CS061 - Programming Assignment 1

Objective

    The purpose of this assignment is to familiarize you with the basics of LC-3 assembly language programming, the SIMPL emulator, and rudimentary debugging.

Your Tasks

    Implement the LC-3 program from the image below.

Note: This is *almost*, but not *quite*, the same program you worked on in Lab 1: it uses a DO-WHILE loop to multiply one number by another by repeated adding.

First (as with all your labs and assignments) go to the Assignment folder in Piazza, and download the assignment 1 zip file to your assignments/assn1 folder in your cs account, and unzip it in place.

Type the code from the image below into your assignment1.asm file
*Again, the registers are assigned differently from your Lab 1 exercise, so **don't** copy that!*
*But **do** use the (AND with 0) technique discussed in the "TIP" paragraph in lab 1.*

```
geany assignment1.asm &
```

 and then launch & run simpl:

```
simpl assignment1.asm &
```

Remember: *whenever you run simpl, you must <u>ALWAYS</u> keep the Text Window open so you can see warnings & error messages from the emulator!*

Now place a <u>*breakpoint*</u> at the <u>beginning</u> of the DO-WHILE loop by right-clicking that line of code and selecting "Mark as Breakpoint".
Now, when you press the "Run" button, simpl will execute each line of code until it reaches the breakpoint, where it will pause and wait for you to press Run again; it will continue doing this until it no longer encounters the breakpoint, at which time it will continue executing until the HALT instruction.

Lastly, create a table in your .asm file (see the image below) to record the register contents for <u>*each*</u> register (R0 through R7) as you repeatedly press the Run button and iterate through the loop, as follows:

- Before entering the loop (i.e. the first time the program halts at the breakpoint)
- After each iteration of the loop (each subsequent breakpoint pause)
- Note that since your breakpoint is at the start of the loop, the register values after the <u>*last*</u> iteration will just be the values when the program has ended, as suggested by the naming of the last row in the image below.

Before you run the code, try to answer this question:
How many rows do you think you will end up with in your table?

Format your table like the one in the image below *(but obviously reporting the **actual** values from **ALL** eight registers, for **ALL** iterations of the loop!)*

The following program performs the action:

    R3 <-- 12 * 6   ; (i.e. multiply 12 by 6 and write the result into Register 3)
using the equivalent of a DO-WHILE loop:

```
;--------------------------------------------------------
; Name: Hayao Miyazaki
; Login: hayam
; Email address: hayam@cs.ucr.edu
; Assignment: assn1
; Lab Section: <021 or 022>
; TA: Sean Foley
;
; I attest that this code was totally given
; to me and that I didn't come up with
; any of it  =P
;--------------------------------------------------------


;--------------------------------------------------------
; REG VALUES       R0  R1  R2  R3  R4  R5  R6  R7
;--------------------------------------------------------
; Pre-loop         X   X   X   X   X   X   X   X
; Iteration 01     X   X   X   X   X   X   X   X
; Iteration 02     X   X   X   X   X   X   X   X
; ...
; Iteration n      X   X   X   X   X   X   X   X
; End of program   X   X   X   X   X   X   X   X
;--------------------------------------------------------

.ORIG x3000                         ; Program begins here
;-----------------
; Instructions
;-----------------
LD   R1,  DEC_6                     ; R1 <-- 6
LD   R2,  DEC_12                    ; R2 <-- 12
LD   R3,  DEC_0                     ; R3 <-- 0

DO_WHILE     ADD R3,  R3,  R2   ; R3 <-- R3 + R2
             ADD R1,  R1,  #-1  ; R1 <-- R1 - 1
             BRp DO_WHILE       ; if ( LMR > 0 ) goto DO_WHILE

HALT                                ; Terminate the program
;---------
; Data
;---------
DEC_0    .FILL     #0      ; Put the value 0 into memory here
DEC_6    .FILL     #6      ; Put the value 6 into memory here
DEC_12   .FILL     #12     ; Put the value 12 into memory here

.END
```

Remember: Replace the instruction that zeros out R3 with the AND technique you learned in lab 1!
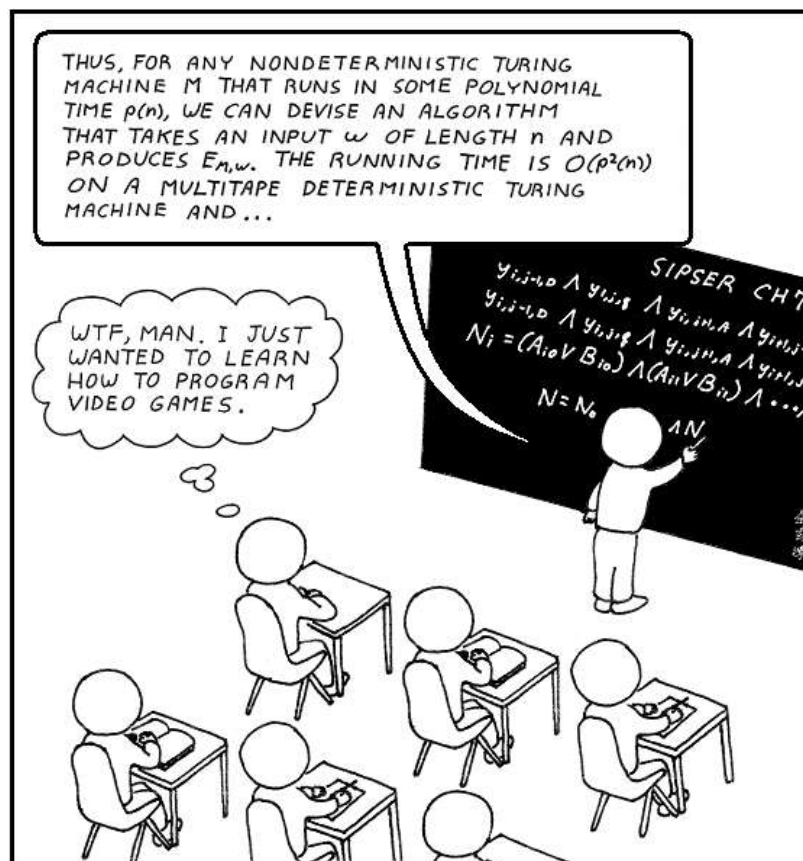
Submission Instructions

Submit your **assignment1.asm** file *(and ONLY that file!)* to the Programming Assignment 1 folder in Gradescope.
Note that there is no autograding for assn1 - you will be notified when it has been graded manually.

Rubric
- There is nothing for you to actually get wrong in this assignment (we give you the code!) - so the only way to mess up and lose points would be to fail to follow instructions.
  So get into the habit of reading the instructions fully & carefully :)
- This assignment has to be graded manually (i.e. we have to look at your code formatting and your register table); we will try to grade submissions at least once or twice a day.
  So don't leave it to the last day - you may need a second chance!! See the point above!
- For this and all assignments, use of the template given to you is **required**.

Comics??!Sweet!!!

Source: https://xkcd.com/