(1) Complete the following figure by specifying number systems we discussed in the classroom.

(2) Transform the following decimal number to the two’s complement binary number (using the 16-bit format: your processor is a 16-bit architecture processor): -200\(_{(10)}\). Show all your work.

- 200\(_{(10)}\) = “0000 0000 1100 1000\(_{(2)}\)”
- Flip the binary bit pattern: 1111 1111 0011 0111
- Add binary one: 1111 1111 0011 1000

**Note:** Not much credit were given for those solutions that did not show all the work.
(3) “li $t0, (1024)” is an illegal instruction (if you try to assemble that instruction using PC-SPIM simulator, that instruction will cause a syntax error). What’s wrong?

“li” instruction takes a hardcoded constant integer for its second parameter. The parenthesis “( )” means “the memory address of”. Therefore, “li $t0, (1024)” should be “li $t0, 1024”.

(4) What is the difference between “li $a0, 1024” and “la $a0, 1024” instructions? Assume that this computer system is a 32-bit system (i.e., all the registers are 32-bit registers and its ALU can deal with up to 32-bit inputs and outputs).

The former (“li $a0, 1024”) transforms “1024” based on the two’s complement integer, while the latter (“la $a0, 1024”) transforms “1024” based on the unsigned integer.

Note 1: The two terms “immediate” and “constant” mean the same thing. Thus, saying, “li instruction stores an immediate value to a register, while la instruction stores a constant value to a register” does not explain the difference.

Note 2: Memory addresses are specified by positive integers (because there is no “negative address” or “fraction address”). Thus, saying, “li instruction stores an (positive) integer value to a register, while la instruction stores a (memory) address to a register” does not explain the difference.

(5) “li $t1, $t0” is an illegal instruction (if you try to assemble that instruction using PC-SPIM simulator, that instruction will cause a syntax error). What’s wrong?

“li” instruction takes a hardcoded constant integer, and copy it (in the binary format) to a register, but it does not copy the content of a register to another register. Therefore, “move” instruction should be used instead.