

Operating Systems



Module Introduction

- CCAT Exam
 - Operating Systems: 5 Questions
 - Computer Fundamentals: 5 Questions
 - Networking: 5 Questions

} Section B

→ Section A
- Quiz
 - Operating Systems: 4 Quizzes (10 marks) ✓
 - Computer Fundamentals: 1 Quiz (10 marks)
 - Operating Systems: Module end quiz (20 marks)
- GitLab Repository
 - OS Galvin Slides ✓
 - Practice MCQ ✓
 - Notes/Diagrams ✓
 - Reference Book:
Operating System Concepts - Galvin.



Operating System Concepts

- **Introduction**

- Introduction to Operating System, What is OS, Booting the System

- **System Architecture Design of OS**

- System Calls, Dual Mode Operation: System mode and Kernel mode

- **Process Management**

- What is Process & PCB?
- States of the process
- CPU scheduling & CPU scheduling algorithms
- Inter Process Communication: Shared Memory Model & Message Passing Model
- Process Synchronization/Co-ordination
- Deadlocks & deadlock handling methods



Operating System Concepts

* Memory Management

- Swapping
- Memory Allocation Methods
- Segmentation
- Paging
- Virtual Memory Management



Operating System Concepts

File Management

- What is file?
- What is filesystem & filesystem structure?
- Disk space allocation methods
- Disk scheduling algorithms



Operating System Concepts

Introduction:

- Why there is need of an OS?
- What is an OS?
- History of OS- Multi-threading, multiprocessing etc
- Functions of an OS
- What is Process?
- States of Process



End User

Editor

IDE

Browser

media player

appl.
Software

Operating System

Prog.

- ① Interface
- ② Resource Manger
- ③ Control Program
- ④ CD/DVD
↳ Core OS + appl prog + Utilities
- ⑤ Kernel = Core OS.

CPU

RAM

HDD

Keyboard

Monitor

Hardware
Computer.

Program. \rightarrow set of instⁿ given computer system.
(executable file).

Software \rightarrow Set of program.

Utility \rightarrow Prog. associated with OS.

IDE \rightarrow Integrated Development Environment.

\uparrow Compiler pre-processor linker -	\downarrow C C++ ,	\downarrow gcc .
---	----------------------------	-----------------------

Operating System Concepts

Q. Why there is a need of an OS?

- Computer is a machine/hardware does different tasks efficiently & accurately.
- Basic functions of computer:
 1. Data Storage
 2. Data Processing
 3. Data Movement
 4. Control
- As any user cannot communicates/interacts directly with computer hardware to do different tasks, and hence there is need of some interface between user and hardware.



Operating System Concepts

Q. What is an Operating System?

- An OS is a **system software** (i.e. collection of system programs) which acts as an interface between user and hardware.
- An OS also acts as an interface between programs and hardware.
- An OS allocates resources like main memory, CPU time, i/o devices access etc... to all running programs, hence it is also called as a **resource allocator**.
- An OS controls an execution of all programs and it also controls hardware devices which are connected to the computer system and hence it is also called as a **control program**.
- An OS manages limited available resources among all running programs, hence it is also called as a **resource manager**.



Operating System Concepts

- From End User: An OS is a software (i.e. collection of programs) comes either in CD/DVD, has following main components:889IKM

1. Kernel: It is a core program/part of an OS which runs continuously into the main memory does basic minimal functionalities of it.

e.g. Linux: `vmlinuz`, Windows: `ntoskrnl.exe`

2. Utility Softwares: e.g. disk manager, Control panel, windows firewall, anti-virus software etc...

3. Application Softwares: e.g. google chrome, shell, notepad, msoffice etc...

CD/DVD → Core OS + appl'n software + Utilities .
 ↑
 Kernel .



Operating System Concepts

Q. What is a Software?

Software is a collection of programs.

Q. What is a Program?

- Program is a set of instructions written in any programming language (either low level or high level programming language) given to the machine to do specific task.

- Three types of programs are there:

1. "user programs": programs defined by the programmer user/developers

e.g. main.c, hello.java, addition.cpp etc....

2. "application programs": programs which comes with an OS/can be installed later

e.g. MS Office, Notepad, Compiler, IDE's, Google Chrome, Mozilla Firefox, Calculator, Games etc....

3. "System Programs": programs which are inbuilt in an OS/part of an OS.

e.g. Kernel, Loader, Scheduler, Memory Manager etc...



Operating System Concepts

Q. What is an IDE (Integrated Development Environment)?

- It is an application software i.e. collection of tools/programs like **source code editor**, **pre-processor**, **compiler**, **linker**, **debugger** etc... required for **faster software development**.

e.g. VS code editor, MS Visual Studio, NetBeans, Android Studio, Turbo C etc....

1. **"Editor"**: it is an application program used to write a source code.

e.g. notepad, vi editor, gedit etc...

2. **"Pre-processor"**: it is an application program gets executes before compilation and does two jobs - it executes all pre-processor directives and removes all comments from the source code.

e.g. cpp

3. **"Compiler"**: it is an application program which convert high level programming language code into low level programming language code i.e. human understandable language code into the machine understandable language code.

e.g. gcc, tc, visual c etc...



Operating System Concepts

4. **"Assembler"**: it is an application program which converts assembly language code into machine language code/object code.

e.g. masm, tasm etc...

- Program written in any programming language is called as a **"source code"**.

5. **"Linker"**: it is an application program which links object file/s in a program with precompiled object modules of library functions exists in a lib folder and creates final single executable file.

e.g. ld. link editor in Linux.



Operating System Concepts

Functions of an OS:

Basic minimal functionalities/Kernel functionalities:

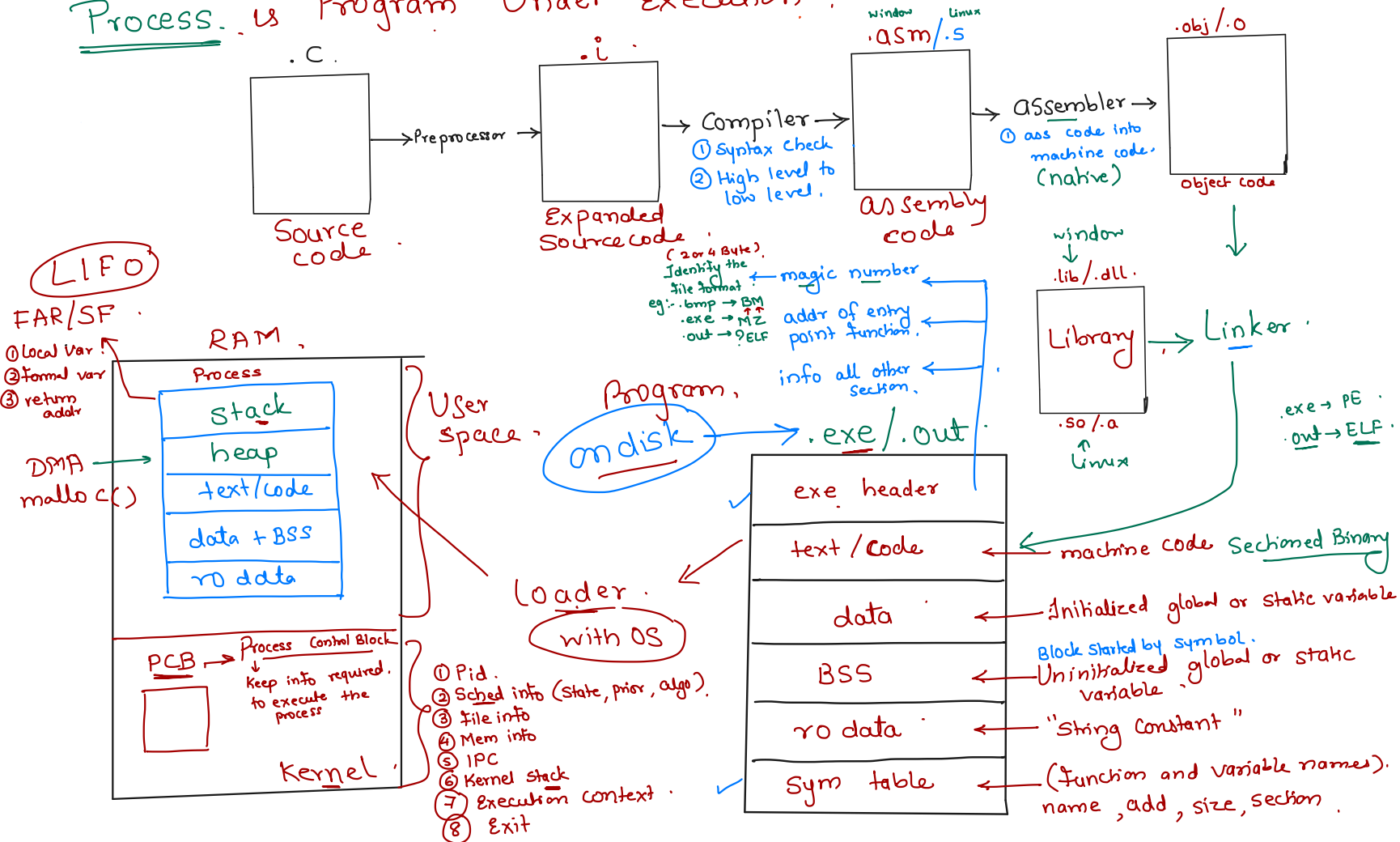
- ✓ 1. Process Management ✓
 - ✓ 2. Memory Management ✓
 - ✓ 3. Hardware Abstraction
 - ✓ 4. CPU Scheduling ✓
 - ✓ 5. File & IO Management ✓
- ✓

Extra utility functionalities/optional:

- 6. Protection & Security
 - 7. User Interfacing
 - 8. Networking
- ✗



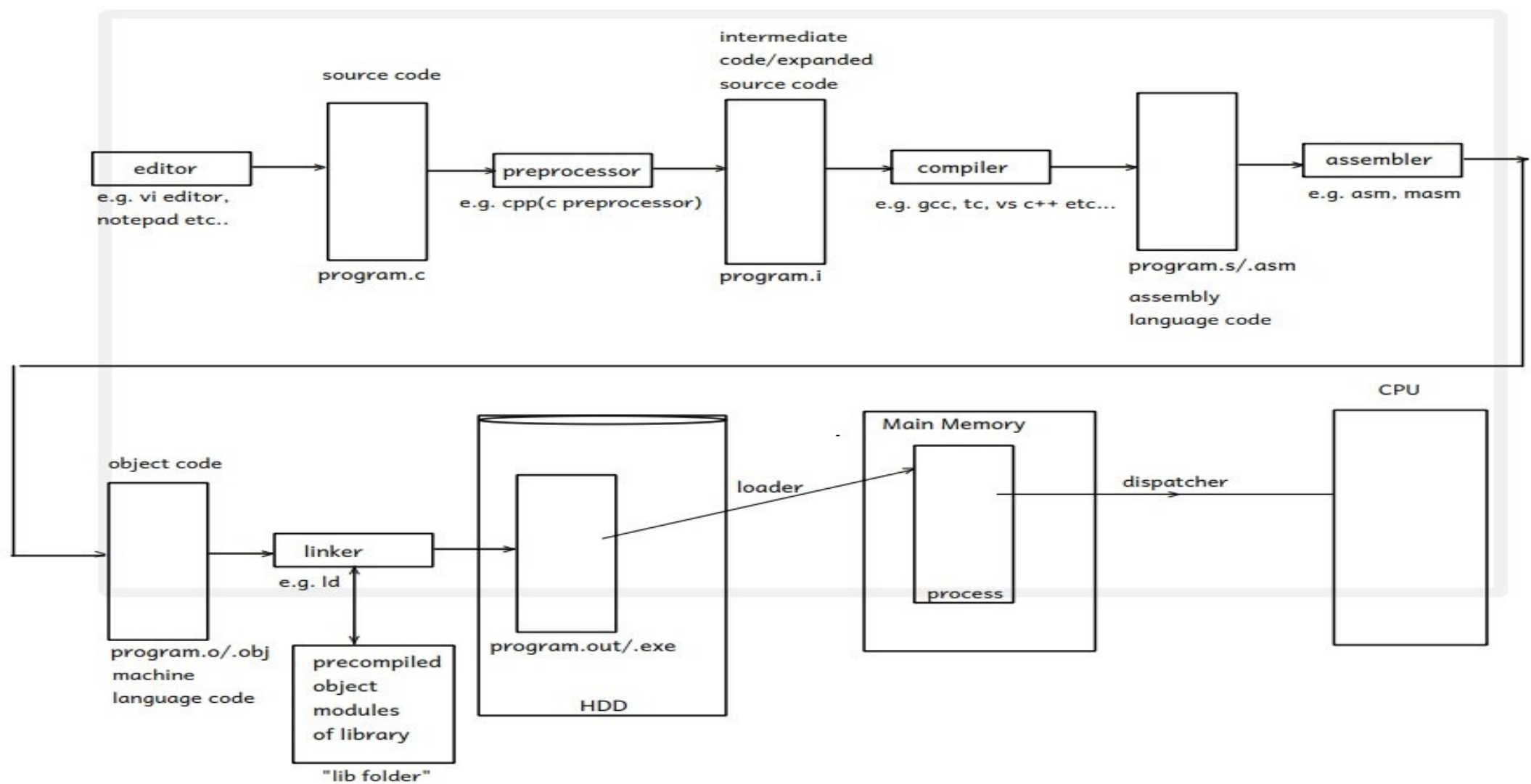
Process is Program Under Execution.




```
int main( )  
{  
    _____  
    _____  
    _____  
    return 0 ;  
}
```

0 → Successful .
any . → unsuccessful .
value .
Exit() .

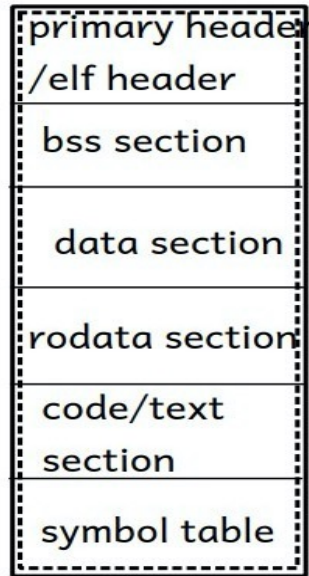
Operating System Concepts



Operating System Concepts

Structure of an executable file
ELF file format in Linux

RAM → Process



program.out

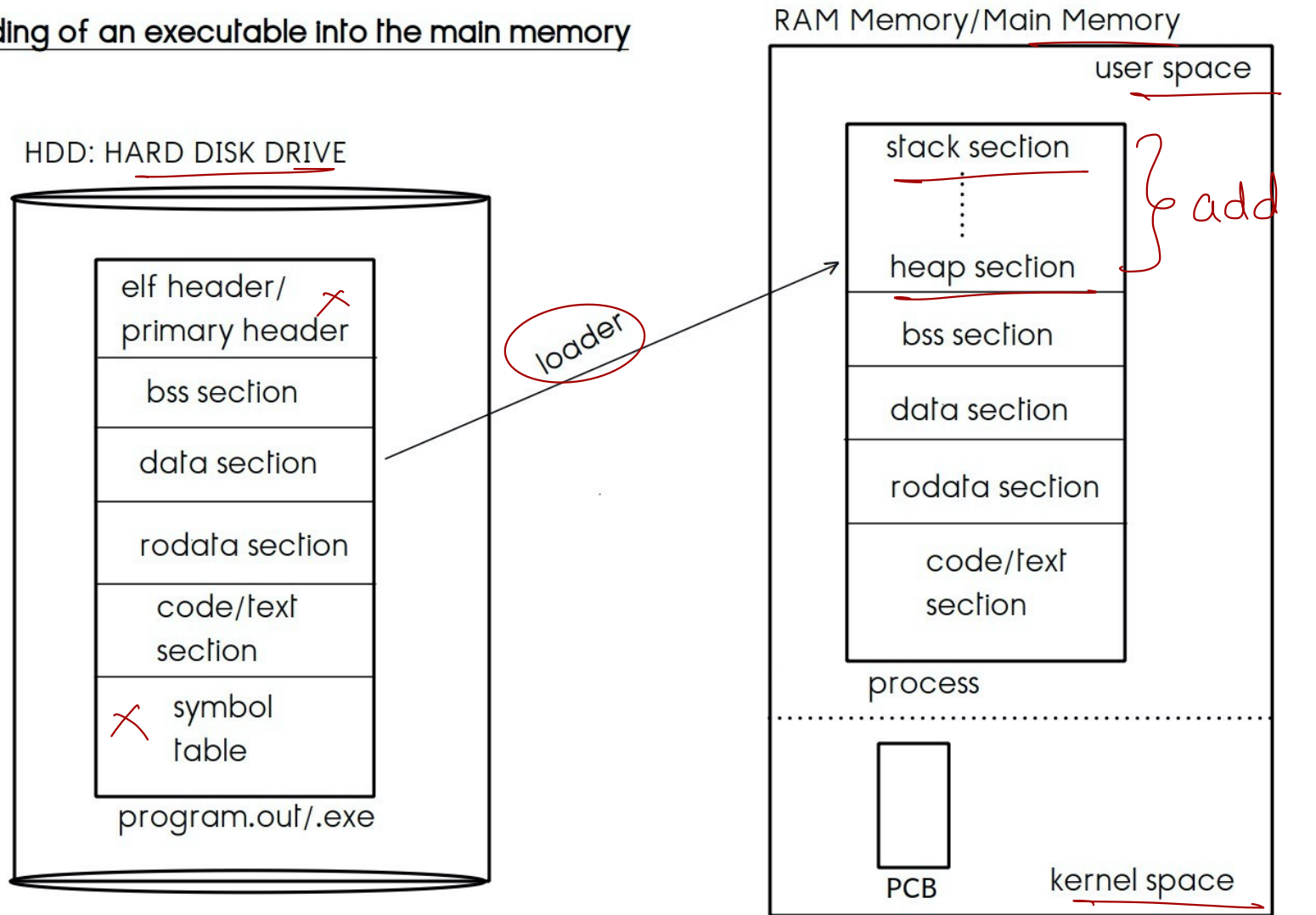
Hard Disk Drive

- 1. primary header/exe header:** it contains information which is required to start an execution of the program.
e.g. - addr of an entry point function --> main() function
- **magic number**: it is a constant number generated by the compiler which is file format specific.
 - magic number in Linux starts with ELF in its eq **hexadecimal format**.
 - info about remaining sections.
- 2. bss(block started by symbol) section:** it contains uninitialized global & static vars
- 3. data section:** it contains initialized global & static vars
- 4. rodata (readonly data) section:** it contains string literals and constants.
- 5. code/text section:** it contains executable instructions
- 6. symbol table:** it contains info about functions and its vars in a tabular format.



Operating System Concepts

Loading of an executable into the main memory



Operating System Concepts

Process

- Program under execution.
- Process execute in RAM.
- Process control block contains information about the process (required for the execution of process).
 - Process id
 - Exit status
 - Scheduling information (State, Priority, Sched algorithm, Time, ...)
 - Memory information (Base & Limit, Segment table, or Page table)
 - File information (Open files, Current directory, ...)
 - IPC information (Signals, ...)
 - Execution context
 - Kernel stack
- PCB is also called as process descriptor (PD), uarea (UNIX), or task_struct (Linux).



Q. What is a Process?

User view:

- A program in execution is called a process.
- Running program is called a process.
- When a program gets loaded into the main memory it is referred to as a process.
- Running an instance of a program is referred to as a process.

System view:

- The process is a file loaded into the main memory which has got bss section, rodata section, code section, and two new sections added for the process:
- **stack section**: contains function activation records of called functions.
- **heap section**: dynamically allocated memory



Operating System Concepts

- file format of an executable file in Windows is PE (Portable Executable), whereas file format of an executable file in Linux is **ELF (Executable & Linkable Format)**.
- file format is a specific way to store data & instructions of a program inside an executable file, and it is different in diff OS.
- ELF file format divides an executable file logically into sections and inside each section specific contents can be kept in an organized manner:
 1. elf header
 2. bss section (block started by symbol)
 3. data section
 4. rodata (read only data)section
 5. code/text section
 6. symbol table



Memory Layout of Program and Process

Program Consist of

exe header/primary header

Block started by symbol (bss) section
(un initialized static / global variables)

Data section (initialized static / global
variables)

Rodata Section(Constant/literals)

code/text section (contains
executable instructions)

Symbol Table

Process Consist of

Skipped

Block started by symbol (bss) section
(un initialized static / global variables)

Data section (initialized static / global
variables)

Rodata Section(Constant/literals)

code/text section (contains executable
instructions)

Skipped

Stack Section

Heap Section



Interaction with an OS : Two Types of Interface (CUI and GUI)

1. CUI/CLI : Command User Interface/Command Line Interface

- By using this kind of interface user can interact with an OS by means of entering commands onto the terminal/command line in a text format.

e.g. In Windows name of the program which provides CUI => cmd.exe command prompt

In Linux name of an application program which provides CUI => shell/terminal

In MSDOS name of the program which provides CUI => command.com (Microsoft Disk Operating System).

2. GUI : Graphical User Interface

- by using this kind of interface user can interact with an OS by means of making an event like click on buttons, left click/right click/double click, menu bar, menu list etc.....

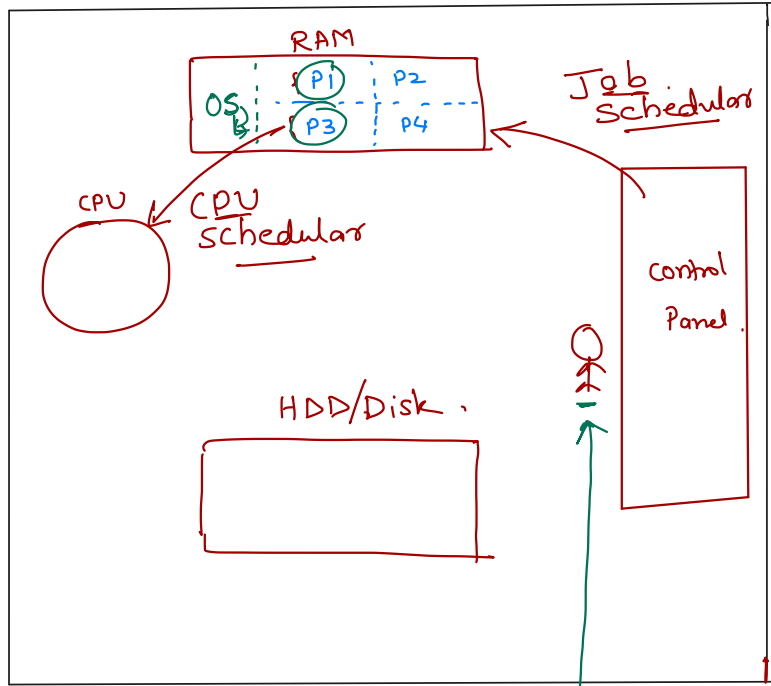
- Windows = User-friendly GUI.

e.g. In Windows name of an application program which provides GUI => explorer.exe

In Linux name of an application program which provides GUI => GNOME/KDE (GNU Network Object Model Environment / Common Desktop Environment).



Old Computers \rightarrow Main-frame computers.



OS history / Evolution.

- ① Resident Monitor.
- ② Batch System.
- ③ Multi-Programming
 - \hookrightarrow multiple prog load RAM.
 - \hookrightarrow better utilization of CPU.
 - \hookrightarrow mixed prog.

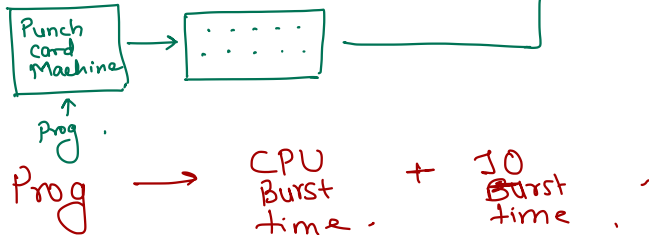
CPU + IO.

- ④ time sharing / multi-tasking
 - \rightarrow Sharing CPU time among multiple tasks present in mem & ready for execution.

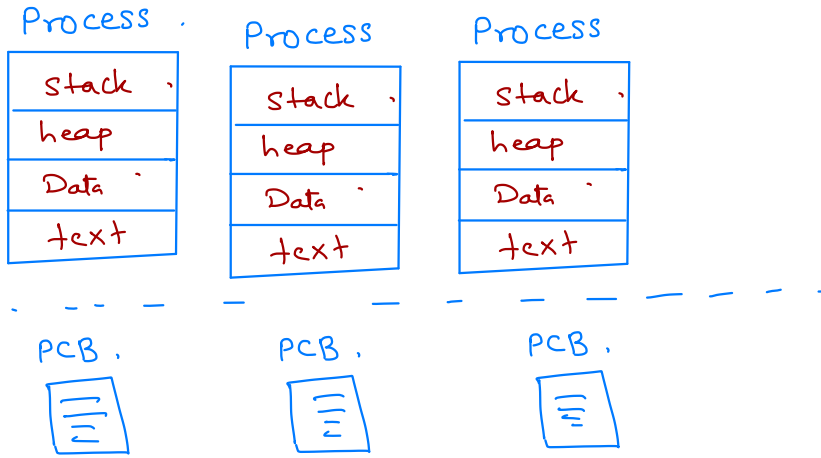
Response $< 1\text{sec}$.

multi-tasking

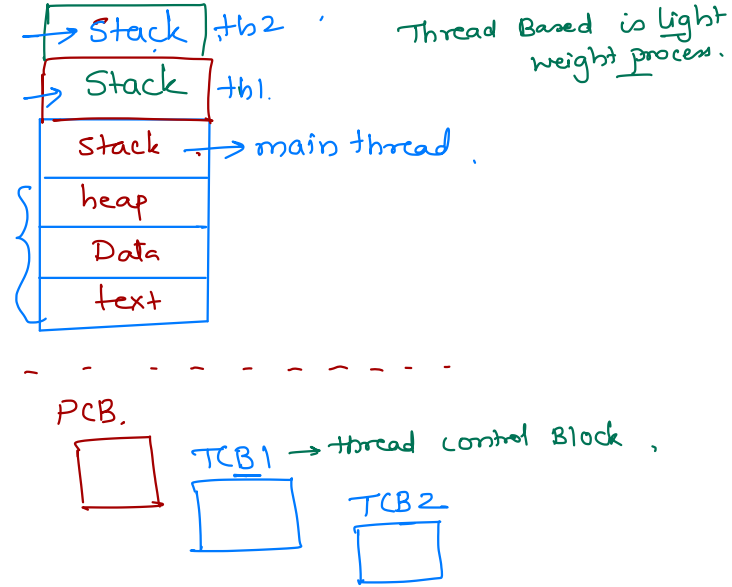
process Based thread Based



Process Based.



Thread Based.



Operating System Concepts

▪ History of Operating System

1. Resident monitor

2. Batch System

- The batch/group of similar programs is loaded in the computer, from which OS loads one program in the memory and execute it. The programs are executed one after another.
- In this case, if any process is performing IO, CPU will wait for that process and hence not utilized efficiently.

3. Multi-programming

- Better utilization of CPU
- Loading multiple Programs in memory
- Mixed program(CPU bound + IO bound)

4. Time-sharing/Multitasking

- Sharing CPU time among multiple process/task present in main memory and ready for execution
- Any process should have response time should be less than 1sec
- Multi-tasking is divided into two types
 - Process based multitasking
 - Thread based multitasking

Thread is a light weight process

- When new thread is created a new stack and new TCB is created.
- Thread Share text, data, heap sections with the parent process



Operating System Concepts

- Process and thread
 - Process is a container for resources.
 - Thread is unit of execution/scheduler.
 - For each process one thread is created by default it is called as main thread

5. Multi-user system

Multiple users runs multiple programs concurrently.

6. Multi-processor/Mutli-core system

System can run on a machine in which more than one CPU's are connected in a closed circuit.

Multiprocessing Advantage is it increased throughput (amount of work done in unit time)

- There are two types of multiprocessor systems:
 - Asymmetric Multi-processing Symmetric Multi-processing
- **Asymmetric Multi-processing**
 - OS treats one of the processor as master processor and schedule task for it. The task is in turn divided into smaller tasks and get them done from other processors.
- **Symmetric Multi-processing**
 - OS considers all processors at same level and schedule tasks on each processor individually. All modern desktop systems are SMP.

