Homework 2
Due 9/21

1. **(1 point)** You are storing the string “ARCHITECTURE ROCKS!” in the memory. Assume that the whole string is one object, and that the memory locations are free starting from location 300.
   a. **(0.2 points)** Draw a table depicting addresses and values stored in memory locations, if we are using Big Endian notation. Discover ASCII values for characters using an ASCII-chart (e.g., find it on the Internet).
   b. **(0.2 points)** Repeat part a) but for Little Endian notation.
   c. **(0.2 points)** What is the number of misaligned 1-byte words?
   d. **(0.2 points)** What is the number of misaligned 2-byte words?
   e. **(0.2 points)** What is the number of misaligned 4-byte words?

2. **(1.5 points)** Assume that we are introducing a new instruction into our architecture so that a sequence of ADD/STORE instructions can be replaced with one ADD instruction in the following manner:

   \[
   \text{ADD R3, R2, R1} \\
   \text{STORE R3, 0(Rb)}
   \]

   becomes

   \[
   \text{ADD 0(Rb), R2, R1}
   \]

   Assume also that adding this new instruction increases clock cycle time by 10%.
   a. **(1 point)** What is the minimum percentage of instructions we must be able to replace to actually benefit from this new instruction (Hint: compare old and new CPUtime). Assume that CPI stays the same.
   b. **(0.5 points)** Now assume that I have another optimization in mind – which will result in clock speedup by 30% (new clock cycle will be 70% of the old clock cycle). This optimization is applied to the original processor, i.e. without optimization a. Calculate the percentage of instructions that have to be replaceable so that option a) is better than option b).

3. **(1 point)** Assume that registers R2, R3 and R4 store values 100, 200 and 400 respectively. Also assume that Mem[96]=20, Mem[100] = 300, Mem[300] = 500, Mem[400]=100 and Mem[600]=38. What value will be stored in register R1 and in R2 in each of the following cases:
   a. **(0.1 points)** ADD R1, R2, R4
   b. **(0.1 points)** ADD R1, R2, #4
   c. **(0.1 points)** ADD R1, R2, (R4)
   d. **(0.1 points)** ADD R1, R2, 300(R2)
   e. **(0.1 points)** ADD R1, R2, (R2+ R3)
   f. **(0.1 points)** ADD R1, R2, (100)
   g. **(0.1 points)** ADD R1, R2, @(R2)
   h. **(0.1 points)** ADD R1, R3, (R2)+ (assume d=4)
   i. **(0.1 points)** ADD R1, R3, -(R2) (assume d=4)
   j. **(0.1 points)** ADD R1, R2, 100(R2)[R2] (assume d=4)
4. **(1.5 points)** An instruction mix has 30% of loads and 20% of stores.
   a. **(0.5 points)** How many memory accesses we have per instruction, on the average (bear in mind that each instruction has to be read from memory)? How many data accesses we have per instruction, on the average? What percentage of all memory accesses are data accesses?
   b. **(0.5 points)** What percentage of memory accesses are read accesses?
   c. **(0.25 points)** What percentage of data accesses are read accesses?
   d. **(0.25 points)** What percentage of data accesses are write accesses?

5. **(1.5 points)** In our architecture, CPI of ALU instructions is 3, CPI of LOAD is 5, CPI of STORE is 6 and CPI of all other instructions is 4. In our typical instruction mix we have 15% of LOAD instructions, 5% of STORE, and 40% of ALU instructions.
   a. **(0.5 points)** What is the average CPI?
   b. **(1 point)** Assume that we are considering an enhancement that replaces sequence of instructions:
      
      ALU R1, R2, R3
      STORE R1, 100(R4)
      
      with
      
      ALU 100(R4), R2, R3
      
      We can still use old format of ALU instruction where necessary. Old ALU instruction format has all three operands in registers and CPI of 3. Newly introduced ALU instruction has CPI of 6 and stores the result in memory. Assume that \( x \)% of STORE instructions are replaceable with a new ALU instruction. How large is new instruction count (expressed as function of \( x \))? How large is new average CPI (expressed as function of \( x \))? For what values of \( x \) do we actually benefit from the enhancement?