Homework – Add two memory locations and subtract the next two.

This is a programming assignment that will be graded and scored as a homework since it is so simple.

Write a program for the LC3. This program should be named ***hwAdd2Sub2Mem.hex.*** This hex file will be a machine language program for the LC3 processor.

This program will be written as an ASCII hex file.

DON'T TRY TO ASSEMBLE IT WITH AN LC3 ASSEMBLER.

The hex file must contain only hex characters; 0-9 and A-F with no x, X, 0x, or 0X.

Each line should be a four-digit hex number. Do not include a 0x or x at the beginning of this number.

The first line should be the starting address in memory, use 3000 to be safe.

The last instruction line should be a HALT (F025) with any data you need immediately following the HALT.

THERE SHOULD ONLY BE ONE HALT IN ANY LC3 PROGRAM!

The lines between the initial 3000 and F025 should do the following:

> Add the values from the two memory locations immediately after your HALT instruction. Subtract the value of the next two memory locations after the first two. Then store the result in the fifth memory location after your HALT instruction.

Write your program in assembly. Convert it manually to hex. Put the hex in hwAdd2Sub2Mem.hex.

Test your program using the simulator. Make sure to test with numbers OTHER than the ones we discussed in class.

You can manually put in values in the simulator by clicking on the memory locations or you can store them as part of your program and change them and reload the program for testing.

Submit your hwAdd2Sub2Mem.hex file to Web-CAT. Make sure you did not name it hwAdd2Sub2Mem.txt or that Notepad or WordPad didn't rename it hwAdd2Sub2Mem.hex.txt.

MACs allow a space to be the first character of a filename. If you are on a MAC and your program looks like it is spelled correctly but Web-CAT is complaining, make sure you don't have a space as the first character in the filename.

You will lose points for every submission over 5. You will lose 10 points for each day late. Submitting more than 2 days late will result in a zero.

Late days start at 9:00 am. So, submitting after 9:00 am on the due date will result in -10 points. Submitting after 9:00 am on the day after that will result in an additional -10 points. You will not be able to submit after 9:00 am on the second day.

For example, if you want to test the data set 5, 4, 3, and 2, you will store the following in memory starting at the memory location after the HALT. Note that the XXXXs below is where your code should be and the 0005, 0004, 0003, and 0002 is the data. Depending on how you write your program you may use a different number of steps. Do not add the comments. Each line must be exactly four hex digits only with NO TRAILING SPACES or TABS.

3000 ← Where the program is loaded into memory.
XXXX ← Your program's first instruction.
XXXX ← Your program's second instruction.
XXXX ← Your program's third instruction.
XXXX ← Your program's fourth instruction.
…. ← You may have any number of instructions. I'm just showing four as an example.
F025 ← Your program's HALT instruction comes after all your instructions.
0005 ← First data memory to add to sum will be the memory following F025.
0004 ← Second data memory to add to sum.
0003 ← Third data memory for subtracting from sum.
0002 ← Fourth data memory for subtracting from sum.
0000 ← Store the result here. You don't have to put 0000 but you do have to use this location.

The XXXX values above are your instructions in hex. You must figure these out. I show four instructions, but your program may (probably will) take more. Note that if you have more than four instructions in your program, that will move the location of the HALT (F025) and the location of your data, which will change the offsets.

If you have questions about what the above means, please ask.

If you run your program with 5, 4, 3, 2 in the data section as shown in the example above, you should get the following result:

5 + 4 – 3 – 2

The sum will be 4 so the value 0004 should overwrite the 0000 in the example above at the fifth memory location after the HALT.

Note that ALL the data items are positive values. DO NOT STORE NEGATIVE VALUES WHILE TESTING.

Values 5, 4, 3, and 2 are all stored in your program as positive values. You program must make the 3 and 2 negative and before adding them to the sum. DO NOT STORE -3 and -2 and then try to simply add those.

If you store the values 5, 4, -3, -2, your program should calculate 5 + 4 - (-3) - (-2) which would be 14.

Test your program with different values of data. That is change the 5, 4, 3, 2 to something else and make sure the correct result shows up in the correct place.