ASSIGNMENT 1

1. HelloWorld.c program is attached with this assignment.

2. Pointers are a type of special variables which stores the address of another variable, i.e. it 'points' to another variable. It can be used to access the address of the variable it stores or the content of that address. A non-pointer variable may have any arbitrary value and the memory location(address) of that variable is allocated by the compiler according to the data type in the background without us knowing what it is (unless we use the address or reference operator) and if we access the variable, we get the value of that variable and not the address. However, for a pointer variable, if we try to access it, we get the address of the variable it 'points' at.

3. Programs in languages like MATLAB are called interpreted code meaning the code is converted to machine language and executed while the program is still running. However, C codes are compiled first, i.e. converted to machine language before the program is run. This allows us to get error and warning messages prior to the program being run. Compiled codes run quicker than interpreted codes.

4. (a) 0x1E: Binary = 0b00011110 msb = 0 Decimal = $1x16^1 + 14x16^0 = 30$
   (b) 0x32: Binary = 0b00110010 msb = 0 Decimal = $3x16^1 + 2x16^0 = 50$
   (c) 0xFE: Binary = 0b11111110 msb = 1 Decimal = $15x16^1 + 14x16^0 = 254$
   (d) 0xC4: Binary = 0b11000100 msb = 1 Decimal = $12x16^1 + 4x16^0 = 196$

6. Since each byte of memory can be addressed by a 16-bit address, total number of bits of memory we have is 2^{16}.

7. (a) ch = 'k' or ASCII Value = 107
   (b) ch = '5' or ASCII Value = 107
   (c) ch = '=' or ASCII Value = 61
   (d) ch = '?' or ASCII Value = 63

8. Range of values for the following data types:

   Unsigned char = 0 to 255
   Short = -32768 to 32767
   Double = 2.3e+308 to 1.7e+308

10. Signed integer is a 4-byte type of data type that permits the variable to take negative values. It can take values from – (2^{31}) to (2^{31}) – 1. However, Unsigned integers, like the name suggest takes only positive or zero values from 0 to (2^{32}) – 1. Unsigned integers are also a 4-byte data type similar to signed variable. In fact, Unsigned and Signed integers both can take approximately 4 billion values just they are shifted differently on the number line.

11. (a) Pros and Cons *chars* vs *integers*:

    Pros of *chars*: If you know that your program will require use of small integers values, then it may be advantageous to use char of int because it takes up much less memory, about 4 times less.

    Cons of *chars*: However, biggest con of char over integers is that if your calculations in the program exceeds the small integer range of char, you may experience something called integer overflow, for example if you add 100 & 240, you will get the result as 84 and not 340 because 340 is outside the char range of representation.

    (b) Pros and Cons *floats* vs *doubles*:

    Pros of *floats*: Floats are 4-byte data types whereas doubles are 8-byte, so if you want to save memory you should use floats over double if you don't require too much precision.

    Cons of *floats*: Doubles are preferred over floats if your calculation requires lot of precision because they are 8-byte and have lot of bits for decimal values.

    (c) Pros and Cons *chars* vs *floats*:

    Pros of *chars*: If you know that your program will require use of small integers values and won't require any floating-point precision then it would make sense to use char instead of float because it takes up much less space. This is because char is 1-byte and floats are 4-byte data types.

    Cons of *chars*: Disadvantage of char is that it only represents small integer values, so for programs that require higher levels of precision, i.e. decimal values then it can't be represented by char even if the number is in the range of char.

16. If we assume the pointer occupies 8-bytes, similar data types would be long int, long long int and double which are all 8-bytes.

17. (a) the initial conditions, all memory contents unknown

    (b) kp = 0xB0, kp takes address of i.

    (c) j = 0xB0, j takes the content of kp (kp is dereferenced).

    (d) i = 0xAE, i takes the value 0xAE at location 0xB0.

    (e) np = 0xB0, content of np takes content of kp, i.e. address of i.

    (f) *np = 0x12, contents of np are set to 0x12.

    (g) j = 0xB0, j takes the content of kp.