions, if any) in a source file named as **a3_superherolib.c**. Remember the .h file only contains the function headers.

Include your main function definition (and of your helper functions, if any) in another source file named as **a3_superheroLookupSystem.c**. We call this the driver file because this is where the program starts.

To determine where to place a helper function, think about its purpose. If it provides functionality with the Superhero structs (e.g., create/clear Superheroes, compare Superheroes), it is a library function; if it provides functionality of the program (e.g., print a fancy banner), it is a program function. Incorrectly placing these functions will lead to mark deductions in the Coding Style category.

Here are some sample input and output (you can assume user input is always valid, but could be incorrect):

```
=====
===== Superheroes Lookup System =====
===== Victor =====
===== 012345678 =====
===== no-reply@sfu.ca =====
=====
Press numbers 1-5 for the following options and then enter:
1) to load a superheroes file.
2) to list superheroes sorted by height.
3) to list superheroes sorted by name.
4) to look up a superhero.
5) to terminate the program.
Option: 9
Unrecognized option.
=====
Press numbers 1-5 for the following options and then enter:
1) to load a superheroes file.
2) to list superheroes sorted by height.
3) to list superheroes sorted by name.
4) to look up a superhero.
5) to terminate the program.
Option: 1
Enter the full name of the superheroes file (with extension): abc.txt
Error in opening the file, check if it is available.
=====
Press numbers 1-5 for the following options and then enter:
1) to load a superheroes file.
2) to list superheroes sorted by height.
3) to list superheroes sorted by name.
4) to look up a superhero.
5) to terminate the program.
Option: 1
Enter the full name of the superheroes file (with extension): superheroes.txt
100 entries read.
=====
Press numbers 1-5 for the following options and then enter:
```

Figure 1. Inputting an incorrect option (9), opening a non-existent file (incorrect input) then an existing file (with 100 entries).

```
Option: 2
Superhero #1
Ember
5'5"
Heat Generation: Ember radiates heat and can ignite her surroundings, using her powers to fight against cold-hearted villains.
=====
Superhero #2
Inkwell
5'5"
Ink Manipulation: Inkwell brings drawings to life, using animated illustrations to aid in battles against evil.
=====
Superhero #3
Mystic Echo
5'6"
Sound Manipulation: Mystic Echo can harness and amplify sound waves, using them for sonic blasts or stealthy stealth missions.
=====
Superhero #4
Nightingale
5'6"
Sonic Voice: Nightingale's voice can create sonic waves, using sound to incapacitate enemies and support allies.
=====
Superhero #5
Pulsar
```

Figure 2. Listing superheroes sorted by height (note those with the same height are sorted by name).

```
Option: 4
What is the superpower, enter in part or as a whole (50 character max.)? strength
No such superhero on record.
```

Figure 3. Looking up for "strength" which has no result (but with "Strength" there will be one – Iron Guardian).

```
Option: 4
What is the superpower, enter in part or as a whole (50 charater max.)? Sound
Superhero #1
Echo
5'8"
Sound Manipulation: Echo can manipulate sound waves, using them to create barriers or disorient opponents.
=====
Superhero #2
Echo
5'9"
Sound Manipulation: Echo can amplify and redirect sound, using sonic waves as weapons and for stealth.
=====
Superhero #3
Gale
```

Figure 4. Looking up for “Sound” (4 results, order doesn’t matter) – yes there are actually name duplicates in the file!

Coding Style [4 marks]

Your program should be properly indented, have clear and meaningful variable names (e.g., no single-letter variable names except loop iterators) and enough white space and comments to make it easy to read. Named constants should be used where appropriate. Each line of code should not exceed 80 characters. White space should be used in a consistent manner. Remember to include your information.

Keep your code concise and efficient. If your code is unnecessarily long or inefficient (e.g., hard-code for all possible cases, extraneous function calls), we might deduct marks. To help you to get into the habit of good coding style, we will read your code and marks will be deducted if your code is not styled properly.

Before You Submit

Before submitting your assignment, make sure you have test-run your code in our CSIL machines using the steps covered in class: **open your assignment folder in VS Code, use the gcc command with options in the terminal to compile your code, and run the executable**. We will use these steps when marking your submission and will deduct marks if your code does not compile/run – to maintain fairness we do not accept reasons like “but it worked on my computer” or “but it ran just fine in my IDE”.

The Makefile provided in this assignment is used by a command in the CSIL machines called “make” to quickly compile your code. It is especially useful if you have multiple source files. To use it, type the following command in the prompt (make sure you are in the directory with all the files of Assignment 3):

```
$ make test1
```

The example above illustrates how Question 1 is compiled into an executable called “test1” when using the Makefile. Replace the “test1” with “test2”, “test3”, ...etc. for other questions. You can then run the executable by typing **./test1** to test your code for Question 1. If you make changes to your code, use the make command again. You can also use **make all** if you want to compile all your code at once.

The header file a3_superherolib.h is there to make the compilation work. Take a look at it for information about how each function should work. You might have to modify it and you will have to submit it this time.

Submission

Submit **only the 3 source files** (a3_superherolib.h, a3_superherolib.c, a3_superheroLookupSystem.c) to CourSys. Refer to the corresponding Canvas assignment entry for details.

Assignment late penalty: 10% per calendar day (each 0 to 24 hour period past due), max 2 days late.

Academic Honesty

It is expected that within this course, the highest standards of academic integrity will be maintained, in keeping with SFU's Policy S10.01, "Code of Academic Integrity and Good Conduct." In this class, collaboration is encouraged for in-class exercises and the team components of the assignments, as well as task preparation for group discussions. However, individual work should be completed by the person who submits it. Any work that is independent work of the submitter should be clearly cited to make its source clear. All referenced work in reports and presentations must be appropriately cited, to include websites, as well as figures and graphs in presentations. If there are any questions whatsoever, feel free to contact the course instructor about any possible grey areas.

Some examples of unacceptable behavior:

- Handing in assignments/exercises that are not 100% your own work (in design, implementation, wording, etc.), without a clear/visible citation of the source.
- Using another student's work as a template or reference for completing your own work.
- Sharing your work with or making your work available on any platform for anyone who would benefit from it (e.g., other students in the course).
- Using any unpermitted resources during an exam.
- Looking at, or attempting to look at, another student's answer during an exam.
- Submitting work that has been submitted before, for any course at any institution.

All instances of academic dishonesty will be dealt with severely and according to SFU policy. This means that Student Services will be notified, and they will record the dishonesty in the student's file. Students are strongly encouraged to review SFU's Code of Academic Integrity and Good Conduct (S10.01) available online at: <http://www.sfu.ca/policies/gazette/student/s10-01.html>.

Use of ChatGPT or Other AI Tools

As mentioned in the class, we are aware of them. I see them as helpers/tutors from which you can look for inspiration. However, these tools are not reliable and they tend to be overly confident about their answers, sometimes even incorrect. It is also very easy to grow a reliance to them and put yourself at risk of not actually learning anything and even committing academic dishonesty. Other issues include:

- When it comes to uncertainty, you won't know how to determine what is correct.
- If you need to modify or fine tune your answer, you won't know what to do.
- You will not be able to learn the materials and thus will not be able to apply what you learn in situations where no external help is available, for example, during exams and interviews.

Bottomline is, if you ask these tools to give you an answer and you use the answer as yours, you are committing academic dishonesty by claiming work that is not done by you as yours. You can, however, ask general questions like "how do I write a for-loop?", "explain mergesort to me" and use the answers to help you come up with your own answers – make sure you have cited it at the top of your code/essay as comments by stating what tool you used what questions you asked, and how you used the answers. Type everything yourself.

Note that different instructors might have different policies regarding the use of these tools. Check with them before you proceed with assignments from other courses.