**All submissions will be on Git Hub**

**Warm-up Exercises (Week 1)**

1. a) Write a program called *hex2Base64* to convert hex to base64. The hex string comes from the user.

Do not use any inbuilt functions for this conversion. Implement the conversion algorithm yourself. The program must display the base64 output on screen.

b) Write another program called *b64ConversionTester* that takes a text file as input, calls *hex2Base64* to convert it to base64 and then calls an inbuilt function (from a popular vetted library) to do the conversion and compares the result. The output must “True” or “False”, where True implies your output matched with the one generated by the system.

The input text file will contain hex strings (small letters), one on each line (user will provide the file name as an argument when calling the program). The hex strings may contain one or more whitespaces in between characters (e.g. ea b7 5 e a 8) and/or 0x as prefix. Your program should be able to handle these. If there are multiple lines in the file, your program should print True/False for each line.

Feel free to use look-up tables/dictionary structures as needed.

Clearly state the assumptions you make on the input (capital letters, length of input, etc.) and justify those assumptions (make a case why these are practical assumptions). For inputs that you think are not valid and may occur in the real world, create appropriate error handling.

Your submission must include a READ ME files for both 1a and 1b detailing the usage format and any other peculiarities of your implementation.

1. XOR: Implement a function that takes two equal length hex strings and XORs them. The output must also be in hex. Print the output to the screen.
2. The hex string given below has been XOR’d against a single character; find the key and decrypt the message. You can use English plaintext frequencies to rank the outputs.

TODO: Create a dummy encrypted input long enough that frequency analysis becomes possible.

1. Implement repeating-key XOR
2. Breaking repeating-key XOR