Question 1 (25 points): Provide a brief answer for each of the questions below.

1. (**5 points**) In Chapter 1 of the textbook we studied that the execution time of a computer program in a given machine can be computed by the following expression:

$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock Cycle}} \quad (1)$$

   This expression can be simplified to:

$$\text{CPU Time} = \text{IC} \times \text{CPI} \times \frac{1}{\text{Frequency}} \quad (2)$$

   where IC is the Instruction Count and CPI is the average number of Clock Per Instruction. Explain which of the three components of the execution time (IC, CPI and frequency) are affected by the choice of algorithm, the choice of programming language, the compiler, and the instruction set architecture.

   The algorithm affects IC (and possibly CPI), the compiler and the programming language affect IC and CPI, the ISA affects IC, CPI and frequency.

2. (**5 points**) Now that you have completed CMPUT229 you should be aware of some shortcomings of a performance analysis that uses the expression above for modern microprocessors. What are these shortcomings?

   The expression above does not take into consideration the effects of the memory hierarchy into performance, and does not include the influence of branch prediction, pipelined execution, virtual memory, or the interference of context switching and interruptions with the memory-hierarchy performance.

3. (**5 points**) What is the role of a sticky bit in a floating-point unit? How does this bit affect the rounding process.

   The sticky bit's role is to extend the precision of the computation by allowing the floating-point unit to differentiate between a number that has 0.1000...000 after the lowest bit that the unit can represent and a number that has 0.10000...001 after the lowest bit. The former number is rounded to the nearest even number, the later is always rounded up when the sticky bit is used.

4. (**5 points**) Explain how increasing the associativity of a cache memory may improve the performance of the system.

   Higher associativity prevents the occurrence of conflict misses that occur when multiple references are competing for the same cache entry and cause misses even when other entries in the same cache are not in use.

5. (**5 points**) Two basic approaches to generate code to traverse an array are: (i) an index-based loop where an index variable is incremented in each iteration of the loop and used to access array elements; (ii) a pointer-based loop where a pointer is used to access array elements and the pointer is incremented in each iteration of the loop. What are the advantages/disadvantages of each of these approaches?

The index-based approach results in a code that is easier to read and to maintain, but results in more instructions inside the loop and therefore in slower code. An ideal solution is to use an index-based approach at the source-code level and then use a compiler that automatically converts the index-based loops into pointer-based loops.