Prelab 0

Programming isn’t just about writing a program for one-time use to solve some problem. Large pieces of software (e.g., Photoshop) are constructed by creating special-purpose functions that can be used as building blocks to create even more powerful functions.

A programming language like C provides the bare minimum set of functions, e.g., input/output (I/O) functions like `scanf` /`printf` and `fscanf`/`fprintf`. It doesn’t need to provide much more than that because the real power of a general-purpose programming language is that it allows the user to expand the language by creating libraries of new functions. In fact, even the I/O functions for C are provided in the form of a library.

All of the prelabs and lab assignments this semester will involve the specification of a function or set of functions that need to be implemented. (This is a common way that programmers are tasked in the real world.) If the specification includes the interface/prototype (name of the function, its parameters, etc.) of a function then the function must be implemented accordingly so that it can be called in the way that is expected. In many cases the function will be specified more generally in terms of what information is available to the function and what is expected to be returned by that function, and it is up to the programmer to determine the details of how the function is to be called.

Regardless of how a function is specified, it is the responsibility of the programmer to document how the function is to be used and to include comments in the code to assist future programmers who may be tasked to make changes to it. It is also the programmer's responsibility to verify the correctness of the implemented function. Usually that involves creating a testing program that calls the function using programmer-generated test data to verify that the function produces correct results.

For this prelab you are tasked to implement a function that will be passed an array of integers; an integer giving the size of the array; and a *query* integer to be searched for in the array. The function will return a nonnegative number representing the number of occurrences of integers in the array that are *greater than or equal* to the query integer. There is no prototype specified for the function, so you're free to choose the name of the function and the order in which the parameters are passed.

That's it. That's all that's required for this prelab. Will you need to create a test program to verify the correctness of your implemented function? Technically, no. If you have practical programming proficiency from 1050 then you should be able to write the code for your function on a piece of paper and *hand-execute* it to verify that it will compile and work correctly when typed into a computer. After all, the computer is just a tool for automating the execution of your code, so it should do exactly what you expect it to do.

If your approach to programming in the past has been to type in a bunch of code and then make changes in response to error messages until it compiles, and then make changes to the code until it gets the correct result using TA-provided test code, then it is highly likely that you don't yet have the level of practical programming proficiency needed to be successful in this class. However, that's not necessarily a problem if you are diligent early on about doing the assigned prelabs *entirely on your own* so that you know for sure that you have the programming proficiency necessary to complete the lab assignments *entirely on your own*.

If you find yourself looking for code on the internet or seeking help on the prelab from a tutor, a classmate, or even a TA in the class, then you're setting yourself up for failure. It's perfectly fine to get help with general concepts, but it is very counterproductive to get prelab-specific help. Why? Because that kind of help won't be available during the labs.