Homework 6

Preamble

Several of these problems require you to work with the actual 16-bit instructions of the LC-3, rather than the more familiar assembly language codes.
Use the table inside the back cover of the text to translate between the two.
e.g. the instruction
ADD R1, R2, x0A
translates to
`0001 001 010 1  0 1010` *(note that bit 5 = 1, indicating the immediate addressing mode)*
And the instruction
`0000 110  0 0000 1100`
translates to
`BRzn x00C`
- i.e. Branch on zero or negative to the label located x00C = #12 locations past the current PC (this is called the "offset").
Now, remember that the value stored in the PC at any given time is ***ALWAYS***
(the address of the instruction being executed + 1)
since `PC <- (PC) +1` *during every Instruction Fetch.*
So if the BRzn instruction were stored in location x3100, the branch (if taken) would be to the instruction stored at
`(x3100 + 1) + x00C =` `x310D`
Given that PCoffset9*, the Direct and Indirect addressing mode offset used in the LEA, LD, LDI, ST, STI, & BR instructions, is 9 bits (two's complement), the furthest we can "reach" with these instructions is about +/- 256 locations.*

***Make sure you understand all this thoroughly before embarking on the following exercises*.**

Exercises

5.5
5.6 - we will be doing a number of exercises involving this scenario (including an assignment), so make sure you master it now.
5.14 - you must show your reasoning
5.15 - rewrite the table showing the addresses as hex values, the first five "data" values as assembly code, and the remaining data as hex values; then step through the code to determine the final contents of registers R1 - R4
5.16
5.17
5.19
5.20
5.24
5.26 - OMIT part c), and use the structure of the LC-3 ADD instruction as a model to explore the effects of the changes described.