Computer Science 210 –Computer Organization

Homework Exercise 1
Due on github 11:59 PM Monday 17 January

This assignment will require you to add to a Python script **binary.py** that you will download from the link on the course website (below the link you clicked to view these insructions). Your name should be the first line of text in the comment at the top of the file.

Part I: Complete the first 4 numbered exercises below. Then you can check your work on the first four exercises with the code you develop in the last seven exercises. Please put your answers to exercises 1-4 in a Python comment at the beginning of your **binary.py** file.

1. Interpret the eight bit string 01101100 as
   a. a two's complement eight bit number
   b. a sign-magnitude eight bit number
   c. an unsigned binary number

2. Interpret the eight bit string 11001101 as
   a. a two's complement eight bit number
   b. a sign-magnitude eight bit number
   c. an unsigned binary number

3. Represent the following integer values as eight bit two's complement numbers:
   a. 56
   b. -43
   c. -1

4. Convert the following values to hexadecimal notation
   a. 56 (decimal)
   b. 438 (decimal)
   c. 11010111 (binary)

Part II: Add the following functions to your **binary.py** file. ***Each function definition should include a docstring that describes what the function does, its arguments, and its returned value.***

5. Define a Python function **`unsignedDecimalToBinary`** that expects an unsigned decimal integer as an argument and returns the corresponding string of binary digits. You should assume that the caller provides a correctly formed integer, so no error handling is necessary

6. Define a Python function **`unsignedBinaryToDecimal`** that expects an unsigned string of binary digits as an argument and returns the corresponding unsigned decimal integer. Again, assume that all arguments are well-formed values.

7. Define a Python function **addOne** that expects an unsigned string of binary digits as an argument and returns an unsigned string of binary digits that is 1 greater than the argument string. Ignore the problem of integer overflow for now.

8. Using the functions you already have defined, define a function **twosCompToDecimal** that expects a twos complement string of binary digits as an argument and returns the corresponding signed decimal integer.

9. Using the functions you already have defined, define a function **decimalToTwosComp** that expects a signed decimal integer as an argument and returns the corresponding twos complement string of binary digits.

10. Define a function **signExtend** that expects a twos complement string of binary digits and the total number of bits as arguments and returns the corresponding twos complement string of binary digits, s