UNIVERSITÀ DEGLI STUDI DI MILANO
DIPARTIMENTO DI INFORMATICA

1. Fundamentals of Computer Science

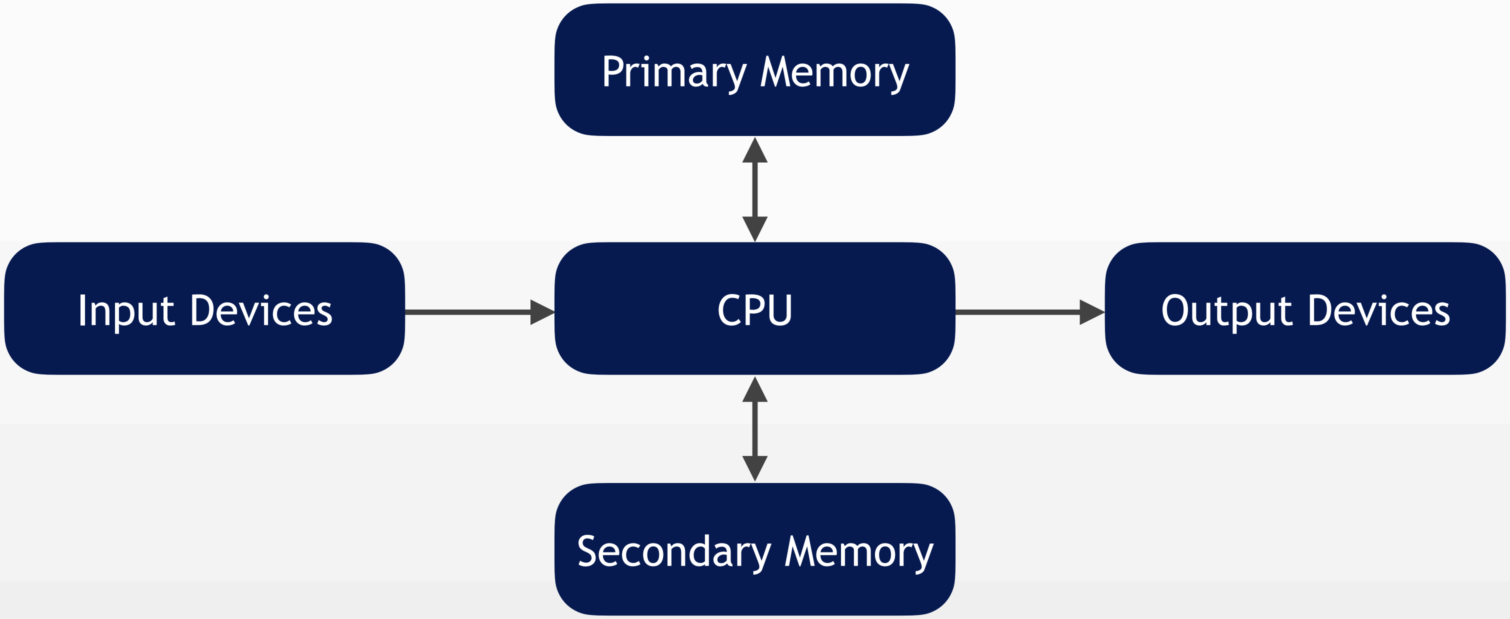
Computer

A **computer** is a machine that stores and manipulates information

Computing is the process of using computer technology to complete a given goal-oriented task

Information is data that is processed, organized, structured, or presented in a way that makes it useful in a given context

Computer Components



I/O Devices

An **input device** is any hardware device that sends data to a computer, allowing you to interact with and control it (*keyboard*, *mouse*, *microphone*, …)

An **output device** is any hardware device used to send data from a computer to another device or user (*monitor*, *speakers*, *printer*, …)

CPU

A **central processing unit** (CPU) is the principal part of any digital computer system, generally composed of the control unit, processor registers and the arithmetic-logic unit (ALU)

Data is transferred via the main memory to the ALU for processing, which involves basic arithmetic functions and logic operations

Primary Memory

Primary memory is the only memory directly accessible by the CPU

It allows computers to store data for immediate manipulation and to monitor what is currently being processed

The main primary memories used in computers are **RAM** and **ROM**

Secondary Memory

Secondary memory is computer memory that is non-volatile, persistent and not immediately accessible by a computer or processor

It allows users to store data and information that can be retrieved, transmitted, and used by apps and services quickly and easily (*hard disks*, *DVD*, ...)

Instruction Cycle

The **instruction cycle**, or **fetch-decode-execute cycle**, is the cycle that the CPU performs in order to process instructions

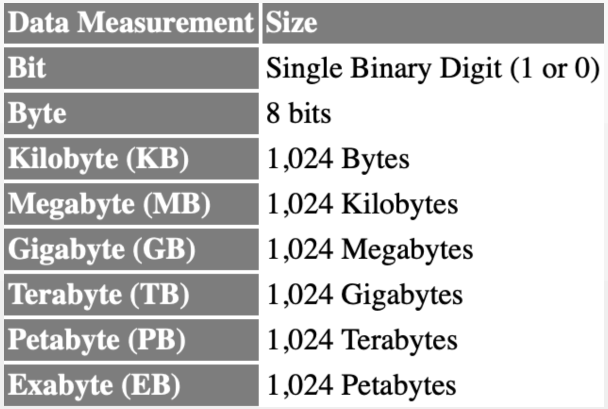
It is composed of the following phases:

- *fetch*: the next instruction is retrieved from the memory address where it is stored
- *decode*: the instruction is interpreted by the decoder
- *execute*: the CPU performs the actions required by the instruction

Information Representation

The smallest quantity of information that a computer can store and handle is called a **bit** (binary digit): it can have two values, *1* or *0*

The more bits, the more different values you can represent: a single bit permits 2 values, 2 bits give 4 values, 3 bits gives 8 values, and in general n bits gives 2^n



Data Measurement	Size
Bit	Single Binary Digit (1 or 0)
Byte	8 bits
Kilobyte (KB)	1,024 Bytes
Megabyte (MB)	1,024 Kilobytes
Gigabyte (GB)	1,024 Megabytes
Terabyte (TB)	1,024 Gigabytes
Petabyte (PB)	1,024 Terabytes
Exabyte (EB)	1,024 Petabytes

Positional Notation

A number N in base b is written as a sequence of digits $\mathbf{a}$

$$N = \sum_{i=-m}^{n} a_i b^i = a_n b^n + \ldots + a_{-m} b^{-m}$$

Base 10: $152_{10} = 1 \cdot 10^2 + 5 \cdot 10^1 + 2 \cdot 10^0$

Base 2: $10011000_2 = 1 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$

Convert Decimal to Binary

Division	*Quotient*	*Remainder*
$152 \div 2$	76	0
$76 \div 2$	38	0
$38 \div 2$	19	0
$19 \div 2$	9	1
$9 \div 2$	4	1
$4 \div 2$	2	0
$2 \div 2$	1	0
$1 \div 2$	0	1

$$152_{10} = 10011000_2$$

Binary Operations

Addition

$$\begin{array}{cccccccccc}
0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & + \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & = \\
\hline
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 &
\end{array}$$

Subtraction

$$\begin{array}{cccccccccc}
1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & - \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & = \\
\hline
0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 &
\end{array}$$

Machine Language

Machine language is a collection of binary digits or bits that the CPU reads and interprets

It is also known as machine code or object code and is extremely tough to comprehend

It is the only language a computer is capable of understanding; different machine code is used by different processor architectures

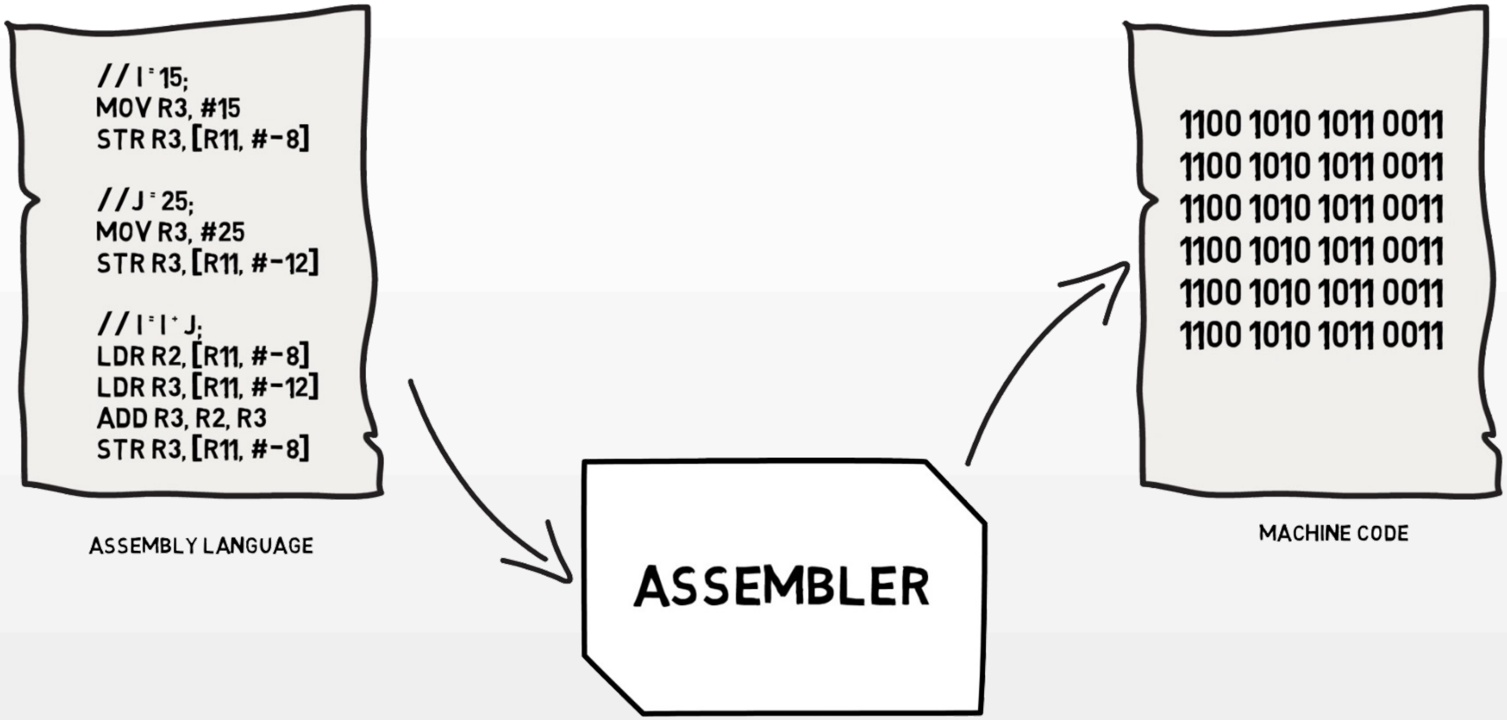
Assembly Language

Assembly language is a human-only language that is not understood by computers

It requires the usage of an **assembler** to convert instructions into machine or object code

Mnemonics such as Mov, Add, Sub, End, and others make up the assembly language, which people can understand, utilize, and apply

Assembly Language



```
// I = 15;
MOV R3, #15
STR R3, [R11, #-8]

// J = 25;
MOV R3, #25
STR R3, [R11, #-12]

// I = I + J;
LDR R2, [R11, #-8]
LDR R3, [R11, #-12]
ADD R3, R2, R3
STR R3, [R11, #-8]
```

```
1100 1010 1011 0011
1100 1010 1011 0011
1100 1010 1011 0011
1100 1010 1011 0011
1100 1010 1011 0011
1100 1010 1011 0011
```

High-Level Language

A **high-level language** is any programming language that enables development of a program in a much more user-friendly programming context and is generally independent of the computer's hardware architecture

It has a higher level of abstraction from the computer, and focuses more on the programming logic rather than the underlying hardware components such as memory addressing and register utilization

Compiler vs. Interpreter

Both a **compiler** and an **interpreter** are used to convert a program written in a high-level language into machine code understood by computers

	Interpreter	Compiler
Translation	by statement	entire program
Time	fast analysis, slow execution	slow analysis, fast execution
Object Code	not generated	generated
Programming Languages	Python, PHP, Javascript, Ruby, …	Java, C, C++, Scala, …

Python

Python is a high-level, interpreted, general-purpose programming language created by Guido van Rossum, and released in 1991

It runs on different platforms and it has a simple syntax similar to English

It supports a lot of libraries for data analysis and machine learning