The requirements are changing day-by-day. Thus, hardware is becoming more and more powerful and is able to perform complex functions. Hardware is nothing but finely designed machinery.

An operating system is a program, which acts as an interface between a user and the hardware (i.e. all computer resources).

E.g.: it is just like our secretary. As the boss gives orders to his secretary and she does all the work for his boss. The secretary herself decides: How to do? What to do? When to do? etc. same way, we pass our orders/request to OS thus to make the computer system convenient to use and secondary goal is use computer hardware in an efficient manner.

Q. **What is an operating system?**

Ans. An operating system is a control program. A control program is responsible for the execution of the programs in an efficient and proper way so as to prevent errors and improper use of the computer; especially concerned with the operation and control of the I/O devices. It controls and coordinates the use of the hardware among the various application programs for the various users.

An o. s. is an important component of a computer system, which controls all other component of the computer system. Major components of a computer system are:

1) The hardware
2) The operating system
3) The application program routine (complier, linkers, database, management systems)
4) The human ware (users).

**Device Driver:**-
Each I/O device has its own characteristics requiring careful programming. A special program routine was written for each I/O device. This special routine was called ‘device driver’.

**Note:** A device driver is a program, which controls and monitors the working of a particular I/O device. E.g. our keyboard will have a separate device driver and mouse will have a separate device driver.

The compliers for COBOL, FORTAN and other languages appeared but made the computer operations complex. The program execution consisted of many steps: loading the complier tape, running the complier, unloading the complier, unloading the complier tape, etc.... If an error occurred, the entire process has to be repeated.

**Note:** The time spent in loading, unloading, mounting etc. is called setup time.

Q. **What are the different Components of OS?**

Ans. An operating system is a program, which acts as an interface between a user and the hardware.

The major components of OS are as follows:

(i) **The Storage Manager:**-
It controls the utilization of the main storage by different user programs. It protects the program area in main storage from illegal access i.e. it checks the authority of the user to access the data. It translates the data from HLL to MLL. It terminates a user program if sufficient storage is not available for the program. The storage manager keeps the track of the deallocation storage and provides its reuse by other programs.
(ii) **The Process Manager:**

The process manager (or processor manager) is responsible for effective utilization of the processor. It keeps the track of each program’s (a) process state and (b) priority associated. Then by making use of one of the job scheduling technique, it allocates the CPU to the highest priority program. It interacts with other OS components in order to carry out the various operations.

(iii) **File System:**

It provides information management support to OS as well as to a program. It manages allocation of space for a file on a secondary storage device, when a program demands for it. It also takes care of file protection against illegal access. It is also responsible for grouping of files under directories and manages them.

(iv) **Input Output Control System:**

Actual input and output for the file data is performed through this very component, IOCS.

- **Logical IOCS** provides the access method and data organization for the data stored in files.
- **Physical IOCS** is responsible for carrying out devices level I/O operations to implement data transfer. It also optimizes devices performance by scheduling jobs waiting for the device.

Some other components of OS are:

- **The console manager** is responsible for changing the operating mode from supervisor (monitor) mode to user mode or vice versa.
- **Log on-Log off Manager** checks user’s authorization like a password or account number.

Q. **What are the types of OS?**

**Ans.**

There are different types of OS available, which requires different types of hardware to run upon.

The OS are of mainly following types:

(i) **Single Program OS:** As the name suggests, this OS is single user operating system, so only one user program can be supported and executed at a time. Example: DOS

(ii) **Multiprogramming OS:** This is a multi-user OS. It supports multi-programming i.e. more than one user can be supported by it, therefore, more than one user program are loaded and active in the main storage at the same time.

**The concept of Spooling**

Spooling is a very important data structure: a job pool. Spooling results in several jobs having been read and writing on disk, ready to run. A pool of jobs on disk allows the OS to decide which job to execute next in order to increase CPU utilization.

Multiprogramming is an attempt to increase CPU utilisation by always having something for the CPU to execute. When several jobs are waiting for the CPU, while I/O is being performed for one program, the CPU can perform computation for another program.

**Note:**

1) When two or more programs can be executed in an interleaved fashion so as to keep the CPU busy most of the time, it is called multiprogramming.

2) An I/O bound program is the one, which requires more of I/O operation than CPU time whereas a CPU bound program requires more of CPU time than I/O.
(iii) **Timesharing OS**: This OS uses the time-sharing technique. Each active user program is given a fair share of CPU time (δ), if the time elapses or an I/O operation is requested, CPU shifts over to the next job waiting and the previous job is put to wait (if the time is over).

**Note**: The delay between job submission and job completion is called **turnaround time**. A turnaround time does not improve the system utilization as multiprogramming does.

(iv) **Real time OS**:  
In real time OS, the jobs have fixed deadlines and the jobs have to be completed within their deadlines. The system performance is measured by its ability to complete the jobs within their specified deadlines. If a job cannot be completed within its deadline, this situation is called deadline overrun.

(v) **Multiprocessing OS**: The multiprocessing OS is capable of handling more than one processor, as the jobs have to be executed on more than one processor. The multiprocessing OS should be capable of load sharing in case of identical processors so that the system’s efficiency improves.

**FUNCTIONS OF OS**

The environment required for the execution of the program is provided by the operating system. The environment provided is responsible for effective and efficient execution of the program which depends upon the service provided by the OS.

- **Types of Services**
  1. Program execution
  2. Handling I/O operations
  3. Manipulation of file system
  4. Error detection and Handling
  5. Resource allocation
  6. Accounting
  7. Information and Resource Protection

**Major OS functions are as follows:-**

1. Processor Management
2. Storage (Memory) Management
3. Information Management

1. **Processor Management**: Multiprogramming improves the overall efficiency of the computer system by more CPU utilization. While CPU is executing a job, it has to wait for the job; if the job requires certain I/O operation, the CPU waits for the I/O operation to get over and that wait time is CPU’s idle time. So another job takes and over utilizes CPU’s idle time.

The benefits of multiprogramming are as follows:

i) Increased CPU utilization
ii) Higher total job throughput

**Note**: Throughput is the amount of work accomplished in a given time interval. It is calculated as:
Throughput = \frac{\text{The number of jobs completed}}{\text{total time taken to complete the jobs}}

∴ Throughput = \frac{2 \text{ jobs}}{2 \text{ min}} = 1 \text{ job/minute}

Note: When two or more tasks are simultaneously in progress (e.g. CPU working as well as the I/O operation) it is referred to as multitasking.

**Job Scheduling**: Job scheduling not only assigns priority to jobs but also admits new jobs for processing at appropriate time.

**Program** is a set of instructions given to computer. Job or process is a program in execution.

**Process State**: A process is a program in execution. During execution, the process changes its states. The state of a process is defined by its current activity.

**Process Control Block**: Each process is represented in the OS by a data block called Process Control Block (PCB). It contains following information.

(i) Process state
(ii) Program counter it indicates the address of next instruction to be executed
(iii) CPU Register: Stores system related information
(iv) Memory Limits
(v) List of open files

<table>
<thead>
<tr>
<th>Pointer</th>
<th>Process State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process number</td>
<td></td>
</tr>
<tr>
<td>Process counter</td>
<td></td>
</tr>
<tr>
<td>Register</td>
<td></td>
</tr>
<tr>
<td>Memory limits</td>
<td></td>
</tr>
<tr>
<td>List of open files</td>
<td></td>
</tr>
</tbody>
</table>

To have the best possible results, the criteria are as follows:

(a) **CPU utilization**: There should be maximum use of it.
(b) **Turnaround time**: There should be minimum possible turn-around time.
(c) **Waiting time**: waiting time should be minimum.
(d) **Response time**: System should give faster response time.
(e) **Throughput**: Throughput should be maximum possible.

There are certain scheduling techniques. They are as follows:-
a. **Non-preemptive scheduling**

In this type of scheduling job always completes before another scheduling decision is made. The scheduling techniques, which use non-preemptive scheduling, are:

(i) First Come First Served (FCFS) scheduling
(ii) Shortest Job Next (SJN) scheduling
(iii) Deadline scheduling

**Explanation:**

(i) **First Come First Served (FCFS) scheduling:**
- This is the simplest scheduling technique, which is managed by FIFO (First in First Out).
- That is, the process, which requires the CPU first, is allocated the CPU first.
- A queue is maintained called **ready queue** in which all the processes, that want CPU time, are entered.
- The CPU executes the job in the ready queue one by one.
- Batch Processing is one obvious example of FCFS scheduling in which all the jobs in the batch are executed one by one.
- The turnaround time for the very first job in the batch is the best and for the last job is the worst.
- Suppose, the last job (job 3) requires only ‘1 second’ for its execution but as to wait for 45 seconds as it is the last job. This is the major **drawback**.

E.g.

<table>
<thead>
<tr>
<th>Job</th>
<th>CPU Burst time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

Using SJN, the scheduling order of these jobs would be job4; job3; job6; job1; job2; job5.
(iii) Deadline scheduling:
- In deadline scheduling, the job with the earliest deadline is selected for scheduling.
- Deadline of a job is the time limit within which a job must be over.
- If a job exceeds deadline, it is said to be deadline overrun.
- It is calculated as:
  \[ K = C - D \]
  Where K is the deadline overrun, C is job completion time and D is the deadline for a job.

b. Preemptive Scheduling:
  In this type of scheduling, decision can be made even while the job is executing. Preemptive
  scheduling may force a job in execution to release the processor, so that execution of some other
  job can be undertaken, in order to improve throughput. The techniques, which use this scheduling,
  are:
  (i) Round Robin scheduling
  (ii) Response Ratio scheduling

Explanation:
(i) Round Robin (RR) scheduling
- RR scheduling is aimed at giving equal opportunity to make progress.
- No program is given second opportunity unless all other program has atleast one opportunity.
- A small unit of time, called a time quantum or time slice \( \delta \), is defined.
- The ready queue is treated as a circular queue. The programs in the ready queue are processed
  according to the time slice given, one by one.
- When a program’s turn comes, it can utilize the processors for duration of time slice (say \( \delta \)).
  However, if it requires I/O before elapses, it is passed over to I/O handler and CPU moves over to
  the next program.
- After the time slice is over, it puts the program in the end of the queue and moves to the next
  program.
- This way, CPU processes all the programs in the ready queue one by one until their execution is
  over.
- Once the execution is over, its output is processed and it is removed from the ready queue

(a) Round Robin Scheduling
Response Ratio Scheduling:
Response Ratio is calculated as follows:

\[ \text{Response time} = \frac{\text{Burst Time}}{\text{Execution Time} + \text{Waiting Time}} \]

- The job with highest Response Ratio is preferred over others.
- When a short job arrives, the response ratio is high, so it is scheduled for execution immediately.
- A longer job would achieve high enough ratio only after a subsequent wait.

Multilevel Queuing:
- Multilevel queuing classifies the jobs into different groups.
- It partitions the ready queue into separate queue.
- Jobs are permanently assigned to one particular queue based upon some property of the job such as memory line or deadline etc.
- Each queue is scheduled using a particular scheduling technique, which might differ from scheduling techniques for other queues.

Multi Processor scheduling:
- In processor management, we mainly focus on single processor system.
- Note: Processing of more than one job simultaneously by multiple processors is called multiprocessing.

2. Storage Management: Memory is a large array of word or bytes, each with its own address. The CPU reads from and writes to specific memory address.

A program must be assigned some memory area and loaded into memory in order to be executed. As the program executes, it access program instructions and data from memory by generating addresses of the allocated memory. When the program terminates, its memory space is declared
free, and the next program is given the same memory area. There can be many user program loaded in the memory along with the system programs and OS. How the memory is managed? is given below.

1. **Contiguous Storage Allocation**: Each program to be executed is allocated a contiguous storage memory. At every job step initiation, the OS has to ensure that sufficient contiguous area is available to accommodate it.

   There are three techniques used for multiprogrammed storage management in contiguous storage allocation system.

   (i) **Fixed Partition Allocation**:
   - In this technique, the main storage is split into different storage areas called **Partitions**.
   - The number and sizes of partitions are **fixed**, thus degree of multiprogramming is also fixed.
   - The number and sizes of partitions can be changed only when no programs are executing in the affected partitions.
   - Example: When 3 programs are executing together. If P1 has the size 40K, P2 has 35K and P3 has 60K; then the size must be >= 60K and number of partitions should be at least 3.

   ![Partition Example](image)

   - As the entire partition is allocated to one job, two parts 20K, 25K of the two partitions is wasted, as the program needs only 40K and 35K. This wastage of space is called **internal fragmentation**.
   - Drawback: One is internal fragmentation. Second is, external fragmentation, during a program execution, after the program has been assigned a partition a if its complied version (pgm after compilation) is too large to fit in the partition.

   (ii) **Visible partitioned Allocation**:
   - In this technique, the partition sizes and boundaries are not fixed.
   - Each job is allocated a partition size according to its requirements.
   - When a job finishes, the system keeps track of the released storage.
   - For a new job, a partition is created to suit its requirements.
   - This technique overcomes internal fragmentation, though external fragmentation would still persist because of the holds that are too small to be used for job initiations.
(iii) Roll in/Roll out:-

- The above techniques allow the activation of a new program only after current program terminates.
- When a High Priority Program (HPP) arrives for the execution and it requires 120K. Two programs P1 and P2 are running with 100K (60K+40K) of memory occupied.
- So, the current running programs are sent to backing store and the freed memory is given to HPP.
- After HPP finishes, memory is released and P1 and P2 are bought back to resume their execution.
- This scheme is called rolling out/rolling in or swapping.
- P1 = 60K, P2 = 40K, HPP = 120K

![Diagram of roll in/roll out](image)

P1 and P2 making rooms for HPP

![Diagram of HPP releasing memory](image)

HPP releasing memory and P1 and P2 reloaded.

2. **Non contiguous storage allocation**
   (i) **Paging**
   1. Partitions suffer from external fragmentation because of available memory is not contiguous.
   2. Paging permits a program’s memory to be non-contiguous allowing a program to be allocated physical memory wherever it is available.
   3. Physical memory is broken into fixed-size blocks called Page Frames. Logical memory is also broken into blocks of the same size called Pages.
   4. When a program is to be executed its pages are loaded into any available frames and the page map tables defined to translate from user pages to memory frames.
   5. The page size is defined by hardware. It is typically power of 2. The paging model of memory is shown as follows:
6. Every address generated by CPU is divided in two parts: a page number (p) and a page offset/displacement (d). The page number is used as an index into a PMT.

(ii) Segmentation
1. Segmentation are logical divisions of programs and hence are normally of variable sizes.
2. Segmentation is a memory management scheme which support user’s view of memory.
3. Each segment has number and length.
4. Each program in executable form can be considered to be consisting of different segments such as code, data and stack. Each of these can be further divided into new segments.
5. A program normally contains main program, some subprograms and few predefined and precompiled functions. Each of these belongs to different segments.
6. An application programmer does not necessarily have to declare different segments in the program. If various segments in his programs does not define explicitly, then the compiler does it by its own. Following are the jobs of compiler:
   (a) Recognize different segments in program.
   (b) Number those segments.
   (c) Define segment table.
   (d) Produce an executable image by assigning two dimensional addresses.
7. Consider the examples as shown in figure Four segments numbered 0 to 3. The SMT (Segment Map Table) has separate entry for each segment giving the size and base of segment.
3. **Information Management**

A computer system works on ‘information’. It stores information, processes information, provides information etc. Thus managing this information is very important and necessary tasks performed by OS. To support this functions, OS’s have one component called ‘Information management’. This information management component is structured as follows:

![Diagram of Hierarchy of Information Management Modules]

1. **Physical IOCS (Input-Output Control System)** is responsible for device management for ensuring device independence (through drivers).
2. **Logical IOCS** is responsible for efficient organization and access of data on I/O devices. It provides basic capabilities for file definition, choice of data organization and access method.
3. **File System** is responsible for protection and controlled sharing of files.

**Q. What is Terminal I/O?**

**Ans.** A terminal or Visual Display Unit (VDU) is an extremely common I/O medium. It would be hard to find any programmer or user who has not seen and used a terminal. Example: Monitor.

**TERMINAL HARDWARE:** Terminal hardware can be divided into two parts: the keyboard, which is an input medium and the video screen, which is an output medium.

The terminal can be dumb terminal or intelligent terminal.

- The **dumb terminal** has a microprocessor in it on which we can run rudimentary software.
- It has very limited memory.
- The dumb terminal is responsible for the basic input and output of characters.
- Even then, it is called ‘dumb’ because it does no processing on the input characters. Example: normal monitor.
• As against this, the intelligent terminal can also carry out some processing (e.g. validation) on the input.
• This requires a more powerful hardware and software for it. Example: ATM machines.

THE GRAPHICAL USER INTERFACE (GUI) SYSTEM

Q. What is GUI? Explain its features.
Ans. We see many pictorial signboards of parking instructions or a hospital in the vicinity. Software developers borrowed this idea and decided to make the OS flexible, fast and user-friendly.

• The pictorial representations are called pictures rather than representations.
• The OS interact with the help of pictures rather than commands.
• Graphical User Interface (GUI) environment gives a flexible working environment to the user for interacting with the hardware.
• The working of the software and hardware is completely transparent to the user in GUI.
• It is fast and easy to use because entire system is picture based.
• It interacts through icons. Icons are pictorial representations of a program.
• The four characteristics of icons are:
  ✓ Visual: the appearance of the icon.
  ✓ Textual: the name below the icon.
  ✓ Operational: the program that is executed when the icon is activated.
  ✓ Positional: the location of the icon on the screen.
• It is multi-tasking. The multi-tasking facility of the GUI allows more than one program to be loaded in the memory.
• For Example: MS- Windows is a GUI environment.
• The GUI environment allows easy communication of data between programs.
• The step-by-step procedure to complete a job is called wizard. In this, the user does not have to remember a set of steps or commands to complete a job.

Q. What are the basic components of the GUI System?
Ans. The working of GUI environment is different from the character based operating system environment. Some components of the GUI environment are described below:

1. Desktop:- The area on the screen, which displays various components of a GUI.
2. Windows:- Every group under a GUI OS has a number of icons. These are enclosed in a frame called a window. Windows separate one group from the other. An icon can also represent a program group. If more than one program is active or opened by a user, the number of windows on the desktop will be equal to the number of programs running. Many programs open their windows in their own window. This is called a child window.
3. Menu Bar:- There is a menu bar on the top of the display, which shows the various menu options. The menu and menu bar are displayed on the top of the desktop.
4. Menu:- The most common way for the user to interact in the GUI environment is to use menu options. Every GUI environment has at least one menu containing options or commands. Menus hold a list of commands that are usually related in some way. Example: the window menu of
Window Explorer contains two groups of related commands indicated by a solid line drawn between them.

5. **Menu Options:** Menu options are displayed as a list when the user selects the menu. These options are commands or utilities that the system provides. To execute a command, it has to be selected from the menu.

The GUI environment has an additional input device called mouse. When any pictorial representation is pointed at, one of the following three actions can follow:

- **Click once,** press the mouse button & release it. This click will only select the program item. It will not activate it.
- **Double click,** press & release the mouse button twice, with a fairly short time between the two clicks. This selects and activates the program item.
- **Drag,** press the mouse button and while continuing to hold it down you move the mouse. The keyboard can also be used to activate the programs and select options.

6. **Toggle:** Some commands are preceded by a check mark as ‘Save settings on exit’

7. **Dialog Box:** They are the standard way to tell the program exactly how you want it to act. They are used to choose options for the commands.

8. **List Box:** If an item has many choices, it becomes difficult to show all these options on the display. The list box contains more than one option, which can be traversed by the scroll bar.

9. **Text Box:** If a choice is not provided amongst the pre-defined choices in list boxes, option buttons, etc. then the choice has to be entered. The place to insert the option is called text box.

10. **Radio button:** When only one item from the group (generally a dialog box) is to be selected, then these options are in the form of radio buttons. The radio buttons, which is selected by default, is enclosed in a dotted circle.

11. **Control Menu:** Every Window opened in an environment has a control menu, which has menu options to maximize, minimize or size a window. It also has option to close the window.

### Some GUI Operating Systems

- Windows-NT
- Open Server UNIX operating system
- OS/2
- Novel operating system (NOS) version 4.1

### Some GUI Packages

- MS-Office
- Oracle Power Objects
- Power Builders
- FoxPro Under Windows (ver.2.6)
Operating System

Worksheet - I

1. What is operating system?
2. What is device driver?
3. Explain different types of operating system.
4. What are the different services of operating system?
5. What is GUI? Explain its components.