



The Islamia University of Bahawalpur Pakistan



# Computer Architecture & Organization



*Instructor: Hafiz Jawad Ahmad*

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

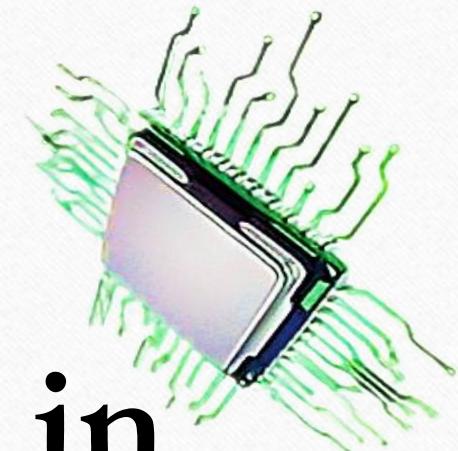
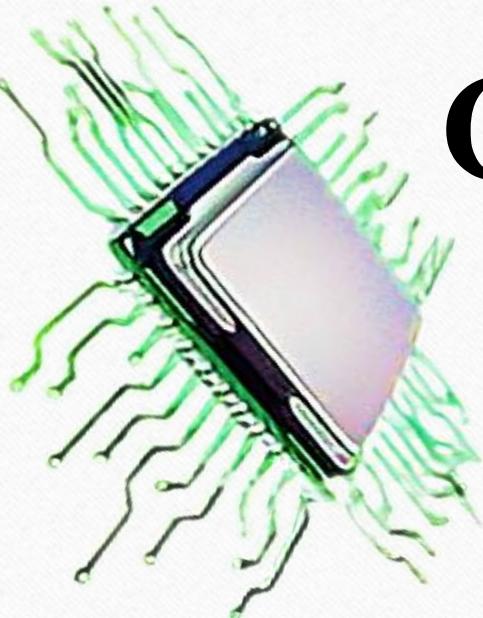
ترجمہ: شروع اللہ کے پاک نام سے جو بڑا مہربان نہایت رحم والا ہے۔

رَبِّ زِدْنِيْ عِلْمًا

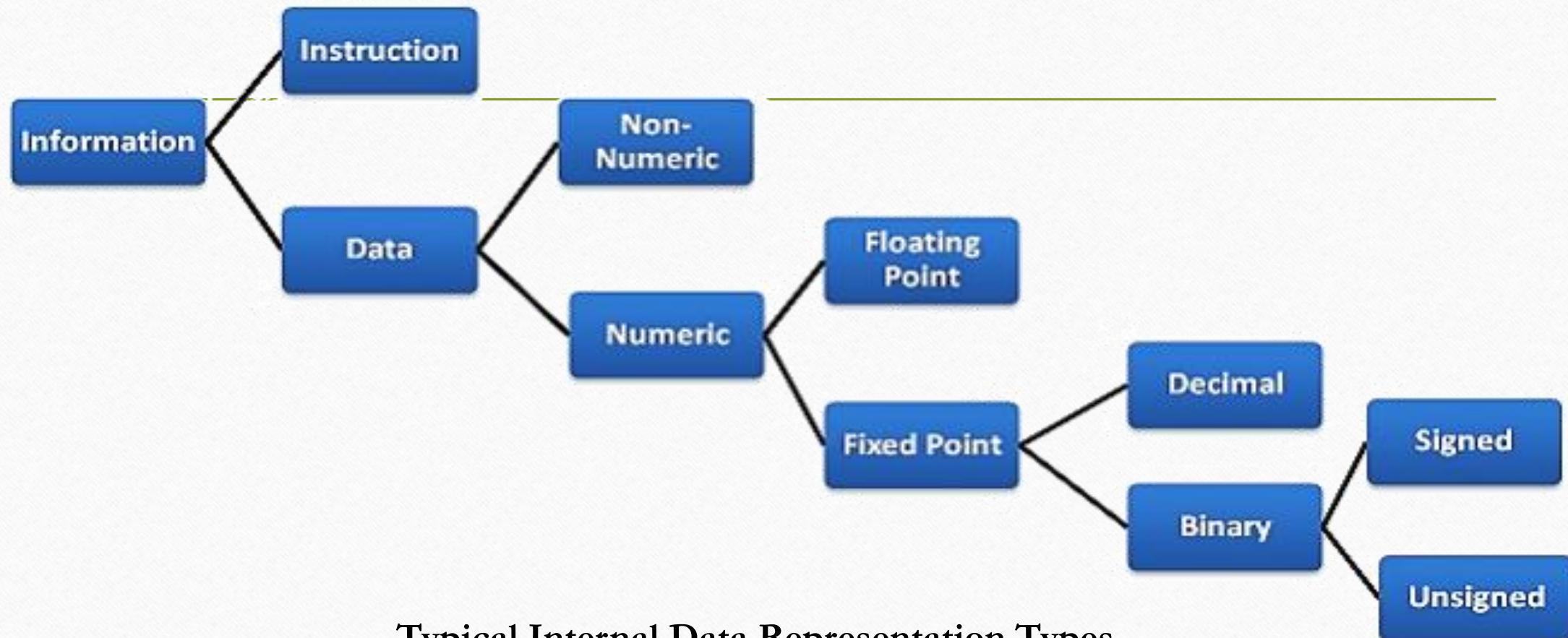
ترجمہ: اے میرے رب! میرے علم میں اضافہ فرم۔

سورہ طہ: (۱۶): پارہ نمبر- (16)

# Data Representation in Computer System



# Data Representation in Computer System



# Data Representation

- 1) Integer Numbers (Signed/ Unsigned)
- 2) Binary Numbers (base 2 (0 or 1))
- 3) Octal Numbers (Base 8 (0 to 7))
- 4) Decimal Numbers (Base 10 (**Floating point numbers**))
- 5) Hexa Decimal (Base16 (0 to 9 & A to F))

## Other Data Types Are

- 1) Alphanumeric Text data (**Alphanumeric A,a,B,b, #,@,%,\*,\$,<,=,\*,/ ,+,-, 0,1 etc.**)
- 2) Picture Data (.jpg, .jpeg, .png, .gif, .bmp, .tiff, .tif, .svg, .webp, .heif, .heic, .raw, .pcx, .pdf, .ico, .eps, .psd, .dng etc.)
- 3) Audio (.mp3, .wav, .aac, .flac, .ogg, .wma, .m4a, .alac, .opus, .aiff, .dsd, .cda, .ra, .mod, .vqf. Etc.)
- 4) Video (.mp4, .avi, .mkv, .mov, .wmv, .flv, .webm, .mpeg, .mpg, .3gp, .rm, .m4v, .ts, .f4v, .qt, .dat, etc.)

Basically, Computer Deals With Binary Values

# Data Representation in Computer System

- **Data Representation in Computer Systems** refers to how information is stored, processed, and transmitted using a computer.
- Computers operate using binary (0s and 1s), and all forms of data
  - **Text** (Alphanumeric A,b, #,@,%,\*,\$, 0,1 etc.)
  - **Numbers** (0,1,2,3,4,5,6,7,8,9 etc.)
  - **Images** (.jpg, .jpeg, .png, .gif, .bmp, .tiff, .tif, .svg, .webp, .heif, .heic, .raw, .pcx, .pdf, .ico, .eps, .psd, .dng etc.)
  - **Audio** (.mp3, .wav, .aac, .flac, .ogg, .wma, .m4a, .alac, .opus, .aiff, .dsd, .cda, .ra, .mod, .vqf. etc)
  - **Video** (.mp4, .avi, .mkv, .mov, .wmv, .flv, .webm, .mpeg, .mpg, .3gp, .rm, .m4v, .ts, .f4v, .qt, .dat etc)

And all are represented in binary form.



# Representing Test

00100000	Space	00110011	3	01000110	F	01011001	Y	01101100	I
00100001	!	00110100	4	01000111	G	01011010	Z	01101101	m
00100010	"	00110101	5	01001000	H	01011011	[	01101110	n
00100011	#	00110110	6	01001001	I	01011100	\	01101111	o
00100100	\$	00110111	7	01001010	J	01011101	]	01110000	p
00100101	%	00111000	8	01001011	K	01011110	^	01110001	q
00100110	&	00111001	9	01001100	L	01011111	_	01110010	r
00100111	.	00111010	:	01001101	M	01100000	~	01110011	s
00101000	(	00111011	:	01001110	N	01100001	a	01110100	t
00101001	)	00111100	<	01001111	O	01100010	b	01110101	u
00101010	*	00111101	-	01010000	P	01100011	c	01110110	v
00101011	+	00111110	>	01010001	Q	01100100	d	01110111	w
00101100	,	00111111	?	01010010	R	01100101	e	01111000	x
00101101	-	01000000	@	01010011	S	01100110	f	01111001	y
00101110	.	01000001	A	01010100	T	01100111	g	01111010	z
00101111	/	01000010	B	01010101	U	01101000	h	01111011	{
00110000	0	01000011	C	01010110	V	01101001	i	01111100	
00110001	1	01000100	D	01010111	W	01101010	j	01111101	}
00110010	2	01000101	E	01011000	X	01101011	k	01111110	-

# List of Memory Units

Sr#	Full Form	Units	Bytes
1	Bit (b)	Binary Digits (0,1)	$2^0$ Bit
2	1 Nibble	4 Bits	$2^1$ Bits
3	1 Byte (B)	8 Bits	$2^3$ Bits
4	1 Kilobit (Kb)	1000 Bits	
5	1 Kilobyte (KB)	1024 Bytes	$2^{10}$ Bytes
6	1 Megabit (Mb)	1000,000 Bits	
7	1 Megabyte (MB)	1024 KB	$2^{20}$ Bytes
8	1 Gigabit (Gb)	1000,000,000 Bits	
9	1 Gigabyte (GB)	1024 MB	$2^{30}$ Bytes
10	1 Terabit (Tb)	1000,000,000,000 Bits	
11	1 Terabyte (TB)	1024 GB	$2^{40}$ Bytes
12	1 Petabit (Pb)	1000,000,000,000,000 Bits	

Sr#	Full Form	Units	Bytes
13	1 Petabyte (PB)	1024 TB	$2^{50}$ Bytes
14	1 Exabit (Eb)	1000,000,000,000,000 Bits	
15	1 Exabyte (EB)	1024 PB	$2^{60}$ Bytes
16	1 Zettabit (Zb)	1000,000,000,000,000,000 Bits	
17	1 Zettabyte (ZB)	1024 EB	$2^{70}$ Bytes
18	1 Yottabit (Yb)	1000,000,000,000,000,000,000 Bits	
19	1 Yottabyte (YB)	1024 ZB	$2^{80}$ Bytes
20	1 Brontobit (YB)	1000,000,000,000,000,000,000,000 Bits	
21	1 Brontobyte (YB)	1024 YB	$2^{90}$ Bytes
22	1 Geopbit	1000,000,000,000,000,000,000,000,000 Bits	
23	1 Geopbyte	1024 BB	$2^{100}$ Bytes
13	1 Petabyte (PB)	1024 TB	$2^{50}$ Bytes

## 2. Bits and Bytes

**Definition:** A bit, short for "binary digit," is the most basic unit of data in computing and digital communications. It represents one of two possible states: 0 or 1. Here's a step-by-step explanation of what a bit is and its significance:

- **0 (off, false, low, -, Close, NO)**
- **1 (on, true, high, +, Open, YES)**

Binary System Computers use the binary system to represent data. This system is based on two symbols: **0** and **1**. Each bit can represent two possible values.

- Bit:** The smallest unit of data in a computer, representing a binary state (**0 or 1**).
- Byte:** A group of 8 bits, used to represent a single character (e.g., 'A').

### Example:

- A byte can represent 256 different values (from 0 to 255).
- The letter 'A' is represented in binary as **01000001**.

# Binary Number System

---

**Definition:** The binary number system is a **base-2** numeral system that uses two symbols, typically **0** and **1**, to represent values.

**Explanation:** Computers use binary because they operate using electrical signals that can be either on (**1**) or off (**0**).

**Example:** The decimal number 5 is represented in binary as 101:

$$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 0 + 1 = 5$$

00111110110000000000000011110000  
00011010010000011110011111000100001  
1111001111001011110011010000001101  
0001001111100010110011110011110000  
0001001100000000010011110001100000  
000100110011000100100111100111100000  
00011010010000011110011111000100001  
0011010000000010011100000011001000  
0001001100000000010011110001100000  
00010011111000001001111001111000000

# Bit Representation of Data

**Definition:** Data representation refers to the methods used to encode information in a format that computers can process.

- **Bit:** 1 bit = 0, 1 are combined to create larger units of data
  - **Nibble:** 4 bits (e.g., 0001) A nibble is a unit of digital information that consists of 4 bits.)
  - **Byte (B):** 8 bits
  - **Kilobit (Kb):** 1,000 bits
  - **Kilobyte (KB):** 1,024 bytes (1 KB = 8,192 bits)
  - **Megabit (Mb):** 1,000,000 bits (or 1,000 Kb)
  - **Megabyte (MB):** 1,024 KB (1 MB = 1,048,576 bytes or 8,388,608 bits)
  - **Gigabit (Gb):** 1,000,000,000 bits (or 1,000 Mb)
  - **Gigabyte (GB):** 1,024 MB (1 GB = 1,073,741,824 bytes or 8,589,934,592 bits)
  - **Terabit (Tb):** 1,000,000,000,000 bits (or 1,000 Gb)
  - **Terabyte (TB):** 1,024 GB (1 TB = 1,099,511,627,776 bytes or 8,800,000,000,000 bits)
- etc.) follow this logarithmic expansion.
- **Bit (b):** 2 1 bit (0,1)
  - **Nibble:** 4 bits
  - **Byte (B):** 8 bits
  - **Kilobit (Kb):** 1,000 bits
  - **Kilobyte (KB):** 1,024 bytes (1 KB = 8,192 bits)
  - **Megabit (Mb):** 1,000,000 bits
  - **Megabyte (MB):** 1,024 KB (1 MB = 8,388,608 bits)
  - **Gigabit (Gb):** 1,000,000,000 bits
  - **Gigabyte (GB):** 1,024 MB (1 GB = 8,589,934,592 bits)
  - **Terabit (Tb):** 1,000,000,000,000 bits
  - **Terabyte (TB):** 1,024 GB (1 TB = 8,796,093,022,208 bits)
  - **Petabit (Pb):** 1,000,000,000,000,000 bits
  - **Petabyte (PB):** 1,024 TB (1 PB = 1,125,899,906,842,624 bits)
  - **Exabit (Eb):** 1,000,000,000,000,000,000 bits
  - **Exabyte (EB):** 1,024 PB (1 EB = 1,152,921,504,606,846,976 bits)
  - **Zettabit (Zb):** 1,000,000,000,000,000,000 bits
  - **Zettabyte (ZB):** 1,024 EB (1 ZB = 1,180,591,620,717,411,303,424 bits)
  - **Yottabit (Yb):** 1,000,000,000,000,000,000,000 bits
  - **Yottabyte (YB):** 1,024 ZB (1 YB = 1,208,925,819,614,629,174,706,176 bits)

# Types of Data

---

- **Integers:** Whole numbers represented in binary.
- **Characters:** Represented using encoding schemes like ASCII or Unicode.

**Example:**

- The integer 10 in binary is 1010.
- The character 'A' in ASCII is represented by the decimal value 65, which is 01000001 in binary.

# Programs

---

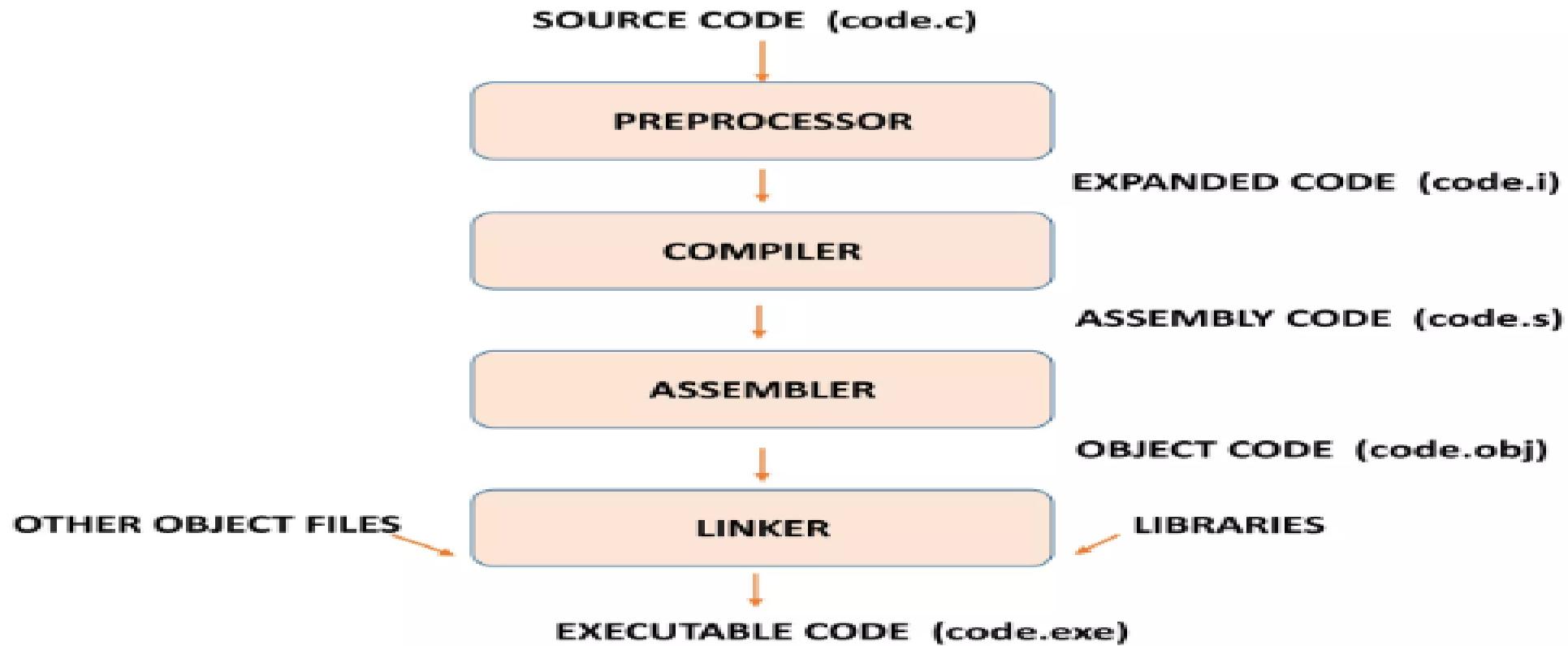
- **Programs:** A program is a set of instructions written in a programming language that tells a computer how to perform a specific task.

# Compilation

---

- **Compilation:** This is the process of converting source code written in a high-level programming language into machine code or intermediate code that a computer's processor can execute.
- The compilation process typically involves several stages, including lexical analysis, syntax analysis, semantic analysis, optimization, and code generation.

# Compilation Process



# Types Of Compiler

---

- **Preprocessor:** This is the first phase through which source code is passed.
- **Compiler:** Converts the high level Language into machine code as a whole
- **Interpreter:** Converts the high level language into machine code line by line
- **Assembler:** Converts assembly language code into machine code

# Compilation Systems

---

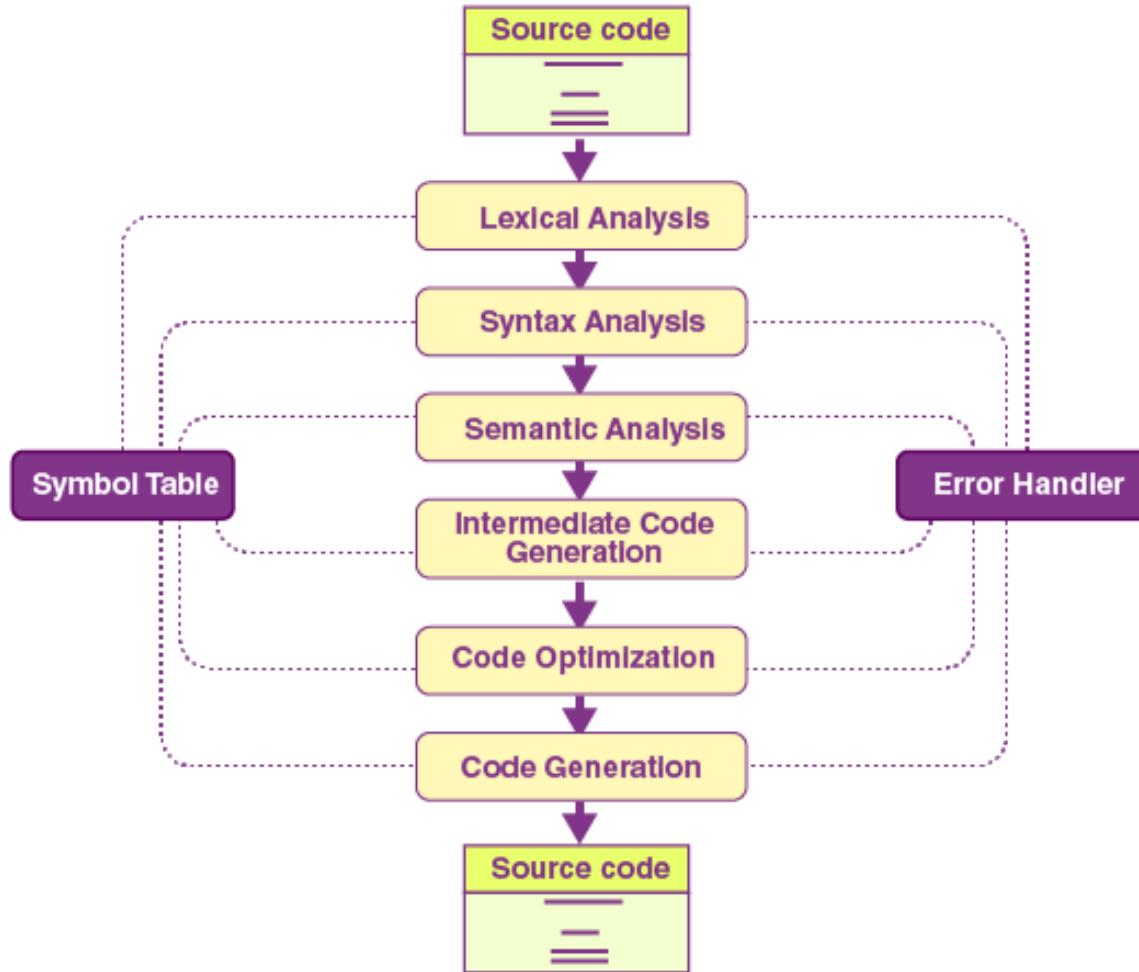
- **Compilation Systems:** These are the tools and processes involved in compiling programs. A typical compilation system includes:
  - **Compiler:** The main program that performs the compilation.
  - **Linker:** Combines different object files or libraries into a single executable file.
  - **Loader:** Loads the executable into memory for execution.
  - **Debugger:** A tool for testing and debugging to find errors in code that is being compiled .

# Stages of Compilation

---

- **Lexical Analysis:** Breaking down the code into tokens.
- **Syntax Analysis:** Checking the code structure against grammar rules.
- **Semantic Analysis:** Ensuring the code makes logical sense.
- **Optimization:** Improving code efficiency.
- **Code Generation:** Producing machine code.
- **Example:** Compiling a C program involves converting the code written in C into machine code that can run on a specific CPU architecture.

# Stages of Compilation



# Instruction Flow

---

- **Instruction Flow:** This refers to the sequence in which instructions are executed by the processor. It can include:
  - **Sequential Execution:** Instructions are executed one after another.
  - **Branching:** The flow can change based on conditions (e.g., if-else statements).
  - **Looping:** Instructions can be repeated based on certain conditions (e.g., for loops, while loops).
  - **Function Calls:** The flow can jump to a different part of the program to execute a function and then return.

# Program

---

- A **program** in computer system architecture refers to a set of instructions that a computer can execute to perform specific tasks. The development and execution of a program involve multiple steps and components within a computer system. Here's a step-by-step explanation of how a program is processed from conception to execution in computer architecture:

# Programming Phase

---

- Coding: Write the actual code in a programming language (such as Python, C++, Java, etc.).
- Syntax Checking: Ensure that the written code adheres to the syntax rules of the programming language.

# Compilation/Interpretation

---

- **Compilation (for compiled languages):** The program's source code is translated into machine code or intermediate code by a compiler.
- **Interpretation (for interpreted languages):** An interpreter processes the high-level code and executes it line by line

# Loading

---

- Once compiled, the machine code (or bytecode in some languages) is loaded into the computer's memory, specifically into RAM.
- The operating system manages this loading process, allocating the necessary resources

# Fetch, Decode & Execute Cycle

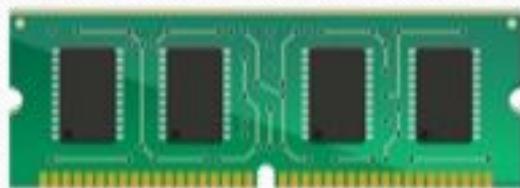
---

- **Instruction Cycle:** This is typically broken down into:
- **Fetch:** The CPU retrieves an instruction from memory.
- **Decode:** The CPU interprets what the instruction means.
- **Execute:** The CPU performs the specified operation.
- **Store (Optional):** If the instruction involves writing back results, the CPU stores them in memory.

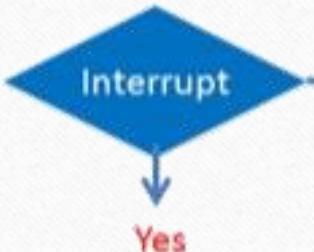
**START**



Disk Memory  
Secondary Memory  
Operating System Loads  
Program Into RAM



Primary Memory  
Main Memory  
RAM



Service Interrupt

**PROGRAM**

- 1 Instruction.
- 2 Instruction.
- 3 Instruction.
- 4 Instruction.
- 5 Instruction.
- 6 Instruction.

Program Is Set Of  
Instructions Stored In The  
Main Memory  
RAM



Processor CPU

