Name: ______________________________ **Date:** _________________________ **Lab Section:** ___________

Objectives: To review important concepts in Chapter 4 - MARIE. ***Answer on this sheet in the given spaces***.

References: ECOA2e Section 4.1-4.6, 4.8.1-4.9.1, 4.9.3, 4.10, 4.11.1, 5.4.2 and associated Chapter Slides
Class Notes (via course home page at **teaching.idallen.com/cst8214**): **text_errata.txt**

Equipment: MARIE Simulator: free download from **`http://computerscience.jbpub.com/ecoa/2e`** and the free Sun Java Run Time Environment

Put all answers ***on this question sheet*** *in the spaces provided.* ***Circle*** *answers on this sheet if indicated. Not all questions may be marked –* ***check all your answers*** *against the answer sheet when it is posted.*

1. Reproduce here the one-line descriptions (no RTL/RTN) of the *revised* Instruction processing Cycle of operations from the *revised* Section 4.9.1 (see the revised version in the Class Notes, file **text_errata.txt**):
   1. ____
   2. ____
   3. ____
2. The third part of the Instruction Cycle is "Decode and Execute". Where is the PC register pointing while this is happening? ____
3. Circle: True/False: The operand field of a MARIE instruction is always used to contain an address.
4. Define an instruction "mnemonic" (p.195): ____
5. Another name for "binary instructions" is (p.195): ____
6. The mnemonics that correspond to machine code are referred to as: ________________________________
7. Circle: True / False: every assembly language instruction corresponds to exactly one machine instruction.
8. The name of the program that converts mnemonic assembly language to its binary equivalent machine code is (p.195, also Section 4.11): ____
9. Give the hex for "Skip if AC less than zero": ____
10. Give the RTL for "Add X": ____
11. Give the RTL for "Jump X": ____
12. Suppose the program in Table 4.3 p.204 started at hex address **AB5h**. Give the seven 16-bit hex values for the contents of memory:____

    ____
13. An assembler reads a source file and produces as output (p.206): ________________________________
14. Circle: True/False: The first pass of a two-pass assembler builds the "symbol table". (p.206-207)
15. Circle: True/False: The second pass of a two-pass assembler is used to fill in addresses using the symbol table. (p.207)

16. This section uses the MARIE simulator. You can download this program at home. Open the MARIE simulator program and do the following using the given seven-line assembly language program:

Label	Instruction
	`load x`
	`add y`
	`store z`
	`halt`
`x,`	`dec 32`
`y,`	`dec -15`
`z,`	`hex 0`

a) Type the adjacent seven-line MARIE assembly language program into the MARIE simulator and save it. Remember where you put it.

b) Tell MARIE to assemble the saved program into object code (machine code) and load it into the MARIE memory. Prepare to single-step through the program.

c) Single-step through each instruction of the program and write the contents of AC, IR, MAR, MBR and PC in the following table after each instruction has been decoded and executed.

d) Be prepared to explain why the registers contain these values.

Remember that the PC always points to the *next* instruction when decoding and executing the *current* instruction out of the IR. The Instruction Cycle is always: *fetch, increment, execute*

Instruction	AC	IR	MAR	MBR	PC
`load x`					
`add y`					
`store z`					
`halt`					

17. Hand-assemble the preceding seven-line assembly language program starting at hex memory address **CF2h**:

Hex Address	Label	Instruction	Machine Code (hexadecimal)
`CF2`			

18. Copy the ten-line assembly-language program from Question 13 on page 239 and hand-assemble it (without using MARIE) starting at hex memory location **A1Fh** (not at location 100 as shown in the text):

Hex Address	Label	Instruction	Machine Code (hexadecimal)
`A1F`			

19. Give the contents of the Symbol Table for the preceding ten-line assembly language program:

`Symbol Name`	**`Hex Address of Symbol`**

20. Answer Question 12 p.239: ____

____

21. Write the Assembly Language equivalents for Question 15a p.239:

i) ____________________ ii) ______________________ iii) ____________________

22. Read *Address Modes*, Section 5.4.2, and refer to Figure 5.3, Table 5.1, and Question 13 p.277. Given the instruction **`LOAD 2000`**, determine the actual value loaded into the accumulator and fill in the Addressing Mode table below if the index register **`R1`** (used by "indexed mode") contains the value **`1200`**:

Memory	
Address	**Contents**
1200	**2500**
...	...
2000	**2200**
...	...
2200	**2600**
...	...
2800	**1200**
...	...
3200	**3600**
...	...
3600	**2400**

Addressing Mode	*Value loaded in accumulator, AC, after executing* **`LOAD 2000`**
Immediate	
Direct	
Indirect	
Indexed (with R1)	

Note that most MARIE instructions are Direct Addressing. MARIE has no Immediate or Indexed Addressing instructions and only two Indirect Addressing instructions: **AddI** and **JumpI**. (MARIE has no **SubI**, **LoadI**, or **StoreI** – a real computer ISA would be more consistent and permit more address modes.)

23. Give the RTL/RTN for a new MARIE **LoadI** instruction:

____

____

____

____

____

24. Give the "value loaded into AC" for the four-element addressing mode table for Question 13, p.277:

Immediate: _____________ Direct: _____________ Indirect: _____________ Indexed: _____________

25. Give the "value loaded into AC" for the four-element addressing mode table for Question 14, p.277:

Immediate: _____________ Direct: _____________ Indirect: _____________ Indexed: _____________

26. Can I read all your answers clearly?