Homework 5: Machine Language

Objective:

Build the two Assembly Language programs described below, which will test your understanding of Assembly programs for our HACK architecture. Highly recommend you go through the simulator tutorial for assembly programs before attempting the homework.

Grading method:

Generally speaking, we prefer implementations that *use as few A- and C- instructions as possible,* documentation/commenting do not count towards that line count. Documentation will be in form of commenting AT LEAST EVERY section of assembly code, describing what the instruction is doing overall or to the registers. This may be over-commenting in high-level languages like Java, but is absolutely necessary here to understand the machine code.

Div.asm:

In this program, you will be implementing the division operation by successively subtracting the dividend and divisor to reach a quotient result. Write an assembly program to perform the division of two integers (stored at R0 and R1) and store the result in R2. You can perform multiplication through successive subtraction. For example:

16 / 3 = 16 – 3 – 3 – 3 – 3 – 3 = 0 (remainder 1)

a. First write Java code using a loop to perform the successive subtraction.
   Assume the following:
   int R0 = 16;
   int R1 = 3;
   int R2 = 0; // R2 holds the result of R0 / R1 (after using successive subtraction in a loop)
   // You do not have to submit the Java code, it is meant to help you translate into assembly.

b. Next, convert the high-level Java code into assembly code (named div.asm). This is the code you will copy/paste to the Word document you submit. Please ensure *every* section of Assembly code is documented with a comment.

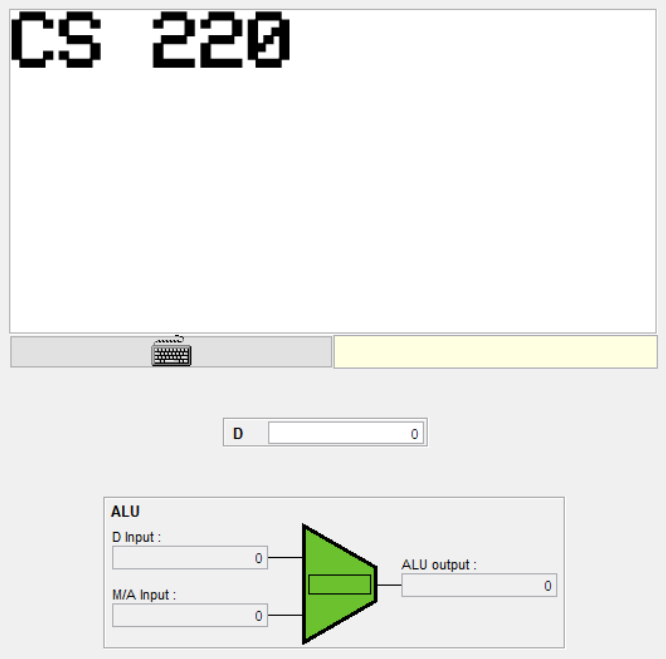
See http://nand2tetris.org/04.php *for test scripts/output files (if you need clarification email your instructor. You will be graded based on this documents requirements).*

CS220.asm:

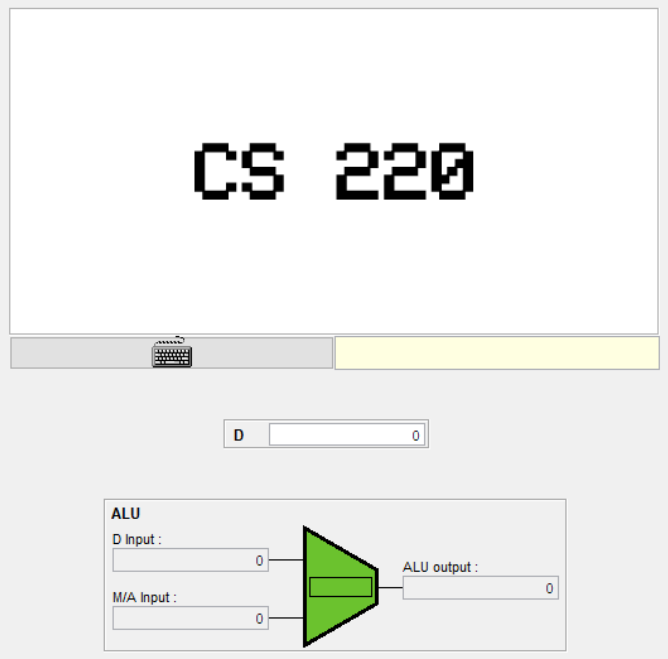
In this program, you will be writing assembly code to produce the text "CS 220" on the simulated monitor of CPUEmulator tool. You may align the text at the top-left portion of the screen to accomplish this task. For more of a challenge, try centering the text both horizontally and vertically.

Here are two screen shots indicating the expected output:

a) Regular Challenge



b) Extra-Credit Challenge w/ Centering (+ 20 points extra credit)



See http://nand2tetris.org/04.php *for test scripts/output files (if you need clarification email your instructor. You will be graded based on this documents requirements).*

Generally speaking, we prefer implementations that *use as few A- and C- instructions as possible,* documentation/commenting do not count towards that line count. Documentation will be in form of commenting AT LEAST EVERY section of assembly code, describing what the instruction is doing overall or to the registers. This may be over-commenting in high-level languages like Java, but is absolutely necessary here to understand the machine code.

What do you turn in?

Create one Word document (or PDF) with the following in order:

1. The **Div.asm** source code (make sure to comment every section of code)
2. A screen shot (entire window) containing the output from running the **Div.asm** code with **R0 initialized to 16** and **R1 initialized to 3**. The correct result of 5 should be shown in R2. Be sure to comment your code and end the program with an infinite loop to prevent NOOP slides.
3. The **CS220.asm** source code (make sure to comment every section)
4. A screen shot containing the output from running the **CS220.asm** code in the CPUEmulator (take screen shot of entire window).

Program	***Working?***	***Well built?***	***Documentation?***
Div.asm	20	/ 15	/ 5
CS220.asm	40	/ 15	/ 5
Extra Credit	20		
Subtotal	60	/ 30	/ 10

See http://nand2tetris.org/04.php *for test scripts/output files (if you need clarification email your instructor. You will be graded based on this documents requirements).*