HISTORY OF LINUX UNIX

- **Linux** began in 1991 as a personal project by Finnish student Linus Torvalds: to create a new free operating system kernel. The resulting Linux kernel has been marked by constant growth throughout its history. Since the initial release of its source code in 1991, it has grown from a small number of C files under a license prohibiting commercial distribution to the 4.15 version in 2018 with more than 23.3 million lines of source code, not counting comments,under the GNU General Public License v2
- Linus Torvalds

- After AT&T had dropped out of the Multics (multiplexed information and computing services) project, the Unix operating system was conceived and implemented by Ken

Thompson and Dennis Ritchie (both of AT&T Bell Laboratories) in 1969 and first released in 1970. Later they rewrote it in a new programming language, C, to make it portable. The availability and portability of Unix caused it to be widely adopted, copied and modified by academic institutions and businesses.

- Ken Thompson (left) and Dennis Ritchie (right)

- In 1977, the Berkeley Software Distribution (BSD) was developed by the Computer Systems Research Group (CSRG) from UC Berkeley , based on the 6th edition of Unix from AT&T. Since BSD contained Unix code that AT&T owned, AT&T filed a lawsuit (*USL v. BSDi*) in the early 1990s against the University of California. This strongly limited the development and adoption of BSD.
- In 1983, Richard Stallman started the GNU project
- with the goal of creating a free UNIX-like operating system . As part of this work, he wrote the GNU General Public License (GPL). By the early 1990s, there was almost enough available software to create a full operating system.

However, the GNU kernel, called Hurd , failed to attract enough development effort, leaving GNU incomplete

Richard Stallman

n 1985, Intel released the 80386

- the first x86 microprocessor with a 32-bit instruction set and a memory management unit with paging.
- In 1986, Maurice J. Bach, of AT&T Bell Labs, published *The Design of the UNIX Operating System* This definitive description principally covered the System V Release 2 kernel, with some new features from Release 3 and BSD.

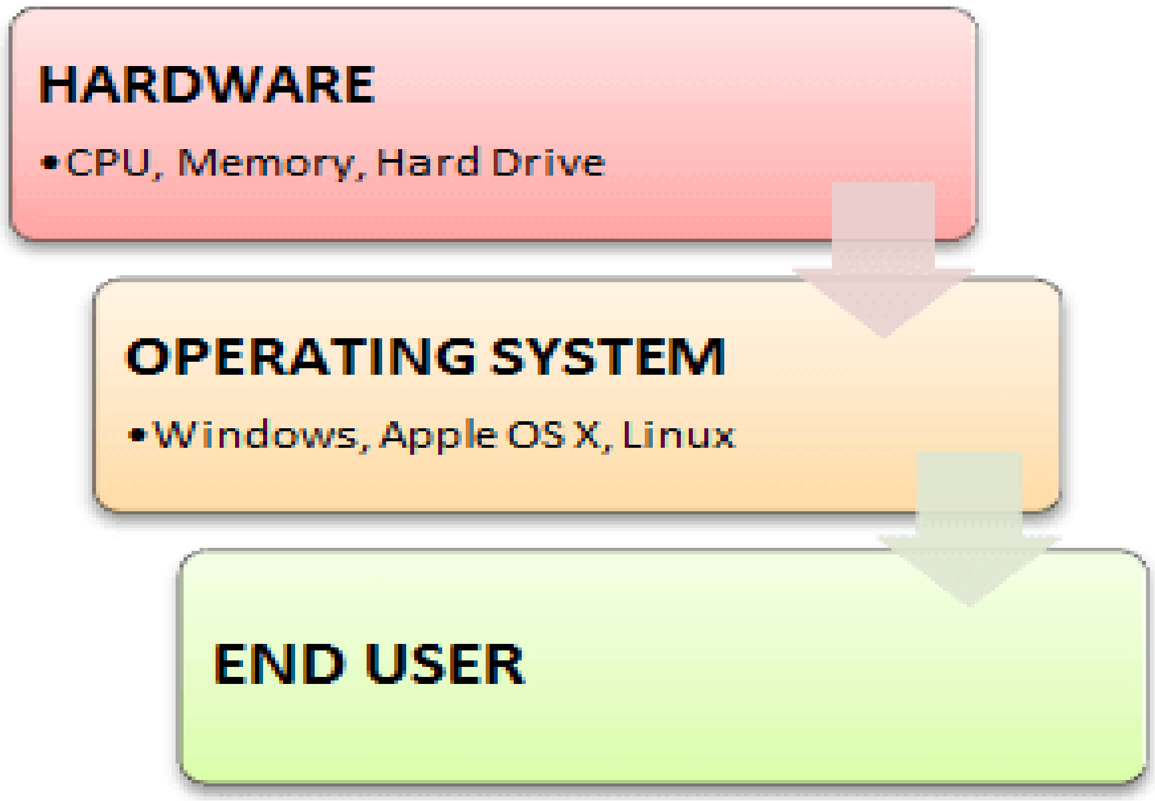
- In 1987, MINIX , a Unix-like system intended for academic use, was released by Andrew S. Tanenbaum to exemplify the principles conveyed in his textbook, *Operating Systems: Design and Implementation*. While source code for the system was available, modification and redistribution were restricted. In addition, MINIX's 16-bit design was not well adapted to the 32-bit features of the increasingly cheap and popular Intel 386 architecture for personal computers.

  Andrew S. Tanenbaum

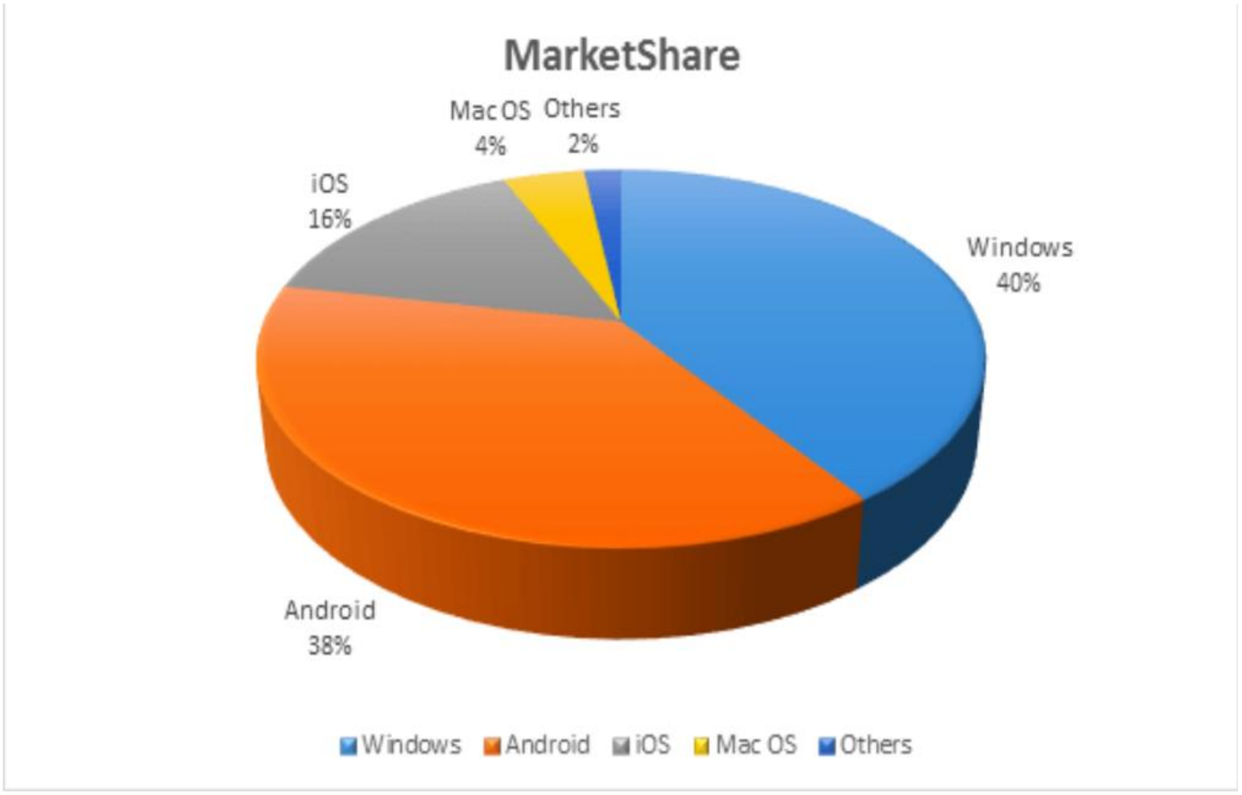
- Torvalds announced in 1996 that there would be a mascot for Linux, a penguin. This was because when they were about to select the mascot, Torvalds mentioned he was bitten by a little penguin (*Eudyptula minor*) on a visit to the National Zoo & Aquarium in Canberra, Australia.

What is operating system?

- An operating system is a software which acts as an interface between the end user and computer hardware. Every computer must have at least one OS to run other programs. An application like Chrome, MS Word, Games, etc needs some environment in which it will run and perform its task.
- The OS helps you to communicate with the computer without knowing how to speak the computer's language. It is **not** possible for the user to use any computer or mobile device without having an operating system.



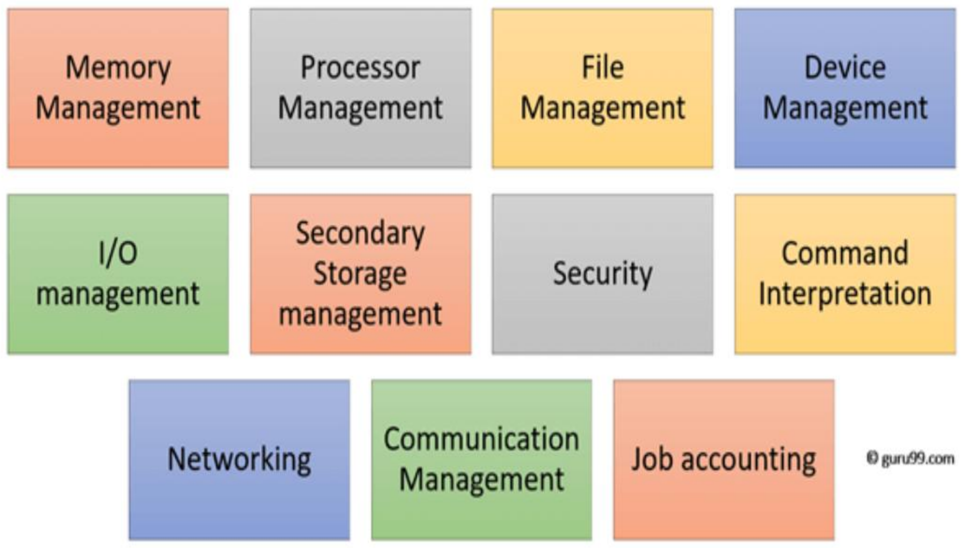
Examples of Operating System with Market Share



list of Operating Systems with the latest MarketShare

1. **OS Name** **Share**
2. Windows 40.34
3. Android 37.95
4. iOS 15.44
5. Mac OS 4.34
6. Linux 0.95
7. Chrome OS 0.14
8. Windows Phone OS 0.06

Functions of an Operating System



Function of an Operating System

1. Process management:-

- Process management helps OS to create and delete processes. It also provides mechanisms for synchronization and communication among processes.

2. Memory management:- Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.

3. **File management**:- It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.

4. Device Management: Device management keeps tracks of all devices. This module also responsible for this task is known as the I/O controller. It also performs the task of allocation and de-allocation of the devices.

5. I/O System Management: One of the main objects of any OS is to hide the peculiarities of that hardware devices from the user.

6. Secondary-Storage Management: Systems have several levels of storage which includes primary storage, secondary storage, and cache storage. Instructions and data must be stored in primary storage or cache so that a running program can reference it.

7. Security:- Security module protects the data and information of a computer system against malware threat and authorized access.

8. Command interpretation: This module is interpreting commands given by the and acting system resources to process that commands.

9. Networking: A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through the network.

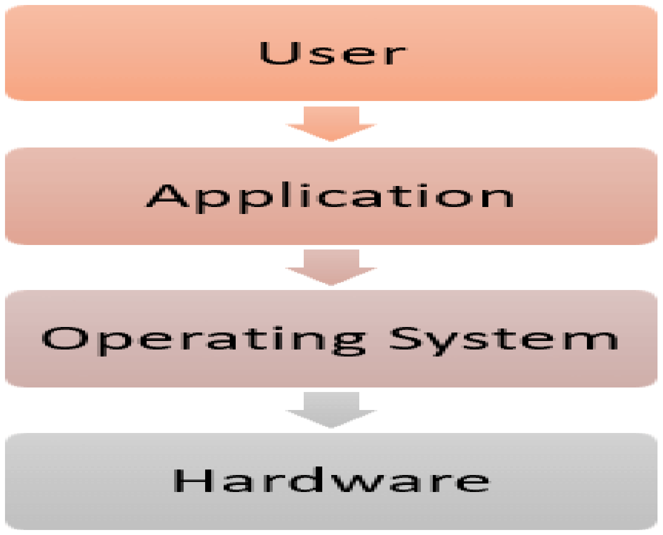
10. Job accounting: Keeping track of time & resource used by various job and users.

11. Communication management: Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

Features of Operating System

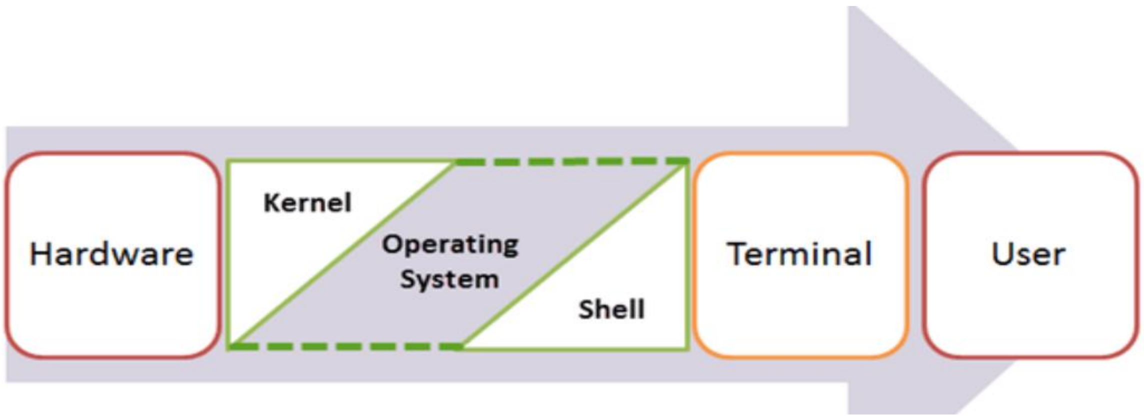
- Protected and supervisor mode
- Allows disk access and file systems Device drivers Networking Security
- Program Execution
- Memory management Virtual Memory Multitasking
- Handling I/O operations
- Manipulation of the file system

- Error Detection and handling
- Resource allocation
- Information and Resource Protection



What is a Kernel?

- The kernel is the central component of a computer operating systems. The only job performed by the kernel is to the manage the communication between the software and the hardware. A Kernel is at the nucleus of a computer. It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.



Types of Operating system

1. **Batch Operating System**
2. **Multitasking/Time Sharing OS**
3. **Multiprocessing OS**
4. **Real Time OS**
5. **Distributed OS**
6. **Network OS**
7. **Mobile OS**

1. Batch Operating System

- Some computer processes are very lengthy and time-consuming. To speed the same process, a job with a similar type of needs are batched together and run as a group.
- The user of a batch operating system never directly interacts with the computer. In this type of OS, every user prepares his or her job on an offline device like a punch card and submit it to the computer operator.

2. Multi-Tasking/Time-sharing Operating systems

Time-sharing operating system enables people located at a different terminal(shell) to use a single computer system at the same time. The processor time (CPU) which is shared among multiple users is termed as time sharing.

3. Multiprocessing Operating System

It is the type of operating system that makes use of more than one CPU. The term multiprocessing is used to describe interconnected computer configuration or computers with two or more CPUs that have the ability to simultaneously execute several programs.
In such a system, instruction from different independent programs can be processed at the same instant of time by different CPU. More than one program in main memory at a time.

4. Real time OS

- A real time operating system time interval to process and respond to inputs is very small. Examples: Military Software Systems, Space Software Systems.

5. Distributed Operating System

- Distributed systems use many processors located in different machines to provide very fast computation to its users.

6. Network Operating System
Network Operating System runs on a server. It provides the capability to serve to manage data, user, groups, security, application, and other networking functions.

7. Mobile OS

Mobile operating systems are those OS which is especially that are designed to power smartphones, tablets, and wearables devices.

Some most famous mobile operating systems are Android and iOS, but others include BlackBerry, Web, and watchOS.

introduction of os process

A process is an instance of a computer program that is being executed or in simple words process is a program in execution

All instruction one by one read from the program and specified operations are performed. A process is also referred as task or job.

- A process is dynamic entity. It changes its states during its time duration of execution. It takes birth (creation), lives (execution) and dies (termination).
- A process has limited life span. Its starts when program is executed. It performs various operations specified by instruction given in program and terminates once all instruction get executed.
- A process contains various resources like memory space, disk, printer, scanner etc as per its requirements

  A process contains memory space which is called its Address Space. All processes of same program have same address space.

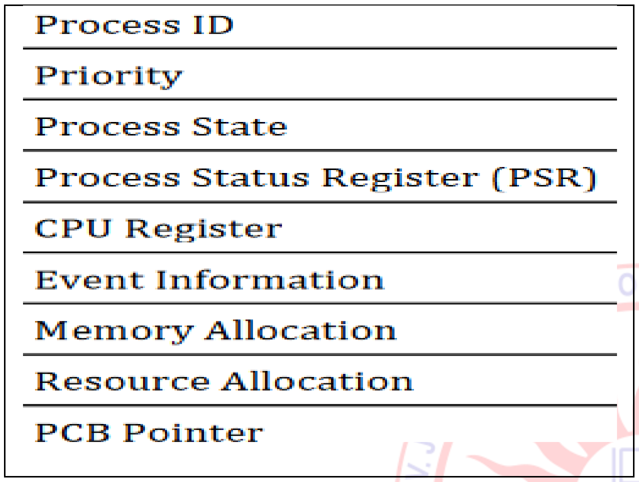
- A process consists of Process ID, Code, Data, Register and Program Counter (PC) value
- Process ID is the unique ID assigned to the process by the OS
- Code is the program called. i.e. instruction
- Data is the data which is used during execution
- Register value is the value in machine register
- PC value is the address in the program counter

Process Control Block (PCB)

- OS maintains a table called Process Table to store all the information about each process.

Process table contains one entry per process called Process Control Block

The information of a process stored in PCB are shown in following figure



Process ID
Priority
Process State
Process Status Register (PSR)
CPU Register
Event Information
Memory Allocation
Resource Allocation
PCB Pointer

- Process ID contains process identification number given by an OS
- Priority field contains the priority of process
- In process state, currently process is in which state is stored
- Process Status Register (PSR) contains the content of process status register, means the status of process

Process States (Process Life Cycle)

- A process is a dynamic entity. It changes its state during time duration of its execution.A process state indicates current activity of process. Each process may be in one of the following states during its life span.

  (1)New

When a process is first created, it occupies 'New' state.

(2)Ready

A process which is in 'Ready' state has been loaded in mainmemory. In this state, process awaits for execution on CPU. It is possible that more than one process may be in 'Ready' state at a time.

3)Running

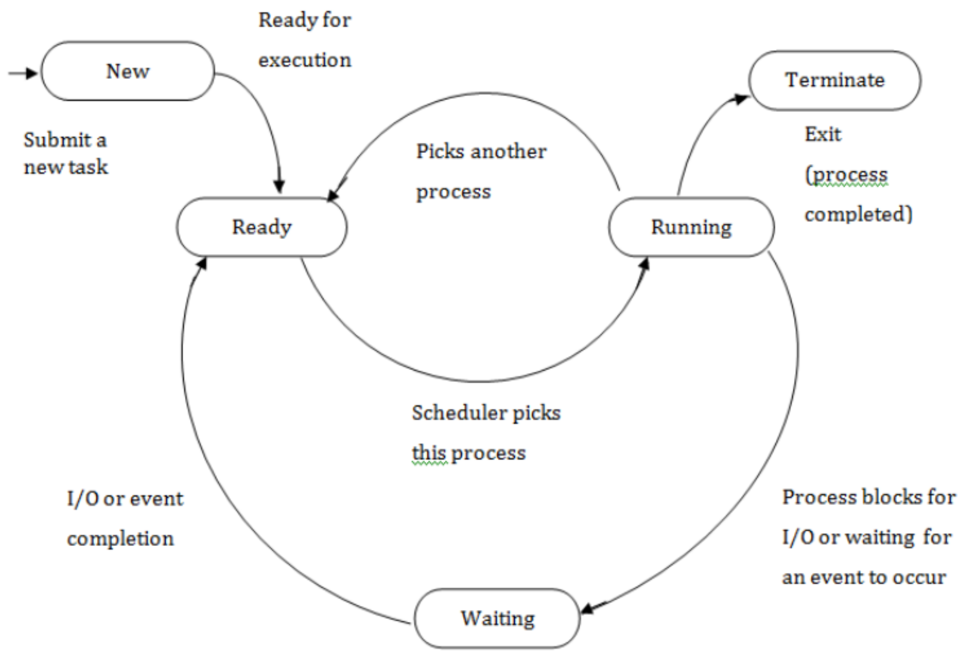
A process which is in running state is currently executing on a CPU. In a one processor system, one and only process can be in this state at a time.

(4)Waiting / Blocked

A process which is in waiting state is waiting for some I/O completion or some event to occur. E.g. consider some process requires printer to print documents but printer is being used by other process. So, first process has to wait and it will be in this state till the printer becomes free.

(5)Terminate

When process terminates it acquires this state. A process can terminate by completing its execution or it can be killed explicitly.



Process Scheduling

Scheduling is the activity of determining which service request (any process) should be handled by a system. Here, it is the task to arrange the process in order for execution. There are two separate modules for this task.

Process Scheduler

Process scheduler schedules the process. All the time of scheduling the process, scheduler use information from the PCB. Process can be scheduled based on various scheduling policies.

Process Dispatcher

Process dispatcher allocates the CPU to the new scheduled process. Process dispatcher also performs context saves before allocating CPU to new process. After context saves new data are loaded and new process starts its execution.

Performance of scheduling method is determined by following performance criteria

Turnaround Time

Time required to complete execution of a process is called turnaround time. It specifies that how long it takes to complete a process execution. Turnaround Time = Process Finish Time – Process Arrival Time.

Waiting Time

It is total time duration spent by a process waiting in Ready queue. Scheduling algorithm affects the time that a process spends waiting in the Ready state. Waiting Time = Turnaround Time

Response Time

It is time between issuing a command/ request and getting output/ result. It is the time between the process arrival and it gets response from the system

– A

The different policies of the process scheduling are given below:

1. FCFS (First Come First Served)
2. SJN (Shortest Job Next)

3. Round Robin
4. Priority based Preemptive
5. Priority based Non –Preemptive

FCFS (First Come First Served)

Selection Criteria

The process that request first is served first. It means that processes are served in the exact order as they come. A process which first comes has the first turn for execution.

Decision Mode

Non Preemptive-Once a process is selected it runs until it blocks for an I/O or some event or terminate. Execution of the current process will not stop in between for another process.

Implementation

It is implemented by using FIFO (First in First out) queue. When first process enters into the system, it starts its execution. All other processes are appended in a queue. When CPU becomes free, a process from the first position in a queue is selected to run.

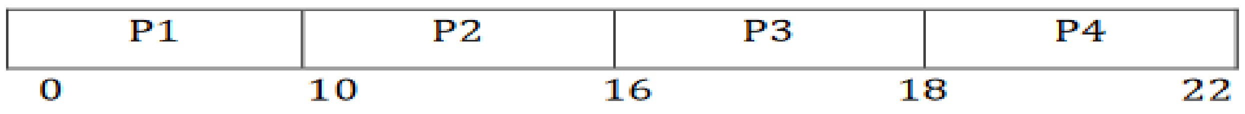
Example

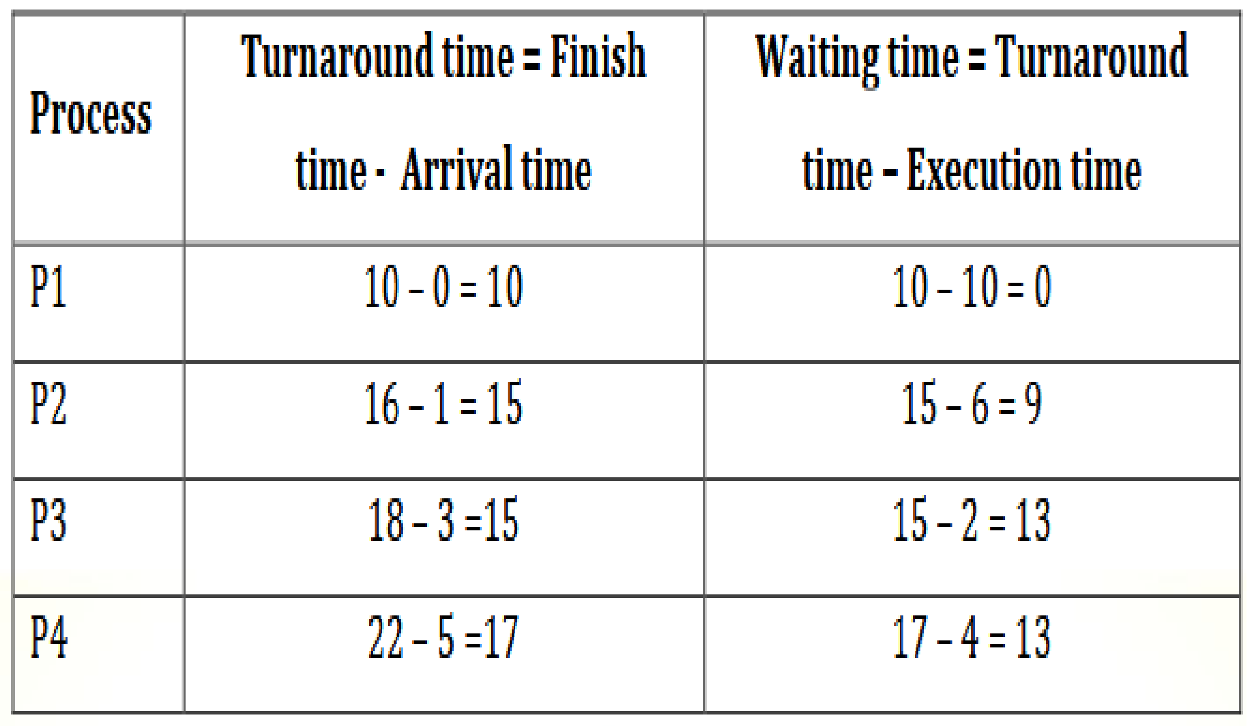
Consider the following set of four processes. Their arrival time and time required to complete the execution are given in following table. Consider time values in milliseconds.

Process	Arrival Time	ExecutionTime
p1	0	10
p2	1	6
p3	3	2
p4	5	4

Solution of example:

- Initially only process P1 is present and it is allowed to run. When P1 completes, all other processes are present. So, next process P2 from Ready queue is selected and allowed to run till it completes. This procedure is repeated till all process completes its execution.





Process	Turnaround time = Finish time - Arrival time	Waiting time = Turnaround time - Execution time
P1	10 - 0 = 10	10 - 10 = 0
P2	16 - 1 = 15	15 - 6 = 9
P3	18 - 3 =15	15 - 2 = 13
P4	22 - 5 =17	17 - 4 = 13

Average turnaround time = (10 + 15 + 15 + 17) / 4

= 14.25 ms

Average waiting time = (0 + 9 + 13 + 13) / 4

= 8.75 ms

Average response time = (0 + 10 + 16 + 18) / 4

= 11 ms

Advantages:

Easy to understand

Easy to implement

Disadvantages:

Not efficient. Average waiting time is too high

All small processes wait for one big process to achieve CPU

SJN(Shortest Job Next)

Selection Criteria

- The process that requires shortest time to complete execution is served first. Means, small processes get the chance first for execution.

Decision Mode

- Non Preemptive-Once a process is selected it runs until it blocks for an I/O or some event or terminate. Execution of the current process will not stop in between for another process.

Implementation

- All processes in a queue are sorted in ascending order based on their required CPU time. When CPU becomes free, a process from the first position in a queue is selected to run.

Example

Consider the same example of FCFS. The arrival time and time required to complete the execution are given in following table. Consider time values in milliseconds.

Process	Arrival Time	Execution Time
P1	0	10
P2	1	6
P3	3	2
p4	5	4

Solution of example:

- Initially only process P1 is present and it is allowed to run. But when P1 completes, all other processes are present. So, process with shortest CPU time is selected and allowed to run till it completes. This procedure is repeated till all process completes its execution

p1	P3	P4	p2

0 10 12 16 2

Process	**Turnaround time = Finish time - Arrival time**	**Waiting time = Turnaround time –Execution time**
P1	10 –0 = 10	10 –10 = 0
P2	22–1 = 21	21–6 = 15
P3	12–3 =9	9–2 = 7
p4	16–5 =11	11–4 = 7

- Average turnaround time = (10 + 21 + 9 + 11) / 4
- = 12.75 ms
- Average waiting time = (0 + 15 + 7 + 7) / 4
- = 7.25 ms
- Average response time = (0 + 10 + 12 + 16) / 4
- = 9.5 ms

Advantages:

- Less waiting time
- Good response for short process

Disadvantages:

- Starvation is possible for long process. Long process may wait forever

Round Robin

This scheduling method primarily used in the time sharing system. CPU time is divided into time slices. Each process is allocated a small time slice while it is running. It does not wait for a process to finish or give up control.

Selection Criteria

Here the selection criteria are same as FCFS. The process that request first is served first. It means that processes are served in the exact order as they come. A process which first comes has the first turn for execution.

Decision Mode

Preemptive-Each selected process is assigned a time interval called 'time quantum' or 'time slice'. Process is allowed to run only for this time interval. Here two things are possible. First, process needs CPU time less than time slice. In this case, process will voluntarily release the CPU.

Second, process needs CPU time longer than time slice. In this case, process will be running at the end of time slice. Now, it will be preempted and moved to the end of the queue. CPU will be allocated to another process

Implementation

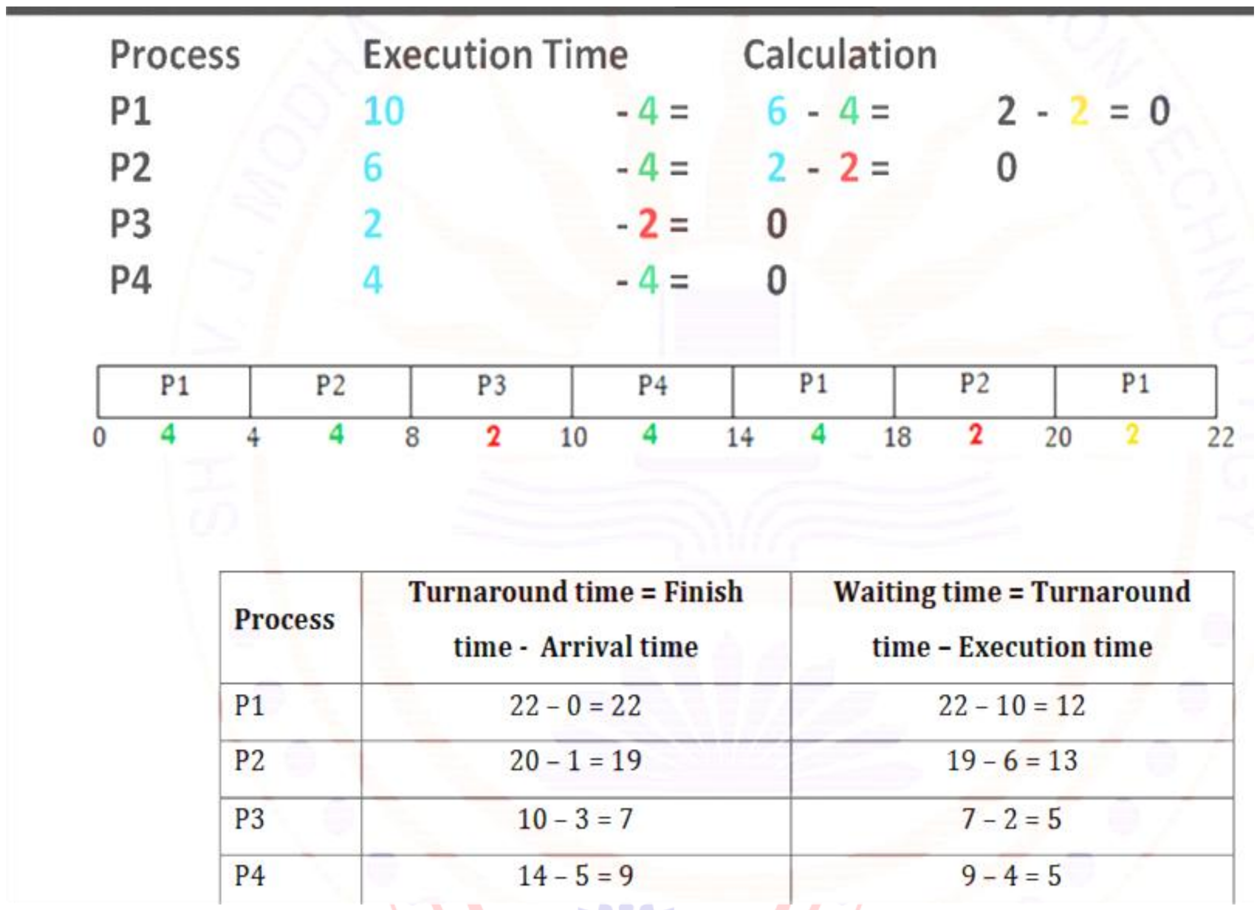
- This can be implemented by using circular FIFO (First in First out) queue. When new process comes or process release CPU or process is preempted, it is moved to the end of the queue. When CPU becomes free, a process from the first position in a queue is selected to run.

  Example

- Consider that time slice is 4ms. Their arrival time and time required to complete the execution are given in following table. Consider time values in milliseconds.

Process	Arrival Time	Execution Time
P1	0	10
P2	1	6
P3	3	2
P4	5	4

- Solution of example:
- Initially only process P1 is present and it is allowed to run. At 4ms process P1 completes its time slice. So, it is preempted and another process P2 is allowed to run. The procedure is repeated till all processes complete their execution.



Process	Execution Time		Calculation	
P1	10	- 4 =	6 - 4 =	2 - 2 = 0
P2	6	- 4 =	2 - 2 =	0
P3	2	- 2 =	0	
P4	4	- 4 =	0	

P1	P2	P3	P4	P1	P2	P1
4	4	2	4	4	2	2

0 4 8 10 14 18 20 22

Process	Turnaround time = Finish time - Arrival time	Waiting time = Turnaround time – Execution time
P1	22 – 0 = 22	22 – 10 = 12
P2	20 – 1 = 19	19 – 6 = 13
P3	10 – 3 = 7	7 – 2 = 5
P4	14 – 5 = 9	9 – 4 = 5

Average turnaround time = (22 + 19 + 7 + 9) / 4

= 14.25 ms

Average waiting time = (12 +13 + 5 + 5) / 4

= 8.75 ms

Average response time = (0 + 4 + 8 + 10) / 4

= 5.5 ms

Advantages:

Simplest method, Most widely used

Disadvantages:

Determination of time slice is too critical. If it is too short it causes frequent context switches and lower CPU efficiency. If it is too long it causes poor response for short processes

Priority Based Preemptive

Selection Criteria

Here the selection criteria are based on priority. The process that has highest priority will get first turn for execution. Priority of all processes is compared with each other to arrange them into a queue.

Decision Mode

Preemptive-When a new process arrive its priority is compared to the current process priority. If the new process has higher priority than the current process then the current process is suspended and the new process started

Implementation

All processes in a queue are sorted based on priority with highest priority process at front end. When CPU becomes free, a process from the first position in a queue is selected to run.

Example

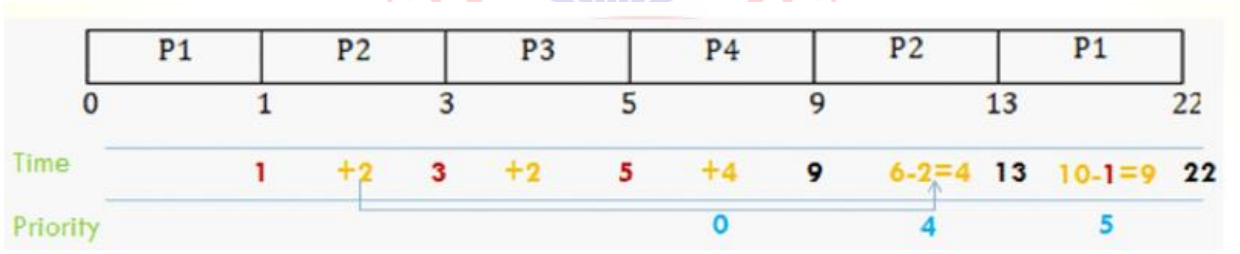
Consider the same example that we have seen earlier. The arrival time and time required to complete the execution and priority for each process are given in following table. Consider time values in milliseconds.

Process	Arival Time	Execution Time	Priority
P1	0	10	5
P2	1	6	4
P3	3	2	2
p4	5	4	0

Solution of example:

Initially only process P1 is present and it is allowed to run. When P2 comes, it has higher priority than P1. So, P1 is preempted and P2 is allowed to run and so on.

Here, small values of priority mean higher priority. So, priorities of processes are: P4 > P3 > P2 > P1. Means P4 has the highest priority and P1 has the lowest priority



Process	Turnaround time = Finish time - Arrival time	Waiting time = Turnaround time –Execution time
P1	22-0=22	22–10 = 12
P2	13-1=12	12–6 = 6
P3	5-3=2	2–2 = 0
p4	9-5=4	4–4 = 0

Average turnaround time = (22 + 12 + 2 + 4) / 4

= 10 ms

Average waiting time = (12 + 6 + 0 + 0) / 4

= 4.5 ms

Average response time = (0 + 1 + 3 + 5) / 4

= 2.25 ms

Advantages:

Priority is considered. So, critical processes can get better response time

Disadvantages:

Starvation is possible for low priority process

Priority Based Non-Preemptive

Selection Criteria

Here the selection criteria are based on priority. The process that has highest priority will get first turn for execution. Priority of all processes is compared with each other to arrange them into a queue.

Decision Mode

Non Preemptive-Once a process is selected it runs until it blocks for an I/O or some event or terminate. Execution of the current process will not stop in between for another process.

Implementation

All processes in a queue are sorted in ascending order based on their priority with highest priority process at front end. When CPU becomes free, a process from the first position in a queue is selected to run.

Example

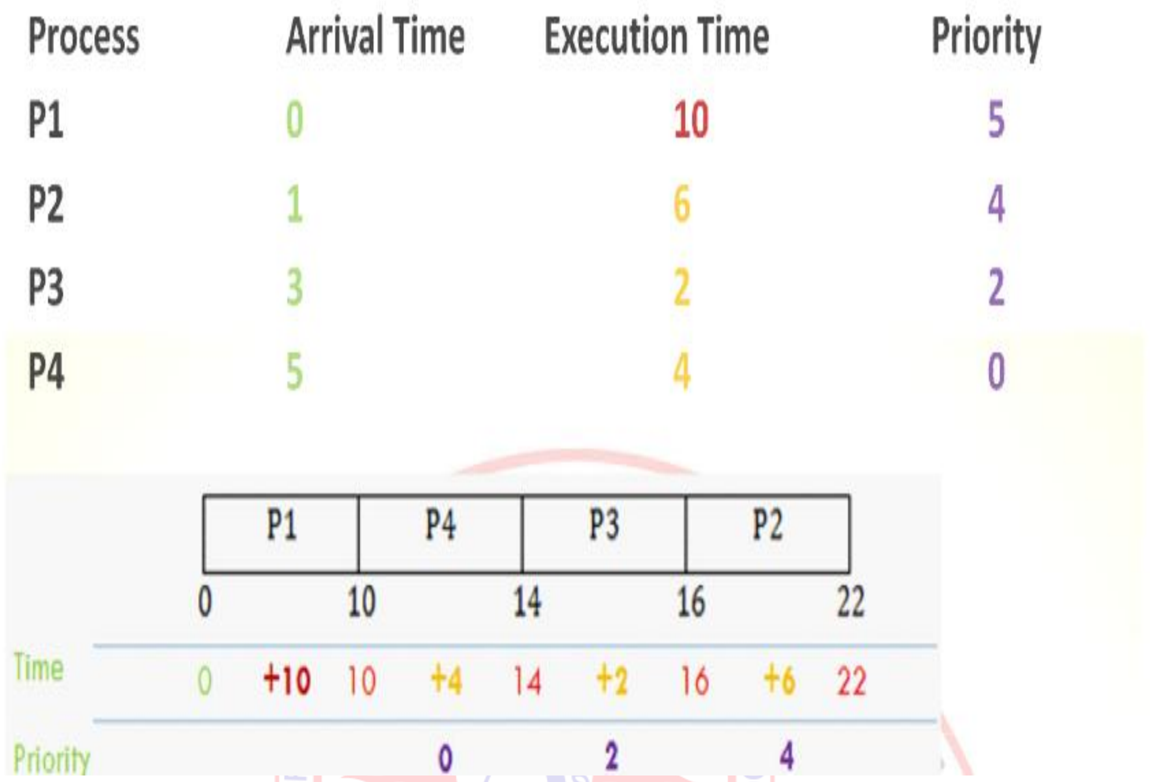
Consider the same example that we have seen earlier. The arrival time and time required to complete the execution and priority for each process are given in following table. Consider time values in milliseconds.

Process	Arival Time	Execution Time	Priority
P1	0	10	5
P2	1	6	4
P3	3	2	2
p4	5	4	0

Solution of example:

Initially only process P1 is present and it is allowed to run. But when P1 completes, all other processes are present. So, process with higher priority is selected and allowed to run till it completes. This procedure is repeated till all process completes its execution.

Here, small values of priority mean higher priority. So, priorities of processes are: P4 > P3 > P2 > P1. Means P4 has the highest priority and P1 has the lowest priority.



Process	Arrival Time	Execution Time	Priority
P1	0	10	5
P2	1	6	4
P3	3	2	2
P4	5	4	0

Process	**Turnaround time = Finish time - Arrival time**	**Waiting time = Turnaround time –Execution time**
P1	10 –0 = 10	10 –10 = 0
P2	22–1 = 21	21–6 = 15
P3	16–3 =13	13–2 = 11
p4	14–5 =9	9–4 = 12

Average turnaround time = (10 + 21 + 13 + 9) / 4

= 13.25 ms

Average waiting time = (0 + 15 + 11 + 5) / 4

= 7.75 ms

Average response time = (0 + 10 + 14 + 16) / 4

= 10 ms

Advantages:

Priority is considered. So, critical processes can get better response time

Disadvantages:

Starvation is possible for low priority process

Preemptive v/s Non-Preemptive

- Preemptive Scheduling
- Here, one process is selected to run then it is allowed to run only for some maximum time duration. After that time duration, another process is selected to execute.
- With the preemptive scheduling, a running process may be replaced by a higher priority process at any time.
- OS removes a low priority process from running state to allow a higher priority process to run.
- When running process is preempted by a higher priority process it remains in Ready state until next turn of execution.

Non-Preemptive Scheduling

- Here, one process is selected to execute then it is allowed to run until it voluntarily gives up the CPU i.e. when it enters into wait state or to get terminates.
- The running process is not forced to release CPU when a higher priority process becomes ready for execution.
- When the running process completes its execution or going for I/O operation than only it release the CPU.

- **MEMORY MANAGEMENT**
- **Physical Memory and Virtual Memory**
- **Memory Allocation**
- **Contiguous Memory Allocation**
- **Noncontiguous Memory Allocation**
- **Virtual Memory Using Paging**
- **Virtual Memory Using Segmentation**

What Is physical memory?

- Physical memory are the RAM chips purchased and placed in a slot on the computer motherboard. The RAM is the first memory used when the computer requires memory usage, such as for loading an application or opening a document.

What is virtual memory?

- Virtual memory is stored on the hard drive. Virtual memory is used when the RAM is filled. Virtual memory is slower than physical memory, so it can decrease the performance of applications.

Main Memory

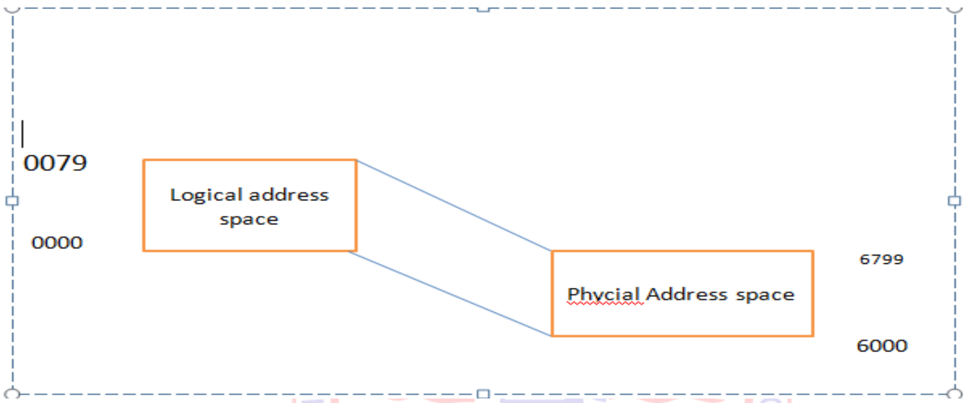
- Main memory is also known as physical memory or primary memory.
- It is internal to the computer system and comes in the form of chips.
- Main memory constructs the middle layer in the memory hierarchy of a computer system.
- It is slower and cheaper compared to cache memory while faster and costly compared to the disk.
- Main memory generally consists of Random Access Memory (RAM)
- Main memory is also known as physical memory or primary memory.
- It is internal to the computer system and comes in the form of chips.
- Main memory constructs the middle layer in the memory hierarchy of a computer system.

- It is slower and cheaper compared to cache memory while faster and costly compared to the disk.
- Main memory generally consists of Random Access Memory (RAM)
- **Memory Allocation**
- Before execution, data have to be in main memory and OS also needs to be stored in main memory. For this purpose, there is a need of memory allocation.
- Memory allocation refers to the operation of allocating memory to various processes and data as per requirement and when there is no need, allocated memory should be free.
- There are two main goal of memory allocation: First, high utilization (maximum possible memory should be utilized) and second, high concurrency (maximum possible process should be in main memory).

Contiguous Memory Allocation

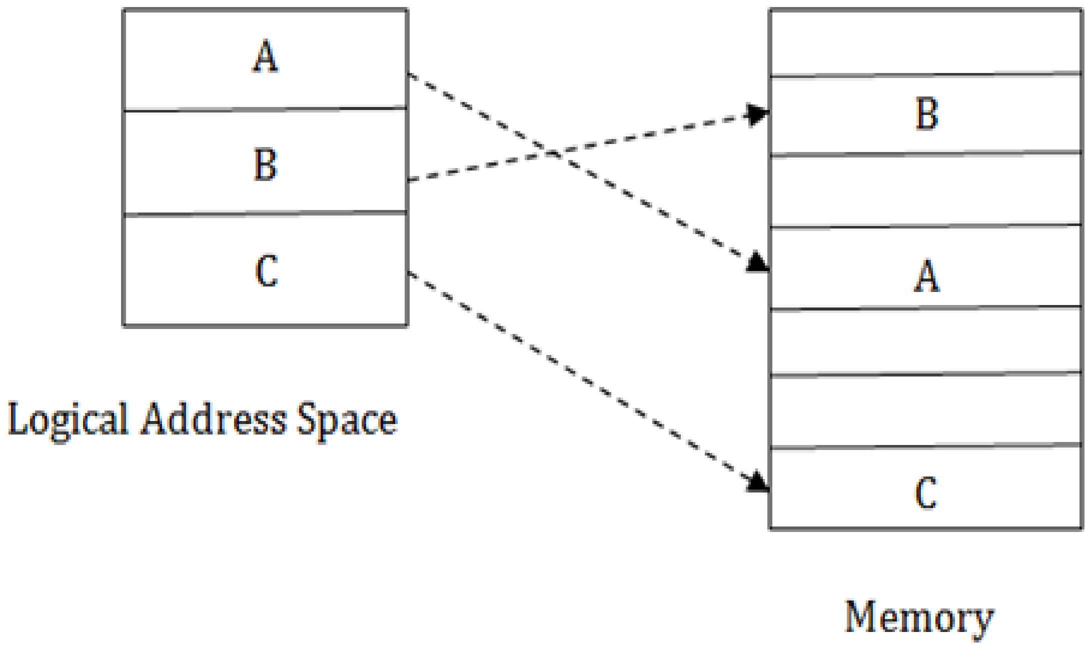
- Here, each process occupies a block of contiguous memory locations in main memory. Entire process is kept together in a contiguous section of memory.
- When process is brought in memory, a memory is searched to find out an area of free memory having enough size to hold a process. Once such area is found, required memory is allocated.

- If a contiguous memory space of the required size is not available in the main memory, the process has to wait until contiguous space of the required size is available.
- Here, physical address space and logical address space both are contiguous, not divided into partition.
- This method is simple for implement but it having poor memory utilization.



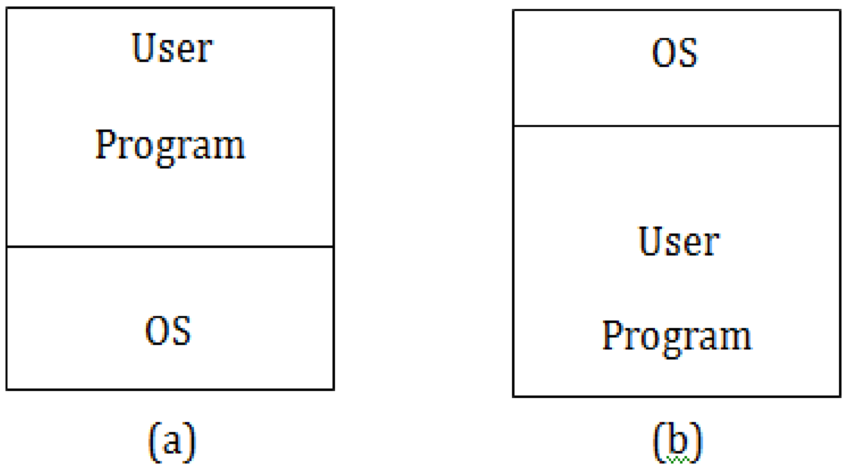
Non-Contiguous Memory Allocation

- Here, logical address space of a process is divided into partitions and for each partition contiguous area of free memory is allocated.
- Physical address space will not contiguous now.
- In the given figure, logical address space of a process is divided into three partitions A, B and C. Each partition is allocated separate area of memory in main memory.
- This method having better memory utilization but it is complex to implement.



Contiguous Memory Allocation
Single Partition Method (Single Process Monitor)

- This is the simplest memory management approach.
- Memory is shared between the process and Operating System. The location of Operating System in the main memory is either at the top level or at the bottom level as shown in figure.

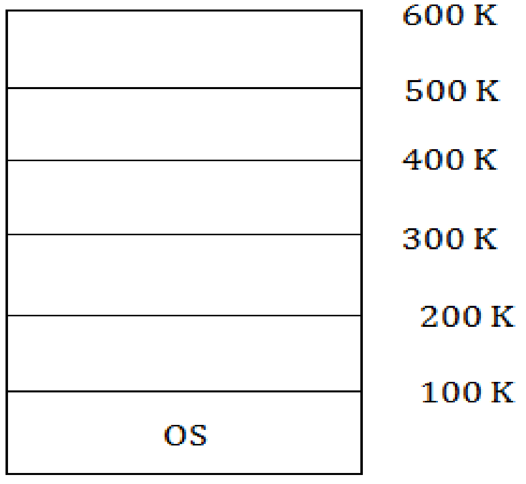


- Such type of system can be found in earlier personal computers running only MS-DOS. Here, only one process can load in main memory. So, it does not support multiprogramming.
- When user types a command, the Operating System copies the requested program from disk to main memory and executes it.
- When the process finishes the Operating System displays a prompt character and waits for a new command.
- In this method, utilization of CPU is lower and memory wastage is also there because only one process can be in memory at a time.

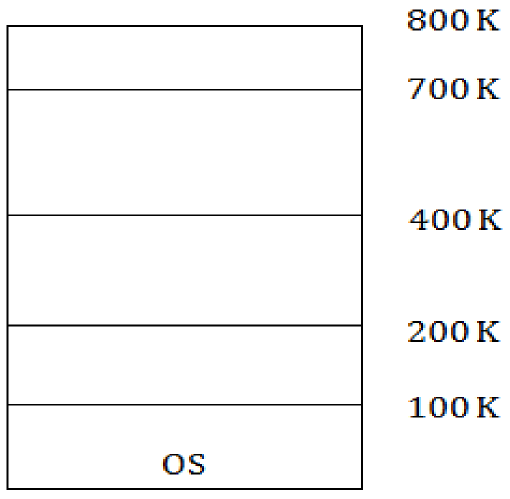
Multiprogramming with Fixed Partition

- This method allows multiple processes to execute simultaneously.

- Memory is divided into fixed size partitions. Size can be equal or unequal. Generally, unequal partition is used for better memory utilization.
- Each partition can allocate to only one process. Whenever, any process needs to be in memory a free partition big enough to hold the process is found and allocate.
- If there is no free partition available of required size than that process needs to wait.
- The number of process residing in the memory is called degree of multiprogramming and it is depend on number of partition.
- When a process terminates, the memory allocated to it will be free and allocated to another process waiting for memory.



Equal Partition



Unequal Partition

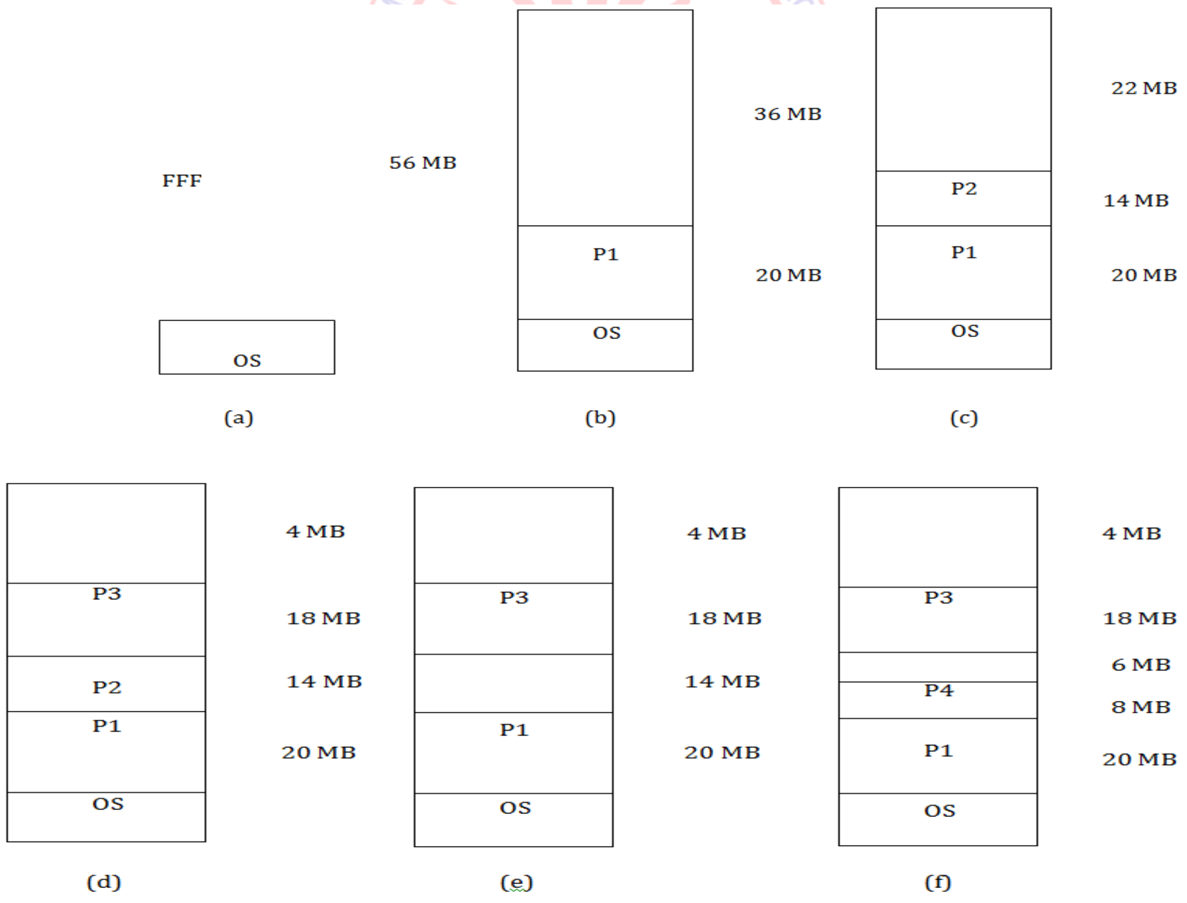
Disadvantages:

- Degree of multiprogramming is fixed.

- Here, partitions are fixed size. So, any space in a partition which is not used by the process is wasted. (Internal Fragmentation)

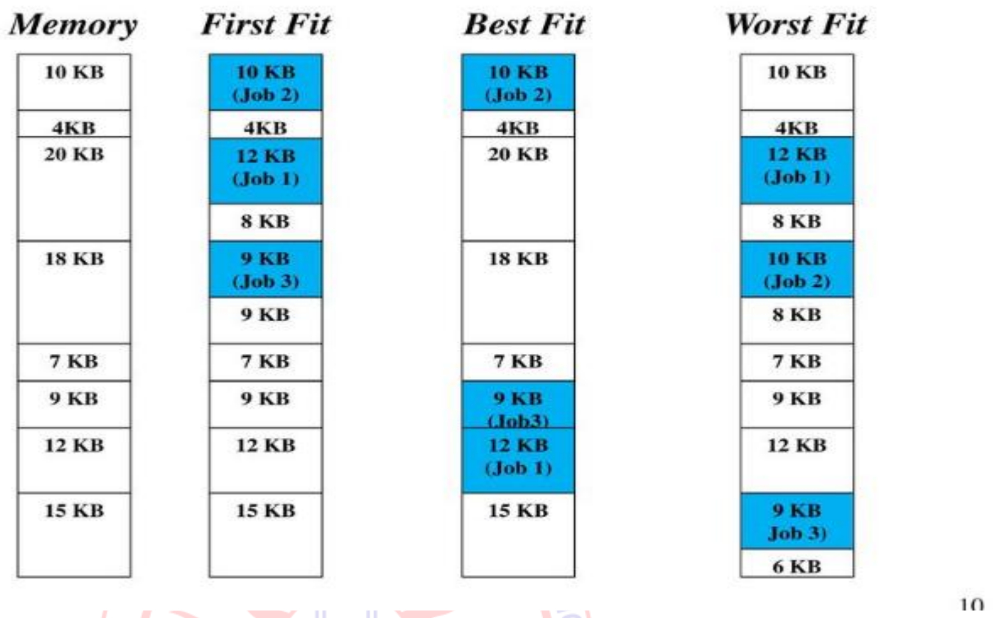
Multiprogramming with Dynamic Partition

- Here, memory is not divided into any fixed size partitions. Also, the number of partitions is not fixed.
- Size of each partition is variable depending on the size of process. Process is allocated exactly as much space as it requires.



- Initially, entire available memory is treated as a single free partition. Whenever any process enters in a system, area of free memory big enough to fit a process is found and allocated.
- If enough free memory is not available to fit a process than process needs to wait until required memory becomes free.
- Whenever any process terminates it release the memory. The three common strategies used to select a hole from the set of available holes:
- **First fit:** Search starts from starting location of memory. First available hole which is large enough to hold the process is selected to allocate.
- **Best fit:** Entire memory is searched here and the smallest hole which is large enough to hold the process is selected for allocation.
- **Worst fit:** Entire memory is searched here also and the largest hole which is large enough to hold the process is selected for allocation.
- **<u>Advantages:</u>**
- Better utilization of memory
- **<u>Disadvantages:</u>**
- External fragmentation: When process releases the memory that holes may be so small that no any process can be loaded in it. But total size of all holes may big enough to hold any process. But as contiguous allocation, these holes

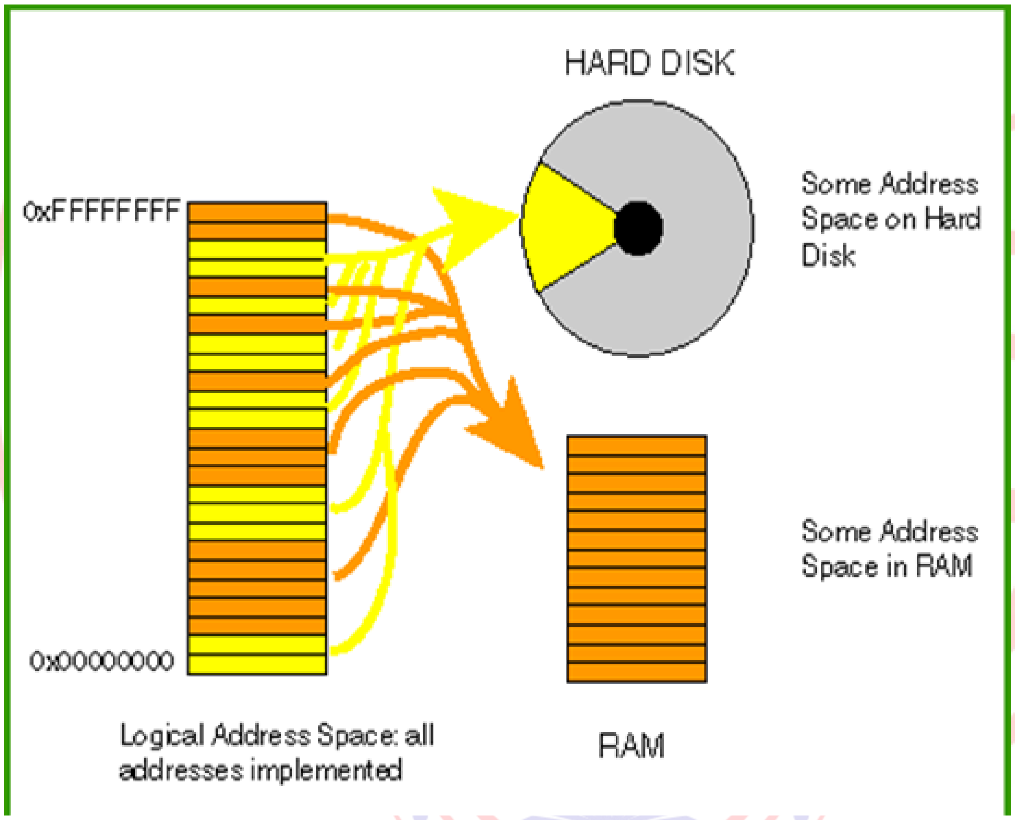
can’t be used. This memory wastage is called external fragmentation.



Non-Contiguous Memory Allocation Virtual Memory

- A virtual memory is a technique that allows a process to execute even though it is not fully loaded into the memory. Virtual memory removes the requirement that an entire process should be in main memory for its execution. The main advantage of this is a process can be larger size than the main memory.
- Here, the logical addresses are referred to virtual address and logical address space as a virtual address space. Virtual address space can be larger than physical address space. This virtual addresses are used by the application programs

to converted to physical addresses by the translation of these virtual addresses.


HARD DISK
Some Address Space on Hard Disk
0xFFFFFFFF
0x00000000
Some Address Space in RAM
Logical Address Space: all addresses implemented
RAM

- **<u>Advantages & Disadvantages:</u>**
- The primary advantage of virtual memory is the ability to load and execute a process that requires larger amount of memory than the size of main memory.
- Another advantage is the ability to eliminate the external fragmentation.
- Here, by loading only a part of a process into memory process execution can be done.

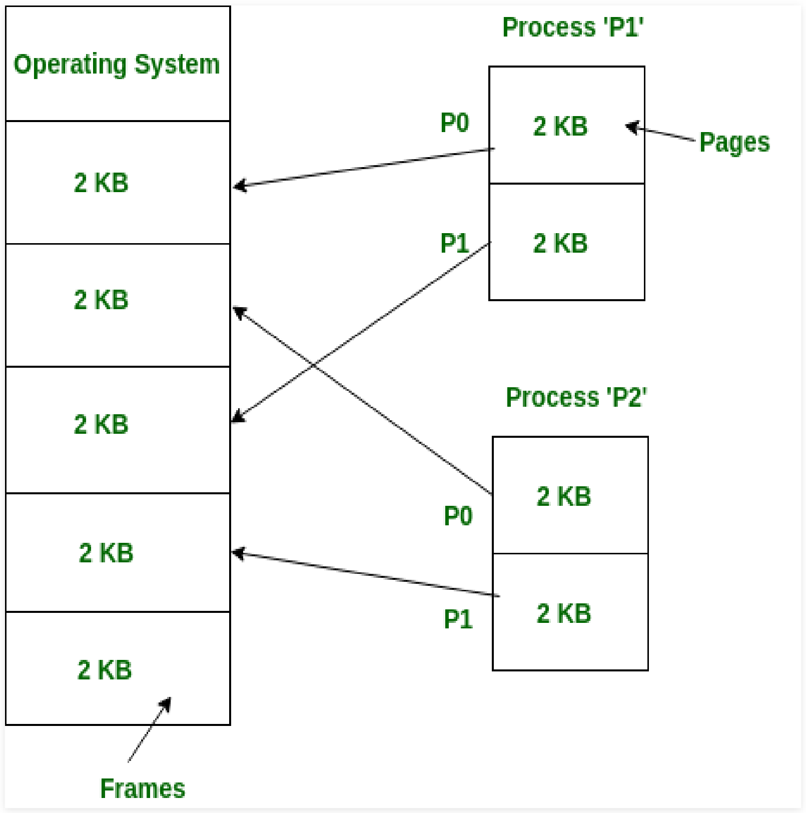
- The disadvantage is that the virtual memory system is slow and requires additional support from the system hardware for address translation.

Paging

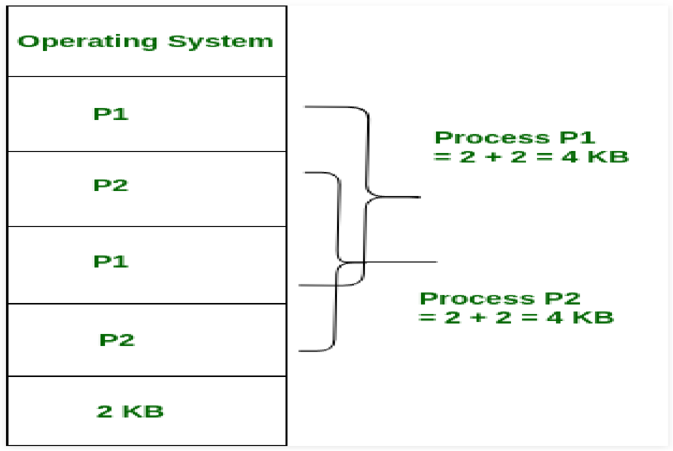
- Paging is one of the allocation methods which uses virtual memory concept and allocate memory in non contiguous way.
- Here, logical address space of a process is divided into blocks of fixed size which is called pages. Also, physical memory is divided into blocks of fixed size which is called frames.
- The pages and frames will be of the same size. The size of these blocks is the power of 2, means 2n. Whenever a process is to be executed its pages are moved from disk to the main memory.
- Operating System maintains a table called page table. The page table contains the page number, the information about the frame in which pages are loaded and also contains the addresses of those pages.
- System can have one page table for the whole system or a separate page table for each application. Page table is created and maintained by the part of the Operating System called Paging Supervisor.
- When a CPU attempts to access a page that is not available in main memory, the page fault occurs.

- When a page fault occurs, virtual memory handler finds the free frame in memory and loads the required page. This is called page-in operation.
- If no free frame exists in memory, some page existing in the memory is moved to the disk to free the frame. This is called page-out operation.

processes P1 and P2 are 2 KB each.



Resolvent main memory,



- **Advantage:**

  There is no external fragmentation because the size of the frame and page are same.

- **Disadvantage:**

  It requires additional memory references to read information from page table. Every instruction or data references require two memory accesses. One for page table and one for instruction and data.

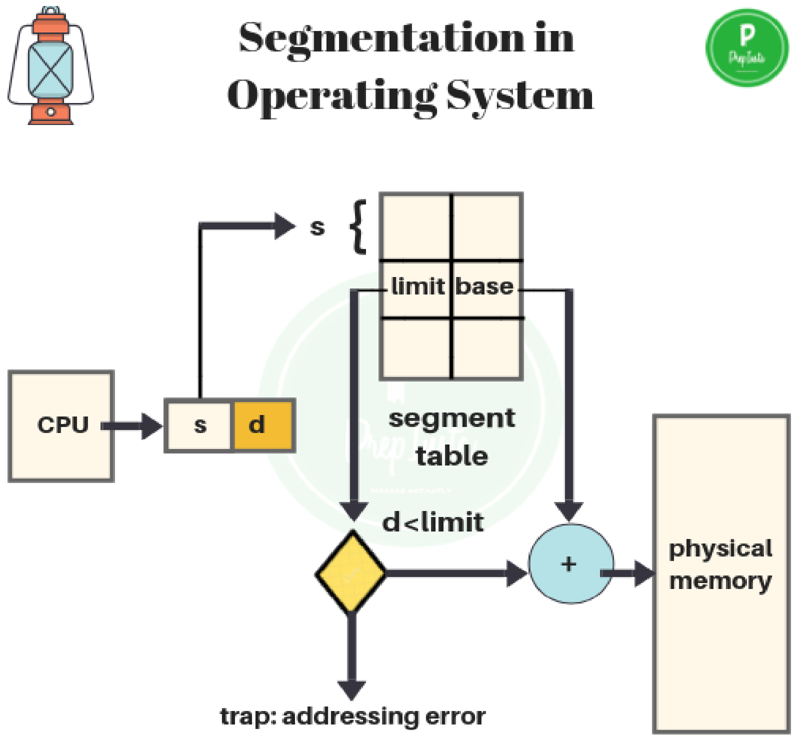
Demand Paging

- In simple paging, all pages of a program can be loaded into memory before the program is initiated. It is possible to improve on the memory utilization by using technique of demand paging.
- Here, page is loaded on demand, not all the pages loaded previously.

- For execution of a particular instruction, a page containing that instruction exists in memory; it will be loaded on the time of demand. Other pages of the program need not exist in memory.
- When a program refers to a page which does not exists in memory, the OS must arrange to load this page in memory. Thus a page is loaded in a demand. Hence, the terms demand paging.

Segmentation

- Segmentation is also a non contiguous memory allocation method which support virtual memory concept.
- Here, logical address space of any process is divided into code, data and stack.
- Code can be main function, other user defined function etc. Data can be local variables, global variables, arrays etc.
- In segmented system, information of a single segment resides in one contiguous area but different segments from the same process can be in non contiguous area of memory.
- Each segment can be considered as a completely independent address space. All segments are of different lengths and length depends on the size of that logical unit. All segments having a unique number to identify them.



- **Advantage:**
- Modification in one segment can’t affect the other, because all segments are independent from each other.
- **Disadvantage:**
- It is difficult to allocate contiguous free area within a segment.

- **Paging v/s Segmentation**
- The difference between the virtual memory implementation using paging and using segmentation is not only about the memory division with fixed and variable partition respectively but in segmentation process is more structured, it’s not only the large array of bytes while in paging, view of process does not change.