ME 333 Homework 1

Marshall Johnson

January 11, 2022

A Crash Course in C — Exercises Cont.

18) Invoking the gcc compiler with a command like gcc myprog.c -o myprog actually initiates four steps. What are the four steps called, and what is the output of each step?

I. Preprocessing — Outputs source code equivalent to original .c program with comments removed. This step also performs actions when encountering preprocessor commands indicated by the # character, such as *define* and *include*.

II. Compiling — Outputs **assembly code** specific to the processor. This turns the C code into instructions for the processor to perform certain corresponding actions.

III. Assembling — Converts assembly instructions into machine-level **binary object code** (dependent on processor).

IV. Linking — Outputs single **executable file** from 1+ object code files.

19) What is main's return type, and what is the meaning of its return value?

The return type of the main function is typically int. Usually the return is 0, which would indicate the program executed successfully (i.e., without error). If an integer is not returned, an error ocurred.

21) Consider three unsigned chars, i, j, and k, with values 60, 80, and 200, respectively. Let sum also be an unsigned char. For each of the following, give the value of sum after performing the addition.

a. sum = i+j = **140**

b. sum = i+k = **4**

c. sum = j+k = **24**

22) **For the variables defned as**

int a=2, b=3, c;
float d=1.0, e=3.5, f;

give the values of the following expressions.

a. f = a/b = **0.000**

b. f = ((float) a)/b = **0.667**

c. f = (float) (a/b) = **0.000**

d. c = e/d = **3**

e. c = (int) (e/d) = **3**

f. f = ((int) e)/d = **3.000**

27) **You have written a large program with many functions. Your program compiles without errors, but when you run the program with input for which you know the correct output, you discover that your program returns the wrong result. What do you do next? Describe your systematic strategy for debugging**

To debug, I would first test each of the functions I have written to make sure they are all working as expected. If I discover one or more that doesn't behave as expected, I would then examine those functions closely to determine where the error exists and resolve accordingly. After debugging individual functions, I would compile and run again. If the error(s) persist, as well as throughout the previously mentioned process, I would look for common errors such as data types, and indexing. I would ensure all data types are what I intended and consistent. I would check specifically for values called by index from arrays and ensure all indices are within the bounds of the array. Additionally, I would check to ensure that any numbers I'm using are within the limits of the data type they are assigned and nothing unexpected is happening.

28) **Erase all the comments in invest.c, recompile, and run the program to make sure it still functions correctly. You should be able to recognize what is a comment and what is not. Turn in your modifed invest.c code.**

See invest_edit.c for modified invest.c program with comments removed.

30) **Consider this array definition and initialization:**

int x[4] = {4, 3, 2, 1};

For each of the following, give the value or write "error/unknown" if the compiler will generate an error or the value is unknown.

- **a.** 3
- **b.** 4
- **c.** 2
- **d.** 6
- **e.** unknown/error
- **f.** unknown/error
- **g.** 2

31) **For the (strange) code below, what is the fnal value of i? Explain why.**

int i,k=6;
i = 3*(5>1) + (k=2) + (k==6);

The final value of i is **5**. The first term becomes 3*1, since the truth value of 5>1 is equal to 1 (True). The next term is 2 and sets k equal to this value. Last, the truth value of k==6 is 0 (False), since k was set to 2 in the previous term. This leaves 3*1 + 2 + 0 = 5.

32) **As the code below is executed, give the value of c in hex at the seven break points indicated, (a)-(g).**

- **a.** 0xF2
- **b.** 0x01
- **c.** 0x0F
- **d.** 0x0E
- **e.** 0x01
- **f.** 0x68
- **g.** 0x00

34) **Write a program to generate the ASCII table for values 33 to 127. The output should be two columns: the left side with the number and the right side with the corresponding character. Turn in your code and the output of the program.**

See ascii.c

35) We will write a simple bubble sort program to sort a string of text in ascending order according to the ASCII table values of the characters.

See bubble.c