Adding Instruction Memory and Program Counter to Your Computer

Begin by reading your lab manual. Then ***use the Instruction Set document on Canvas*** as a road-map for answering the following questions.

1 Straight Line Assembly Program

Based on the **instruction set document on Canvas** (Here we have only **4** registers, $0-$3, as oppose the the MIPS ISA which has 32 registers) write an assembly program using our own instructions from the instruction set.pdf document which fulfills the following tasks and turn in the assembly code:

- Initialize address 0x4 and 0x5 of Data Memory with values 0x10 and 0x0F respectively.
- Add the numbers you stored in 0x4 and 0x5 in Data Memory and store the sum in 0x11.
- Flip the sign of the number (two's complement) stored in 0x5 in Data Memory and store it in 0x12 of data memory.
- Subtract the numbers stored in 0x4 and 0x12. Store the result in address 0x13 of Data Memory.

Turn in the assembly program you wrote.

Please note that the numbers are in two's complement format. Also, since you not have yet created the hardware unit for branching, you are not allowed to use j-type instruction like beq and bne in your code for this prelab.

2 Generate the Machine Codes using the Assembler

Use the assembler available on Canvas to generate the machine codes out of your assembly program. (Do not the blank line in your assembly program.)

Verify the output of the assembler by comparing the output machine codes with instruction structure described in the instruction set document.

Turn in the the output file of the assembler.

Note: You can use gcc compiler or an online C IDE from `https://www.tutorialspoint.com/online_c_compiler.php` (Using the C IDE from Advanced IDEs catalog not the Online Terminals one) for your code development, compilation and running the assembler.