Ava Zahedi
ME 333
HW for Class 3

Appendix A: 18-19, 21-22, 27-28, 30-32, 34-35

18. Invoking a compiler initiates 4 steps:
    1. Preprocessing: takes in our .c code and outputs an equivalent .c source code that is cleaned up by removing comments, substitutes in constants and macros, and adds content from included header files.
    2. Compiling: turns the preprocessed .c code into assembly based on the computer's specific processor.
    3. Assembling: converts assembly into processor-dependent machine-level binary object code.
    4. Linking: takes one or more object code files and links them together to produce a single executable file.

19. The main function's return type is int, where returning a 0 means the program ran successfully and returning a nonzero value means the execution had an error.

21. Unsigned chars i, j, and k with values 60, 80, and 200. sum is also an unsigned char.
    a) sum = i+j = 140
    b) sum = i+k = 4
    c) sum = j+k = 24

22.
int a=2, b=3, c;
float d=1.0, e=3.5, f;

    a) f = a/b = 0.0
    b) f = ((float) a)/b = 0.6666…7
    c) f = (float) (a/b) = 0.00
    d) c = e/d = 3
    e) c = (int) (e/d) = 3
    f) f = ((int) e)/d = 3.00

27. I would start by going back to the source code and reading through it to make sure I didn't miss anything obvious. Then, I would add print statements throughout my program to identify where the problem occurs. I can track my values throughout the execution and check where they no longer match my expectations. Once I've localized where the issue is coming from, I can carefully debug that section of the code and continue narrowing down the problem until I can solve it. Throughout this process I might Google questions that come up while I'm debugging.

28. Modified invest.c attached

30. int x[4] = [4,3,2,1];
    a) x[1] = 3
    b) *x = 4
    c) *(x+2) = 2
    d) (*x)+2 = 6
    e) *x[3] = error
    f) x[4] = unknown
    g) *(&(x[1])+1 = 4

31.
int i,k=6;
i = 3*(5>1) + (k=2) + (k==6);

5>1 is true which evaluates to 1.
k=2 assigns k to 2.
k==6 is false after k has been reassigned to be equal to 2, so this evaluates to 0.

Therefore, we have i = 3*1 + 2 + 0 which gives a final value of i = 5.

32.
unsigned char a=0x0D, b=0x03, c;
In binary: a = 0b1101, b = 0b0011
    a) c = -a; // c = 243 = **0xF3**
    b) c = a & b; // c = 0b0001 = **0x01**
    c) c = a | b; // c = 0b1111 = **0x0F**
    d) c = a ^ b; // c = 0b1110 = **0x0E**
    e) c = a >> 3; // c = 0b0001 = **0x01**
    f) c = a << 3; // c = 0b1101000 = **0x68**
    g) c &= b; // c = c & b = 0b0000 = **0x00**

34. ascii_table.c attached

35. bubble.c attached