Question 8 is worth 20 points.

Question 0 __________
Question 1 __________
Question 2 __________
Question 3 __________
Question 4 __________
Question 5 __________
Question 6 __________
Question 7 __________
Question 8 __________

TOTAL __________
0) The quarter is exactly half-way over. What are the (2) things you would like to see improved in this class, to make it ‘better’ for you—either from a grading or from a learning aspect.

1) Write the decimal number -5.375 in single precision IEEE 754 floating pt. I want to see all 32 bits. Please group your exponent bits together in groups of 4 and your significand in groups of 4.

2) Computer (4 GHz) has 3 instruction classes: GHz = 10^9 Hz
   class A  CPI=5
   class B  CPI=1
   class C  CPI=20
The same program is compiled with 2 compilers: Billions of instructions in each class (billion=10^9):

<table>
<thead>
<tr>
<th>Compiler</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiler1</td>
<td>1</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>Compiler2</td>
<td>33</td>
<td>2</td>
<td>9</td>
</tr>
</tbody>
</table>

Determine the execution time for each version of the program.

Determine the MIPS figure for each version of the program.
A programmer was assigned by her boss the job of improving performance of a program. The program consists of 8 independent sections with the following percents of total execution time for each section:

- Section A: 10%
- Section B: 2%
- Section C: 15%
- Section D: 32%
- Section E: 6%
- Section F: 11%
- Section G: 40%
- Section H: 1%

i) She was able to speed up code section C by a factor of 4. (This means the new section C uses 1/4 the number of cycles as the old section C.) What is the speedup for the entire program? Your answer should be > 1 and be rounded to the hundredths place. X.XX

ii) What is the speedup for the entire program if section D is eliminated? (Assume the slow version of section C.) Your answer should be > 1 and be rounded to the hundredths place. X.XX

iii) What is the speedup for the entire program if (i) and (ii) are both implemented? (This means the faster code section C AND the elimination of code section D.) Your answer should be > 1 and be rounded to the hundredths place. X.XX
4) Multiply a times b using the 2’s Complement multiplication algorithm. You will get points for each intermediate step so show ALL your work. Circle the results of your shifts. (4 circled lines). Assume negative numbers, and that overflow is possible.

\[ a = 0110, \quad b = 1001 \]

Circle your 8-bit answer

5) We have a multi-cycle implementation machine that is running at 4 GHz. We have a program that has 9,100,000 instructions with this mix:

- loads \( 40\% \)
- stores \( 10\% \)
- arithmetic \( 25\% \)
- branches and jumps \( 25\% \)

What is the CPI? _____________

What is the total execution time? _______________
Consider the instruction: `beq $t0, $t3, EXIT` through the implementation on page 7 of this test. After the instructions is decoded:

What are Instruction bits 25-21 __ __ __ __ __

What are Instruction bits 20-16 __ __ __ __ __

What is ALUSrc __ (0, 1, or X <- don't care)

What is ALUOp __

7. Multicore

10% of the time a particular program is completely sequential and observes no explicit parallelism.

Assume you have MACHINE-1 => 100-core multicore processor. Each of these cores is a ‘thin’ computing unit.

Assume now you have MACHINE-2 => 90-core multicore processor, PLUS ‘1’ single ‘fat’ core, which achieves 2x sequential performance over the conventional ‘thin’ core. Hence this 90-core + 1-fat core has the same area as the 100-core processor. (The ‘fat’ core requires 10x more physical resources than the ‘thin’ core, to achieve this 2x performance speedup in sequential.)

What is the Speedup observed for this particular program between MACHINE-1 and MACHINE-2.
A) Write a loop in MIPS that adds up all the ints stored in an array of size 100. I have started it for you. Use my registers ($t0, $s1, $s2) as declared in the comments. Use only MIPS core instructions. Feel free to use registers $t1 - $t7.

    # $t0 holds the address of the first int in the array
    addi $s1, $t0, 400    #first word not in the array
    addi $s2, $zero, 0    #sum initialized to zero
    LP:

B) Unroll your loop (just the code that starts at LP) so that it executes 50 times and has no data hazards. Write your code to the right of your code from part A).

C) Given the registers you see used in the starter code of A), what register(s) must be saved to the stack frame in order to follow MIPS calling conventions?