Operating System

Q.1 What is the main purpose of an OS? Discuss different types?

→ An operating system is a software that manages computer hardware, software resources, and provides common services for computer programs.

→ it manages the computer's memory, processes, devices, files and security aspects of the system.

→ it also allows us to communicate with the computer without knowing how to speak the computer's language

Types of OS

- Batch OS
- Distributed OS → geo separated → load balancing
- Multitasking OS → preemptive / time sharing → Response time
- Network OS
- Real time OS → hard / → soft
- Mobile OS.
- Multiprogrammed – non preemptive → idleness
- clustered OS → LAN. → load balancing
- Embedded OS → specific function.

Q.2 What is Socket, Kernel and monolithic kernel?

→ Socket is defined as an endpoint for communication. a Socket is identified by a IP address concatenated with a port number.

→ Kernel is the central core component of an OS that manages operations and hardware.

→ Establishes communication b/w user level application and hardware.

→ Manages memory, CPU time, Disk, Task Management

→ Decides state of incoming processes.

Monolithic Kernel (provides good performance but lots of lines of code)

→ all operating system services operate in kernel space.

e.g. Unix, Linux, VMS, XTS-400 etc.

Q.3 Difference b/w process, program and thread?
Different types of process.

process: process is a instance of an executing program.

e.g. we write program in text and we execute this program, it becomes a process which performs all tasks mentioned in program.

program: program is a set of instructions to perform a certain task e.g. chrome.exe, notepad.exe.

Thread: → Thread is a path of execution within a process.

→ also known as lightweight process.

→ achieve parallelism by dividing a process into multiple threads.

e.g. word processor uses multiple threads: one thread to format the text, another thread to process inputs.

Types of process

a) Based on Event-Specific

a) CPU bound process spend majority of time simply using CPU

b) I/O Bound Process → reading from files

b) nature

a) independent Process

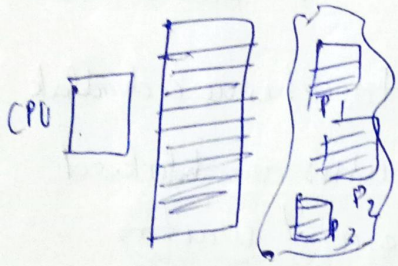
b) Cooperative Process
depends Occurrence of any event and outcome affects.

→ Single level Thread and Multilevel threads
→ there are two levels at which threads operate. System threads and User-level threads.

Q.4 Define virtual Memory and Thrashing

<u>Virtual Memory</u> → programs can be larger than physical memory

CPU → ~~defen~~ demanding page 0 of process 1
→ page fault (if not present)
→ User → OS → User
→ Trap ⇒ effective Mem. access time
= p (page fault service time)
+ (1−p) (main mem Access time)



p — probability of page fault.

Local Address space

<u>Thrashing</u>

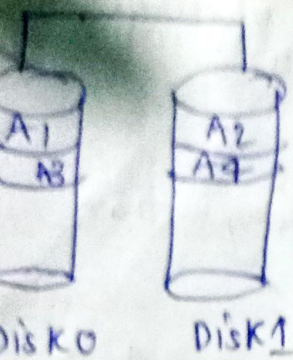
Thrashing is a condition or a situation when the system is spending a major portion of its time in servicing the page faults, but actual processing done is very negligible.
→ High degree of multiprogramming and lack of frames
→ Many and too freq. page faults → thrashing.

Q.5 What is <u>RAID</u>? Different types.

- Redundant Array of Independent Disks
- Redundant Array of Inexpensive Disks

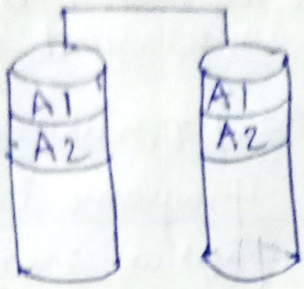
→ performance in case of Read and write
→ security & Availability.

RAID 0



performance↑
data striping

RAID 1



Availability↑
Mirroring

⇒ RAID 1+0
(Nested RAID)

⇒ RAID 3
divide the data into blocks and store parity in another disk also.

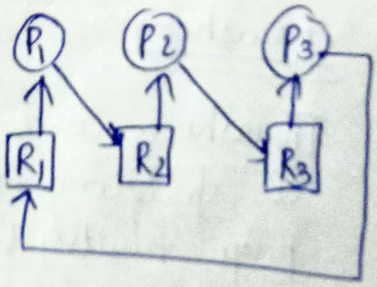
⇒ RAID 4, RAID 5
↓
parity ko bhi distribute kar do

⇒ RAID 6
calculate two parities.

Q.6 What is deadlock? Different conditions to achieve a deadlock

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.

Necessary condition for Deadlock



1) Mutual Exclusion if two process cannot use same resource at same time

2) Hold and Wait. process waits for resource while holding ~~and a~~ another resource at same time

3) No preemption process which once scheduled will be executed till completion.

4) Circular wait All processes must be waiting for resource in a cyclic manner.

Q.7 What is fragmentation? Types of Fragmentation.

As programs are loaded and deleted from memory, they generate free space or a hole in memory. These small blocks cannot be allotted to new arriving processes, resulting in inefficient memory use.

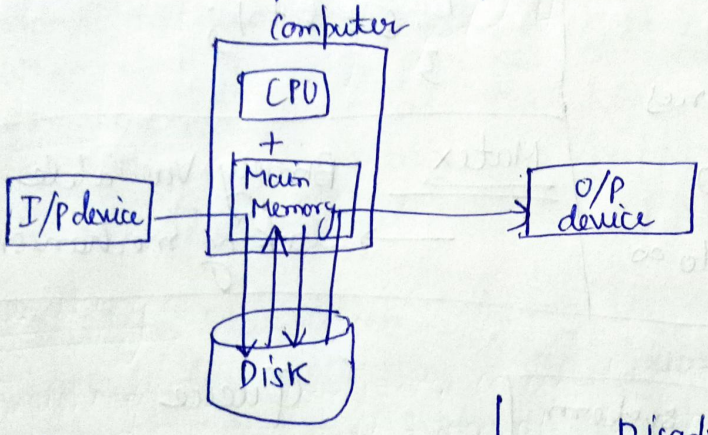
Types of Fragmentation

1) Internal → best fit block.

2) External → variable size memory block → solⁿ compaction and paging

Q.8 What is spooling?

Simultaneous peripheral operations online



Adv

→ no interaction of I/P & O/P device with CPU

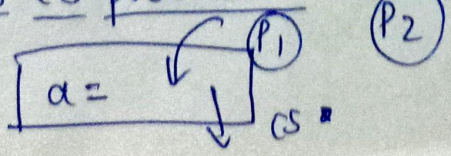
→ CPU utilization is more as CPU is busy most of the time

Disadv

In starting spooling was uniprogramming.

Q 9 What is Semaphore and Mutex? Define Binary semaphore?

Solves CS problem



3 criterias

1. Mutual exclusion
2. Progress
3. Bounded wait

```
do
{
  wait(S);
  //critical section;
  Signal(S);
  // remainder section
} while(T)
```

Types of semaphores

1. Binary 1 & 0
2. Counting $-\infty$ to ∞

Semaphore is an integer variable that solves the critical section problem. After initialization, it can only be accessed by two atomic operations

1. wait (S)

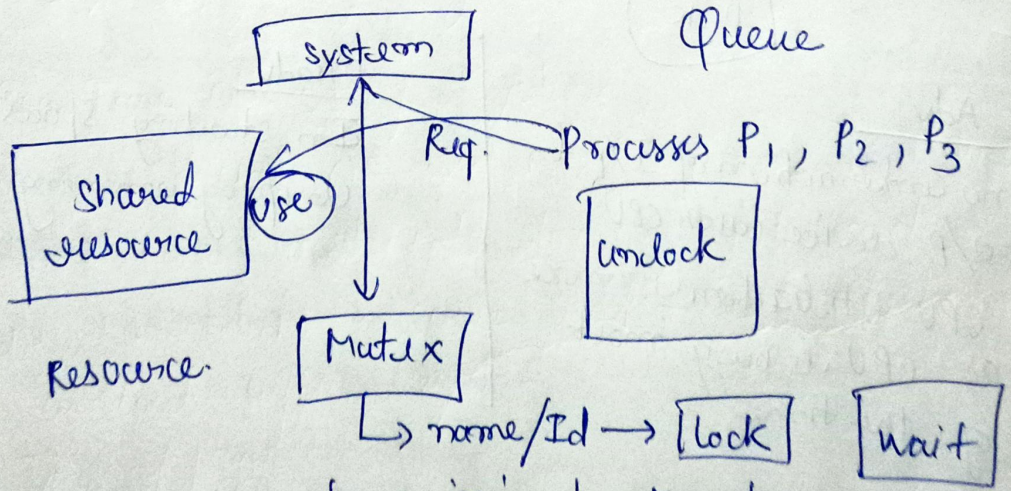
```
{ while (S<=0); // trap
  S = S-1;
}
```

2. Signal (S)

```
{
  S = S+1;
}
```

Mutex Binary Variable → locking mechanism

⇒



→ Mutex gives permission to the processes for accessing shared resource.

Q.10 What Belady Anomaly?

is the name given to phenomenon where increasing the number of page frames results in an increase in the number of page faults for a given memory access pattern.

Solⁿ to Belady Anomaly

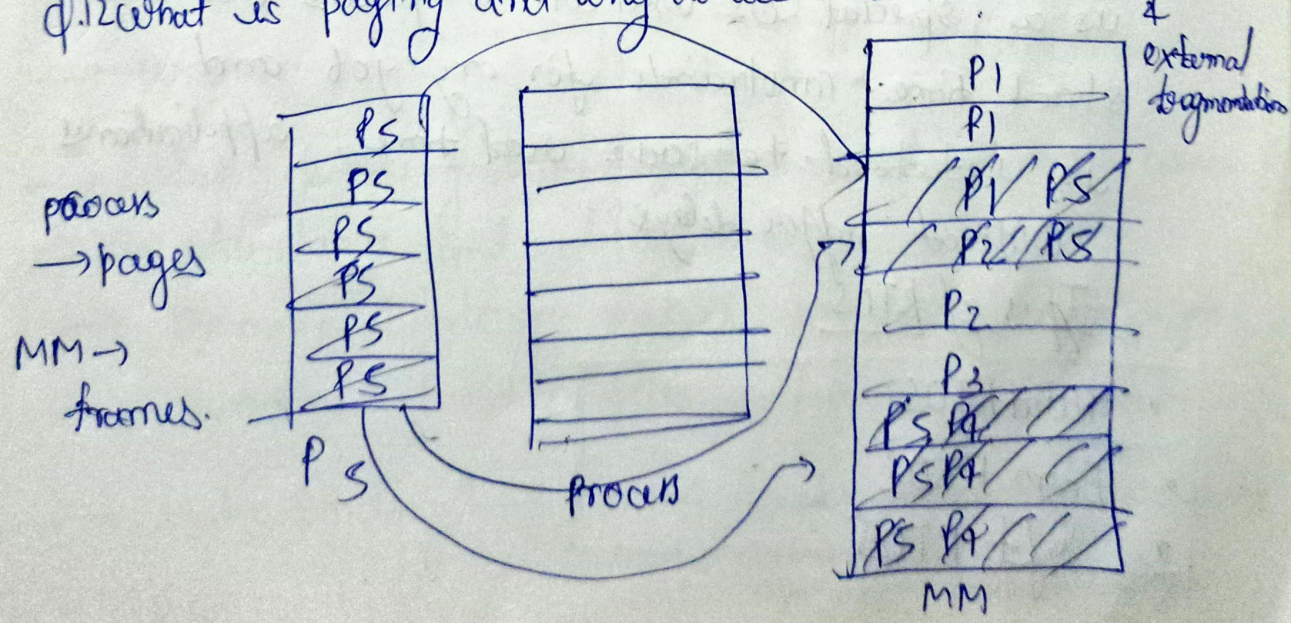
use of stack based Algo optimal page Replacement Algo and LRU algo can eliminate this issue as these algo assign priority to pages.

Q.11 What is Starving and Aging in OS

Starvation (also called lived lock) is the problem that occurs when low priority processes get jammed for an unspecified time as the high priority keep executing. So starvation happens if a method is indefinitely delayed.

Solⁿ to starvation: Aging is a technique of gradually increasing the priority of processes that wait in the system for a long time.

Q.12 What is paging and why do we need it? → internal & external fragmentation



Q.13 Demand Paging and Segmentation?

Demand Paging is a method of virtual Memory Management which is based on principle that pages should only be brought into memory if the executing process demands them.

Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each each ch is called segment which can be allocated to a process.

→ details about each segment → segment table.
↓
storedin of the segments

→ segment table
→ Base address of segment
→ length of segment.

Q.14 Real Time Operating System
is a special OS used in computers that has strict time constraints for any job and is intended to serve real time applications without suffer delays.

Types of RTOS

- Hard RTOS
- Firm RTOS
- Soft RTOS

Q.15 Difference b/w Main Memory and Secondary Memory?

Primary Memory	Auxiliary Memory
→ temporary Memory	→ permanent memory
→ fast access time	→ it has slow access time
→ small in size	→ big in size
→ directly communicate to CPU	→ it is not directly communicate to CPU.
→ more expensive	→ cheap than primary
→ volatile in nature	→ Non-valatile in nature
→ RAM, ROM	→ HDD, FDD, DVD, pendrive etc.
→ we can add catch also	

Q.16 Static (early) and Dynamic (late) Binding?

→ Binding means associating the code of a function with the definition of that function.

→ Static binding is performed at compile time

→ Dynamic Binding is performed at run time.

→ static binding and Dynamic Binding both have advantages and disadvantages.

→ Dynamic Binding makes program bit slower.

→ Dynamic Binding is flexible in nature

→ static Binding → normal function, function overloading and ~~function overriding~~ operator overloading.

Dynamic Binding → virtual functions, function overriding

```
class User {
  public :
    virtual void getPermissions() {
        cout<< "Users can see limited info"<< endl;
    }
};

class SuperUser: public User {
  void getPermission () {
      cout<< "Superusers can see all the info"<<endl;
  }
};

int main ()
{   User u;
    SuperUser s;
    list<User *> users;
    users.push_back (&u);
    users.push_back (&s);
    for ( User* usePtr : users)
        usePtr -> getpermission ();
}
```

O/P

——— → different.
———

Q.17 FCFS Scheduling ? (First Come First Serve)

criteria : "Arrival time"
mode : "Non-Preemptive"

Arrival and Burst time is given

Turn around time (TAT) = Completion time − arrival time

Waiting time = TAT − Burst time

Response time = process ko CPU first time kab mila − arrival time

for non-preemptive it is equal to WT

Q.18 SJF Scheduling ? (Shortest Job first)

criteria : "Burst time"

Mode : "Non-preeptive"

if (Burst time equal) then check arrival time

Q.19 SRTF Scheduling ? Shortest Remaining Time first (SJF with Preemption)

Criteria : "Burst time"

Mode : "Preemptive"

* Burst time update is imp

* Sab processes ko one one unit ke liya chalana hai or fir dekhna hai ki ussa koi chota burst time wala process toh nahi hai agar hai toh usse repeat this process. One unit time ke liya chalo and

Q20 LRTF (Longest Remaining time first) Scheduling?

Criteria : Burst time
Mode : Preemptive

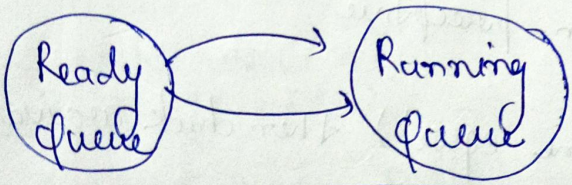
Just like SRTF
But longer burst time is preffered.

Q.21 Priority Scheduling?

is a method of scheduling processes that is based on priority. The processes with higher priority should be carried out first, in case of equal priority on the basis of round robin or FCFS. ~~de~~ priority depends on memory and time requirements etc.

Q22 Round-Robin Scheduling?

Criteria : "Time Quantum"
Mode : "Preemptive"



Context switching at Time Quantum

⇒ PCB → Process Control Blocks

⇒ no. of context switching is important.

Q.23 Producer Consumer Problem ?

The producer Consumer Problem that is used for multiprocess synchronisation. producer produces data and put in into buffer and consumer consume data from buffer

Problems

→ producer should produce data only when buffer is not full. if the buffer is full, then the producer should not be allowed to ~~pull~~ put data into buffer

→ consumer should consume data only when buffer is not empty. if buffer is empty then consumer shouldn't be allowed to take any data from buffer.

→ producer and consumer should not access the buffer at same time.

→ we can solve this problem using semaphores.

Q.24 Banker's Algo ?

→ Banker's Algo used to avoid deadlock and allocate resources safely to each process in computer system.

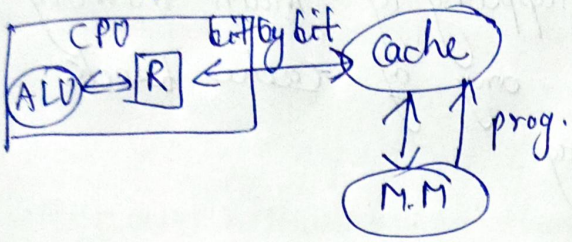
→ we have maximum no. of resources which a process can use and also currently allocated no. of resources.

→ we have check weather process is requesting for resources is valid → donot exceeds max. no. of resources
→ and available resources should be more.

→ if it is a valid request then we have check for safe sequence → serial wise execution is possible.

Q.25 Explain Cache ?

→ it is the memory which is very nearest to the CPU, all recent instructions are stored into cache memory.



→ if required data is not available in cache then offers as cache miss

→ Hit rate = $\frac{\text{no. of hits}}{\text{total no. of bus cycles (hits + miss)}}$ × 100 %.

Q.26 Diff. between direct mapping and associative mapping?

<u>Cache mapping</u> The process of bringing data of main memory blocks into cache block is known as cache mapping.

mapping techniques

① Direct Mapping

② Associative

③ Set-Associative

Direct Mapping Each block from main Memory has only one possible place in the cache organization.
The address is divided into 3 field, Tag, Block & word.

Associative Mapping Here mapping of main memory block can be done with any of cache block only two fields word & tag.

Set-Associative mapping adv. of both mapping.
here cache consists of a number of sets, each of which consists of no. of blocks.

Direct	Associative	Set-Associative
needs only one comparison	with all tag bits	blocks per set.
3 fields	2 fields	3 fields
Search time is less	Searchtime is more	Search time increases with no. of blocks

Q.27 **Diff b/w Multiprocessing and Multitasking**

→ CPU is not allocated to process for fix time	→ Time Quantum is used
→ Can/Can't be other tasks during CPU Processing.	→ we can do other tasks during CPU Processing.
→ ~~difficult~~ to use with slow CPU	→ difficult to use with slow CPU.
→ low Memory	→ Big size Memory
→ less switching overhead	→ High switching overhead.