Computing Machinery I
Assignment 3
8% of your final score
Due May 31th @ 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 decimal numbers to binary and vice-versa.

Details
1. The first program converts a binary number to decimal. It prompts the user for N,a number entered in binary. For reading N, you will need to call the C library `scanf()` function. Read it as a long integer. Let M be any number; we will refer to the least significant bit in M as M[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^i:
   ```
   Pow_2 = 1

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

2. The second program converts a decimal number or binary. The user enters a decimal value N. Use the following algorithm:

   Decimal to binary:
   ```
   If N is negative, decimal_N = -decimal_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
```

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 decimal) and *assign3b.asm* (decimal to binary)
- Create a script file for each program. Call them *assign3a.script* and *assign3b.script*. **Both script files must contain a GDB session.**
- 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).

Computing Machinery I
Assignment 3 Rubric

Student:_____

Item	Max Points	Points
Code compiles	5	
Code runs	5	
Calculating 2^i	5	
Binary to decimal	30	
Decimal to binary	30	
Handling negative numbers (both programs)	10	
Code readability (formatting and documentation)	15	
Total Points	**100**	