Quiz 3 - Wednesday

July 31, 2024

Instructions

- You have time to finish the quiz before the lab session is over. **No need to submit anything for these quizzes**, but you **must** show your codes and results to your TAs before you leave the class. TAs may ask you questions and request you to work with your codes, e.g. asking you to run and compile, add/remove, and debug your program. TAs will provide you with feedback and your grade right away, but the grades will be released on Avenue a few days later.
- During the quizzes using AI generative models, like ChatGPT, is **NOT** allowed, as I believe quizzes are easy enough to be handled. But you can search on the internet, take a look at the lecture notes, talk to your friends, or even having your TA's help.
- The quiz starts at the beginning of the lab sessions, and only those who are present in the class in person can take the quiz. Please don't be late and make sure you are in class 15 min before the quiz starts.

1. **Splitting code into multiple files (1.5 points)**

   (a) Create a header file "q1.h" to include that implements both uncompleted functions in the following snippet: get_nums should take in a reference to your array, populate it with 10 random ints between 0 and 9, and return nothing. my_average should take in an array of ints and return their average.

   ```
        #include <stdio.h>
       #include <stdlib.h>

       int main() {
           int arr[10];
           get_nums(arr);
           float av = my_average(arr);
           printf("The average of the array is %f\n", av);
           return 0;
   }
   ```

   (b) Create a makefile to compile and clean these files, defining at least 4 macros.

2. **Pointers (0.5 points)**
   Fix the following code snippet so that the output is 5. What did you have to change and why?

```c
#include <stdio.h>
#include <stdint.h>

void f(int* b) {
    b = 5;
}

int main() {
    int a = 2;
    f(*a);
    printf("%d\n", a);
    return 0;
}
```

3. **Dynamic memory allocation (1 point)**
   (a) Begin by creating an array with manually allocated space for one float in it- set this value to 0.0
   (b) With a while loop, ask the user how many elements they want to add to the array. Allocate this additional space in your array and then get the values from the user
   (c) Once the user identifies that they want to add 0 elements, exit the while loop
   (d) Print out the range (maximum value-minimum value) of values in the array and free the memory you previously allocated

4. **File input (1 point)**
   Using fopen, read the contents of sample_txt.txt, remove all the vowels (including 'y'), and rewrite this vowelless version of the text to a new file called consonant.txt

5. **Structs (1 point)**
   (a) Create a struct "person" to represent a variety of information about someone (first name, last name, age, boolean representing whether they own a pet)
   (b) Write a function "print_person" that takes in an element of person and prints it all out in a readable format in one line (ex. "John Doe, age 53, does not own a pet" or "Jane Smith, age 17, owns a pet")
   (c) Write a function "birthday" that takes in an element of person and adds one to their age