PROG24310: Programming Languages

Evaluation: 10 points, 10% of your final grade.
Due date: See SLATE.
Late submission: 10% per day penalty, up to 3 days.

In this exercise you are given file **quotes.txt** that has Computer Science quotes. Your goal is to sort the quotes by length using **Selection Sort** algorithm and save them in the file **output.txt**.
You must use the program template **main.txt** file given to you.

Main Requirements:

1. You absolutely must use the program template **main.txt** file. You can't modify prototypes or code inside **main()** function! You must add required header files and create functions using the function prototypes set in the template.
   You'll get a **-3** points penalty if this requirement not met.

2. Please check **output.txt** file – that's the file your program must generate (**OUT_FILE** macros).

3. Please check **image1.png** file – that's the console output your program must show!

4. You must use defined **MAX_QUOTES** (maximum number of quotes that can be read from the **quotes.txt** file) and **MAX_LEN** (maximum length of the string input in **fgets()** function.

5. Your program must correctly allocate memory on the HEAP for each quote (i.e. for each element of the array of pointers **quotes**).
   You'll get a **-3** points penalty if this requirement not met.

6. You must check and deal with potential problems with memory allocations or file IO. For instance, malloc/calloc can return 0, file(s) cannot be opened or created, there are more quotes in the file than **MAX_QUOTES**, etc.

7. Your program must still work if the number of quotes in the file exceeds **MAX_QUOTES**.
   Please provide a notification about that, but do NOT terminate the program – i.e. stop reading quotes from the file in the **read_in()** function and continue with the rest of the program.
   Please test this by settings **MAX_QUOTES** to 5,6,7.
   Please check **image2.png** that shows what happens when **MAX_QUOTES** is set to 7.

8. You must use **Selection Sort** algorithm to sort by length of the quotes. **No other sorting algorithm will be accepted!** Quotes with the same length must be sorted lexicographically.

9. Your solution should have optimal time and space complexity, and modular design.
   There should be no compilation warnings or commented out "debugging" code.

Hints:

1. After quote is read from the file, you must remove new line '\n' and return '\r' characters.

2. Each pointer of the array of pointers **quotes** should point to a memory allocated on the heap. See **Exercise 7.3** for many useful hints.

3. The length of the string/quote determines how much memory must be allocated for it. Don't forget about null-terminating character! I.e. for a string "hi" you'd need to allocated 2+1 bytes. Also, you might want to use calloc() function.

4. For **selection sort** algorithm it's enough to "swap" pointers! I.e. if pointer **ptr[0]** points to string "hello" and pointer **ptr[1]** points to string "hi", then you simply "swap" them so pointer **ptr[0]** points to shorter string "hi" – see how swapping works in **Exercise 7.2 and 7.3**.

5. In the given **output.txt** file check lines **9** and **10**! If two quotes have the same length, then you'd need to compare them **lexicographically**. Check function **strcoll().**

6. Check the Selection Sort algorithm and C code: https://www.geeksforgeeks.org/selection-sort/ In the **selection_sort()** function you'd need to use the length of the string/quote for sorting.

Submission:

This is an individual assignment. Even partially copied code will be subject to regulations against academic integrity. Do NOT discuss your solution with anybody else. Posting this assignment and/or solution(s) on the Internet is a violation of the Academic Integrity policy.

- Please make sure your code is POSIX-compliant (works in NetBeans/Cygwin)!
- Please save **main.c** as a text with extension **.txt**
- You must upload **main.txt** file to the Assignment Dropbox.
- Your submission must be unique or have references.
- Please **self-evaluate and self-grade your code** in the comments section of the Dropbox
- **Late submissions are penalized 10% / day (1 point / day).**
  Per FAST policy, submissions won't be accepted after 3 days.

Grading Scheme:

- See **Main requirements** and also **Course_Introduction.pdf**.
  Deductions will be applied if partial functionality is provided.

- You'll get **zero grade** if your code doesn't compile in POSIX environment. No exceptions.
  Compilation warnings are considered to be **major mistakes**.

- "Debugging code" or commented out "old code" is a **minor mistake.**

- Please check submission requirements for any additional deductions.