EECS 388: Hints for Lab 01 Assignment Problems

Problem 2:

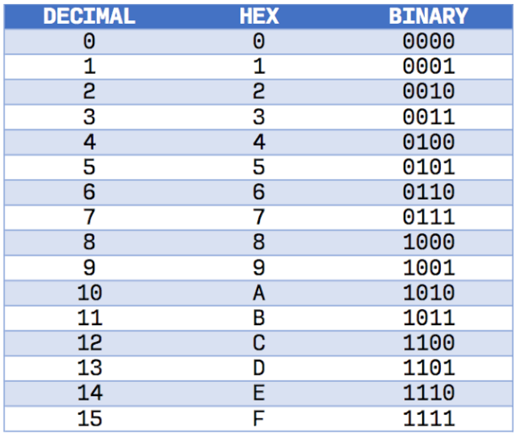
1. Get the Height as user input
2. Set a height range for tall/medium/short
3. Utilize if…else condition to check the input (Check if…else section in your lab sheet)
4. Print the output

Problem 3:

1. Get the limit value as user input
2. Use loop (For/While loop) to iterate over all the numbers till the limit value reached (Check the loop section in your lab sheet)
3. Use if…else to check if the number can be divided by something other than 1 or the number itself (Check if…else section in your lab sheet)
4. The number is prime if it can only be only divided by 1 or the number itself and no other number.
5. Otherwise the number is not prime.

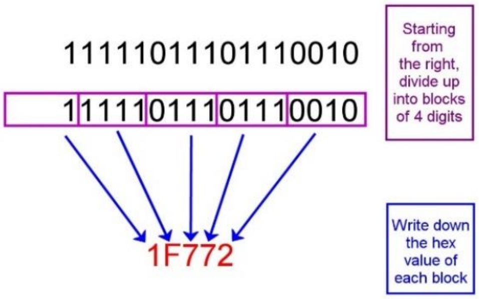
Problem 4:

1. Remember the following table:



DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

2. Here is an example of how to convert a binary to Hex:


11111011101110010
1 1111 0111 0111 0010
1F772
Starting from the right, divide up into blocks of 4 digits
Write down the hex value of each block

$11111011101110010_2 = 1F772_{16}$

3. Divide the input binary string from right side by taking four of them at once.
4. Assign the corresponding HEX value.
5. Do exactly opposite for the Hex to Bin conversion.

Problem 5:

1. Address user inputs for two cases only: String and Integer
2. Copy the input array to another temporary array
3. Use nested for loops for cross check between the two arrays (original and the copied one)
4. Find the duplicates by flagging it with a counter
5. Print the duplicate string/integer value with number of duplicates

Problem 6:

1. Take the string as user input
2. The string is an array of characters
3. “Hello World” is an array of 11 characters. Note, white spaces are characters too.
4. Note that the first element of the array can be accessed by calling them by their memory address using pointer. For example, you can declare a pointer for the address of your string (character array) str to be char* ch. So if char* ch = &str, you can access the each element of the array using pointer arithmetic. That is, you can access the first element by *ch and the second element by *(ch+1). Thus you can count the iteration that you need to reach the end of line (‘\0’ in C)

Problem 7:

1. The calculator should take inputs in one line. Such as, you enter 5+6 and hit enter and it should give you 11
2. Add the additional functions using math.h (check the link given in the lab sheet)
3. Refer to User Input section of the lab sheet for understanding how to take multi-type input together