C Programming Exercises, SC 3260/5260, Spring 2018

If you haven't done so already, clone the repo and build all the examples:

git clone https://github.com/sc3260s18/Cprogramming.git
cd Cprogramming
make

After making edits to a source (.c) file, you should type "*make*" to rebuild the executable file. Examine each file carefully and then attempt the exercises listed below.

hello_world.c

What happens if you remove the semi-colon from the end of the printf() statement and re-compile the code?

What happens if you comment out the "return 0;" line at the end of the program, re-compile, and re-run?

What happens if you comment out the #include line and re-compile?

What happens if you change the name of the main() function and re-compile?

variable_types.c

Try printing out the value of some of the ints you have defined.

Try printing out the value of some of the floats or doubles you have defined.

flow_control.c

Create a floating-point variable and test it directly in an if-statement. What happens? (don't forget to rebuild your executable after editing your file!)

The equality operator (==) is used to test whether two values are equal. What happens if you accidently use the assignment operator (=) instead?

loops.c

Note that array x starts with an index of 0. This is true of all arrays in C. What happens if you try accessing an array element outside of the array limits?

In the for loop, what happens if you initialize i to a value that is greater than n?

pointers.c

Create another variable j and set it equal to i immediately after initializing i at the beginning of the program. What will the value of j be at the end of the program?

functions.c

In line 9 an int type (iloc) is converted to a float type – this is called type casting. This must be done explicitly in C when performing mathematical operations on variables that are not of the same type.

In line 27 can you compute the value of 27 without using iloc and xloc?

What *appears* to be missing (it's actually not missing, but based on what we've discussed so far it may appear to be missing) on line 64 with the myArray argument?

structures.c

Notice the way the variable name was declared on line 4. Pointers are closely associated with arrays in C, and in this case we are declaring an array (of unknown length) of characters (i.e. a string) by declaring a pointer to a char.

Do you notice anything new about the printf statements in lines 15-19?

pass_command_line_options.c

What new header file is being used? What happens if we remove this line from the program and try to rebuild the executable?

What happens if you pass values from the command line that are of the incorrect type?

Can you make sense of the argument types that are being passed to main?

Try modifying the format (size of field) of my_arg_float that's being printed at the end of the program.

dynamically_allocated_arrays.c

What happens if you don't allocate enough space to your array?

How large can variable n become before malloc fails?

file_IO.c

Can you provide a file path to fopen in order to write to a different directory?