Operating Systems

CE206

Ruchika Patil
Assistant Professor,
Computer department
SPIT

CREDITS

Course (Category) Code	Course Name	Teaching Scheme (Hrs/week)					Credits Assigned			
		L	T	P	O	E	L	T	P	Total
(PC)	Operating Systems	3	0	2	5	10	3	0	1	4
CE206		Examination Scheme								
		Component	ISE		MSE		ESE			Total
		Theory	75		75		150			300
		Laboratory	50		--		50			100

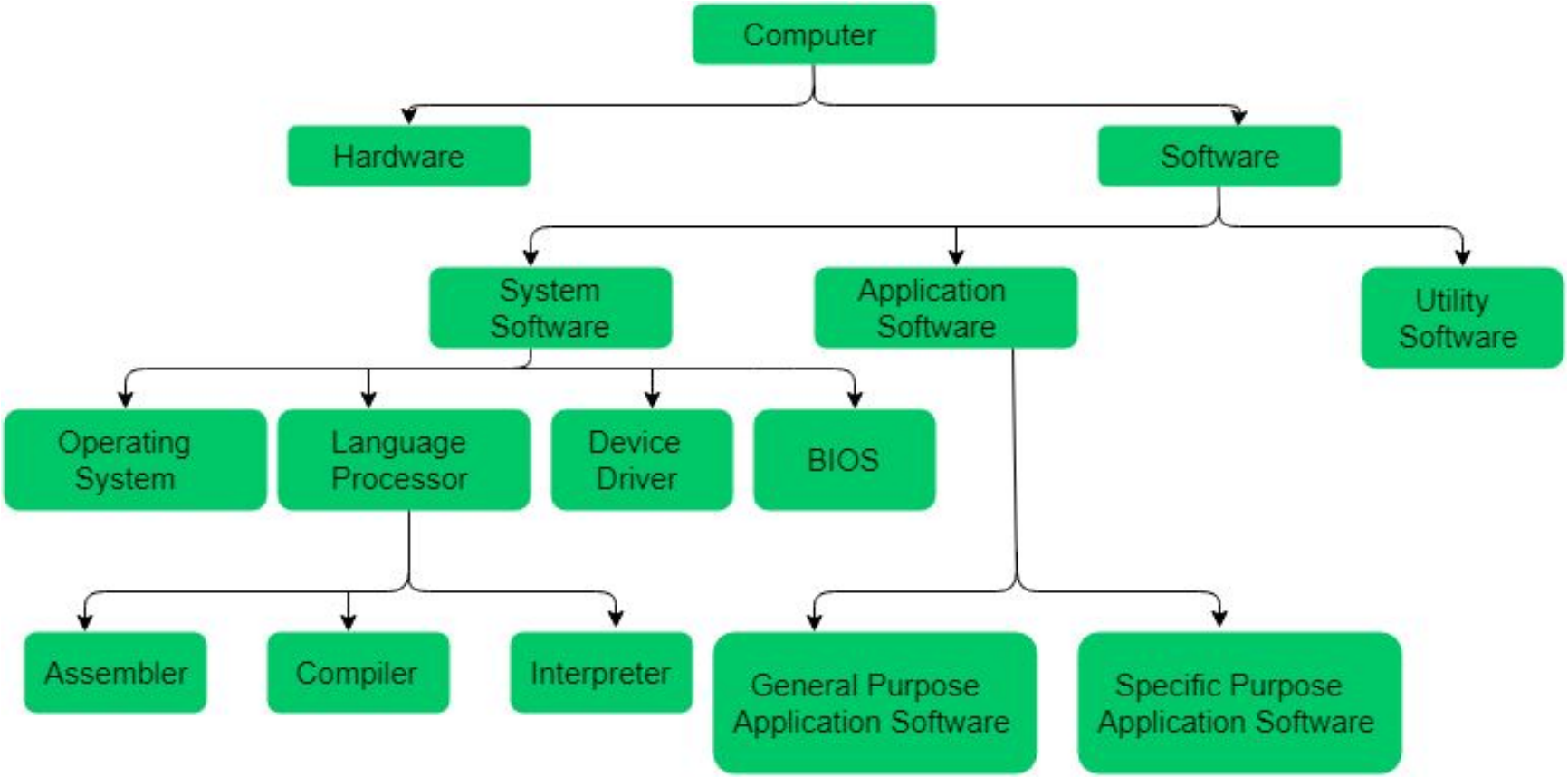
Course Outcomes (CO)

Pre-requisite Course Codes, if any.	Computer Architecture and Organization
Course Objective: To understand structure of OS, process synchronization, memory management and file system.	
Course Outcomes (CO): At *the End of the course students will be able to*	
CE206.1	Comprehend the primitive concepts of Operating System services and System Programming functionality.
CE206.2	Articulate process scheduling algorithms in effective execution of processes.
CE206.3	Acquaint with efficient process synchronization techniques in effective execution of programs.
CE206.4	Analyze virtual memory management algorithms in effective allocation of main memory usage.
CE206.5	Evaluates various algorithms of File Storage & I/O management for performance and quality criterion.

Books

1) System Programming by D M Dhamdhere
2) System Programming by John Donovan
3) Operating System Concepts by Abraham Silberschatz, Peter B Galvin, Greg Gagne
4) Operating Systems: Internals and Design Principles by William Stallings
5) Modern Operating Systems by Andrew S. Tanenbaum, Herbert Bos
6) Design of the UNIX Operating Systems by Maurice J. Bach

Introduction to System Software and Operating Systems


Computer
Hardware
Software
System Software
Application Software
Utility Software
Operating System
Language Processor
Device Driver
BIOS
Assembler
Compiler
Interpreter
General Purpose Application Software
Specific Purpose Application Software

Software

Computer software, or just software, is a general term primarily used for set of instructions such as computer programs and other kinds of information read and written by computers.

System Softwares:

- System softwares is a set of program that manages the resources of computer system.
- Acts as the computer hardware and application software interface.
- Provides platform to other softwares.
- The end-user usually does not directly communicate with the system software; instead, the user interacts with the user interface created by the system software.
- Programs written in low level language.
- General Purpose Software
- Examples: Operating systems, Utilities- BIOS, Device drivers

SOFTWARES OF COMPUTER

- **SYSTEM SOFTWARE**
- System software provides basic functionality to the computer. System software is required for the working of computer itself. The user of computer does not need to be aware about the functioning of system software, while using the computer. For example, when you buy a computer, the system software would also include different device drivers.
- The ***purposes Of system software are:***
- To provide basic functionality to computer,
- To control computer hardware
- To act as an interface between *user, application software and computer hardware.*

System softwares can be broadly classified into:

System control programs

- controls the execution of programs,
- manage the storage & processing resources of the computer & perform other management & monitoring function.

System support programs

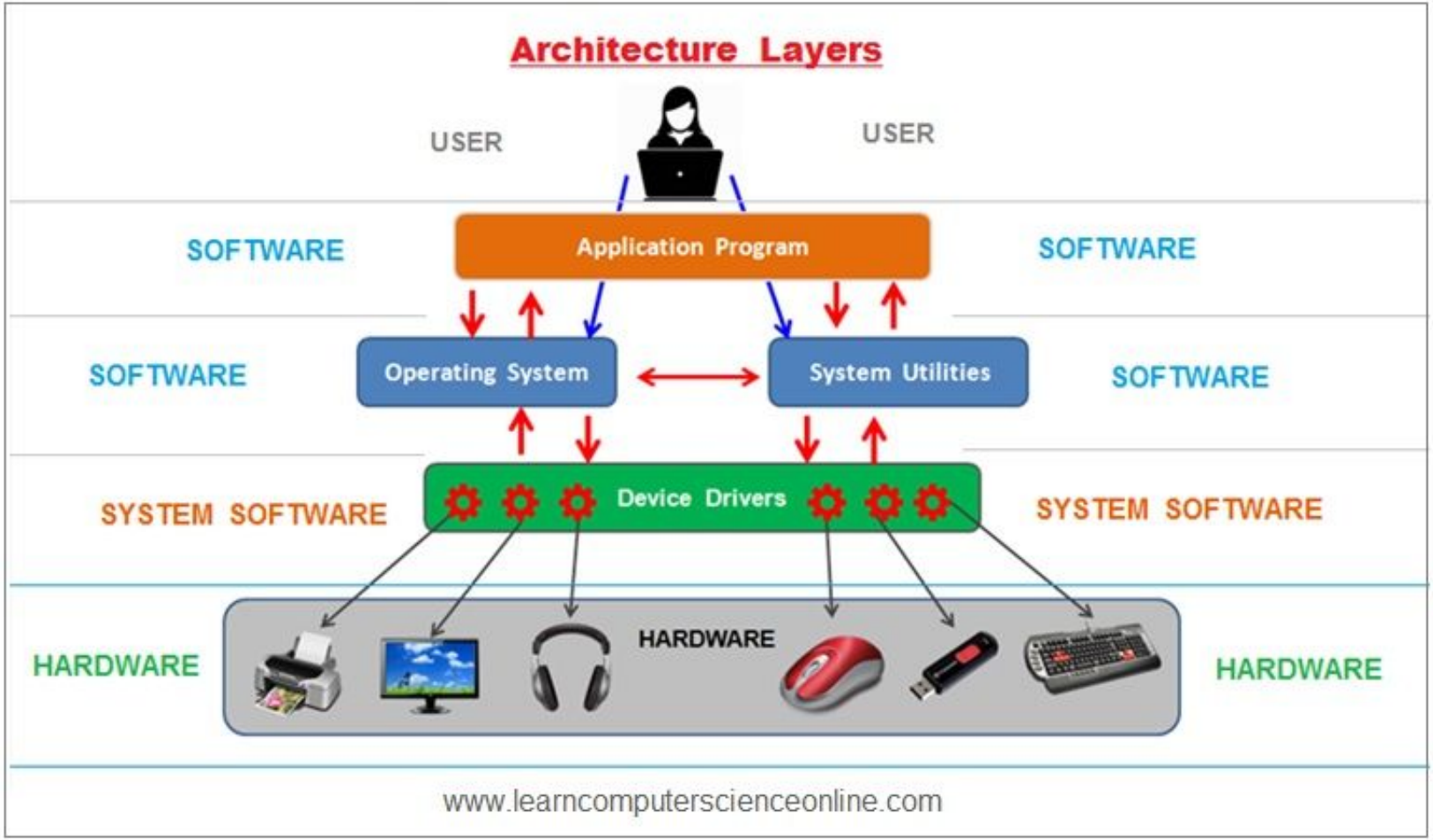
- provide routine service functions to the other computer programs & computer users

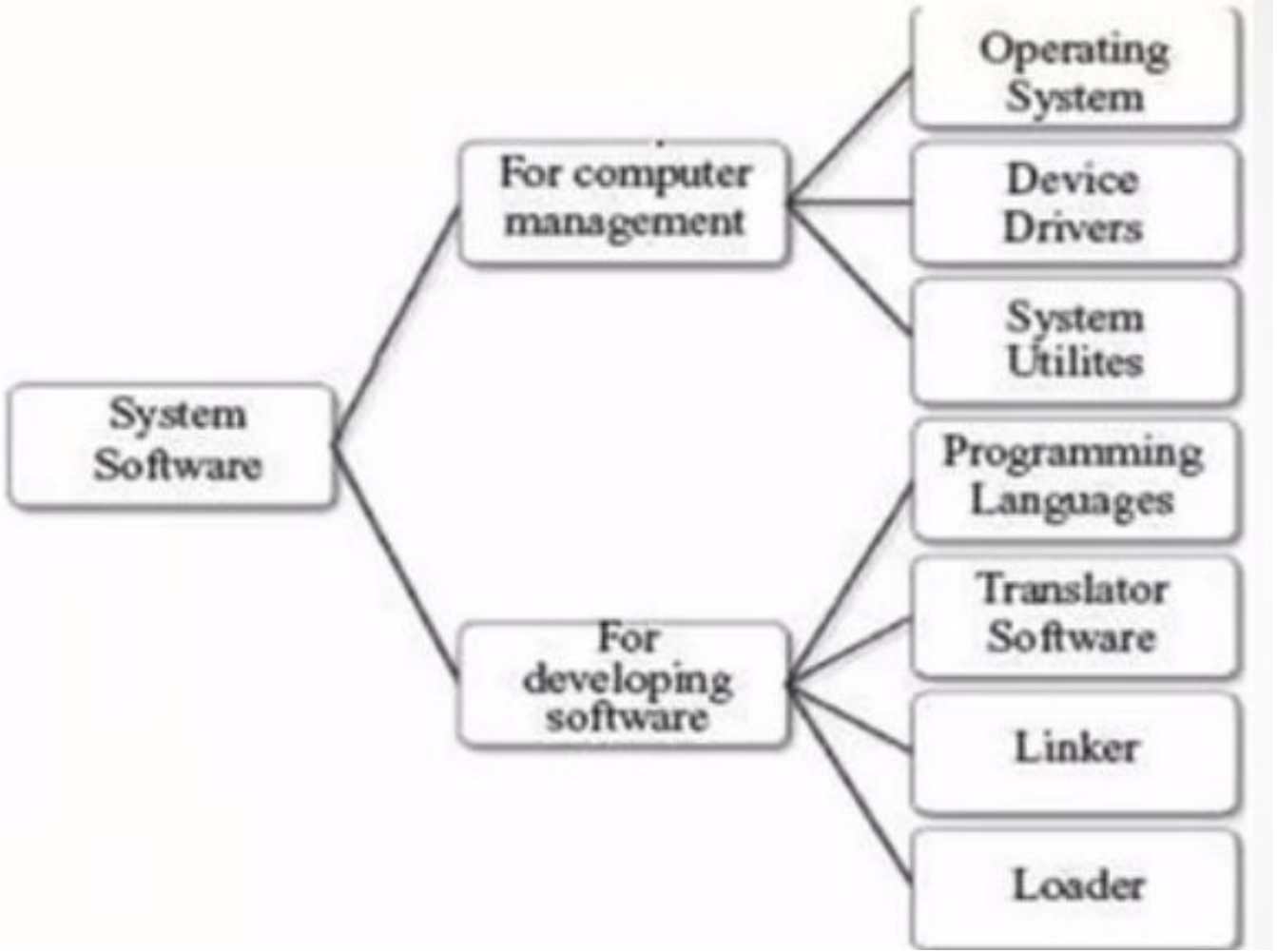
Eg. libraries, performance monitors & job accounting.

Application Software:

- Application software is a program that performs a specific task for the end user.
- It runs on the platform provided by system software.
- It acts as a platform between the system software and the end user.
- Word-processing, Spreadsheet, Database, etc.

Computer System - Architecture Layers



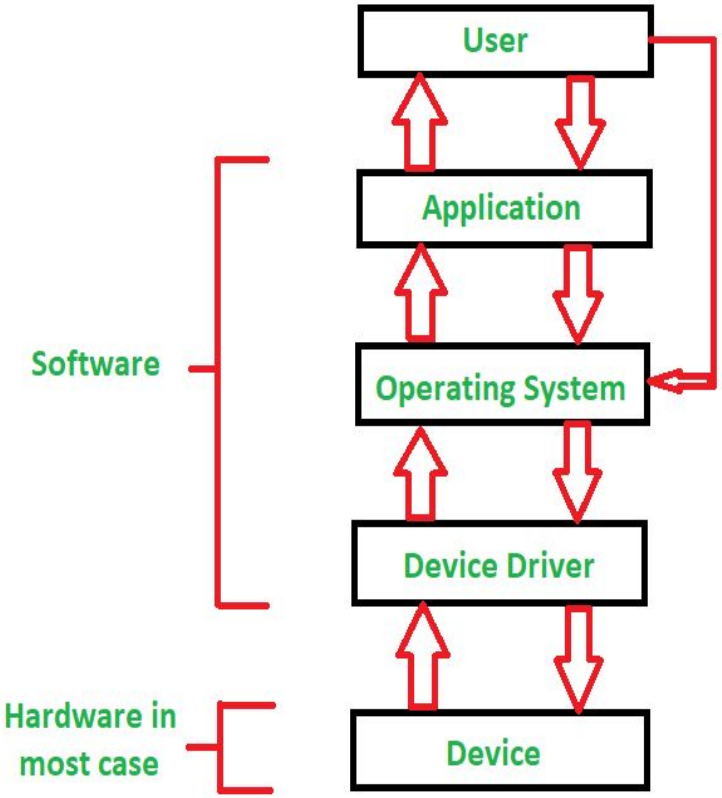

System Software
For computer management
For developing software
Operating System
Device Drivers
System Utilites
Programming Languages
Translator Software
Linker
Loader

Operating System:

A program that acts as an intermediary between a user of a computer and the computer hardware

Device Drivers:

Device Driver in computing refers to a special kind of software program or a specific type of software application that controls a specific hardware device that enables different hardware devices to communicate with the computer's Operating System



System utilities

- The Utility Software is system software that helps to maintain the proper and smooth functioning of a Computer System
- Assists the Operating System to manage, organize, maintain, and optimize the functioning of the computer system.
- EXAMPLES: Antivirus software, file management tools, data compression tools, disk management tools, disk cleaners, network managers etc.

System Programming

Typical Computer Structure

Main components:

- CPU
- Main Memory
- Secondary: disk
- IO devices:

  Input:

  • Keyboard

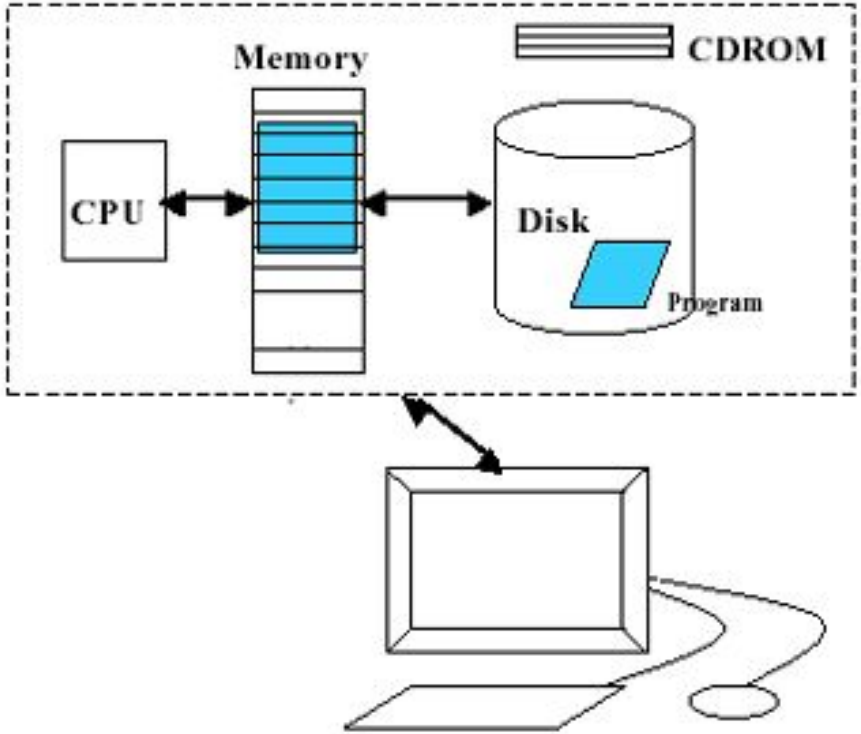
  • Mouse

  • CDROM

  Output:

  • Display

  • Printer

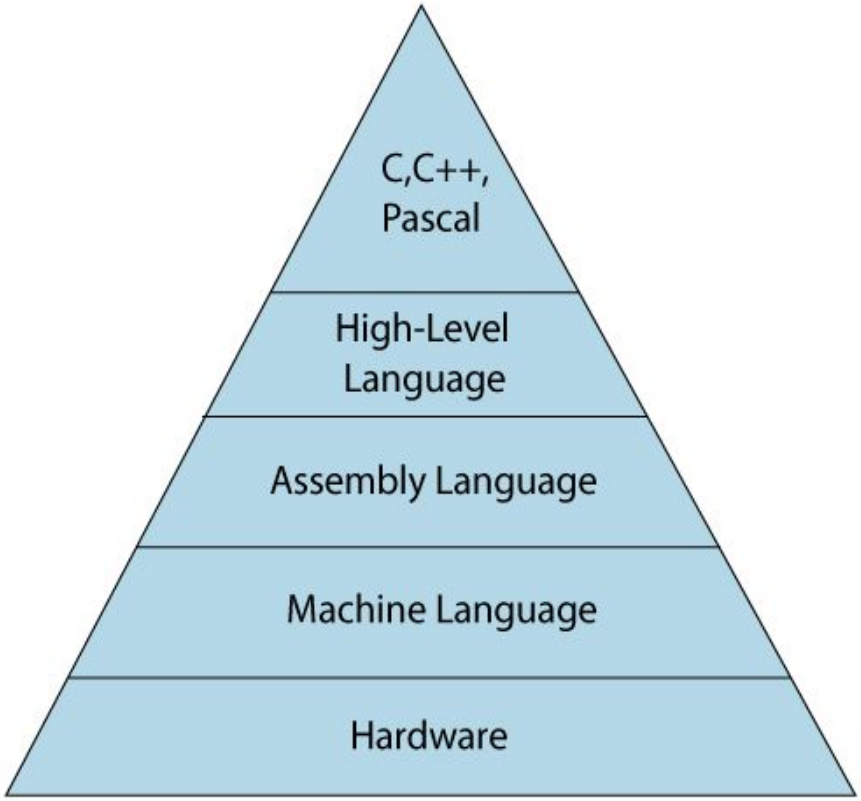


Programming Language

A programming language defines a set of instructions that are compiled together to perform a specific task by the CPU (Central Processing Unit).

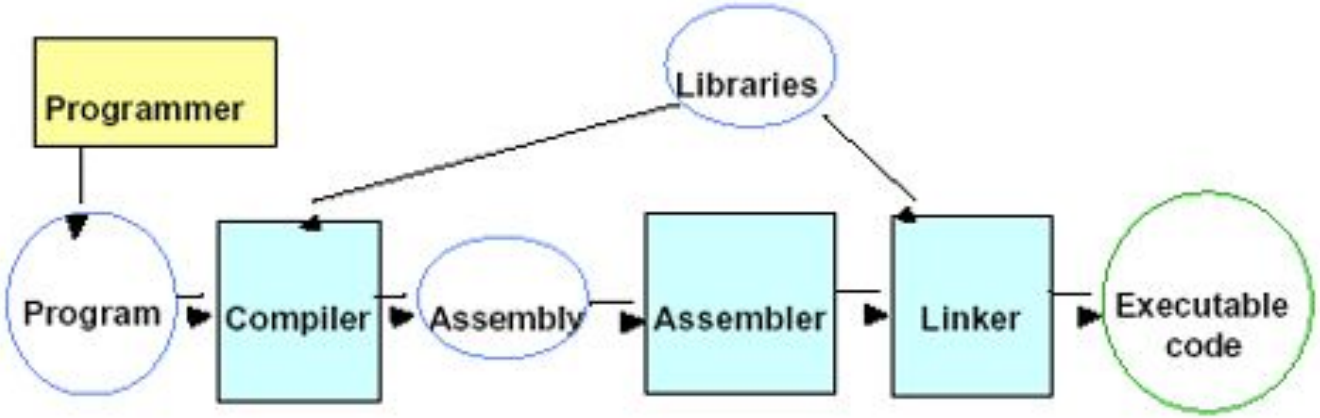
```c
#include <stdio.h>

int main(){
  char str[] = "Hello World\n";
  printf("%s", str);
}
```



https://www.javatpoint.com/classification-of-programming-languages

“Building” a Program

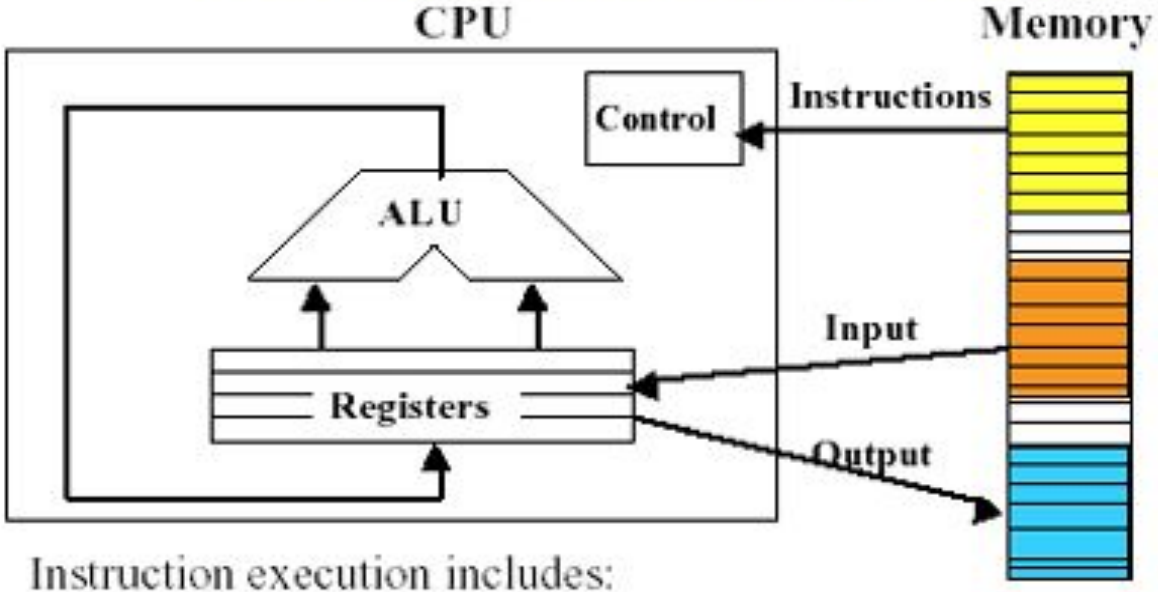



```
z = x + y;
```

```
load    r1, x
load    r2, y
add     r3, r1, r2
store   r3, z
```

```
1100 0110
1010 1111
0101 1000
1010 1111
0101 1000
0000 1001
1100 0110
0000 1001
```

Execution of the Instructions

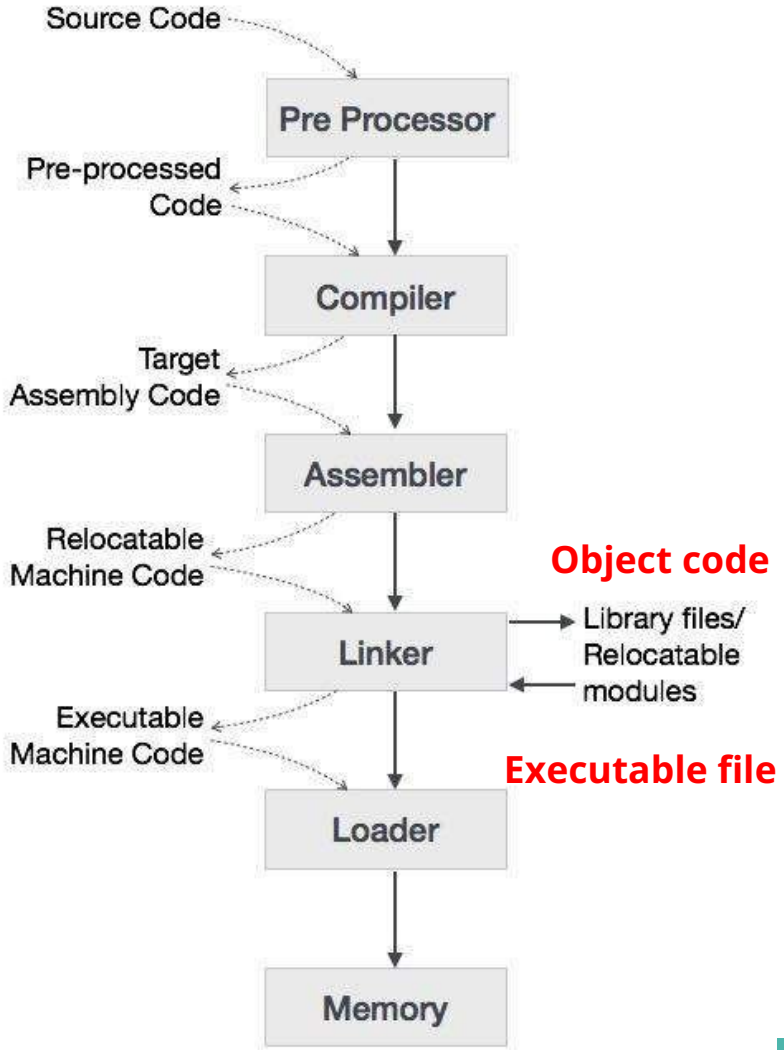


Instruction execution includes:

1. Load instruction from memory
2. Decode instruction
3. Load data from memory/registers
4. Execute the operation
5. Store result in register/memory

z = x + y;

```
load    r1, x
load    r2, y
add     r3, r1, r2
store   r3, z
```


Source Code
Pre Processor
Pre-processed Code
Compiler
Target Assembly Code
Assembler
Relocatable Machine Code
Object code
Linker
Library files/ Relocatable modules
Executable Machine Code
Executable file
Loader
Memory

Preprocessor

A preprocessor, generally considered as a part of compiler, is a tool that produces input for compilers. It deals with macro-processing, augmentation, file inclusion, language extension, etc.

Softwares of Computer

Translator Software:

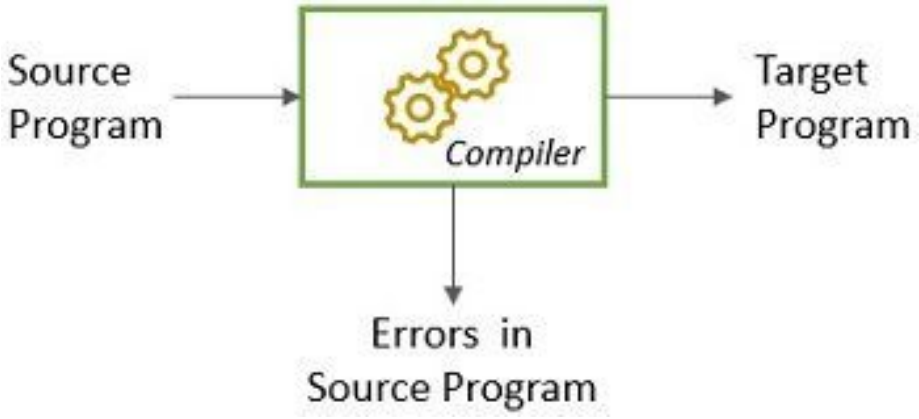
- **Used to convert a program written in high level language/assembly to a form that machines can understand.**
- **The translated program is called Object code**
- **Three kinds of Translator software:**

❑ **Assembler**
❑ **Compiler**
❑ **Interpreter**

Compiler

- A compiler is a program that converts high-level language to assembly language.

- Specifies the error at the end of compilation with line numbers



Interpreter

- Translates high-level language into low-level machine language.
- A compiler reads the whole source code at once, creates tokens, checks semantics, generates intermediate code, executes the whole program and may involve many passes. In contrast, an interpreter reads a statement from the input, converts it to an intermediate code, executes it, then takes the next statement in sequence.

Interpreter

- If an error occurs, an interpreter stops execution and reports it. whereas a compiler reads the whole program even if it encounters several errors.

Assembler

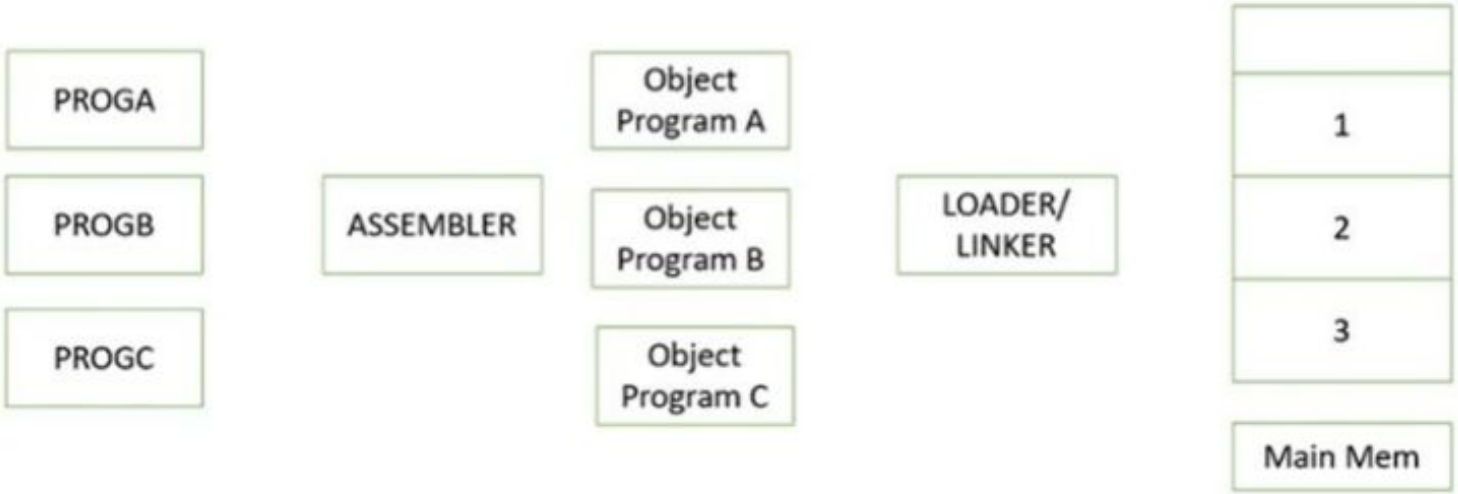
- An assembler translates assembly language programs into machine code.
- The output of an assembler is called an object file, which contains a combination of machine instructions as well as the data required to place these instructions in memory.

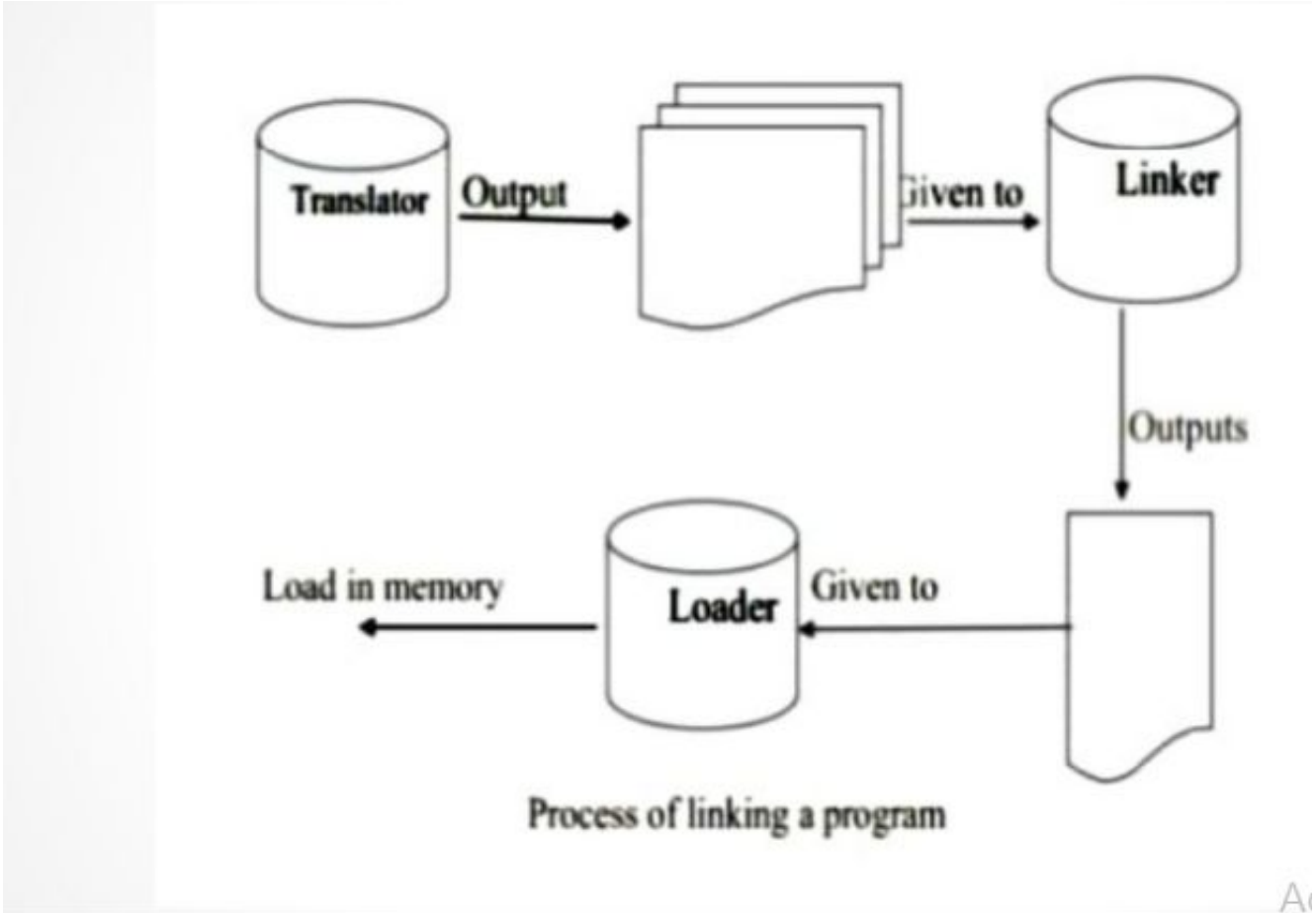
Linker

- Linker is a computer program that links and merges various object files together in order to make an executable file.
- The major task of a linker is to search and locate referenced module/routines in a program and to determine the memory location where these codes will be loaded, making the program instruction to have absolute references.
- Produces .exe file

Program Linking

- three controls sections.
  - Assembled together or assembled independently.





Process of linking a program

Loaders:

- Loader is utility program which takes object code as input prepares it for execution and loads the executable code into the memory. Thus loader is actually responsible for initiating the execution process.

- The loader is responsible for the activities such as allocation, linking, relocation and loading

Types of linking:

1. Static Linking –

- It is performed during the compilation of source program.
- Linking is performed before execution in static linking.
- It takes collection of relocatable object file and command-line arguments and generates a fully linked object file that can be loaded and run.

Types of linking:

2. Dynamic linking –

- Dynamic linking is performed during the run time.
- This linking is accomplished by placing the name of a shareable library in the executable image.
- There are more chances of errors and failures.
- It require less memory space as multiple programs can share a single copy of the library

Functions of Loader:

1) It allocates the space for program in the memory, by calculating the size of the program. This activity is called allocation.

2) It resolves the symbolic references (code/data) between the object modules by assigning all the user subroutine and library subroutine addresses. This activity is called linking.

Functions of Loader:

3) There are some address dependent locations in the program, such address constants must be adjusted according to allocated space, such activity done by loader is called relocation.

4) Finally it places all the machine instructions and data of corresponding programs and subroutines into the memory. Thus program now becomes ready for execution, this activity is called loading.

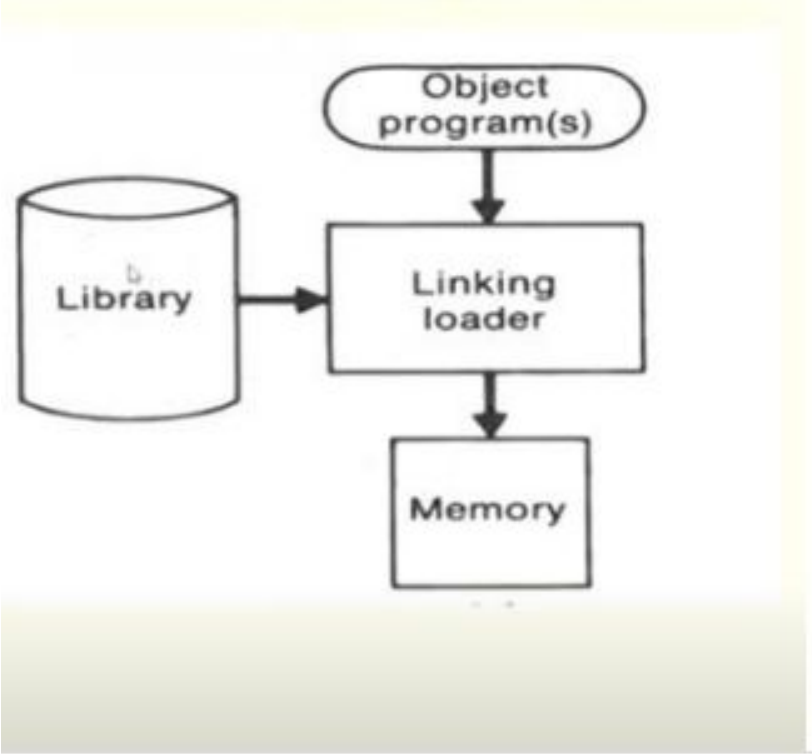
Design of Linker

1. Linking Loader

2.Linkage Editor

Dynamic Linking

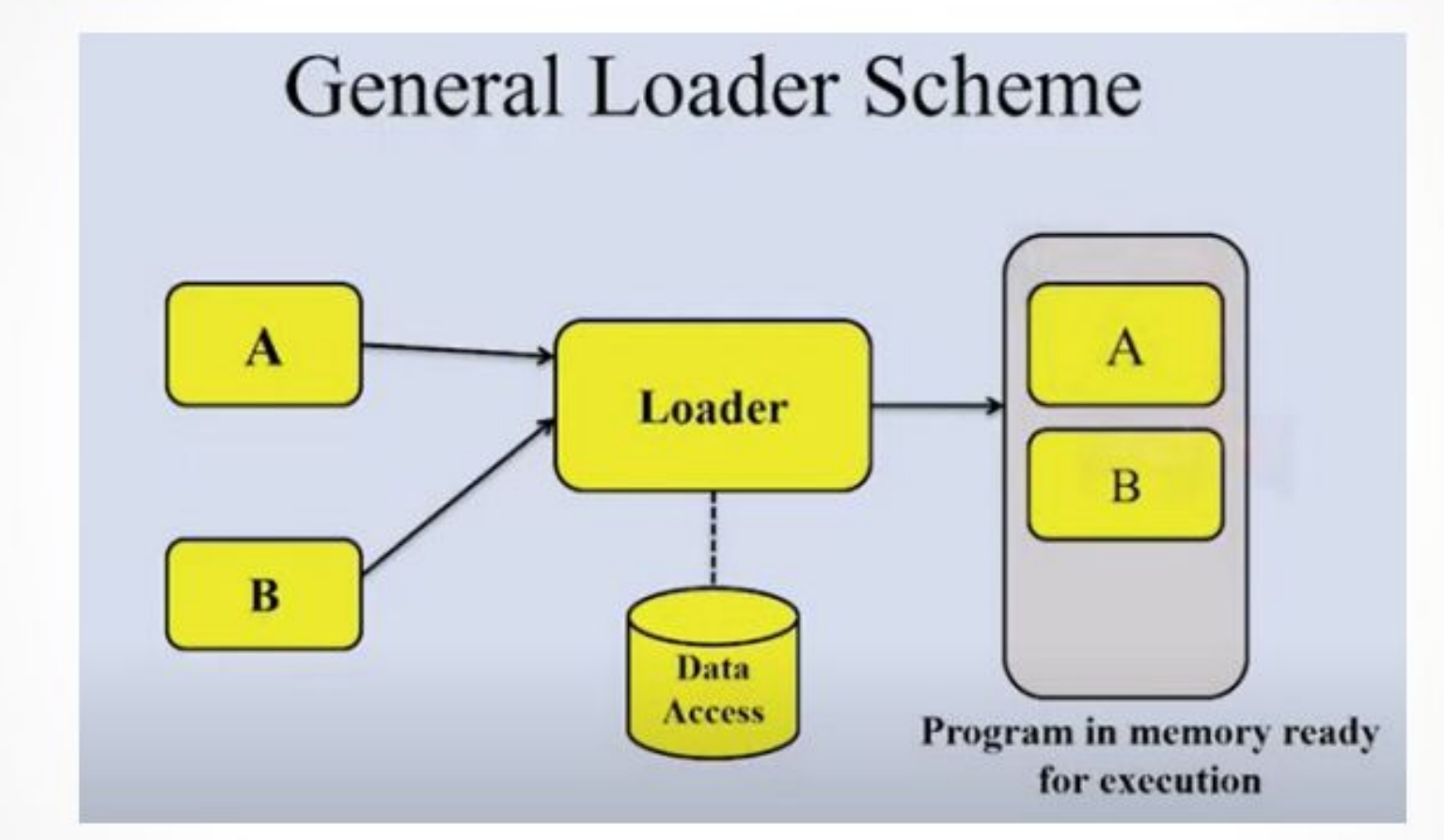
Linking Loader



Each time program calls the library function it will loaded into memory for execution by linking loader

Loader

- Loader is a part of operating system and is responsible for loading executable files into memory and execute them.
- It calculates the size of a program (instructions and data) and creates memory space for it. It initializes various registers to initiate execution.


General Loader Scheme
A
B
Loader
Data Access
A
B
Program in memory ready for execution

Macro processors

A macro processor is a program that reads a file (or files) and scans them for certain keywords. When a keyword is found, it is replaced by some text. The keyword/text combination is called a macro

Computer System Components

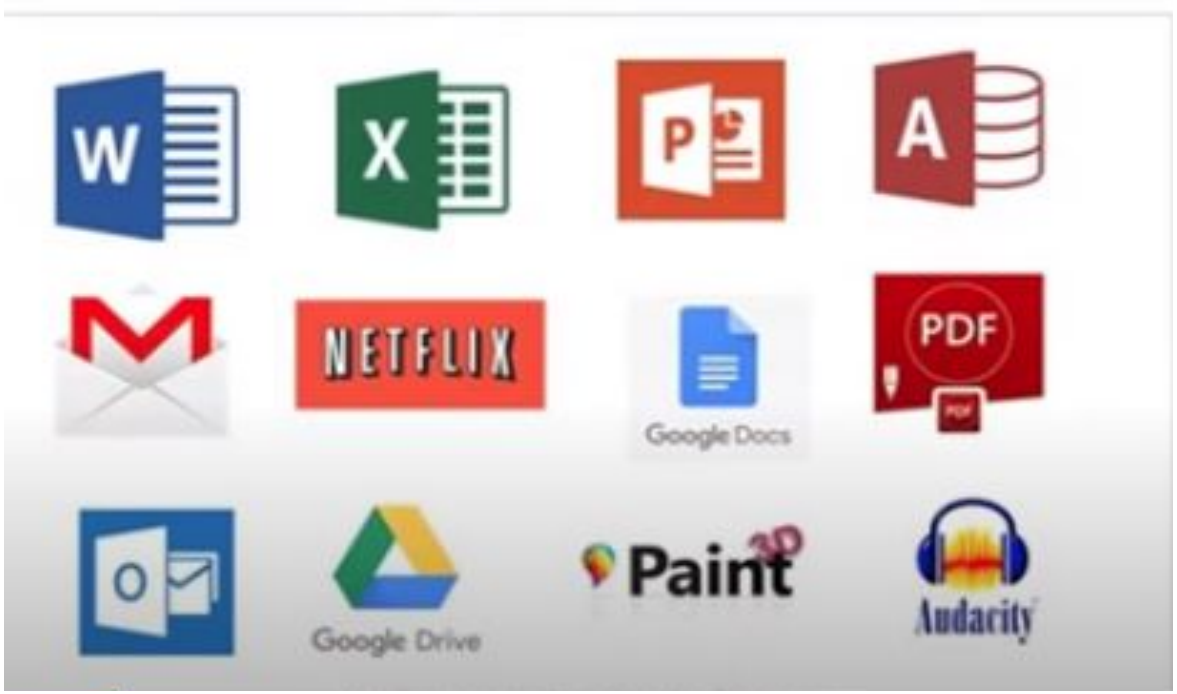
1.Hardware – provides basic computing resources (CPU, memory, I/O devices).

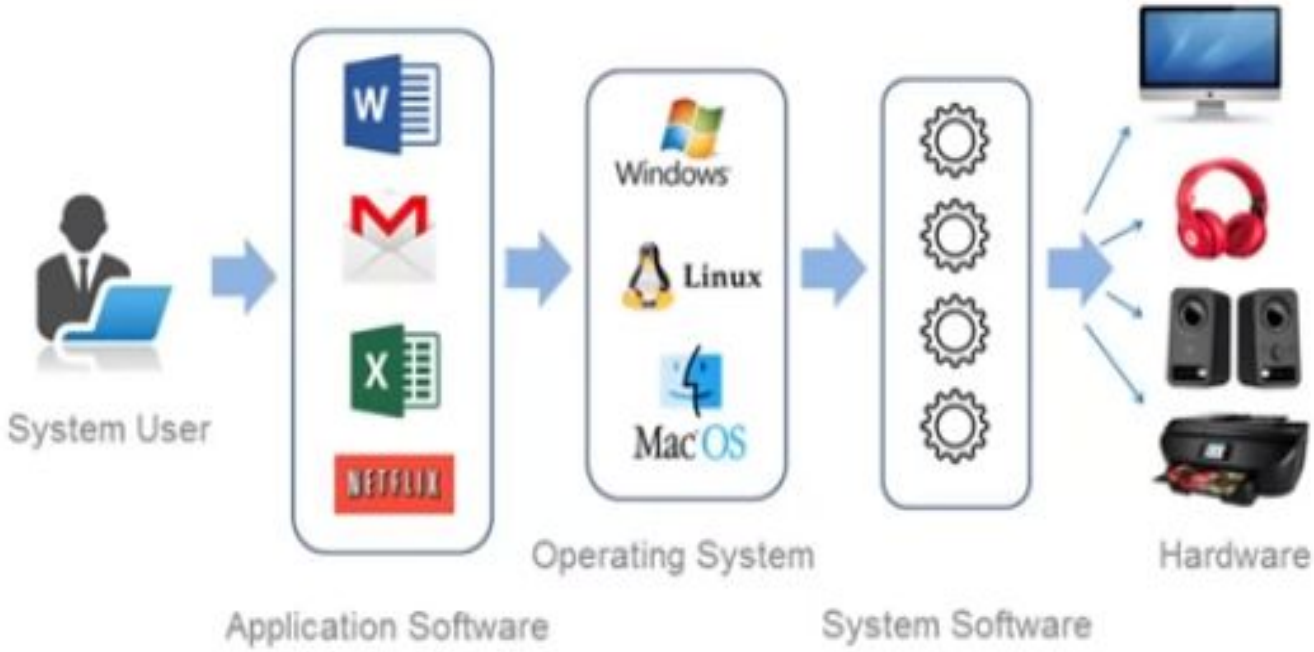
2.Operating system – controls and coordinates the use of the hardware among the various application programs for the various users

3.Applications programs – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).

4.Users (people, machines, other computers).

General purpose application software




System User
Windows
Linux
Mac OS
NETFLIX
Application Software
Operating System
System Software
Hardware

A Simple Program

What is the output of the program?

```
#include <stdio.h>

int main(){
  char str[] = "Hello World\n";
  printf("%s", str);
}
```

Displaying on the screen