INDEX

Unit –1 Introduction to Operating System

05 to 38

1.1 Introduction to Operating System
1.1.1 What is an Operating System?
1.1.2 Operating System Software
1.1.3 Types of Operating Systems

1.2 Memory Management: Early System
1.2.1 Single User Contiguous Scheme
1.2.2 Fixed Partitions
1.2.3 Dynamic Partitions
1.2.4 Allocation and De-allocation methods
1.2.5 Reloadable Dynamic Partitions

1.3 Memory Management: Virtual Memory
1.3.1 Paged Memory Allocation
1.3.2 Demand Paging
1.3.3 Page Replacement Algorithms
1.3.4 Segmented Memory Allocation
1.3.5 Segmented/ Demand Paged Memory Allocation
1.3.6 Virtual Memory

Unit – 2 Processor Management

39 to 55

2.1 Job Schedulers
2.2 Job and Process Status
2.3 Process Control Blocks
2.4 Process Scheduling Policies
2.5 Process Scheduling Algorithms

Unit – 3 Deadlock and process Synchronization

56 to 83

3.1 What is a Deadlock?
3.2 Seven cases for deadlock
3.3 Conditions for Deadlock
3.4 Strategies for handling Deadlocks
3.5 Starvation (Dining Philosophers Problem)
3.6 Process Synchronization
3.7 What is Parallel Processing?
3.8 Typical Multi Processing Configurations
3.9 Process Synchronization Software
3.10 Semaphores
3.11 Process Cooperation

Unit – 4 Device Management and File Management

84 to 102

4.1 Device Management
4.2 Types of System devices
4.3 Communication among Devices
4.4 Management of I/O Requests
4.5 Device Handler Seek Strategies
4.6 File Management
4.7 Physicals of Rage Allocation
4.8 Data Compression
4.9 Access Control Verification Module

❖ **Self Test Examination** 103 to 110
❖ **Paper 2019** 111 to 114

Unit –1 Introduction to Operating System

1.1 Introduction to Operating System:

1.1.1What is an Operating System?

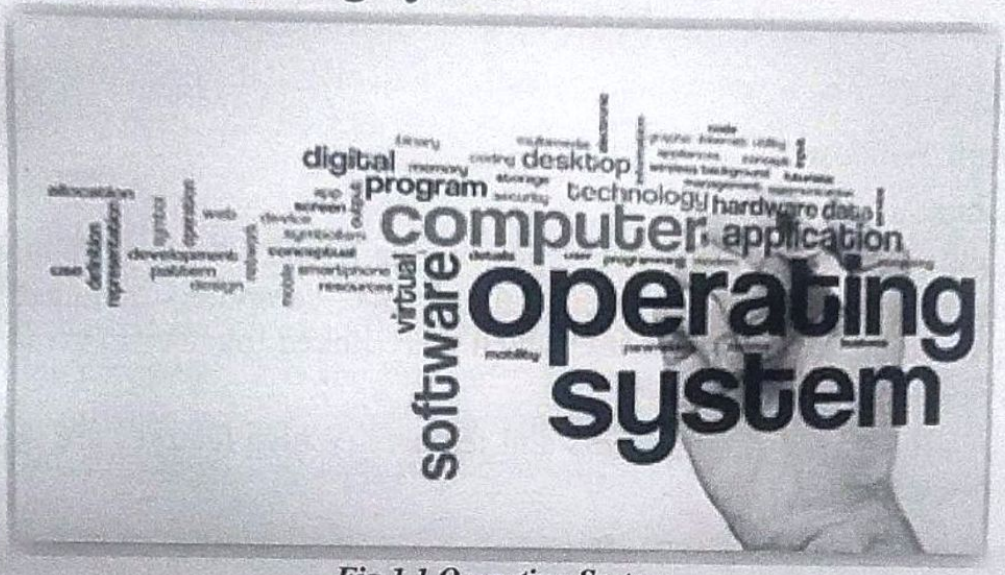


Fig.1.1 Operating System

An **Operating system (OS)** is 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 likes 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.

An operating system brings powerful benefits to computer software and software development. Without an operating system, every application would need to include its own UI, as well as the comprehensive code needed to handle all low-level functionality of the underlying computer, such as disk storage, network interfaces and so on. Considering the vast array of underlying hardware available, this would vastly bloat the size of every application and make software development impractical.

Instead, many common tasks, such as sending a network packet or displaying text on a standard output device, such as a display, can be offloaded to system software that serves as an intermediary between the applications and the hardware. The system software provides a consistent and repeatable way for applications to interact with the hardware without the applications needing to know any details about the hardware.

"The Software is the Non-Touchable Parts of the Computer , and Software's are those which are used for Performing an Operation So that Software's are just used for Making an Application but hardware's are those which are used for Performing an Operation ."

Operating system is software that is required in order to run application programs and utilities. It works as a bridge to perform better interaction between application programs and hardware of the computer.

Examples for OSs include:

- Android
- iOS
- Mac OS X
- Microsoft Windows
- And Linux

Some operating systems were developed in the 1950s, when computers could only execute one program at a time. Later in the decade, computers included many software programs, sometimes called libraries, which were linked together to create the beginning of today's operating systems.

The OS consists of many components and features. Which features are defined as part of the OS varies with each OS.

1.1.2 Operating System Software:

An operating system, or "OS," is software that communicates with the hardware and allows other programs to run. It is comprised of system software, or the fundamental files your computer needs to boot up and function. Every desktop computer, tablet, and smartphone includes an operating system that provides basic functionality for the device.

Common desktop operating systems include Windows, OS X, and Linux. While each OS is different, most provide a graphical user interface, or GUI, that includes a desktop and the ability to manage files and folders. They also allow you to install and run programs written for the operating system. Windows and Linux can be installed on standard PC hardware, while OS X is designed to run on Apple systems. Therefore, the hardware you choose affects what operating system(s) you can run.

Mobile devices, such as tablets and smartphones also include operating systems that provide a GUI and can run applications. Common mobile OSes include Android, iOS, and Windows Phone. These OSes are developed specifically for portable devices and therefore are designed around touchscreen input. While early mobile operating systems lacked many features found in desktop OSes, they now include advanced capabilities, such as the ability to run third-party apps and run multiple apps at once.

Since the operating system serves as a computer's fundamental user interface, it significantly affects how you interact with the device. Therefore, many users prefer to use a specific operating system. For example, one user may prefer to use a computer with OS X instead of a Windows-based PC. Another user may prefer an Android-based smartphone instead of an iPhone, which runs the iOS.

When software developers create applications, they must write and compile them for a specific operating system. This is because each OS communicates with the hardware

differently and has a specific application program interface, or API, that the programmer must use. While many popular programs are cross platform, meaning they have been developed for multiple OSes, some are only available for a single operating system. Therefore, when choosing a computer, make sure the operating system supports the programs you want to run.

➢ **Functions of operating system:**

Operating System means that Resource Manager, that manage all the Resources those are attached to the System, like Memory, Processor, Input/output Devices.

Storage Management:

It manages all the Storing and Accessing Files and Directories Reading/ Writing Operations.

Operating system manages overall activities of a computer and the input/output devices attached to the computer. It is the first software you see when you turn on the computer, and the last software you see when the computer is turned off. It is the software that enables all the programs you use. At the simplest level, an operating system does two things:

The first, it manages the hardware and software resources of the computer system. These resources include the processor, memory, disk space, etc. The second, it provides a stable, consistent way for applications to deal with the hardware without having-to know all the details of the hardware.

The first task is very important i.e. managing the hardware and software resources, as various processes compete to each other for getting the CPU time and memory space to complete the task. In this regard; the operating system acts as a manager to allocate the available resources to 'satisfy the requirements of each process.

The second task i.e. providing a consistent application interface is especially important. A consistent application program interface (API) allows a user (or *S/W* developer) to write an application program on any computer and to run this program on another computer, even if the hardware configuration is different like as amount of memory, type of CPU or storage disk. It shields the user of the machine from the low-level details of the machine's operation and provides frequently needed facilities.

Process Management: It manages all the User and system Process.

Memory Management: Operating System also manages the Computer Memory that is provided to the process.

Extended Machine: It is behaves like an Extended Machine that Provides us Sharing of Files between Multiple Users.

Mastermind: It performs Many Functions that's why we can say that Operating System is a Mastermind.

➢ **How Operating System Work:**

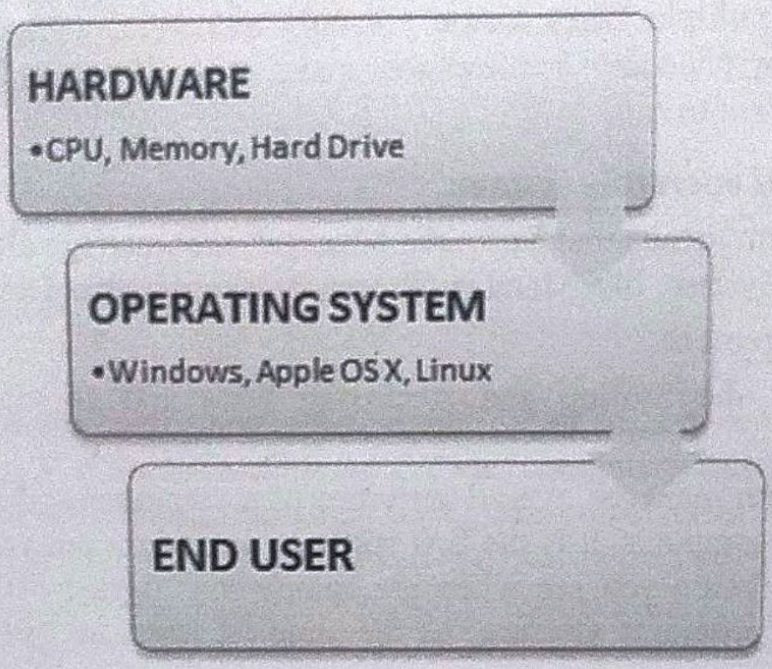


Fig.1.2 Operating System Work

When you turn on the computer, the operating system program is loaded into the main memory. This program is called the kernel. Once initialized, the system program is prepared to run the user programs and permits them to use the hardware efficiently. Windows 98/XP is an excellent example that supports different types of hardware configurations from thousands of vendors and accommodates thousands of different I/O devices like printers, disk drives, scanners and cameras.

Operating systems may be classified based on if multiple tasks can be performed simultaneously, and if the system can be used by multiple users. It can be termed as single-user or multi-user OS, and single-tasking or multi-tasking OS.A multi-user system must be multi-tasking. MS-DOS and Windows 3x are examples of single user operating system. Whereas UNIX is an example of multi-user and multitasking operating system.

For Example if we want to Perform Some Paintings on the Screen, then we must use the Application Software as Paint and Hardware as a Mouse for Drawing an Object. But how the System knows what to do when Mouse Moves on the Screen and When the Mouse Draws a Line on the System so that Operating System is Necessary which Interact between or which Communicates with the Hardware and the Software. For Better understanding you can see the Working of the Operating System.

➢ **Operating System Components:**

Kernel:

This provides basic-level control over all of the computer hardware devices. Main roles include reading data from memory and writing data to memory, processing execution

orders, determining how data is received and sent by devices, such as the monitor, keyboard and mouse; and determining how to interpret data received from networks. Monolithic kernels have a simpler design and consist of a single code that communicates with all hardware and software.

Microkernels implement user and kernel services in different address spaces, reducing their size, but forcing the use of message passing to execute services.

User Interface (UI):

This component allows interaction with the user, which may occur through graphical icons and a desktop or through a command line. The UI is further divided into Command Line Interface (CLI), consisting of a text-based interface where advanced users can prompt specific commands by typing them, and a Graphical User Interface (GUI).

The latter is a visual interface that allows the end user to issue commands by interacting with symbols, icons, and menus using an input device such as a mouse or touchpad.

Application Programming Interfaces (API):

This component allows application developers to write modular code. An API defines how other systems or components can use a certain application.

1.1.3 Types of Operating Systems:

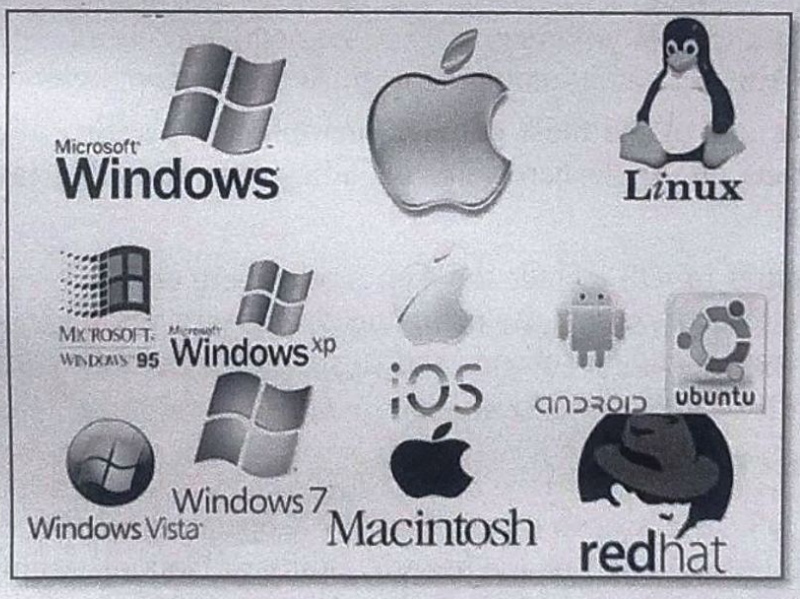


Fig.1.3 Types of Operating System

(1) Microsoft Windows:

Microsoft Windows has existed in one form or another since 1985, and it remains the most popular operating system for home and office computers. Its latest versions, including Windows 10, are also used on some tablets, and the OS is used on some web and number-crunching server computers as well. Computers from a wide variety of manufacturers can use Windows.

Initial versions of Windows worked with an earlier Microsoft operating system called MS-DOS, providing a modern graphical interface on top of DOS's traditional text-based commands. Signature features of Microsoft Windows's user interface include windows themselves – rectangle-shaped, on-panel screens that represent individual applications. The Windows Start menu has helped generations of users find programs and files on their devices.

Efforts to use versions of the Windows OS for smartphones have been less successful.

(2) Linux Operating System:

Unlike many other operating systems, development on Linux isn't led by any one company. The operating system was created by Finnish programmer Linus Torvalds in 1991. Nowadays, programmers from all over the world collaborate on its open source code and submit tweaks to the central kernel software and other programs.

A wide assortment of commercial and open source software is available for Linux, and various Linux distributions provide custom user interfaces and tools for installing software onto machines running the operating system. A favourite of many programmers, Linux is widely used on corporate and scientific servers, including cloud computing environments. Linux can be run on a wide variety of hardware and is available free of charge over the internet.

(3) Apple iOS:

Apple's iOS is one of the most popular smartphone operating systems, second only to Android. It runs on Apple hardware, including iPhones, iPad tablets and iPod Touch media players.

Signature features of iOS include the App Store where users buy apps and download free software, an emphasis on security including strong encryption to limit what unauthorized users can extract from the phone, and a simple, streamlined interface with minimal hardware buttons.

(4) Apple mac OS:

Apple's mac OS, successor to the popular OS X operating system, runs on Apple laptops and desktops. Based in part on the historic family of Unix operating systems dating back to research in the 1960s at AT&T's Bell Labs, mac OS shares some features with other Unix-related operating systems including Linux. While the graphical interfaces are different, many of the underlying programming interfaces and command line features are the same.

Signature elements of mac OS include the dock used to find programs and frequently used files, unique keyboard keys including the Command key, and the stoplight-colored buttons used to resize open program windows. Mac OS is known for its user-friendly

features, which include Sire, a natural-voice personal assistant, and Face Time, Apple's video-calling application.

(5) Google's Android OS:

Android is the most popular operating system in the world judging by the number of devices installed. Largely developed by Google, it's chiefly used on smartphones and tablets. Unlike iOS, it can be used on devices made by a variety of different manufacturers, and those makers can tweak parts of its interface to suit their own needs.

Users can download custom versions of the operating system because large portions of it are open source, meaning anyone can legally modify it and publish their own. However, most people prefer to stick with the version that comes on their devices.

Android, like iOS, comes with an application and media store called the Play Store built by Google. Some phone manufacturers and other organizations also offer their own stores to install software and media.

As computers have progressed and developed, so have their operating systems. Below is a list of operating systems categories and examples of operating systems that fall into these categories. Many computer operating systems fall into more than one of the below types.

GUI - Short for Graphical User Interface, a GUI operating system contains graphics and icons and is commonly navigated by using a computer mouse. See the GUI definition for further information. Examples of GUI operating systems are:

- Windows 7
- Windows 8
- Windows 10

Multi-user - A multi-user operating system allows multiple users to use the same computer at the same time and at different times. See the multi-user definition for a complete definition. Examples of operating systems that would fall into this category are:

- Linux
- Unix
- Windows 7 & 8 & 10

Multiprocessing - An operating system capable of supporting and utilizing more than one computer processor.

Multitasking - An operating system that is capable of allowing multiple software processes to run at the same time.

Multithreading - Operating systems that allow different parts of a program to run concurrently.

➢ Other Types of Operating system:

- Batch Operating System
- Multitasking/Time Sharing OS
- Multiprocessing OS
- Real Time OS
- Distributed OS
- Network OS
- Mobile OS

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 submits it to the computer operator.

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.

Multiprocessing OS:

Multiprocessing refers to a computer system's ability to support more than one process (program) at the same time. Multiprocessing operating systems enable several programs to run concurrently. UNIX is one of the most widely used multiprocessing systems, but there are many others, including OS/2 for high-end PCs. Multiprocessing systems are much more complicated than single-process systems because the operating system must allocate resources to competing processes in a reasonable manner.

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.

Distributed Operating System:

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

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.

Mobile OS:

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

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

1.2 Memory Management: Early System:

"Memory is the primary and fundamental power, without which there could be no other intellectual operation."

—Samuel Johnson (1709–1784)

Types of memory allocation schemes:

- Single User System
- Fixed partitions
- Dynamic partitions
- Allocation and De-allocation methods
- Relocatable dynamic partitions

1.2.1 Single User Contiguous Scheme:

Contiguous memory allocation is a memory allocation method that allocates a **single contiguous section of memory** to a process or a file. This method takes into account the size of the file or a process and also estimates the maximum size, up to what the file or process can grow? Taking into account the future growth of the file and its request for memory, the operating system allocates sufficient contiguous memory blocks to that file. Considering this future expansion and the file's request for memory, the operating system will allocate those many contiguous blocks of memory to that file.

- Commercially available in 1940s and 1950s.
- Each program was loaded in its entirety into memory and allocated as much contiguous memory space allocated as it needed.
- If the program was too large and didn't fit the available memory space, it couldn't be executed,
- Although early computers were physically large, they had very little memory.
- Computers have only a finite amount of memory.
- If a program doesn't fit, then either the size of the main memory must be increased or the program must be modified.

- Make the program smaller –Use methods that allow program segments (partitions made to the program) to be overlaid.
- Transfer segments of a program from secondary storage into main memory for execution.
- Two or more segments take turns occupying the same memory locations.

Operating System	10K
Program 1 (40K)	50K
Unused Main Memory	

➢ **Advantages and Disadvantages of Single Contiguous Allocation:**

Advantages:

1. Simple Allocation
2. Entire scheme requires less memory
3. Easy to implement and use

Disadvantages:

1. Memory is not fully utilised
2. Processor (CPU) is also not fully utilised
3. User Program is being limited to the size available in the memory

1.2.2 Fixed Partitions:

The earliest and one of the simplest technique which can be used to load more than one processes into the main memory is Fixed partitioning or Contiguous memory allocation.

In this technique, the main memory is divided into partitions of equal or different sizes. The operating system always resides in the first partition while the other partitions can be used to store user processes. The memory is assigned to the processes in contiguous way.

This is the oldest and simplest technique used to put more than one processes in the main memory. In this partitioning, number of partitions (non-overlapping) in RAM is fixed but size of each partition may or may not be same. As it is contiguous allocation, hence no spanning is allowed. Here partition are made before execution or during system configure.

In fixed partitioning,

1. The partitions cannot overlap.
2. A process must be contiguously present in a partition for the execution.

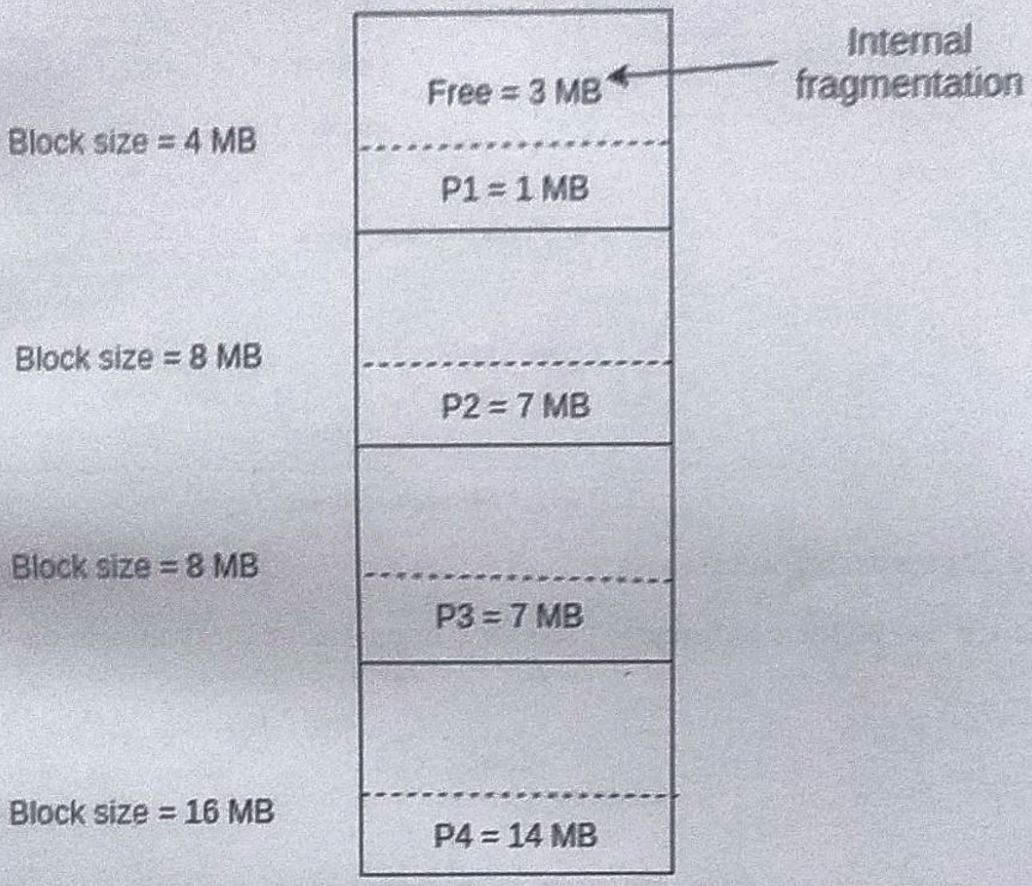


Fig.1.4 Internal Fragmentation

As illustrated in above figure, first process is only consuming 1MB out of 4MB in the main memory.

Hence, Internal Fragmentation in first block is (4-1) = 3MB.

Sum of Internal Fragmentation in every block = (4-1)+(8-7)+(8-7)+(16-14)= 3+1+1+2 = 7MB.

Suppose process P5 of size 7MB comes. But this process cannot be accommodated in spite of available free space because of contiguous allocation (as spanning is not allowed). Hence, 7MB becomes part of External Fragmentation.

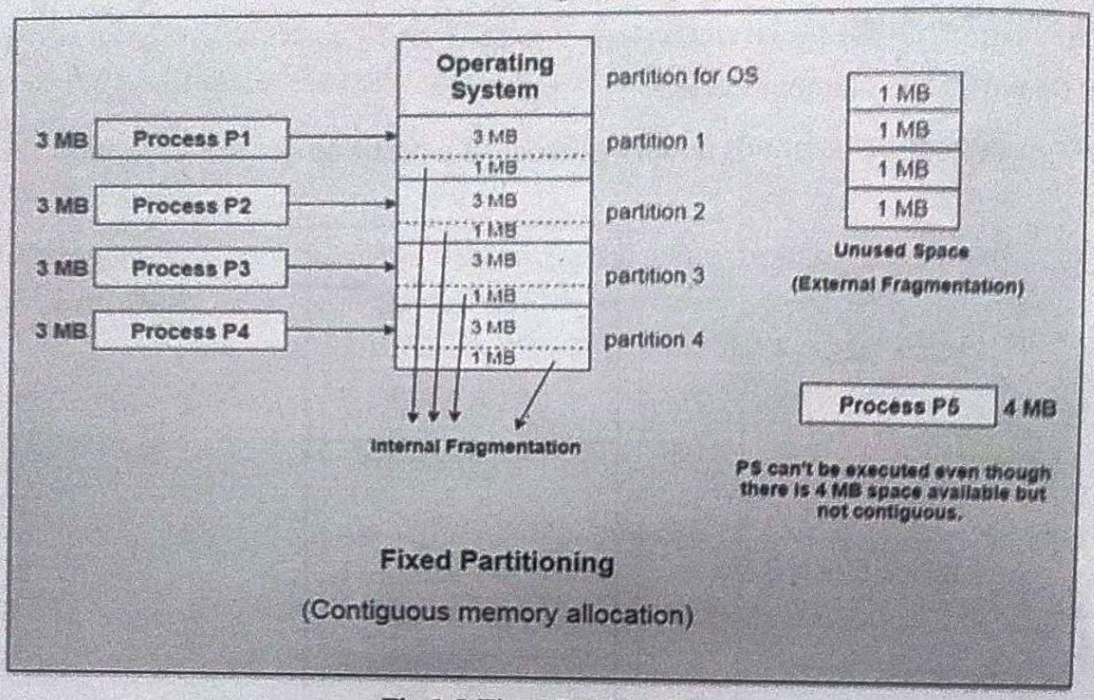
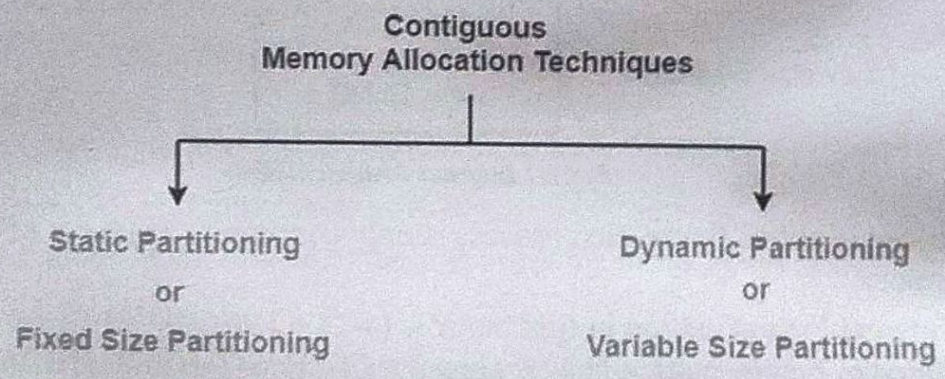


Fig.1.5 Fixed Partitioning

1.2.3 Dynamic Partitions:

There are two popular techniques used for contiguous memory allocation-



Dynamic partitioning tries to overcome the problems caused by fixed partitioning. In this technique, the partition size is not declared initially. It is declared at the time of process loading.

The first partition is reserved for the operating system. The remaining space is divided into parts. The size of each partition will be equal to the size of the process. The partition size varies according to the need of the process so that the internal fragmentation can be avoided.

- Dynamic partitioning is a variable size partitioning scheme.

- It performs the allocation dynamically.
- When a process arrives, a partition of size equal to the size of process is created.
- Then, that partition is allocated to the process.

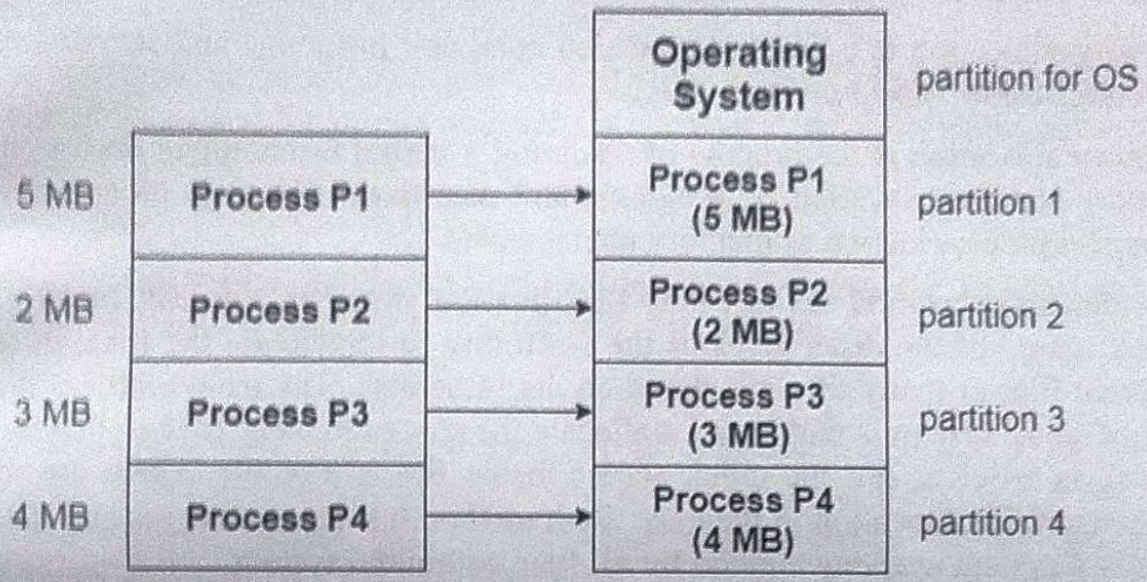


Fig.1.6 Partition Allocated Process

➢ **Advantages of Fixed Partitioning –**

1. **Easy to implement:** Algorithms needed to implement Fixed Partitioning are easy to implement. It simply requires putting a process into certain partition without focussing on the emergence of Internal and External Fragmentation.
2. **Little OS overhead:** Processing of Fixed Partitioning require lesser excess and indirect computational power.

➢ **Disadvantages of Fixed Partitioning –**

1. **Internal Fragmentation:** Main memory use is inefficient. Any program, no matter how small, occupies an entire partition. This can cause internal fragmentation.
2. **External Fragmentation:** The total unused space (as stated above) of various partitions cannot be used to load the processes even though there is space available but not in the contiguous form (as spanning is not allowed).
3. **Limit process size:** Process of size greater than size of partition in Main Memory cannot be accommodated. Partition size cannot be varied according to the size of incoming process's size. Hence, process size of 32MB in above stated example is invalid.
4. **Limitation on Degree of Multiprogramming:** Partition in Main Memory is made before execution or during system configure. Main Memory is divided into fixed

number of partition. Suppose if there are partitions in RAM and are the number of processes, then condition must be fulfilled. Number of processes greater than number of partitions in RAM is invalid in Fixed Partitioning.

1.2.4 Allocation and De-allocation methods:

Memory allocation is a process by which computer programs and services are assigned with physical or virtual memory space.

Memory allocation is the process of reserving a partial or complete portion of computer memory for the execution of programs and processes. Memory allocation is achieved through a process known as memory management.

The allocation method defines how the files are stored in the disk blocks. The direct access nature of the disks gives us the flexibility to implement the files. In many cases, different files or many files are stored on the same disk. The main problem that occurs in the operating system is that how we allocate the spaces to these files so that the utilization of disk is efficient and the quick access to the file is possible. There are mainly three methods of file allocation in the disk. Each method has its advantages and disadvantages. Mainly a system uses one method for all files within the system.

- Contiguous allocation
- Linked allocation
- Indexed allocation

The main idea behind contiguous allocation methods is to provide:

- Efficient disk space utilization
- Fast access to the file blocks

➢ **Contiguous allocation:**

In this scheme, a file is made from the contiguous set of blocks on the disk. Linear ordering on the disk is defined by the disk addresses. In this scheme only one job is accessing the disk block b after that it accesses the block b+1 and there are no head movements. When the movement of the head is needed the head moves only from one track to another track. So the disk number that is required for accessing the contiguous allocation is minimal. Contiguous allocation method provides a good performance that's why it is used by the IBM VM/CMS operating system. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: **b, b+1, b+2,..., b+n-1**. This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file. For a contiguous allocation the directory entry the address of the starting block and Length of the allocated portion.

The file **'A'** in the following figure starts from block 19 with **length = 6 blocks**. Therefore, it occupies **19, 20, 21, 22, 23, 24** blocks.

- Each file in the disk occupies a contiguous address space on the disk.
- In this scheme, the address is assigned in the linear fashion.
- They are very easy to implement the contiguous allocation method.
- In the contiguous allocation technique, external fragmentation is a major issue.

Advantages:

1. In the contiguous allocation, sequential and direct access both is supported.
2. This is very fast and the number of seeks is minimal in the contiguous allocation method.

Disadvantages:

1. Contiguous allocation method suffers internal as well as external fragmentation.
2. In terms of memory utilization, this method is inefficient.

➢ **Linked allocation:**

The problems of contiguous allocation are solved in the linked allocation method. In this scheme, disk blocks are arranged in the linked list form which is not contiguous. The disk block is scattered in the disk. In this scheme, the directory entry contains the pointer of the first block and pointer of the ending block. These pointers are not for the users. For example, a file of six blocks starts at block 10 and end at the block. Each pointer contains the address of the next block. When we create a new file we simply create a new entry with the linked allocation. Each directory contains the pointer to the first disk block of the file. When the pointer is nil then it defines the empty file.

Advantages:

1. In terms of the file size, this scheme is very flexible.
2. We can easily increase or decrease the file size and system does not worry about the contiguous chunks of memory.
3. This method free from external fragmentation this makes it better in terms of memory utilization.

Disadvantages:

1. In this scheme, there is large no of seeks because the file blocks are randomly distributed on disk.
2. Linked allocation is comparatively slower than contiguous allocation.
3. The pointer is extra overhead on the system due to the linked list.

➢ **Indexed Allocation:**

In this scheme, a special block known as the index block contains the pointer to all the blocks occupied by a file. Each file contains its index which is in the form of an array of disk block addresses. The itch entry of index block point to the itch block of the file. The address of the index block is maintained by the directory. When we create a file all pointers is set to nil. A block is obtained from the free space manager when the first itch

block is written. When the index block is very small it is difficult to hold all the pointers for the large file. to deal with this issue a mechanism is available. Mechanism includes the following:

- Linked scheme
- Multilevel scheme
- Combined scheme

Advantages:

1. This scheme supports random access of the file.
2. This scheme provides fast access to the file blocks.
3. This scheme is free from the problem of external fragmentation

Disadvantages:

1. The pointer head is relatively greater than the linked allocation of the file.
2. Indexed allocation suffers from the wasted space.
3. For the large size file, it is very difficult for single index block to hold all the pointers.

1.2.5 Relocatable Dynamic Partitions:

In relocatable dynamic partition, memory manager in operating system relocates the program that is all empty blocks are gathered to form one single block of large memory enough to accommodate some or all of the jobs waiting in the queue.

Consider the following example:

Job	Memory utilized	Turnaround time
Job1	100K	3
Job2	10K	1
Job3	30K	2
Job4	20K	1
Job5	25K	2
Job6	5K	1
Job7	25K	1
Job8	50K	2
Job9	90K	3
Job10	100K	3

Let us assume the memory size is 220k with 10k allocated to operating system programs. The above table gives the data of number of jobs, memory utilized by individual job and their respective turnaround time. There are 10 jobs to be loaded into memory. Initially with 220k of memory, we can load job1 to job6 with free memory of 20k. At first

turnaround time, job2 of 10k, job4 of 20k, and job6 of 5k are completed leaving job 1, job 3, and job 5 as it is with free memory of 55k, in other words job1, job3, and job5 are still processing. Job 7 of 25k now enters into the memory partition leaving only 30k free memory where neither of the remaining jobs (job 8, job 9, and job 10) can accommodate.

At second turnaround time job3 of 30k, job5 of 25k, and job7 of 25k are completed leaving job 1 of 100k with free memory of 110k. Now job 8 of 50K is entered where 50k is allocated leaving 60K of free memory space where either job 9 or job 10 can accommodate.

Note: for job3 and job5 turnaround time is finished and at the same time turnaround time 1 of job 7 is also finished.

At third turnaround time, job1 of 100k is completed leaving free memory space of 160k. Job9 now can be allocated with 90k leaving remaining 70K of free memory where job10 cannot accommodate. Once job 8 finishes turnaround time of 2, it is completed leaving 120k of free memory space. Now job10 can be allocated with 100k of memory leaving 20K of free memory space. Process repeats until all the memory locations are freed leaving 10k allocated to operating system processes.

Diagrammatic representation of above example is shown below:

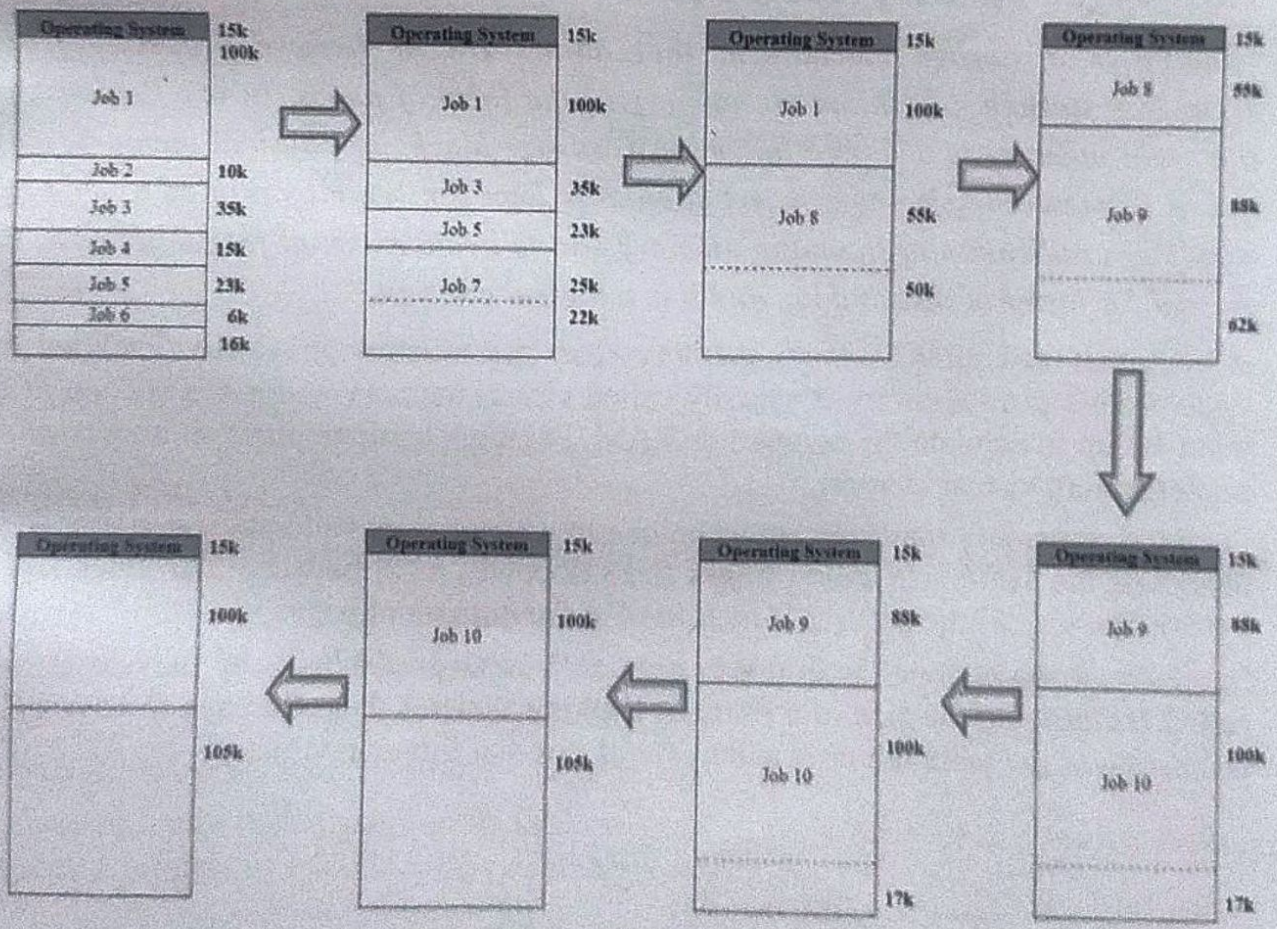


Fig.1.7 Diagrammatic Representation

Advantages Relocatable Memory Management:

What distinguishes this management from the previous ways of management is the possibility of implementing the compacting process for the free partitions, which implementation leads to increasing the degree of multiplicity of programs by reducing the spaces as much as possible and trying to assemble them for loading another job.

Disadvantages Relocatable Memory Management:

1. The need to provide special registers and thus exploit locations in the memory to save the values of displacement.
2. Waste the time of CPU unit in:
3. Performing the compacting process
4. Calculating the values of displacement each job
5. Calculating the physical address when implementing each single instruction of the job instructions.

1.3 Memory Management: Virtual Memory:

1.3.1 Paged Memory Allocation:

"Paging is a storage mechanism that allows OS to retrieve processes from the secondary storage into the main memory in the form of pages. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called frames. The size of a frame should be kept the same as that of a page to have maximum utilization of the main memory and to avoid external fragmentation. Paging is used for faster access to data, and it is a logical concept."

A computer can address more memory than the amount physically installed on the system. This extra memory is actually called virtual memory and it is a section of a hard that's set up to emulate the computer's RAM. Paging technique plays an important role in implementing virtual memory.

Paging is a memory management technique in which process address space is broken into blocks of the same size called **pages** (size is power of 2, between 512 bytes and 8192 bytes). The size of the process is measured in the number of pages.

Similarly, main memory is divided into small fixed-sized blocks of (physical) memory called **frames** and the size of a frame is kept the same as that of a page to have optimum utilization of the main memory and to avoid external fragmentation.

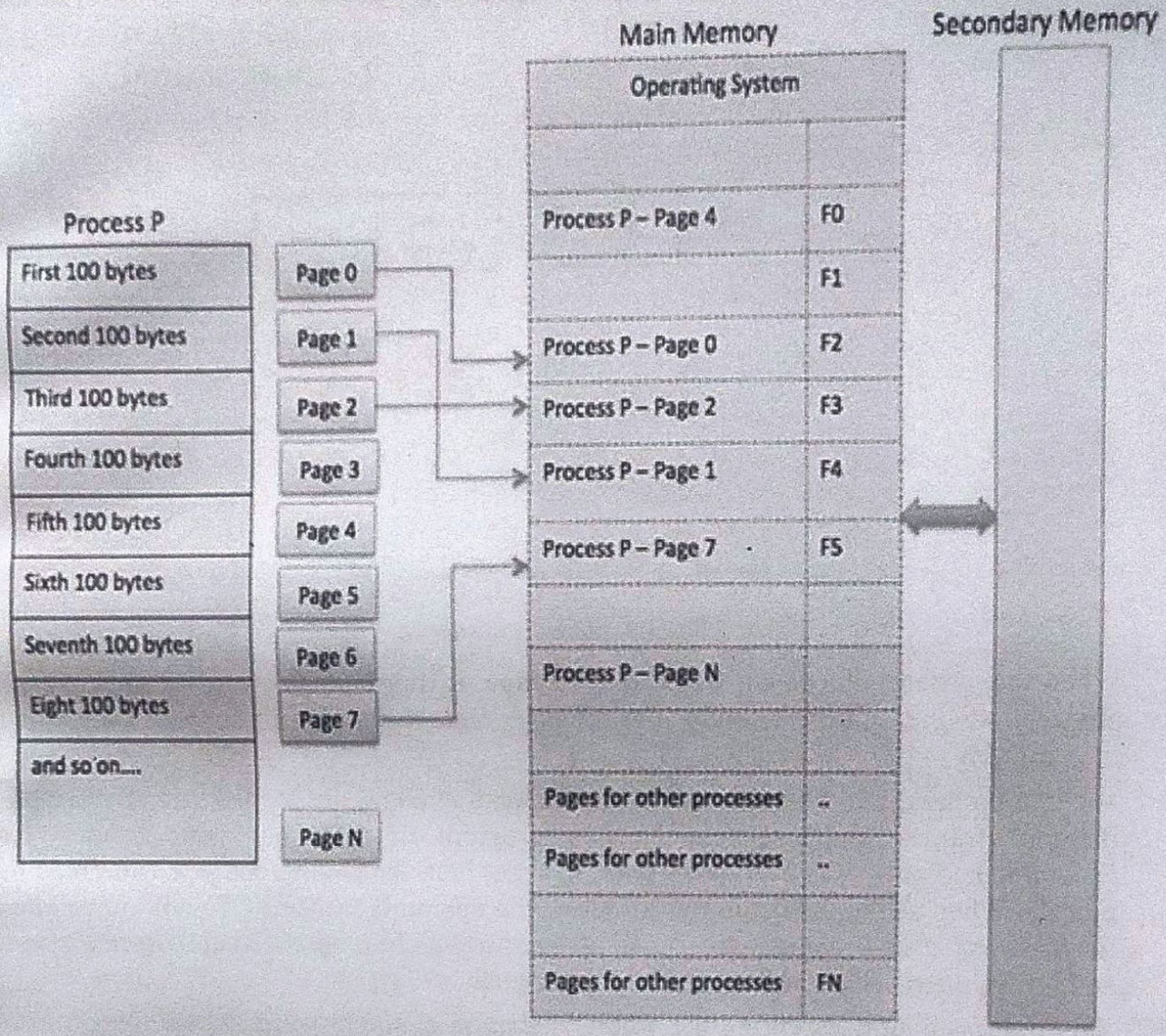


Fig.1.8 Paged Memory Allocation

Address Translation:

Page address is called **logical address** and represented by **page number** and the **offset**.

Logical Address = Page number + page offset

Frame address is called **physical address** and represented by a **frame number** and the **offset**.

Physical Address = Frame number + page offset

A data structure called **page map table** is used to keep track of the relation between a page of a process to a frame in physical memory.

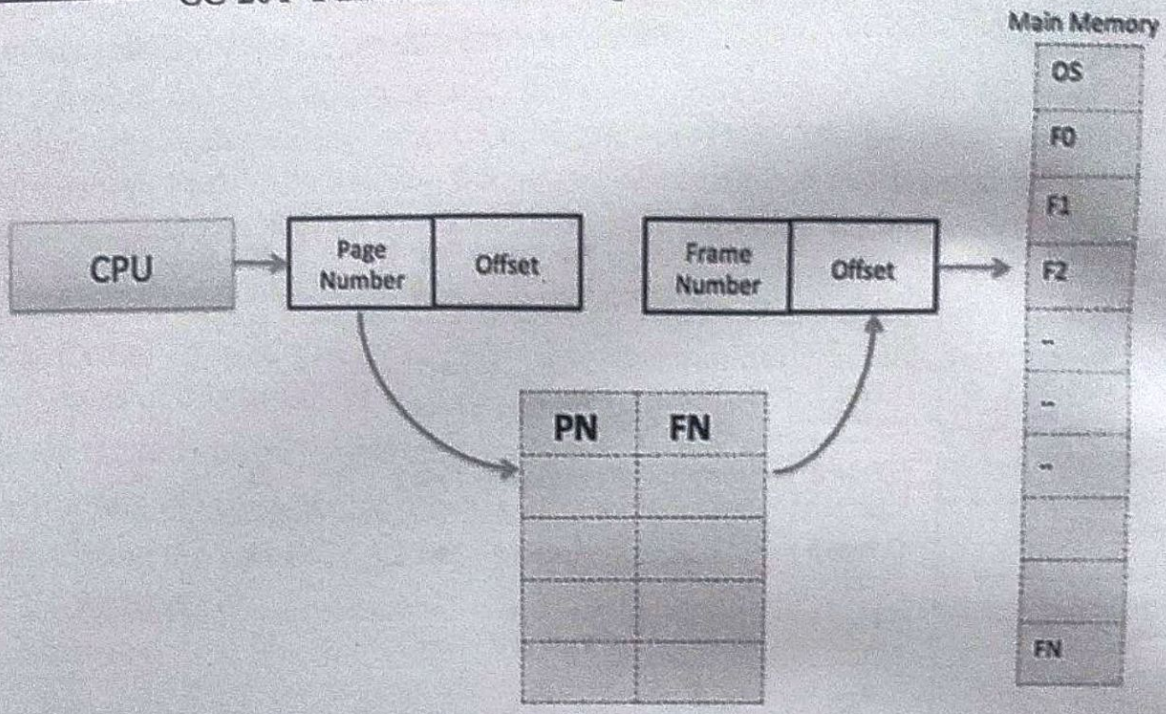


Fig.1.9 Address Translation

When the system allocates a frame to any page, it translates this logical address into a physical address and create entry into the page table to be used throughout execution of the program.

When a process is to be executed, its corresponding pages are loaded into any available memory frames. Suppose you have a program of 8Kb but your memory can accommodate only 5Kb at a given point in time, then the paging concept will come into picture. When a computer runs out of RAM, the operating system (OS) will move idle or unwanted pages of memory to secondary memory to free up RAM for other processes and brings them back when needed by the program.

This process continues during the whole execution of the program where the OS keeps removing idle pages from the main memory and write them onto the secondary memory and bring them back when required by the program.

Advantages:

Here, are advantages of using Paging method:

- Easy to use memory management algorithm
- No need for external Fragmentation
- Swapping is easy between equal-sized pages and page frames.

Disadvantages:

- May cause Internal fragmentation
- Complex memory management algorithm
- Page tables consume additional memory.
- Multi-level paging may lead to memory reference overhead.

1.3.2 Demand Paging:

"As mentioned above, the memory management function called paging specifies storage locations to the CPU as additional memory, called virtual memory. The CPU cannot directly access storage disk, so the MMU emulates memory by mapping pages to frames that are in RAM."

- Demand Paging uses a memory as a cache for the disk.
- The page table (memory map) indicates if the page is on disk or memory using a valid bit.
- Once a page is brought from disk into memory, the OS updates the page table and the valid bit.
- For efficiency reasons, memory accesses must reference pages that are in memory the vast majority of the time–Else the effective memory access time will approach that of the disk.
- Key Idea: Locality---the working set size of a process must fit in memory, and must stay there. (90/10 rule.)

Before we launch into a more detailed explanation of pages and frames, let's define some technical terms.

- **Page:** A fixed-length contiguous block of virtual memory residing on disk.
- **Frame:** A fixed-length contiguous block located in RAM; whose sizing is identical to pages.
- **Physical memory:** The computer's random access memory (RAM), typically contained in DIMM cards attached to the computer's motherboard.
- **Virtual memory:** Virtual memory is a portion of an HDD or SSD that is reserved to emulate RAM. The MMU serves up virtual memory from disk to the CPU to reduce the workload on physical memory.
- **Virtual address:** The CPU generates a virtual address for each active process. The MMU maps the virtual address to a physical location in RAM and passes the address to the bus. A virtual address space is the range of virtual addresses under CPU control.

Physical address: The physical address is a location in RAM. The physical address space is the set of all physical addresses corresponding to the CPU's virtual addresses. A physical address space is the range of physical addresses under MMU control. A demand paging system is quite similar to a paging system with swapping where processes reside in secondary memory and pages are loaded only on demand, not in advance. When a context switch occurs, the operating system does not copy any of the old program's pages out to the disk or any of the new program's pages into the main memory Instead, it just begins executing the new program after loading the first page and fetches that program's pages as they are referenced.

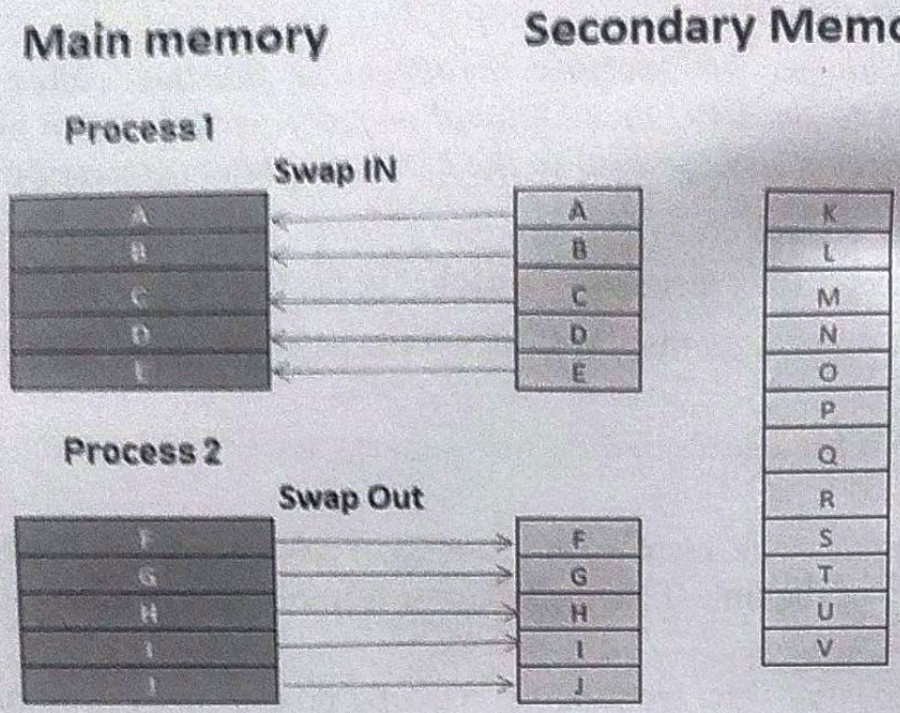


Fig.1.10 Demand Paging

A demand paging mechanism is very much similar to a paging system with swapping where processes stored in the secondary memory and pages are loaded only on demand, not in advance.

So, when a context switch occurs, the OS never copy any of the old program's pages from the disk or any of the new program's pages into the main memory. Instead, it will start executing the new program after loading the first page and fetches the program's pages, which are referenced.

During the program execution, if the program references a page that may not be available in the main memory because it was swapped, then the processor considers it as an invalid memory reference. That's because the page fault and transfers send control back from the program to the OS, which demands to store page back into the memory.

Advantages:

Following are the advantages of Demand Paging –

- On the programmer level, paging is a transparent function and does not require intervention.
- No external fragmentation.
- No internal fragmentation on updated OS's.
- Frames do not have to be contiguous.
- Large virtual memory.
- More efficient use of memory.
- There is no limit on degree of multiprogramming.

Disadvantages:

- Number of tables and the amount of processor overhead for handling page interrupts are greater than in the case of the simple paged management techniques.

- Paging causes internal fragmentation on older systems.
- Longer memory lookup times than segmentation; remedy with TLB memory caches.

1.3.3 Page Replacement Algorithms:

Page Replacement Algorithm:

Page replacement algorithms are the techniques using which an Operating System decides which memory pages to swap out, write to disk when a page of memory needs to be allocated. Paging happens whenever a page fault occurs and a free page cannot be used for allocation purpose accounting to reason that pages are not available or the number of free pages is lower than required pages.

When the page that was selected for replacement and was paged out, is referenced again, it has to read in from disk, and this requires for I/O completion. This process determines the quality of the page replacement algorithm: the lesser the time waiting for page-ins, the better is the algorithm.

A page replacement algorithm looks at the limited information about accessing the pages provided by hardware, and tries to select which pages should be replaced to minimize the total number of page misses, while balancing it with the costs of primary storage and processor time of the algorithm itself. There are many different page replacement algorithms. We evaluate an algorithm by running it on a particular string of memory reference and computing the number of page faults.

Types of Page Replacement Methods:

Here, are some important Page replacement methods

- FIFO
- Optimal Algorithm
- LRU Page Replacement

➢ **FIFO Page Replacement:**

FIFO (First-in-first-out) is a simple implementation method. In this method, memory selects the page for a replacement that has been in the virtual address of the memory for the longest time.

Features:

- Whenever a new page loaded, the page recently comes in the memory is removed. So, it is easy to decide which page requires to be removed as its identification number is always at the FIFO stack.
- The oldest page in the main memory is one that should be selected for replacement first.
- Easy to implement, keep a list, replace pages from the tail and add new pages at the head.

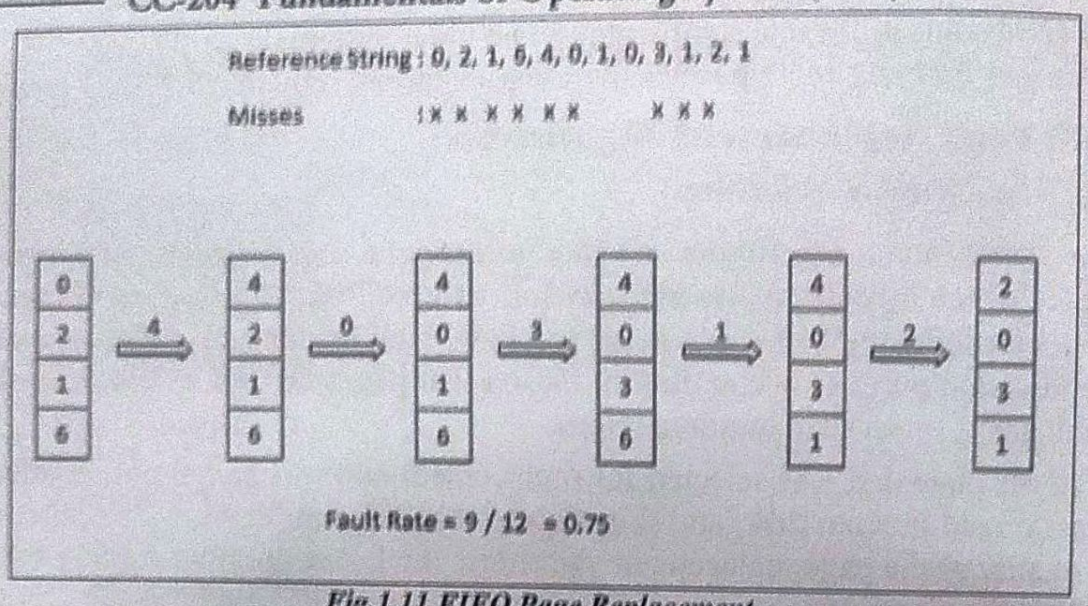


Fig.1.11 FIFO Page Replacement

➢ Optimal Algorithm:

The optimal page replacement method selects that page for a replacement for which the time to the next reference is the longest.

Features:

- Optimal algorithm results in the fewest number of page faults. This algorithm is difficult to implement.
- An optimal page-replacement algorithm method has the lowest page-fault rate of all algorithms. This algorithm exists and which should be called MIN or OPT.
- Replace the page which unlike to use for a longer period of time. It only uses the time when a page needs to be used.

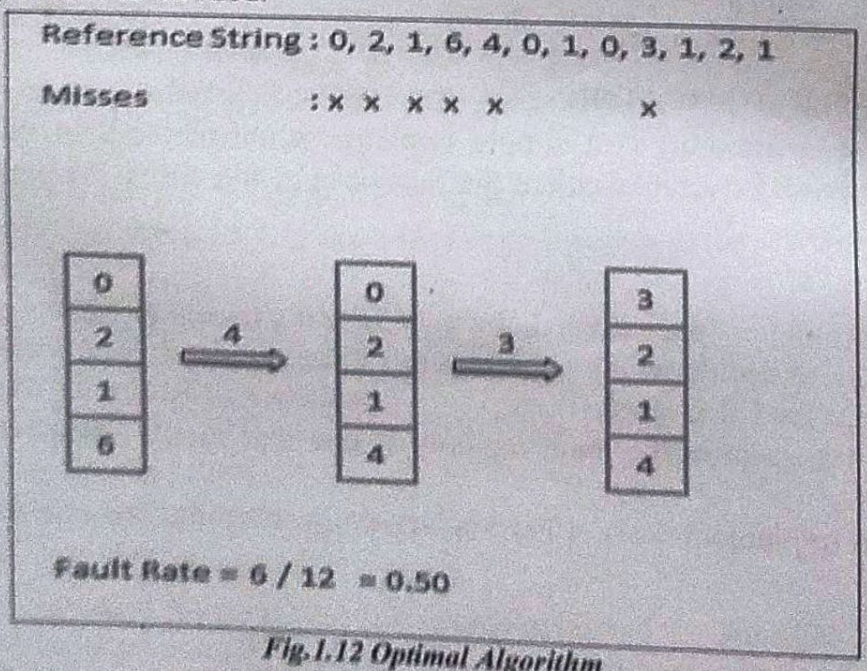


Fig.1.12 Optimal Algorithm

➢ **Least Recently Used (LRU) algorithm:**

The full form of LRU is the Least Recently Used page. This method helps OS to find page usage over a short period of time. This algorithm should be implemented by associating a counter with an even- page.

- Page, which has not been used for the longest time in the main memory, is the one that will be selected for replacement.
- Easy to implement, keep a list, replace pages by looking back into time.

Features:

- The LRU replacement method has the highest count. This counter is also called aging registers, which specify their age and how much their associated pages should also be referenced.
- The page which hasn't been used for the longest time in the main memory is the one that should be selected for replacement.
- It also keeps a list and replaces pages by looking back into time.

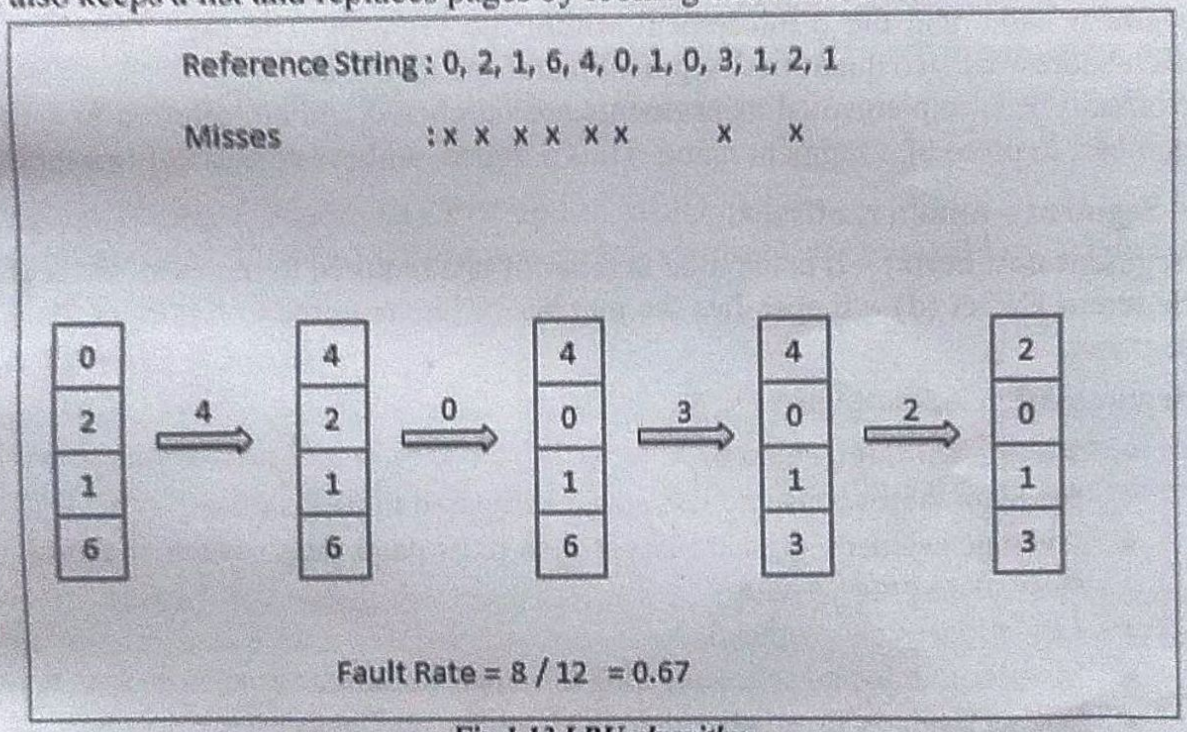


Fig.1.13 LRU algorithm

1.3.4 Segmented Memory Allocation:

"Segmentation is a memory management technique which supports user's view of memory. This technique of division of a computer's primary memory into sections called segments."

The process known as *segmentation* is a virtual process that creates address spaces of various sizes in a computer system, called segments. Each segment is a different virtual address space that directly corresponds to process objects.

When a process executes, segmentation assigns related data into segments for faster processing. The segmentation function maintains a segment table that includes physical addresses of the segment, size, and other data.

Types of Segmentation:

1. **Virtual memory segmentation:** Each processor job is divided into several segments, It is not essential all of which are resident at any one point in time.
2. **Simple segmentation:** Each process is divided into many segments, and all segments are loaded into the memory at run time, but not necessarily contiguously.

Basic method for Segmentation:

In a computer system using segmentation, a logical address space can be viewed as multiple segments. The size of the segment may grow or shrink that is it is of variable length.

During execution, each segment has a name and a length. The address specifies both the segment name and the displacement within the segment. The user, therefore, specifies each address by two quantities; segment name and an offset.
Normally it is implemented as segments are numbered and are referred to by a segment number, in place of a segment name. Thus a logical address consists of two tuples:

< Segment – number, offset >

Segment number(s) – It is the total number of bits required to represent the segment.
Segment Offset (d) – It specifies the number of bits required to represent the size of the segment.

Segmentation Advantages:

- No internal fragmentation.
- Segment tables consume less space compared to page tables.
- Average segment sizes are larger than most page sizes, which allows segments to store more process data.
- Less processing overhead.
- Simpler to relocate segments than to relocate contiguous address spaces on disk.
- Segment tables are smaller than page tables, and takes up less memory.

Segmentation Disadvantages:

- Uses legacy technology in x86-64 servers.
- Linux only supports segmentation in 80x86 microprocessors: states that paging simplifies memory management by using the same set of linear addresses.
- Porting Linux to different architectures is problematic because of limited segmentation support.
- Requires programmer intervention.

1.3.5 Segmented/ Demand Paged Memory Allocation:

Key Differences: Paging and Segmentation:

Size:

- **Paging:** Fixed block size for pages and frames. Computer hardware determines page/frame sizes.
- **Segmentation:** Variable size segments are user-specified.

Fragmentation:

- **Paging:** Older systems were subject to internal fragmentation by not allocating entire pages to memory. Modern OS's no longer have this problem.
- **Segmentation:** Segmentation leads to external fragmentation.

Tables:

- **Paging:** Page tables direct the MMU to page location and status. This is a slower process than segmentation tables, but TLB memory cache accelerates it.
- **Segmentation:** Segmentation tables contain segment ID and information, and are faster than direct paging table lookups.

Availability:

- **Paging:** Widely available on CPUs and as MMU chips.
- **Segmentation:** Windows servers may support backwards compatibility, while Linux has very limited support.

1.3.6 Virtual Memory:

Virtual Memory is a storage mechanism which offers user an illusion of having a very big main memory. It is done by treating a part of secondary memory as the main memory. In Virtual memory, the user can store processes with a bigger size than the available main memory.

Therefore, instead of loading one long process in the main memory, the OS loads the various parts of more than one process in the main memory. Virtual memory is mostly implemented with demand paging and demand segmentation.

- Virtual memory is a technique which allows pages or segments moving between main memory and secondary storage.
- Virtual memory is based on paging and/or segmentation
- Information sharing can be implemented without a large I/O overhead
    - Multiple users, each has a copy of shared information
    - With virtual memory, a single copy is loaded on demand
- Virtual memory works well in a multi programming environment

How Virtual Memory Works?

In the modern world, virtual memory has become quite common these days. It is used whenever some pages require to be loaded in the main memory for the execution, and the memory is not available for those many pages.

So, in that case, instead of preventing pages from entering in the main memory, the OS searches for the RAM space that are minimum used in the recent times or that are not referenced into the secondary memory to make the space for the new pages in the main memory.

Let's understand virtual memory management with the help of one example.

For example:

Let's assume that an OS requires 300 MB of memory to store all the running programs. However, there's currently only 50 MB of available physical memory stored on the RAM.

- The OS will then set up 250 MB of virtual memory and use a program called the Virtual Memory Manager (VMM) to manage that 250 MB.
- So, in this case, the VMM will create a file on the hard disk that is 250 MB in size to store extra memory that is required.
- The OS will now proceed to address memory as it considers 300 MB of real memory stored in the RAM, even if only 50 MB space is available.
- It is the job of the VMM to manage 300 MB memory even if just 50 MB of real memory space is available.

Advantages of Virtual Memory:

Here, are pros/benefits of using Virtual Memory:

- Virtual memory helps to gain speed when only a particular segment of the program is required for the execution of the program.
- It is very helpful in implementing a multiprogramming environment.
- It allows you to run more applications at once.
- It helps you to fit many large programs into smaller programs.
- Common data or code may be shared between memory.
- Process may become even larger than all of the physical memory.
- Data / code should be read from disk whenever required.
- The code can be placed anywhere in physical memory without requiring relocation.
- More processes should be maintained in the main memory, which increases the effective use of CPU.
- Each page is stored on a disk until it is required after that, it will be removed.
- It allows more applications to be run at the same time.
- There is no specific limit on the degree of multiprogramming.
- Large programs should be written, as virtual address space available is more compared to physical memory.

Disadvantages of Virtual Memory:

Here, are drawbacks/cons of using virtual memory:

- Applications may run slower if the system is using virtual memory.
- Likely takes more time to switch between applications.
- Offers lesser hard drive space for your use.
- It reduces system stability.
- It allows larger applications to run in systems that don't offer enough physical RAM alone to run them.
- It doesn't offer the same performance as RAM.
- It negatively affects the overall performance of a system.
- Occupy the storage space, which may be used otherwise for long term data storage.

Exercises

1. What is an operation system? List out and explain few common functions of an operation system.
2. Write a note on components of an operating system.
3. Define the terms: multiprocessing, multitasking, multithreading
4. List out and explain various types of operating system.
5. Describe various types of memory allocation schemes in brief.
6. Write a note on single user contiguous scheme with its merits and demerits.
7. Explain static/fixed sized and dynamic/variable sized partitions.
8. Explain contiguous, linked and indexed allocation.
9. Discuss relocatable dynamic partitions.
10. Write a note on paged memory allocation.
11. Discuss demand paging.
12. What is the role of page replacement algorithms? Discuss any one page replacement algorithm.
13. Write a note on FIFO page replacement algorithm with its merits and demerits.
14. Write a note on LRU page replacement algorithm with its merits and demerits.
15. Write a note on optimal page replacement algorithm with its merits and demerits.
16. Compare various page replacement algorithms.
17. Write a note on segmented memory allocation. Discuss its advantages and disadvantages over paged memory allocation.
18. Differentiate between paging and segmentation.
19. Write a note on virtual memory. How it works?
20. List out advantages and disadvantages of virtual memory.

MCQs/Fill in the Blanks/True-False:

1. The _____ helps you to communicate with the computer without knowing how to speak the computer's language. **(Operating System)**
2. The Software is the Non-Touchable Parts of the Computer. **(True)**
3. Android, iOS, Mac OS X, Microsoft Windows, Linux are the examples of _____. **(Operating System)**
4. _____ is a type of system software that communicates with the hardware and allows other programs to run. **(Operating System)**

5. _____ manages all the User and system Process. **(Process Management)**
6. _____ provides basic-level control over all of the computer hardware devices. **(Kernel)**
7. _____ component allows interaction with the user, which may occur through graphical icons and a desktop or through a command line. **(User Interface)**
8. CLI stands for _____, while GUI stands for _____. **(Command Line Interface, Graphical User Interface)**
9. _____ component allows application developers to write modular code. **(API)**
10. _____ operating system capable of supporting and utilizing more than one computer processor. **(Multiprocessing)**
11. _____ operating system that is capable of allowing multiple software processes to run at the same time. **(Multitasking)**
12. _____ operating systems that allow different parts of a program to run concurrently. **(Multithreading)**
13. To speed the same process, a job with a similar type of needs are _____ together and run as a group. **(Batched)**
14. The user of a _____ operating system never directly interacts with the computer. **(Batch)**
15. _____ operating system enables people located at a different terminal (shell) to use a single computer system at the same time. **(Time-sharing)**
16. The processor time (CPU) which is shared among multiple users is termed as _____. **(Time-sharing)**
17. _____ refers to a computer system's ability to support more than one process (program) at the same time. **(Multiprocessing)**
18. _____ operating systems enable several programs to run concurrently. **(Multiprocessing)**
19. _____ is one of the most widely used multiprocessing systems. **(UNIX)**
20. A _____ operating system time interval to process and respond to inputs is very small. **(Real time)**

21. Military Software Systems and Space Software Systems are the examples of _____ operating system. **(Real time)**
22. _____ systems use many processors located in different machines to provide very fast computation to its users. **(Distributed)**
23. _____ Operating System runs on a server that provides the capability to serve to manage data, user, groups, security, application, and other networking functions. **(Network)**
24. _____ operating systems are those OS which is especially that are designed to power smartphones, tablets, and wearable's devices. **(Mobile)**
25. Android, iOS, BlackBerry are the examples of _____ operating system. **(Mobile)**
26. _____ memory allocation is a memory allocation method that allocates a single contiguous section of memory to a process or a file. **(Contiguous)**
27. In _____ technique, the main memory is divided into partitions of equal or different sizes. **(Fixed partition)**
28. In _____ technique, the partition size is not declared initially. It is declared at the time of process loading. **(Dynamic partition)**
29. _____ is a variable size partitioning scheme. **(Dynamic partitioning)**
30. _____ is a process by which computer programs and services are assigned with physical or virtual memory space. **(Memory allocation)**
31. _____ is the process of reserving a partial or complete portion of computer memory for the execution of programs and processes. **(Memory allocation)**
32. In _____ scheme, a file is made from the contiguous set of blocks on the disk. **(Contiguous allocation)**
33. In the _____ allocation, sequential and direct access both are supported. **(Contiguous)**
34. Contiguous allocation method suffers internal as well as external fragmentation. **(True)**
35. In _____ scheme, disk blocks are arranged in the linked list form which is not contiguous. **(Linked allocation)**

36. In linked allocation scheme, there is large no of seeks because the file blocks are randomly distributed on disk. **(True)**
37. In _____ scheme, the index block contains the pointer to all the blocks occupied by a file. **(Indexed allocation)**
38. In relocatable dynamic partition, memory manager in operating system relocates the program that is all empty blocks are gathered to form one single block of large memory enough to accommodate some or all of the jobs waiting in the queue. **(True)**
39. _____ is a storage mechanism that allows OS to retrieve processes from the secondary storage into the main memory in the form of pages. **(Paging)**
40. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called _____. **(Frames)**
41. The size of a frame should be kept the same as that of a page to have maximum utilization of the main memory and to avoid external fragmentation. **(True)**
42. Page address is called logical address and represented by _____ and the _____. **(Page number, Offset)**
43. Frame address is called physical address and represented by a _____ and the _____. **(Frame number, Offset)**
44. A data structure called _____ is used to keep track of the relation between a page of a process to a frame in physical memory. **(Page map table)**
45. The CPU cannot directly access storage disk, so the MMU emulates memory by mapping pages to frames that are in RAM. **(True)**
46. Demand Paging uses a memory as a cache for the disk. **(True)**
47. The page table (memory map) indicates if the page is on disk or memory using a valid bit. **(True)**
48. _____ is a fixed-length contiguous block located in RAM; whose sizing is identical to pages. **(Frame)**
49. _____ algorithms are the techniques using which an Operating System decides which memory pages to swap out, write to disk when a page of memory needs to be allocated. **(Page replacement)**

50. While executing a program, if the program references a page which is not available in the main memory because it was swapped out a little ago, the processor treats this invalid memory reference as a _____ and transfers control from the program to the operating system to demand the page back into the memory. **(Page fault)**
51. In _____ page replacement method, memory selects the page for a replacement that has been in the virtual address of the memory for the longest time. **(FIFO)**
52. The _____ page replacement method selects that page for a replacement for which the time to the next reference is the longest. **(Optimal)**
53. In _____ page replacement method, a page which has not been used for the longest time in the main memory, is the one that will be selected for replacement. **(LRU)**
54. The process known as segmentation is a virtual process that creates address spaces of various sizes in a computer system, called _____. **(Segments)**
55. In Operating Systems, _____ is a memory management technique in which, the memory is divided into the variable size parts. **(Segmentation)**
56. The details about each segment are stored in a table called as _____ table. **(Segment)**
57. _____ memory is a storage mechanism which offers user an illusion of having a very big main memory. **(Virtual)**
58. Virtual memory is a technique which allows pages or segments moving between main memory and secondary storage. **(True)**
59. Virtual memory helps to gain speed when only a particular segment of the program is required for the execution of the program. **(True)**
60. Virtual memory allows larger applications to run in systems that don't offer enough physical RAM alone to run them. **(Ture)**

Unit – 2 Processor Management

Introduction

In single user system, the processor is busy when user's job is in execution, rest of the time it is idle. But modern operating systems are multiprogramming or multitasking systems.A multiprogramming or multitasking OS is a system executing many processes concurrently. Multiprogramming requires that the processor be allocated to each process for a period of time and de-allocated at an appropriate moment. In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called **process scheduling**. An Operating System does the following activities for processor management –

- Keeps tracks of processor and status of process.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

Before discussing process scheduling, lets define some terms.

Program: A Program is a set of instructions. It is an inactive unit, such as file stored on a disk.

Process: A Process is a program in execution. It is an active unit which requires a set of resources, including a processor and special registers to perform its function. It is also called a **task**.

Thread: A Thread is a portion of process that can run independently. It is a lightweight process.

Processor: Also called the CPU, is the part of machine that performs the calculations and executes the program.

Process scheduling is a task of operating system to schedule the processes of different states like ready, running, waiting.Process scheduling ensures maximum utilization of central processing unit (CPU) because a process is always running at the specific instance of time. At first, the process that are to be executed are placed in a queue called **Job queue**. The processes which are already placed in the main memory and are ready for CPU allocation, are placed in a queue called **Ready queue**. If the process is waiting for input / output, then that process is placed in the queue called **Device queue**.

2.1 Job Schedulers:

An operating system uses a program scheduler to schedules the processes of computer system. **Schedulers** are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types –

- High-Level Scheduler

- Low-Level Scheduler
- Middle-Level Scheduler

➢ High Level Scheduler

It is also known as **Job-scheduler** or **Long-Term scheduler**. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

Job scheduler is only concerned with selecting jobs from a queue of incoming jobs and placing them in the process queue, whether batch or interactive based on each jobs characteristics. The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

➢ Process Scheduler / Low-Level Scheduler

It is also known as **Process Scheduler** or **Short-term Scheduler** or **CPU scheduler**. After a job has been placed in the ready queue by the job scheduler, the work of process scheduler starts. Following are the main functions of Process Scheduler

- Process scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.
- It determines which job will get the CPU, when, and for how long.
- It also decides when processing should be interrupted.
- It determines which queues the job should be moved to during its execution.
- It recognizes when a job has concluded and should be terminated.

To schedule CPU, Process Scheduler uses common trait among most computer programs: they alternate between CPU cycles and I/O cycles.

```
{
printf("\nEnter the first integer: ");
scanf("%d", &a);
printf("\nEnter the second integer: ");
scanf("%d", &b);
```

} I/O cycle

```
c = a+b
d = (a*b)-c
e = a-b
f = d/e
```

} CPU cycle

```
printf("\n a+b= %d", c);
printf("\n (a*b)-c = %d", d);
printf("\n a-b = %d", e);
printf("\n d/e = %d", f);
}
```

} I/O cycle

Although the duration and frequency of CPU cycles vary from program to program, there are some general tendencies that can be exploited when selecting a scheduling algorithm. For example, I/O-bound jobs (such as printing a series of documents) have many brief CPU cycles and long I/O cycles, whereas CPU-bound jobs (such as finding the first 300 prime numbers) have long CPU cycles and shorter I/O cycles.

➢ Middle Level Scheduler

It is also known as **Medium-Term Scheduler**. In a highly interactive environment there is also a third layer of the Processor Manager called Middle-Level scheduler. Sometimes when system is overloaded, it removes the processes from the memory. Thus,It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.The processes that are swapped out and eventually swapped back in are managed by the middle-level scheduler. The swapping of processes is performed to ensure the best utilization of main memory.

2.2 Job and Process Status:

When a process executes, it passes through different states. These are called the job status or the process status. Following are the various states of a process in its life time.

1. HOLD

When a job is accepted by the system it is put on Hold state and placed in a queue. In some systems, the job spooler (or disk controller) creates a table with the characteristics of each job in the queue and notes the important features of the job, such as an estimate of CPU time, priority, special I/O devices required, and maximum memory required. This table is used by the Job Scheduler to decide which job is to be run next.

2. READY

From HOLD, the job moves to READY when it's ready to run but is waiting for the CPU. The process is waiting to be assigned to a processor. The transition from HOLD to READY is initiated by the job scheduler.

3. RUNNING

Once the process has been assigned to a processor by the Process scheduler, the process state is set to running and the processor executes its instructions. The transition from READY to RUNNINGand from RUNNING to READY is handled by the Process Scheduler.

4. Waiting

Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available. The transition from RUNNING to WAITING is handled by the Process Scheduler. If the process is in Waiting state, it then moves to READY state before resuming its execution.The transition from WAITING to READY is handled by the Process Scheduler and is initiated by a signal from the I/O device manager that the I/O request has been satisfied and the job can continue.

5. FINISHED.

Once the process finishes its execution, or it is terminated by the operating system, it is moved to the FINISHED state where it waits to be removed from main memory.Eventually, the transition from RUNNING to FINISHED is initiated by the Process Scheduler or the Job Scheduler either when

- the job is successfully completed and it ends execution or
- the operating system indicates that an error has occurred and the job is being terminated prematurely

Figure: A typical process changes status as it moves through the system from HOLD to FINISHED.

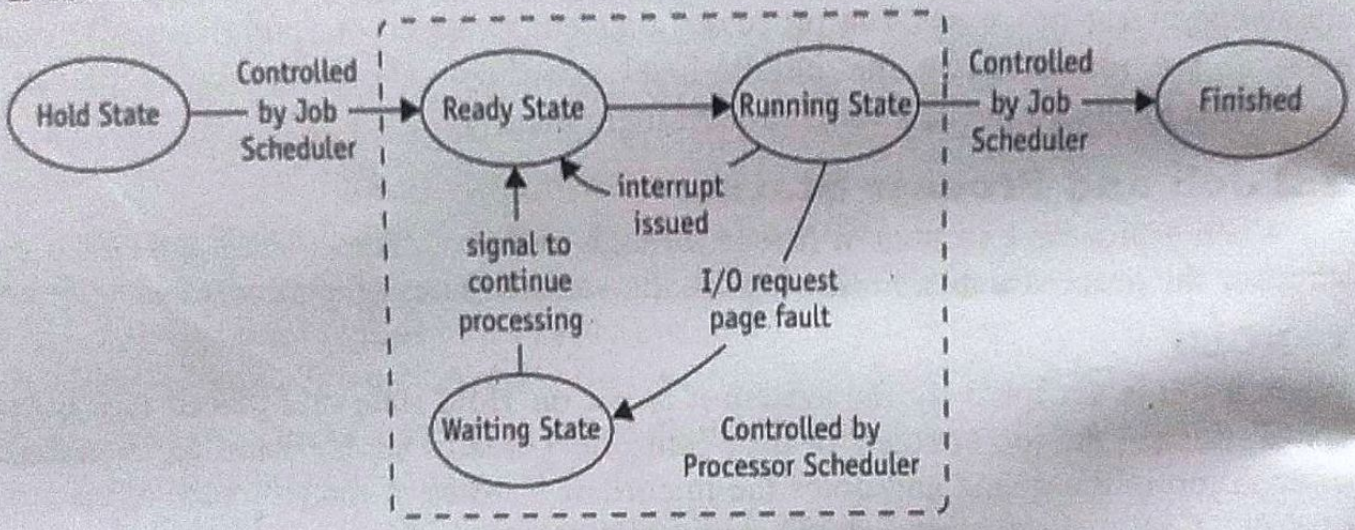


Fig.2.1System from Hold to Finished

2.3 Process Control Blocks:

A Process Control Block is a data structure maintained by the Operating System for every process. The PCB is identified by an integer process ID (PID). A PCB keeps all the information needed to keep track of a process.

Figure: Contents of each jobs Process Control Block

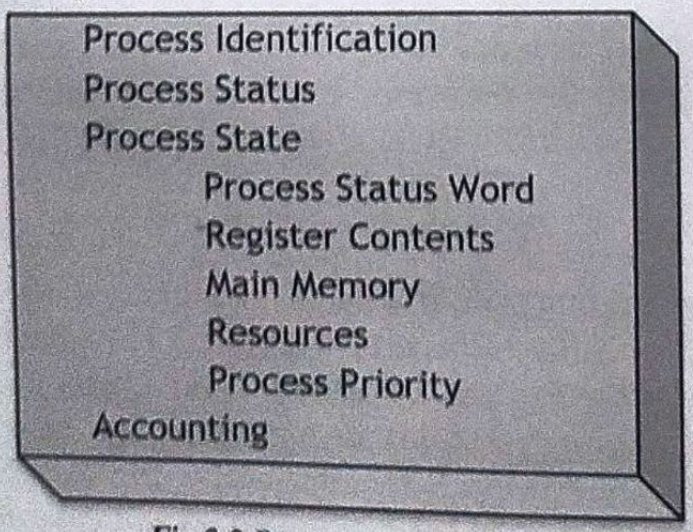


Fig.2.2 Process Control Blocks

The process control block typically contains:

Process Identification

Each job is uniquely identified by the user's identification and a pointer connecting it to its descriptor.

Process Status

This indicates the current status of the job—HOLD, READY, RUNNING, or WAITING—and the resources responsible for that status.

Process State

This contains all of the information needed to indicate the current state of the job such as:

- **Process Status Word**—the current instruction counter and register contents when the job is not in running state, but is either on HOLD or is READY or WAITING.
- **Register Contents**—the contents of the register if the job has been interrupted and is waiting to resume processing.
- **Main Memory**—pertinent information, including the address where the job is stored and, in the case of virtual memory, the mapping between virtual and physical memory locations.
- **Resources**—information about all resources allocated to this job. Each resource has an identification field listing its type and a field describing details of its allocation, such as the sector address on a disk. These resources can be hardware units (disk drives or printers, for example) or files.
- **Process Priority**—used by systems using a priority scheduling algorithm to select which job will be run next.

Accounting

Accounting information includes

- Amount of CPU time used from beginning to end of its execution.
- Total time the job was in the system until it exited.
- Main storage occupancy—how long the job stayed in memory until it finished execution.
- Secondary storage used during execution.
- System programs used, such as compilers, editors, or utilities.
- Number and type of I/O operations, including I/O transmission time, that includes utilization of channels, control units, and devices.
- Time spent waiting for I/O completion.

- Number of input records read (specifically, those entered online or coming from optical scanners, card readers, or other input devices), and number of output records written.

2.4 Process Scheduling Policies:

In multiprogramming environment, many processes are being executed simultaneously. Hence, to schedule these processes is the task of Processor manager. A scheduling policy is required by a multiprocessor operating system to allocate available time and processors to a job or a process, either statically or dynamically. Scheduling of processes is done on basis of following criteria.

- **Maximize CPU Utilization**
  CPU would be working most of the time (Ideally 100% of the time).CPU utilization should be maximum.
- **Maximize Throughput**
  Throughput is the total number of processes completed per unit of time. Run as many job as possible in a given amount of time to maximize throughput.
- **Minimize Turnaround Time**
  Turnaround time is the amount of time taken to execute a particular process, i.e. The interval from time of submission of the process to the time of completion of the process. Turnaround time should be minimum to increase system efficiency.
- **Minimize Waiting Time**
  Waiting Time is the amount of time a process has been waiting in the ready queue to acquire get control on the CPU. Waiting time of process is to be reduced.
- **Minimize Response Time**
  Response time is the time taken from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution. Response time should be minimum.
- **Ensure fairness to all jobs.**
  Give everyone an equal amount of CPU and I/O time.This could be done by not giving special treatment to any job, regardless of its processing characteristics or priority.

In general CPU utilization and Throughput are maximized and other factors are minimized for proper optimization.

Although the Job Scheduler selects jobs to ensure that the READY and I/O queues remain balanced, there are instances when a job claims the CPU for a very long time before issuing an I/O request. If I/O requests are being satisfied, this extensive use of the CPU will build up the READY queue while emptying out the I/O queues, which creates an unacceptable imbalance in the system. To solve this problem, the Process Scheduler

often uses a timing mechanism and periodically interrupts running processes when a predetermined slice of time has expired. When that happens, the scheduler suspends all activity on the job currently running and reschedules it into the READY queue. The CPU is now allocated to another job that runs until one of three things happens:

- the timer goes off
- the job issues an I/O command
- or the job is finished.

Then the job moves to the READY queue, the WAIT queue, or the FINISHED queue, respectively. An I/O request is called a **natural wait** in multiprogramming environments which allows the processor to be allocated to another job. Hence, we can say that scheduling can be preemptive or non preemptive.

- **Preemptive Scheduling Policies**
  A scheduling strategy that interrupts the processing of a job and transfers the CPU to another job is called a preemptive scheduling policy. It is widely used in time-sharing environments.

- **Non-Preemptive Scheduling Policies**
  Under non-preemptive scheduling, once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.

2.5 Process Scheduling Algorithms:

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. These algorithms are either **non-preemptive or preemptive**.

Here are the process scheduling algorithms that have been widely used.

1. First Come, First Served

First Come, First Served (FCFS), is a non-preemptive scheduling algorithm. As the name itself suggests, the processes that arrive first, are executed first. In short, the process that request the CPU first, gets CPU allocated first. It is simple to implement, as it uses FIFO (First In First Out) queue.This algorithm is used in Batch Systems. It's easy to understand and implement programmatically, using a Queue data structure, where a new process enters through the **tail** of the queue, and the scheduler selects process from the **head** of the queue.

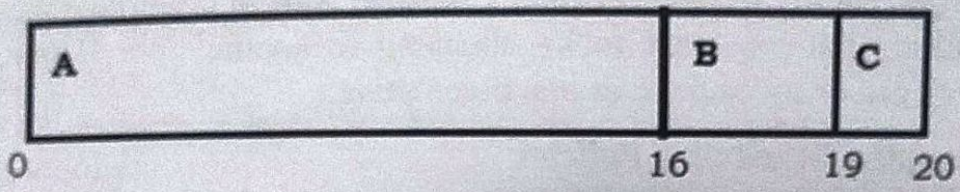
In a strictly FCFS system there are no WAIT queues ,although there may be systems in which control is switched on a natural wait (I/O request) and then the job resumes on I/O completion.

The following examples presume a strictly FCFS.

Consider there are three processes, A, B and C. All of these arrive at the same time 0 and the CPU Burst time is given.

Process	Burst Time
A	16
B	3
C	1

Using an FCFS algorithm with an arrival sequence of A, B, C, the time line (Gantt Chart) is shown as below.



Turnaround time and waiting time for processes will be as follows.

Process	Turnaround Time	Waiting Time
A	16	0
B	19	16
C	20	19

Therefore,

Average turnaround time = (16+19+20)/3 = 18.33 ms

Average waiting time = (0+16+19)/3 = 11.67 ms

FCFS is the simplest and easiest to implement algorithm .But the greatest drawback of first-come, first-served scheduling is that it is not preemptive. Because of this, it is not suitable for interactive jobs. Another drawback is that a long-running process will delay all jobs behind it.

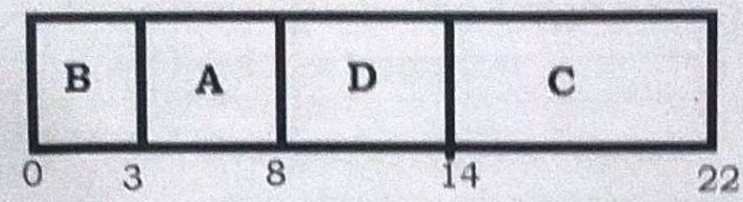
2. Shortest Job Next

It is a non-preemptive scheduling algorithm. Process which have the shortest burst time are scheduled first. If two processes have the same burst time then FCFS is used to break the tie.This algorithm can't be implemented in an interactive system because it requires advance knowledge of the CPU time required to finish each job. It is easier to implement in batch environment.

Consider there are four processes, A, B, C and D. All of these arrive at the same time and the CPU Burst time is given.

Process	Burst Time
A	5
B	3
C	8
D	6

The timeline is given below for the above processes



Turnaround time and waiting time for processes will be as follows.

Process	Turnaround Time	Waiting Time
A	8	3
B	3	0
C	22	14
D	14	8

Therefore,

Average turnaround time = (8+3+22+14)/4 =47/4= 11.75 ms

Average waiting time = (3+0+14+8)/4 =25/4= 6.25 ms

3. Priority Scheduling

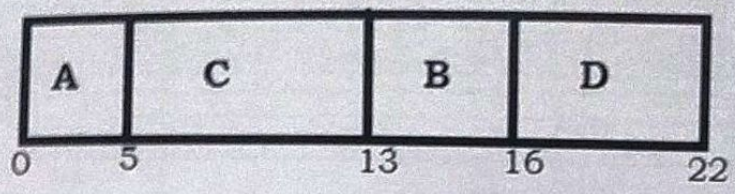
Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned a priority. Process with highest priority is to be executed first and so on. Processes with same priority are executed on first come first served basis.Priorities can also be determined by the Processor Manager based on

- Memory requirements. Jobs requiring large amounts of memory could be allocated lower priorities than those requesting small amounts of memory, or vice versa.
- Number and type of peripheral devices. Jobs requiring many peripheral devices would be allocated lower priorities than those requesting fewer devices.
- Total CPU time. Jobs having a long CPU cycle, or estimated run time, would be given lower priorities than those having a brief estimated run time.
- Amount of time already spent in the system. This is the total amount of elapsed time since the job was accepted for processing. Some systems increase the priority of jobs that have been in the system for an unusually long time to expedite their exit. This is known as **aging.**

Consider there are four processes, A, B, C and D. All of these arrive at the same time 0, their CPU Burst time and Priorities are is given. Here 1 is considered as the highest priority. Note: when A and C have same priorities, hence A will be executed first breaking the tie on FCFS basis

Process	Burst Time	Priority
A	5	1
B	3	2
C	8	1
D	6	3

The timeline is given below for the above processes



Turnaround time and waiting time for processes will be as follows.

Process	Turnaround Time	Waiting Time
A	5	0
B	16	13
C	13	5
D	22	16

Therefore,

Average turnaround time = (5+16+13+22)/4 =56/4= 14 ms

Average waiting time = (0+13+5+16)/4 =34/4= 8.5 ms

4. Shortest Remaining Time

Shortest remaining time (SRT) is the preemptive version of the SJN algorithm.Th processor is allocated to the job closest to completion but it can be preempted by a newe ready job with shorter time to completion. It is not possible to implement in interactiv systems where required CPU time is not known.It is often used in batch environment where short jobs need to be given preference.

Consider there are four processes, A, B, C and D.

Process	Burst Time	Arrival Time
A	6	0
B	3	1
C	1	2
D	4	3

The timeline is given below for the above processes

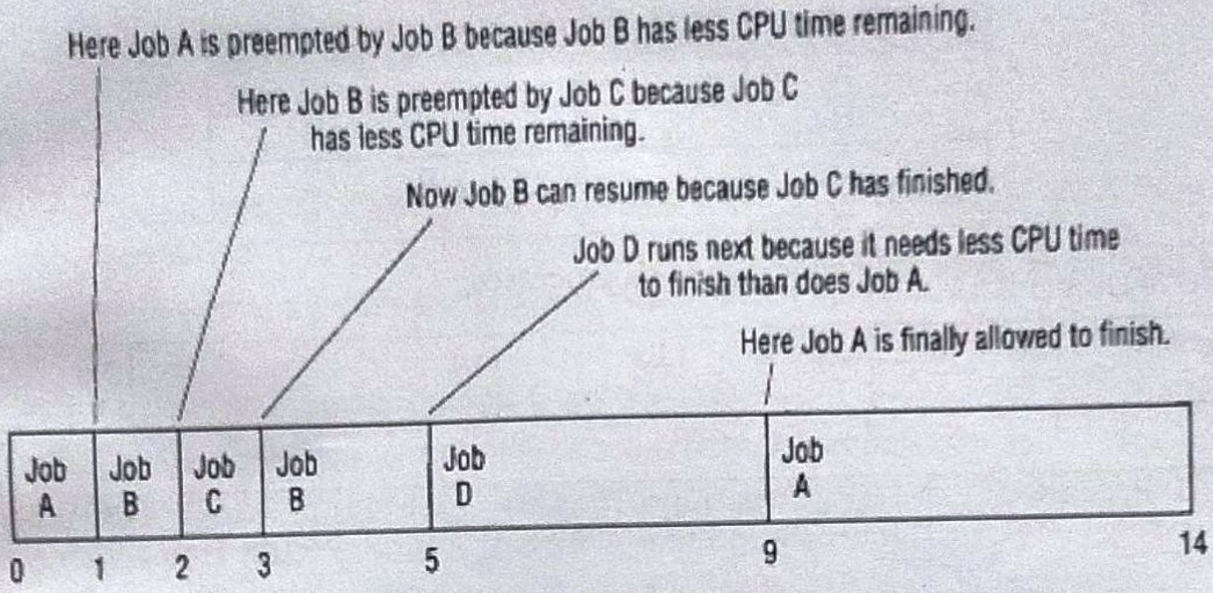


Fig.2.3 Shortest Remaining Time

Turnaround time and waiting time for processes will be as follows.

Process	Turnaround Time (Completion Time – Arrival Time)	Waiting Time (Turnaround Time – Burst Time) – Arrival Time
A	14	8
B	4	1
C	1	0
D	6	2

Average turnaround time = (14+4+1+6)/4 =25/4=6.25 ms

Average waiting time = (8+1+0+2)/4 =11/4= 2.75 ms

1. Round Robin

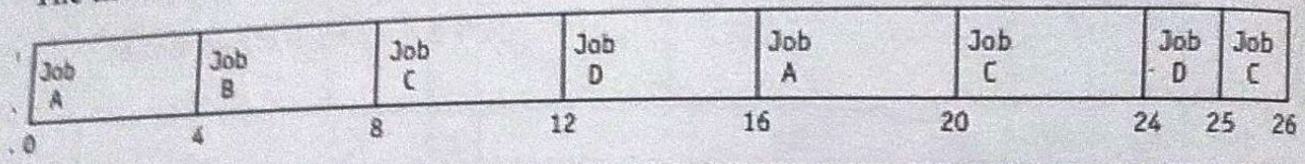
Round robin is a preemptive process scheduling algorithm that is used extensively in interactive systems. It's easy to implement and isn't based on job characteristics but on a predetermined slice of time that's given to each job to ensure that the CPU is equally shared among all active processes and isn't monopolized by any one job. This time slice is called a **time quantum** or **time slice** and its size is crucial to the performance of the system. It usually varies from 100 milliseconds to 1 or 2 seconds.

Jobs are placed in the READY queue using a first-come, first-served scheme and the Process Scheduler selects the first job from the front of the queue, sets the timer to the time quantum, and allocates the CPU to this job. If processing isn't finished when time expires, the job is preempted and put at the end of the READY queue and its information is saved in its PCB.

Consider there are four processes, A, B, C and D.

Process	Burst Time	Arrival Time
A	8	0
B	4	1
C	9	2
D	5	3

The timeline is given below for the above processes



Process	Turnaround Time (completion time – Arrival Time)	Waiting Time (Turnaround time – Burst Time) – Arrival Time
A	20	12
B	7	3
C	24	15
D	22	17

Average turnaround time = (20+7+24+22)/4 =73/4=18.25 ms

Average waiting time = (12+3+15+17)/4 =47/4= 11.75 ms

In the above example, Job A was preempted once because it needed 8 milliseconds to complete its CPU cycle, while Job B terminated in one time quantum. Job C was preempted twice because it needed 9 milliseconds to complete its CPU cycle, and Job D was preempted once because it needed 5 milliseconds. In their last execution or swap int memory, both Jobs D and C used the CPU for only 1 millisecond and terminated before their last time quantum expired, releasing the CPU sooner.

Conclusion:

In this chapter we have learnt that the Processor Manager must allocate the CPU amon all the system's users. Here we have learnt different states of process, and the concept o Process Control Block (PCB). We have seen how job scheduling is done, the selectio of incoming jobs based on their characteristics, and process scheduling, the instant-by- instant allocation of the CPU. There are several process scheduling algorithms like FCFS, SJN, PriorityScheduling, SRT and Round Robin. The Process Scheduler relies on a process scheduling algorithm, based on a specific policy, to allocate the CPU and move jobs through the system.

☺ ☺ ☺ ☺ ☺

Exercises

Review Questions

1. What is meant by Process Scheduling? State and explain different types of schedulers.
2. Explain the different states of process. Also explain state transition in detail with diagram.
3. Write a detailed note on Process Control Block.
4. What are the criteria's for deciding process scheduling policy?
5. Explain : Preemptive and Non-Preemptive process scheduling
6. Given the following information

Process	Burst Time	Arrival Time
A	10	0
B	12	1
C	3	2
D	1	3
E	15	4

Draw a time-line for each of the following algorithm.

a) FCFS
b) SJN
c) SRT
d) Round-Robin (Time quantum – 5ms)

Also find the average turn around time and average waiting time for each of the following algorithm.

7. Given the following

Process	Burst Time	Arrival Time
A	8	0
B	4	1
C	9	2
D	5	3

Draw a time-line for each of the following algorithm.

a) FCFS
b) SJN

Exercises

Review Questions

1. What is meant by Process Scheduling? State and explain different types of schedulers.
2. Explain the different states of process. Also explain state transition in detail with diagram.
3. Write a detailed note on Process Control Block.
4. What are the criteria's for deciding process scheduling policy?
5. Explain : Preemptive and Non-Preemptive process scheduling
6. Given the following information

Process	Burst Time	Arrival Time
A	10	0
B	12	1
C	3	2
D	1	3
E	15	4

Draw a time-line for each of the following algorithm.

a) FCFS
b) SJN
c) SRT
d) Round-Robin (Time quantum – 5ms)

Also find the average turn around time and average waiting time for each of the following algorithm.

7. Given the following

Process	Burst Time	Arrival Time
A	8	0
B	4	1
C	9	2
D	5	3

Draw a time-line for each of the following algorithm.

a) FCFS
b) SJN

c) SRT

Also find the average turnaround time and average waiting time for each of the following algorithm.

8. Consider the following information

Process	Burst Time	Priority
A	10	3
B	1	1
C	2	4
D	1	5
E	5	2

Draw a time-line for each of the following algorithm.

a) FCFS

b) SJN

c) Priority Scheduling (Assuming 1 as the highest priority)

Also find the average turnaround time and average waiting time for each of the following algorithm.

Multiple Choice Questions

1. _____ is a set of instruction.

(a) Program (c) Thread
(b) Process (d) None of the above.

2. A _____ is a program in execution.

(a) Program (c)Thread
(b) Process (d)None of above

3. A _____ is a portion of process that can run independently

(a) Program (c) Thread
(b) Process (d) None of above

4. The processes which are already placed in the main memory and are ready for CPU allocation, are placed in a queue called _____.

(a) Job Queue (c) Device Queue
(b) Ready Queue (d) None of the above

5. _____ is also known as Job-scheduler.

(a) Process Scheduler
(b) Middle level Scheduler
(c) Low-level scheduler
(d) High-level Scheduler

6. _____ selects a process among the processes that are ready to execute and allocates CPU to one of them.

(a) Process Scheduler
(b) Middle level Scheduler
(c) Low-level scheduler
(d) High-level Scheduler

7. A _____ is a data structure maintained by the Operating System for every process.

(a) Thread Control Block
(b) Process Control Block
(c) Both (a) and (b)
(d) None of the above

8. _____ is the total number of processes completed per unit time

(a) Throughput
(b) Turnaround Time
(c) Waiting Time
(d) Response time

9. _____ is the amount of time taken to execute a particular process.

(a) Throughput
(b) Turnaround Time
(c) Waiting Time
(d) Response time

10. _____ is the amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

(a) Throughput
(b) Turnaround Time
(c) Waiting Time
(d) Response time

Answers

1. (a)	2. (b)	3. (c)	4. (b)	5. (d)
6. (a)	7. (b)	8. (a)	9. (b)	10. (c)

Solved Examples of Process Scheduling Algorithms

1. Given the following information. The below set of processes arrive at time zero (0). Burst Time in given in millisecond (ms).

Process	Burst Time
P1	5
P2	24
P3	16
P4	10
P5	3

Draw a time-line for each of the following algorithm.

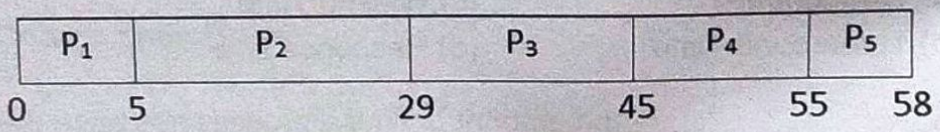
- FCFS
- SJN

Also find the average turn around time and average waiting time for each of the following algorithm.

1. Solution

➢ **FCFS**

Following is the time line (Gantt chart) for the above set of processes using FCFS.



Turnaround time and waiting time for each process are calculated as below

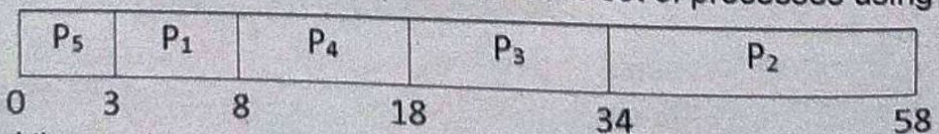
Process	Turnaround Time = Completion time – Arrival Time	Waiting Time = Turnaround time – Burst Time
P1	5-0 = 5	5-5=0
P2	29-0=29	29-24=5
P3	45-0=45	45-16=29
P4	55-0=55	55-10=45
P5	58-0=58	58-3=55

Average turnaround time = (5+29+45+55+58)/5 =192/5=38.4 ms

Average waiting time = (0+5+29+45+55)/5 =134/5= 26.8 ms

➢ **SJN**

Following is the time line (Gantt chart) for the above set of processes using SJN.



Turnaround time and waiting time for each process are calculated as below

Process	Turnaround Time = Completion time – Arrival Time	Waiting Time = Turnaround time – Burst Time
P1	8-0 = 8	8-5=3
P2	58-0=58	58-24=34
P3	34-0=34	34-16=18
P4	18-0=18	18-10=8
P5	3-0=3	3-3=0

Average turnaround time = (8+58+34+18+3)/5 =121/5=24.2 ms

Average waiting time = (3+34+18+8+0)/4 =63/5= 12.6 ms

2. Below is the list of processes with their burst time(in ms) and arrival time.

Process	Burst Time	Arrival Time	Priority
P1	21	0	2
P2	3	1	1
P3	6	2	4
P4	2	3	3

Draw a time-line for each of the following algorithm.

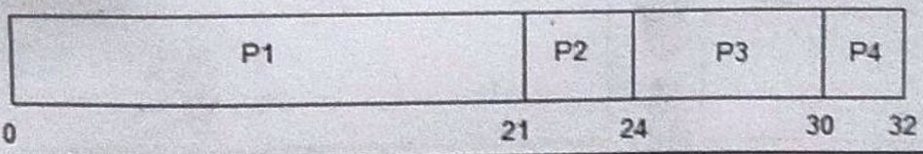
- FCFS
- SRT
- Priority Scheduling
- Round Robin (Time Quantum – 5ms)

Also find the average turn around time and average waiting time for each of the following algorithm.

2. Solution

➢ **FCFS**

Timeline for FCFS is as below



Process	Turnaround Time	Waiting Time = Turnaround time – Burst Time
P1	21	21-21=0
P2	24	24-3=21
P3	30	30-6=24
P4	32	32-2=32

Average turnaround time = (21+24+30+32)/4 =107/4=26.75 ms

Average waiting time = (0+21+24+32)/4 =77/4= 19.25 ms