Exercise for Slot2 - CSI106
Exercise 1: Fill in the Blanks

"A computer consists of three main components: the CPU, main memory, and the input/output subsystem. Within the CPU, there are three main parts: **the arithmetic logic unit (ALU), control unit, and registers**. **Arithmetic operations on integers and reals**."

Exercise 2: Classify Instruction Types

Classify the following instructions into two main types: CISC or RISC.

1. ADD – **CISC**
2. MOV – **RISC**
3. MUL – **CISC**
4. DIV – **CISC**
5. XOR – **RISC**

Exercise 3: Calculate Necessary Bits

A computer has 256 MB of memory. Each word in this computer is four bytes. Calculate the number of bits needed to address a word in memory.

The memory address space is 256MB, or 2^{28} $(2^8 \times 2^{20})$.
Each word is four (2^2) bytes, which means that we have 2^{26} words
We need $log_2\, 2^{26} = 26$ bits to address each words

Exercise 4: Compare Computer Architectures

Compare CISC and RISC architectures. Highlight the main differences between them and the advantages/disadvantages of each architecture.

	CISC	RISC
Architecture	**Complex Instructions:** CISC processors typically have a wide variety of complex instructions that can perform multiple operations in a single instruction. **Variable-Length Instructions:** Instructions in CISC architectures can vary in length, making decoding more complex. **Memory Access:** CISC architectures often have instructions that directly access memory, reducing the	**Simplified Instruction Set:** RISC processors have a simplified instruction set, focusing on basic operations that can be executed efficiently in a single clock cycle. **Fixed-Length Instructions:** Instructions in RISC architectures are typically fixed in length, simplifying the decoding process. **Load/Store Architecture:** RISC architectures often follow a load/store model,

	number of instructions needed for certain tasks. **Emphasis on Hardware:** CISC architectures tend to rely more heavily on hardware to execute complex instructions efficiently. **Examples:** x86 architecture (Intel processors), Motorola 68k series.	where only specific load and store instructions access memory directly. **Emphasis on Software:** RISC architectures prioritize simple hardware operations, relying more on software optimization for performance. **Examples:** ARM, MIPS, PowerPC.
Advantage	**Code Density:** CISC instructions can often achieve more functionality with fewer instructions, leading to denser code. **Versatility:** Complex instructions can be useful for certain tasks, especially those involving memory access or manipulation.	**Simplicity:** The simplified instruction set and fixed-length instructions make RISC architectures easier to decode and execute, often resulting in faster performance. **Efficient Pipelining:** RISC architectures are conducive to efficient pipelining, allowing for higher clock speeds and better performance. **Compiler Optimization:** RISC architectures benefit greatly from compiler optimization, as simpler instructions are easier to optimize.
Disadvantage	**Complexity:** The wide variety of complex instructions can make the architecture and instruction decoding complex, potentially slowing down instruction execution. **Limited Pipelining:** Complex instructions can hinder pipelining efficiency, impacting performance.	**Code Size:** RISC instructions may sometimes require more instructions to accomplish certain tasks, leading to larger code sizes compared to equivalent CISC implementations. **Memory Access Overhead:** RISC architectures may require more memory accesses for certain operations due to the load/store model, potentially impacting performance in memory-bound tasks.