Computing Machinery I
Assignment 3
10% of your final score
Due May 27th @ 11:59PM MST

Objective
The objective of this assignment is to practice binary multiplication, division, and shifting in ARMv8 assembly.

Skills Needed for this Assignment:

- Ability to work with basic arithmetic, loops, and if-else constructs in assembly
- Ability to print to standard output using the `printf()` and `scanf()` functions
- Ability to use shifting instructions to implement multiplication
- Ability to assemble programs using *gcc* and use *m4* to process macros
- Ability to use *gdb* to debug and display assembly language programs

Overview
Your programs will convert hexadecimal numbers to binary and vice-versa.

Details
1. The first program converts a binary number to hexadecimal. It prompts the user for N. For reading N, you will need to call the C library `scanf()` function. Read it as a long integer. To convert the binary number to hex, you will take the least/most (you decide) significant 4 bits and convert it to a symbol between 0 and F, shift the number to right/left (you decide) by 4 bits and repeat. You must also handle negative numbers properly. If the binary number is negative, then convert it to its absolute value (for example, the number -127 is converted to +127). Then, the positive number is converted to hex. For displaying hex numbers use the – or + signs to indicate if the number is positive or negative. You are not required to store the hex number; displaying it to the screen is sufficient.

   For converting a four bit group to a single hex digit, you may use a binary to decimal conversion algorithm. There is one at the end of this assignment.

   The user enters a decimal value N. The output will be its corresponding value in hexadecimal. Example: user inputs -10 your program outputs -0x000000000000000A

2. The second program converts a hexadecimal number to binary. Read the hexadecimal number one character at a time. You may use the ASCII-to-integer conversion function atoi() in C. Here too, no storage of the number is required.

   The user enters a hexadecimal value N made of 4 digits. The output will be a 16 binary digits. Example: user inputs 000A your program outputs 0000000000001010

   You may use the Decimal to binary algorithm at the end of this assignment, but you can use other algorithms if you like.

You do not need to check for a valid input (especially for program 1 where the entered number must be binary). Make sure your code is properly formatted into columns, is readable and fully documented, and includes identifying information at the top of each file. You must comment each line of assembly code. Your code should also be well designed: make sure it is well organized, clear, and concise.

Submission

- **Note:** The TA may provide further submission instructions
- Name your programs *assign3a.asm* (binary to hexadecimal) and *assign3b.asm* (Hexadecimal to binary)
- Create a script file for each program. Call them *assign3a.script* and *assign3b.script*
- Submit a *README* file providing extra instructions or information for your TA (optional)
- Submit your work to the appropriate dropbox on D2L.

Late Submission Policy
Late submissions will be penalized as follows:
-12.5% for each late day or portion of a day for the first two days
-25% for each additional day or portion of a day after the first two days
Hence, no submissions will be accepted after 5 days (including weekend days) of the announced deadline.

Academic Misconduct
This assignment is to be done by individual students: your final submission must be your own original work. Teamwork is not allowed. Any similarities between submissions will be further investigated for academic misconduct. While you are encouraged to discuss the assignment with your colleagues, this must be limited to conceptual and design decisions. Code sharing by any means is prohibited, including *looking* at someone else's paper or screen. The submission of compiler generated assembly code is absolutely prohibited. Any re-used code of excess of 5 lines in C and 10 lines in assembly (10 assembly language instructions) must be cited and have its source acknowledged. Failure to credit the source will also result in a misconduct investigation.

D2L Marks
Marks posted on D2L are subject to change (up or down).

Conversion algorithms
Let N be a number; we will refer to the least significant bit in N as N[0]. Implement the following algorithms to convert N to a decimal value:

Binary to decimal:

```
If N is negative, pos_N = 2s complement of N
Else, pos_N = N
decimal_N = 0

For (i = 0; i < 64; i++) {
   If pos_N(0) is 1 {
         Decimal_N = decimal_N + 2^i
         pos_N >> 1
   }
}

If N is negative, decimal_N = -decimal_N
```

Calculating 2^n:

```
Pow_2 = 1

For (i = 0; i < n; i++) {
   Pow_2 << 1
}
```

Decimal to binary

```
If N is negative, pos_N = 2s complement of N
Else, pos_N = N
binary_N = 0
quotient = N
i = 0

while quotient != 0 {
   quotient, remainder = quotient / 2
   // Divide quotient  by 2 producing a new quotient and a remainder
   remainder = remainder << i
   binary_N = binary_N | remainder
   i++
}

If N is negative, binary_N = 2s complement of binary_N
```

Computing Machinery I
Assignment 3 Rubric

Student:_____

Item	**Max Points**	**Points**
Code compiles	5	
Code runs	5	
Binary to hexadecimal	30	
Hexadecimal to binary	30	
Handling negative numbers (both programs)	15	
Code readability (formatting and documentation)	15	
Total Points	**100**	