Basic Concepts of Assembly Language

Learning Assembly Language

A. Why Learn Assembly Language

Intro

Assembly language bridges the gap between human-readable code and the raw binary instructions of machine code. By working this close to the hardware, you gain unparalleled insight into how computers truly operate; from memory management to CPU behavior. While modern developers often rely on high-level languages, mastering assembly unlocks critical skills for system programming, reverse engineering, and squeezing out peak performance in resource-constrained environments like embedded systems.

Learning assembly doesn't just teach you a language, it reshapes how you think about programming.

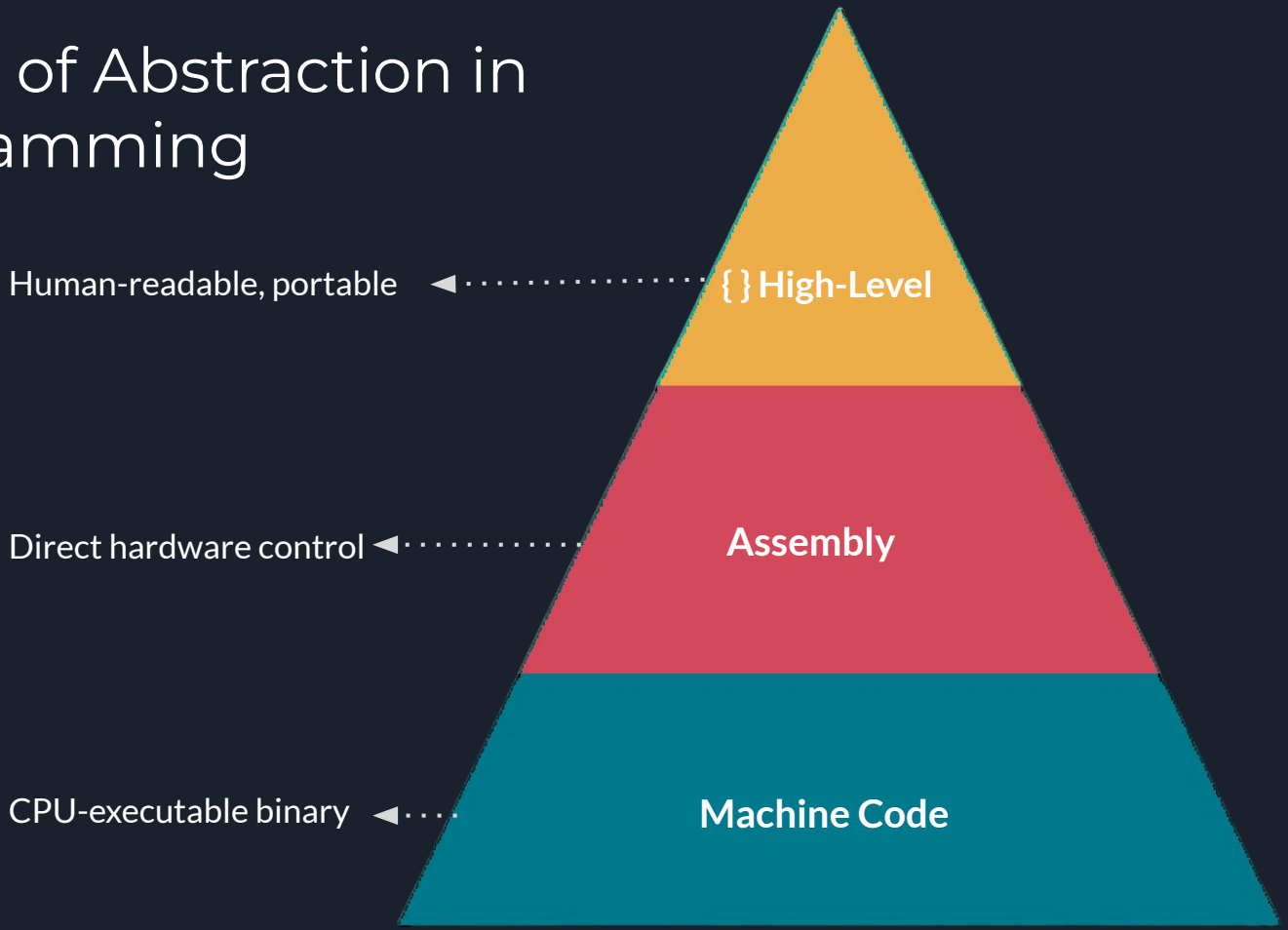
It's used in:

- System programming (OS kernels, drivers)
- Reverse engineering
- Performance optimization
- Embedded systems

Why is it important to learn it?

- Learning it makes you a better programmer with a deeper understanding of memory, registers, and instructions.

Levels of Abstraction in Programming



B. How Data Are Represented

Points:

- Computers use binary (0 and 1) to store and process data.
- A bit is the smallest unit of data; 8 bits = 1 byte.
- **Numbers:**
  - Unsigned integers → only positive values (e.g., 0–255)
  - Signed integers → use two's complement to store negative numbers
  - Floating-point numbers → use binary scientific notation
- Characters:
  - Represented using standards like ASCII or Unicode

Example Chart: Decimal → Binary → ASCII Character

Character	ASCII Decimal	Binary (8-bit)
A	65	01000001
B	66	01000010
C	67	01000011
a	97	01100001
b	98	01100010
1	49	00110001
2	50	00110010
Space	32	00100000

C. Boolean expressions

This topic is divided into 3 subparts:

1. What is binary logic
2. How to use truth tables
3. Boolean functions

1. What is Binary Logic?

- Computers operate using binary values: 0 = False, 1 = True
- Basic Boolean logic gates:

- **AND** → True only if both inputs are 1
- **OR** → True if at least one input is 1
- **NOT** → Reverses the value (1 → 0, 0 → 1)

2. How to use truth tables

A	B	A AND B	A OR B	NOT A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Truth tables show the output of Boolean operations for all possible input combinations.

3. Boolean functions

- A Boolean function takes one or more binary inputs and gives a binary output.
- Example: Majority(A, B, C)

  → Returns 1 if 2 or more inputs are 1

- Widely used in hardware circuits and logic design