1. **What is OS? Functions of OS? Different types.**

An **operating system** (**OS**) is [system software](https://en.wikipedia.org/wiki/System_software) that manages [computer hardware](https://en.wikipedia.org/wiki/Computer_hardware), [software](https://en.wikipedia.org/wiki/Computer_software) resources, and provides common [services](https://en.wikipedia.org/wiki/Daemon_(computing)) for [computer programs](https://en.wikipedia.org/wiki/Computer_program). It acts as an interface between the user and computer hardware.

Functions:-

* Controls the peripherals such as scanners and printers.
* **Process management**: Process management helps OS to create and delete processes. It also provides mechanisms for synchronization and communication among processes.
* **Memory management:** Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.
* **File management**: It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.
* Maintains security and access rights of users.

1. What is kernel, monolithic kernel?

The kernel is **a core component of an operating system and serves as the main interface between the computer's physical hardware and the processes running on it**.

* 1. Monolithic Kernel- A monolithic kernel is an operating system architecture where the entire operating system is working in kernel space. The execution of the monolithic kernel is quite fast as the services such as memory management, file management, process scheduling, etc., are implemented under the same address space. Eg. microsft windows, linux.
  2. Micro Kernel- It is stable, has less services in the kernel space and puts rest in user space. This means that it makes use of both the kernel space and user. Eg. mac os x and windows n t.
  3. Hybrid Kernel- With the speed and design of a monolithic kernel and the modularity and stability of a microkernel, this type is a combination of monolithic and micro kernel.

Eg. windows nt.

* 1. Exo Kernel- This kernel follows an end-to-end principle with least hardware abstractions.
  2. Nano Kernel- This type offers hardware abstraction without system services.

1. process vs program vs thread. Different types of thread.

Program is an executable file containing the set of instructions written to perform a specific job on our computer.

In the Operating System, a Process is something that is currently under execution. So, an active program can be called a Process. For example, **when you want to search something on web then you start a browser**. So, this can be process.

|  |
| --- |
| Thread means a segment of a process. |

For example, in a notepad program, one thread will be taking user inputs and another thread will be printing a document.

* 1. **User Level thread (ULT) –** Is implemented in the user level library, they are not created using the system calls. Thread switching does not need to call OS and to cause interrupt to Kernel.

**Eg.** Java Threads.

* 1. **Kernel Level Thread-** The kernel-level thread is implemented by the operating system. The kernel knows about all the threads and manages them.

**Eg.** Window Solaris.

1. Process Control Block

There is a Process Control Block for each process, enclosing all the information about the process. It is also known as the task control block. It is a data structure, which contains the following: (9)

* **Process State**: It can be running, waiting, etc.
* **Process ID** and the **parent process ID**.
* CPU registers-This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.
* Program Counter. **Program Counter** holds the address of the next instruction to be executed for that process.
* **CPU Scheduling** information: Such as priority information and pointers to scheduling queues.
* **Memory Management information**: For example, page tables or segment tables.
* **Accounting information**: The User and kernel CPU time consumed, account numbers, limits, etc.
* **I/O Status information**: Devices allocated, open file tables, etc.

States of processes:-

* NEW- The process is being created.
* READY- The process is waiting to be assigned to a processor.
* RUNNING- Instructions are being executed.
* WAITING- The process is waiting for some event to occur(such as an I/O completion or reception of a signal).
* TERMINATED- The process has finished execution.

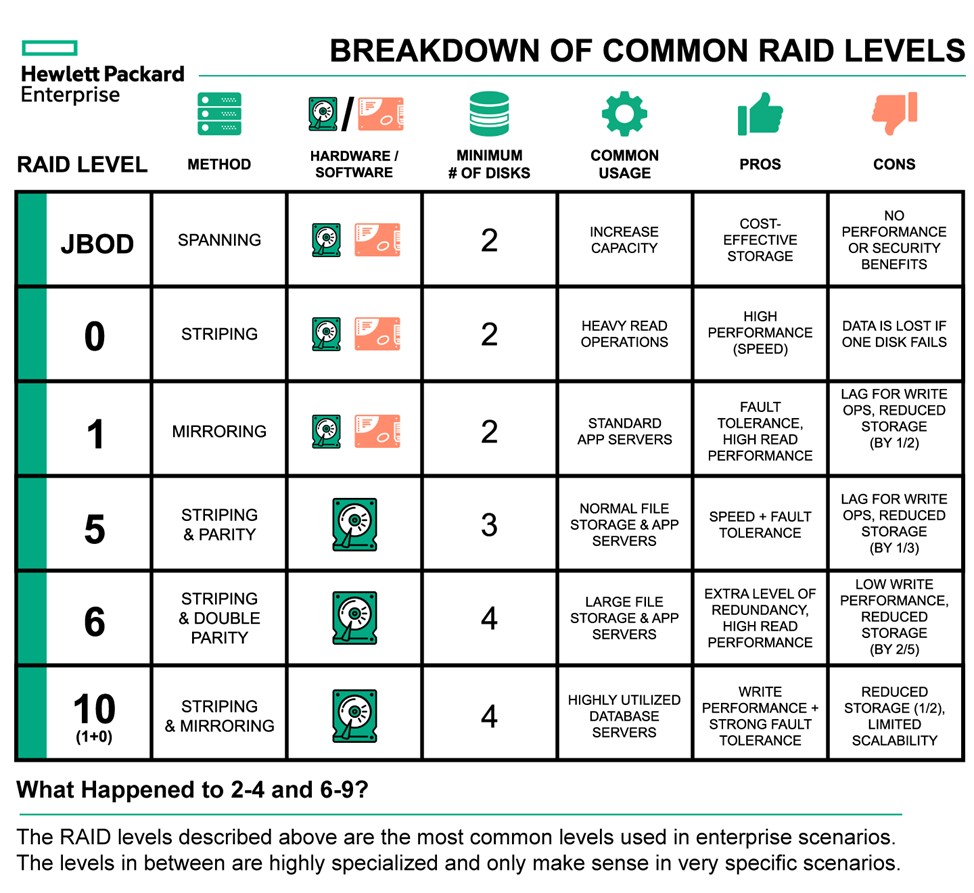
Context Switching **involves storing the context or state of a process so that it can be reloaded when required and execution can be resumed from the same point as earlier**.

1. What is Raid?

RAID or redundant array of independent disks is a data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for data redundancy, performance improvement, or both.

RAID works by placing data on multiple disks and allowing input/output operations to overlap in a balanced way, improving performance. Because various disks increase the mean time between failures (MTBF), storing data redundantly also increases fault tolerance.

Types of raid?



1. Difference between multiprocessing and multitasking.

|  |  |
| --- | --- |
|  | The availability of more than one processor per system, that can execute several set of instructions in parallel is known as multiprocessing. |

Multiprogramming is **multiple programs running simultaneously on one CPU**; multitasking is multiple tasks (processes) running simultaneously on one CPU, and multithreading is multiple threads (light weight processes) running simultaneously on one CPU.

|  |  |  |  |
| --- | --- | --- | --- |
| The processing is slower, as a single job resides in the main memory while execution. | Multitasking follows the concept of context switching. | It allows a single process to get multiple code segments. | A large amount of work can be done in a short period of time. |

Multiprogramming Multitasking Multithreading MP

 When you open your Web browser and then open Word at the same time, you are causing the operating system to do multitasking.

1. Different types of operating systems.
   1. Batch OS: -

The batch operating system does not have a direct link with the computer. A different system divides and allocates similar tasks into batches for easy processing and faster response. Batch operating systems are used for tasks such as managing payroll systems, data entry and bank statements.

* 1. The time-sharing operating system, also known as a multitasking OS, works by allocating time to a particular task and switching between tasks frequently. Examples of time-sharing operating systems include Multics and Unix.
  2. This system is based on autonomous (self-organisable) but interconnected computers communicating with each other via communication lines or a shared network. Distributed operating systems are used for tasks such as telecommunication networks, airline reservation controls.
  3. Network operating systems are installed on a server providing users with the capability to manage data, user groups and applications. Examples of network operating systems include Microsoft Windows server 2003 and 2008, Linux and Mac OS X.
  4. **Real-Time Operating System –**   
     The time interval required to process and respond to inputs is very small. This time interval is called **response time**.

A real-time operating system is an operating system for real-time applications that processes data and events which have critically defined time constraints.

**Real-time systems** are used when time constraints are very strict like missile systems, air traffic control systems, robots, etc.

**Hard Real-Time Systems:**   
These OSs are meant for applications where time constraints are very strict and even the shortest possible delay is not acceptable. These systems are built for saving life like automatic parachutes or airbags which are required to be readily available in case of any accident.

**Soft Real-Time Systems:**

These OSs are for applications where for time-constraint is less strict. **Examples of Real-Time Operating Systems are:** weapon systems, robots, air traffic control systems, etc.

1. Scheduling Algorithms: -

Scheduling algorithms schedule processes on the processor in an efficient and effective manner. It maximizes CPU utilization by increasing throughput.

**Preemptive Scheduling** is a CPU scheduling technique that works by dividing time slots of CPU to a given process. The time slot given might be able to complete the whole process or might not be able to it.

**Non-preemptive Scheduling** is a CPU scheduling technique the process takes the resource (CPU time) as long as it gets terminated or is pushed to the waiting state.

**SCHEDULING CRITERIA**

**CPU utilisation –**   
The main objective of any CPU scheduling algorithm is to keep the CPU as busy as possible.

**Throughput –**   
A measure of the work done by CPU is the number of processes being executed and completed per unit time. This is called throughput.

**Turnaround Time (TAT):**

1. Difference b/w Completion Time and Arrival Time of process is called as its Turnaround Time.

**Waiting Time (WT):**

1. The time spent by a process waiting in the ready queue for getting the CPU.
2. The time difference b/w Turnaround Time and Burst Time is called Waiting Time. (means itna time tak process extra gya but burst time tak hi tha kam to)

**Response Time**

Response time is the time spent when the process is in the ready state and gets the CPU for the first time.

* 1. FCFS
  2. **First come first serve** (FCFS) scheduling algorithm simply schedules the jobs according to their arrival time.
  3. Advantages: -

Simple, easy to understand.

* 1. Disadvantages: -

Due to the non-preemptive nature of the algorithm, the problem of starvation may occur.

* 1. SJF

SJF is **an algorithm which schedules processes according to their burst time, process having smallest burst time would be executed first.**

Advantages

SJF is frequently used for long term scheduling.

It reduces the average waiting time over FIFO (First in First Out) algorithm.

1. Disadvantages

This algorithm may cause starvation.

Requires knowledge of how long a process or job will run.

* 1. SRTF

This Algorithm is the **preemptive version** of **SJF scheduling**. In SRTF, the execution of the process can be stopped after certain amount of time.

**Advantages:**   
SRTF algorithm makes the processing of the jobs faster than SJF algorithm.

**Disadvantages:**   
The context switch is done a lot more times in SRTF than in SJF, and consumes CPU’s valuable time for processing.

* 1. Round Robin

Round Robin is the **pre-emptive process scheduling algorithm**. In this algorithm a process is executed for a fixed time period called time quantum then it gets pre-empted and another process gets executed for same time period and gets prompted and so on.

* It doesn’t face the issues of starvation or convoy effect.
* All the jobs get a fair allocation of CPU.

Disadvantages

* This method spends more time on context switching
* Its performance heavily depends on time quantum.
  1. Priority Scheduling

**Priority Scheduling** is a scheduling algorithm that is based or priority of processes, scheduler executes the processes which have higher priority.

Processes are executed on the basis of priority so high priority does not need to wait for long which saves time.

* If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time.
  1. Multi-Level Queue

In this scheduling algorithm multiple algorithms with common characteristics come together to form a group and then schedule jobs as a whole.

* 1. Multi-Level Feed Back

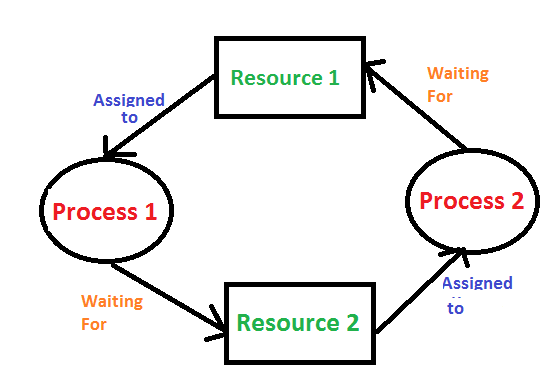
This scheduling algorithm is similar to multilevel queue scheduling except that the processes here can change their queue too i.e., if a process is in queue1, then after partial execution, it can switch to queue2.

1. Deadlock

**DEADLOCK AND SUFFICENT CONDITIONS**

***Deadlock***is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

Process 1 is holding Resource 1 and waiting for resource 2 which is acquired by process 2, and process 2 is waiting for resource 1.



**Deadlock can arise if**the **following four conditions hold simultaneously (Necessary Conditions)**  
***Mutual Exclusion:*** One or more than one resource are non-shareable (Only one process can use at a time)   
***Hold and Wait:***A process is holding at least one resource and waiting for resources.   
***No Preemption:*** A resource cannot be taken from a process unless the process releases the resource.   
***Circular Wait:*** A set of processes are waiting for each other in circular form.

**DEADLOCK HANDLING METHODS**

## **1. Deadlock Ignorance**

Deadlock Ignorance is the most widely used approach among all the mechanisms. In this approach, the Operating system assumes that deadlock never occurs. It simply ignores deadlock. It is also known as ostrich approach.

## **2. Deadlock prevention**

Deadlock happens only when Mutual Exclusion, hold and wait, No preemption and circular wait holds simultaneously. If it is possible to violate one of the four conditions at any time then the deadlock can never occur in the system.

**3. Deadlock Avoidance-**

* In this method we maintain a set of data using which a decision is made whether to take the new request or not.
* If entertaining the new request causes the system to move in an unsafe state, then it is discarded.
* Banker’s Algorithm is a deadlock avoidance algorithm.
* It maintains a set of data using which it decides whether to entertain the request of any process or not.
* It follows the safety algorithm to check whether the system is in a safe state or not.

**4. Deadlock Detection and Recovery**

## **Deadlock Detection and Recovery-**

* This strategy involves waiting until a deadlock occurs.
* After deadlock occurs, the system state is recovered.

**Resource Allocation Graphs:-**

Deadlock is detected using resource allocation graphs algorithms.

Resource Allocation Graph (RAG) is a graph that represents the state of a system pictorially.

Process Synchronization (Just Definition Not Coding)

1. **WHAT IS PROCESS SYNCHRONIZATION AND ITS NEED?**

* **Process Synchronization** is the task of coordinating the execution of processes in a way that no two processes can have access to the same shared data and resources.
* When multiple processes are running together, and more than one processes try to gain access to the same shared resource or data at the same time. This can lead to the inconsistency of shared data. So the change made by one process not necessarily reflected when other processes accessed the same shared data. To avoid this type of inconsistency of data, the processes need to be synchronized with each other.
* **TYPES OF PROCESSES ON THE BASIS OF PROCESS SYNCHRONIZATION**

**A)Independent Process** : Execution of one process does not affects the execution of other processes.

* + - 1. **Cooperative Process** : Execution of one process affects the execution of other processes.

1. **RACE CONDITION**

A Race condition is a scenario that occurs in a multithreaded environment due to multiple threads sharing the same resource or executing the same piece of code. If not handled properly, this can lead to an undesirable situation, where the output state is dependent on the order of execution of the threads.

* + 1. **CRITICAL SECTION**

1. **Definition**

Critical section is a code segment that can be accessed by only one process at a time. Critical section contains shared variables which need to be synchronized to maintain consistency of data variables.

1. **Solution’s requirements:-**

In Critical section problem:

No assumptions may be made about speeds or the number of CPUs.

**Mutual Exclusion**- No two processes may be simultaneously inside their critical sections.

**Progress**- Processes running outside its critical section can't block other processes getting enter into critical section.

**Bounded Waiting**- Processes do not wait forever to enter its critical section.

1. **Various Solutions Available For Critical Section**
2. **Semaphores And Mutex**
   1. **Semaphores vs Mutex**

* A **mutex** is a binary variable whose purpose is to provide locking mechanism. It is used to provide mutual exclusion to a section of code.
* There is misconception that binary **semaphore** is same as mutex variable but both are different in the sense that binary semaphore apart from providing locking mechanism also provides two atomic operation signal and wait, means after releasing resource semaphore will provide signalling mechanism for the processes who are waiting for the resource.
* Mutex is used for thread but semaphore is used for process.
  1. **Semaphores**
* Semaphore is an integer variable which is used to solve the critical section problem and to achieve process synchronization in the multiprocessing environment.
* **Binary Semaphore –**   
  This is also known as mutex lock. It can have only two values – 0 and 1. Its value is initialized to 1. It is used to implement the solution of critical section problems with multiple processes.
* **Counting Semaphore –**   
  Its value can range over an unrestricted domain. It is used to control access to a resource that has multiple instances.

1. **Classical IPC Problems: -**

Producer Consumer Problem- The **Producer-Consumer problem** is a classic synchronization problem in operating systems.

The problem is defined as follows: there is a fixed-size buffer and a Producer process, and a Consumer process.

The **Producer** process creates an item and adds it to the shared buffer. The **Consumer** process takes items out of the shared buffer and “consumes” them.

**Memory Mana gement and Left**

**Cache Memory** is a special very high-speed memory. It is used to speed up and synchronizing with high-speed CPU. Cache memory is costlier than main memory or disk memory but economical than CPU registers.

Dynamic binding or late binding is **the mechanism a computer program waits until runtime to bind the name of a method called to an actual subroutine**. It is an alternative to early binding or static binding where this process is performed at compile-time.

**1. Primary / Main memory:**  
Primary memory is the computer memory that is directly accessible by CPU. It is comprised of DRAM and provides the actual working space to the processor. It holds the data and instructions that the processor is currently working on.

**2. Secondary Memory / Mass Storage:**  
The contents of the secondary memory first get transferred to the primary memory and then are accessed by the processor, this is because the processor does not directly interact with the secondary memory.

* 1. In a multiprogramming computer, the operating system resides in a part of memory and the rest is used by multiple processes. The task of subdividing the memory among different processes is called memory management.

### Why Memory Management is required:

* To Allocate and de-allocate memory before and after process execution.
* To keep track of used memory space by processes.
* To minimize fragmentation issues.
* To proper utilization of main memory.
  1. **Logical Address** is generated by CPU while a program is running. The logical address is virtual address as it does not exist physically, therefore, it is also known as Virtual Address. This address is used as a reference to access the physical memory location by CPU. Logical address space can be defined as the size of the process. The size of the process should be less enough so that it can reside in the main memory.

The hardware device called Memory-Management Unit is used for mapping logical address to its corresponding physical address.

**Physical Address** identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address. Physical address space in a system can be defined as the size of the main memory.

* 1. Fragmentation

Fragmentation is **an unwanted problem in the operating system in which the processes are loaded and unloaded from memory, and free memory space is fragmented**.

In non-contiguous memory allocation, **different parts of a process is allocated different places in Main Memory**.

In the Contiguous Memory Allocation, **each process is contained in a single contiguous section of memory**.

**Non-contiguous memory allocation** is of different types,

1. Paging
2. Segmentation
3. Segmentation with paging

**Internal fragmentation:**

In this fragmentation, the process is allocated a memory block of size more than the size of that process. Due to this some part of the memory is left unused and this cause internal fragmentation.

**Example:** Suppose there is fixed partitioning (i.e. the memory blocks are of fixed sizes) is used for memory allocation in RAM. These sizes are 2MB, 4MB, 4MB, 8MB. Some part of this RAM is occupied by the Operating System (OS).

Now, suppose a process P1 of size 3MB comes and it gets memory block of size 4MB. So, the 1MB that is free in this block is wasted and this space can’t be utilized for allocating memory to some other process. This is called **internal fragmentation**.

#### How to remove internal fragmentation?

This problem is occurring because we have fixed the sizes of the memory blocks. This problem can be removed if we use dynamic partitioning for allocating space to the process. In dynamic partitioning, the process is allocated only that much amount of space which is required by the process. So, there is no internal fragmentation.

**External Fragmentation**

In this fragmentation, although we have total space available that is needed by a process still we are not able to put that process in the memory because that space is not contiguous. This is called external fragmentation.

**Example:**Suppose in the above example, if three new processes P2, P3, and P4 come of sizes 2MB, 3MB, and 6MB respectively. Now, these processes get memory blocks of size 2MB, 4MB and 8MB respectively allocated.

So, now if we closely analyze this situation then process P3 (unused 1MB)and P4(unused 2MB) are again causing internal fragmentation. So, a total of 4MB (1MB (due to process P1) + 1MB (due to process P3) + 2MB (due to process P4)) is unused due to internal fragmentation.

Now, suppose a new process of 4 MB comes. Though **we have a total space of 4MB** still **we can’t allocate this memory**to the process. This is called **external fragmentation**.

This problem is occurring because**we are allocating memory continuously**to the processes. So, if we remove this condition external fragmentation can be reduced. This is what done in**paging & segmentation**(non-contiguous memory allocation techniques) where memory is allocated non-contiguously to the processes.

**External fragmentation:**

1. SPOOL is an acronym for **simultaneous peripheral operations on-line**.  It is a kind of buffering mechanism or a process in which data is temporarily held to be used and executed by a device, program or the system. Data is sent to and stored in memory or other volatile storage until the program or computer requests it for execution.
2. In Operating Systems, Segmentation is **a memory management technique in which the memory is divided into the variable size parts**. Each part is known as a segment which can be allocated to a process. The details about each segment are stored in a table called a segment table.

Left Definitions: -

1. Paging- it is a storage mechanism used to retrieve processes from the secondary storage into the main memory in the form of pages.

Partition of secondary memory- pages.

Partition of main memory- frames.

1. Page Table- page table is a data structure which maps the page referenced by the CPU to the frame number where the page is stored. It is stored in main memory and each process has its own page table.
2. A Translation look aside buffer can be defined as **a memory cache which can be used to reduce the time taken to access the page table again and again**. It is a memory cache which is closer to the CPU and the time taken by CPU to access TLB is lesser then that taken to access main memory.
3. **Multilevel Paging** is a paging scheme which consist of two or more levels of page tables in a hierarchical manner. It is also known as hierarchical paging. The entries of the level 1 page table are pointers to a level 2 page table and entries of the level 2 page tables are pointers to a level 3 page table and so on.
4. **Inverted Paging- It is a technique in which instead of page table for each process there is only a page table (global page table) for all pages for all processes. No. of entries in Inverted page table is equal to the no. of frames in Main Memory.**
5. **Page Replacement Algorithms-** In an operating system that uses paging for memory management, a page replacement algorithm is needed to decide which page needs to be replaced when a new page comes in. all os uses this.
6. **Virtual Memory-** It is a storage scheme that provides an illusion to the user of having big main memory, it is done by treating some part of secondary memory as main memory. Instead of loading a big process into the main memory os loads different parts of more than one processes into main memory, it increases degree of multiprogramming.
7. **Demand Paging-** The process of loading the page into main memory on demand (whenever page fault occurs) is known as demand paging. In this technique pages of process which are least used gets stored in secondary memory. A page is copied to main memory when its demand is made or page fault occurs. There are many page replacement algorithms which are used to decide which pages will be replaced.
8. **Swapping-** The process of temporarily swapping the process from main memory to the secondary memory is so than main memory can be available for other processes.
9. **Thrashing-** It is a condition or situation when the system is spending more time in servicing page faults and the actual processing done is very negligible.

First Fit algorithm scans the linked list and whenever it finds the first big enough hole to store a process, it stops scanning and load the process into that hole.

Next Fit algorithm is similar to First Fit algorithm except the fact that, Next fit scans the linked list from the node where it previously allocated a hole.

The Best Fit algorithm tries to find out the smallest hole possible in the list that can accommodate the size requirement of the process.

The worst fit algorithm scans the entire list every time and tries to find out the biggest hole in the list which can fulfill the requirement of the process.