Tut - 1

Apoorv Garg

Q. Explain the role of time sharing in information technology.

Ans. Time sharing is a tet technique which enables many people, located at various terminals, to use a particular system at the same time.

- It is a logical extension of multi-programming.
- Processor's time is shared among multiple users simultaneously.
- The OS uses CPU scheduling and multiprogramming to provide each user with a small portion of a time
- Time sharing provides advantage of quick response avoid duplication of software and reduces CPU idle time.

Q. Differentiate between multiprogramming and multiprocessing system.

Ans.

multiprogramming OS	multiprocessing OS.
1- It is interleaved execution of 2 or more process by a style CPU system	1. It is the simultaneous execution of 2 or more process by a system having more than 1 CPU.
2. It occurs by context Switching	2. It occurs by parallel processing.
3 It takes more time.	3 It takes less time
4. The idea is to reduce the CPU idle time for as long as possible.	4. The idea is to allow multiple process to run simultaneously via time sharing.

Q. What is the purpose of system calls and how do system calls relate to the OS and to the concept

of dual mode operation?

Ans. System calls are a way for user programs to request some service from O.S.
- System calls operations are exposed to end users in the form of simple library calls APIs - read(), write(), open() etc.
- The most of such operations is executed by OS kernal code after shifting the mode from user to kernel.
- Kernel level data structures are highly critical for correct functionality of the system. This is why they are safely accured by code running in kernel mode.
- User programmes are not allowed to work directly with them since there is likelihood of our programs carrying important data structure in kernel.
- Also OS is there as an interface between user program and hardware. It interacts with hardware on behalf of operations requested by the user program. A read() API call would result in execution of correspondig system call in OS which will further invoke several file system algos to do the I/O from disk.

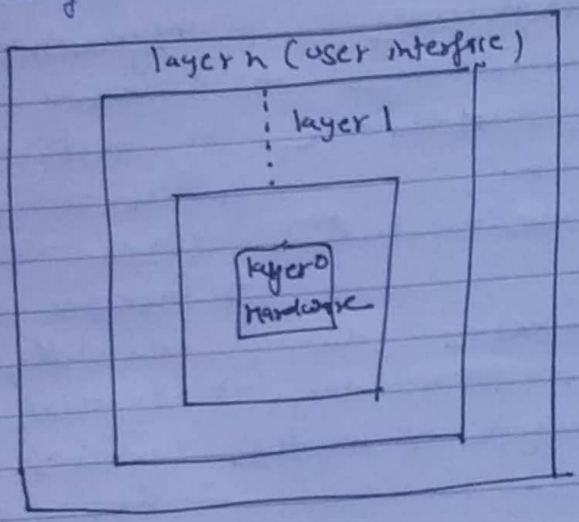
Q. Describe layered approach of an OS.

Ans A system can have different designs and modules. One of them is layered approach, in which the OS is broken into number of layers, the bottom layer (layer 0) being hardware and highest layer being the user interface.

- Describe the hierarchy

- An OS layer have the abstraction of data and functions with in a layer. In a typical layer system, say layer m, consist of data structure and a set of routine that can be invoked by higher level layers and in turn layer m, can call operations on lower level layers.

→ In this approach construction is simple in understanding and debugging.



Q. What is an RTOS? Define its basic kernal service.

Ans. A real time operating system is an OS intended to serve real time applications that process data as it comes in, typically without buffer delays.

→ The kernel of an RTOS provides an "abstraction layer" that hides from application software the hardware details of the processor upon which the software application will run.

Q. differentiate between a parallel OS and a distributed OS

Parallel OS	Distributed OS
- Tightly coupled system	Loosely coupled system
Shared memory	Distributed memory.

Parallel OS

- Global clock control
- Processor interconnection in order of Tbps
- Main focus is on performance, scientific computing

Distributed OS

- No global clock control.
- In order of Gbps.
- main focus is on performance, reliability, resource sharing.

Q. Differentiate between symmetric, asymetric multiprocessing.

Ans.

Symmetric multiprocessing

- Processing of programs by multiple processors that share a common OS & memory.
- All processors are treated equally.
- Processors communicate with each other by shared memory.
- Not as easy to design or handle
- Comparatively costly
- All processors have same architecture.

Asymmetric multiprocessing

- Processing of programs by multiple processors that function according to master slave relationship
- Processors are not treated equally.
- Processors communicate ~~c~~ master processor.
- Easier to design and handle
- Cheaper
- Architecture can be different for each processor.

Tut-2

Apoorv Garg

Q. Given memory partitions of 100KB, 500KB, 200KB, 300KB and 600KB. How would each of the first fit, best fit and worst fit algos place processes of 212 KB, 417 KB, 112 KB and 426 KB? Which algo makes the most efficient use of memory?

Ans :- First fit :-
212 Kb is put in 500 Kb partition
417 Kb " " " 600 Kb partition
112 Kb " " " 288 Kb partition
(new partition 288 Kb = 500 Kb - 212 Kb)
426 Kb must wait.

→ Best fit :-
212 Kb is put in 300 Kb partition
417 Kb " " " 500 Kb "
112 Kb " " " 200 Kb "
426 Kb " " " 600 Kb "

→ Worst fit :-
212 Kb is put in 600 Kb partition
417 Kb " " 500 Kb "
112 Kb " " 388 Kb " (new position 388 Kb = 600 - 212 Kb)
426 Kb must wait.

→ Best fit makes the most efficient use of memory.

Q. What are distinction among logical, relative and physical address?

Ans. Logical address - It is a reference to a memory location independent of current assignment of data to memory. A translation must be made to a physical address

before the memory access can be achieved.

Relative address – It is an example of logical address, in which the address is expressed as a location relative to some known points, usually the beginning of the program.

Physical address – It is an actual location in main memory.

Q. What is thrashing? Explain with the help of a suitable example.

Ans. A process is said to be thrashing if it is spending more time paging than executing.

→ If a process don't have the no of frames it needs to support pages in active use, it will result in page fault. At this point it must replace a page. However, if all pages are in active use, it must replace a page that will be needed again right away. Consequently it quickly faults again, and again as it goes on replacing pages that it must bring back immidiately. This high paging activity is Thrashing.

Q. Differentiate between Contagious and non-contagious memory.

Ans.

Contagious memory	Non contagious memory
1. It assigns memory blocks having consecutive address	1. It assigns a process to memory blocks

Contiguous memory	Non contiguous memory
to a process	located in different locations.
→ Overhead is minimum	→ overhead is more.
→ Process executes faster	→ Process requires more time to execute.
→ Easier for os to control.	→ Difficult to control.

Q. What is fragmentation? what are its type? Discuss any one software based and one hardware based solution for the same.

Ans – Some times it happens that memory blocks can not be allocated to process due to their small size and memory block remain unused. This problem is known as fragmentation.

It is of three types.

1. External fragmentation
2. Internal fragmentation
3. Data fragmentation

Solution to fragmentation.

→ Best fit block
→ Compaction
→ Paging
→ Segmentation.

Q. Consider the page reference string 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 1, 6, 3, 2, 1, 2, 3 6. How many page faults would occur for following replacement algo.

1. LRU
2. FIFO
3 Optimal replacement.

Frames	1	2	3
LRU	20	18	15
FIFO	20	18	16
Optimal	20	15	11

Q. What is virtual memory? what hardware supports are needed to implement it? Illustrate with help of example the FIFO page replacement may encounter more page faults than LRU algos.

Ans. 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 disk that's set up to emulate the computer's RAM.

→ Hardware support for virtual memory consists of specialised h/w for this purpose. This is what's normally called a memory management unit, which implement a translation lookaside buffer in hardware.

→ A page replacement algo is said to satisfy the inclusion property is called a stack algorithm if the set of pages in a k-frame memory is always a subset of the pages in a (k+1) frame memory.

For page reference stream

0, 1, 2, 3, 1, 4, 0 1 2 3 4

the no. of page faults can actually increase when more frames are attached to process. In this there are 9 page faults for 3 frames but for 4 frames there are 10 page faults.

This happens in FIFO because it is not a stack algorithm.

Q When do page fault occur? What actions are taken on page fault?

Ans. A page fault occur when a program attempts to access a block of memory that is not stored in physical memory or RAM.

→ The OS verified the memory access, aborting the program if it is invalid. If it is valid, a free frame is located and I/O is requested to read the needed page into the free frame. Upon completion of I/O, the process table and page table are updated and the instruction is restarted.

Tut-3 Apoorv Garg

Q. What is the difference between a programme and a process?

Ans.

Program	Process.
- A collection of instructions that perform a specific task when executed by a computer.	- A process is the instance of computer program that is being executed.
- Has a longer lifetime	- Has a shorter lifetime.
- Do not require resources.	- Require resources such as memory, IO devices and CPU.

Q. What is the problem of starvation? Also explain its solution?

Ans Starvation is a ~~programm~~ condition where a process does not get the resources it needs for a long time because the resources are being allocated to other process.

- Solution to starvation is Aging.
- Aging is a technique of gradually increasing the priority of process that wait in the system for a long time.

Q. Describe the action taken by a thread library to context switch b/w user level threads.

Ans Context switching b/w user threads is quite similar to switching b/w kernel threads, although it is dependent on thread library and how it maps user threads to kernel threads. In general context switching b/w user threads involves taking a user thread of its LWP and replacing it with another thread. This act typically involves saving and restoring the state eg, the register,

Q. Consider the following set of processes, with the length

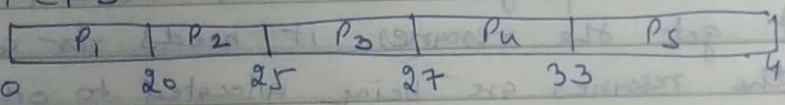
of the CPU, burst time given in milliseconds.

Process	Burst time	Priority
P_1	20	3
P_2	5	1
P_3	2	3
P_4	6	4
P_5	8	2

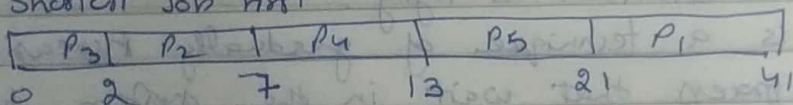
Process are assumed to have arrived at order P_1, P_2, P_3, P_4, P_5 all at time 0

a) Draw 4 Gantt chart for

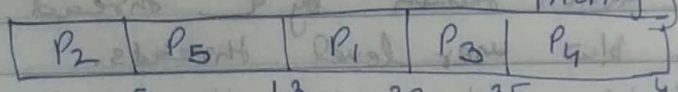
i) FCFS



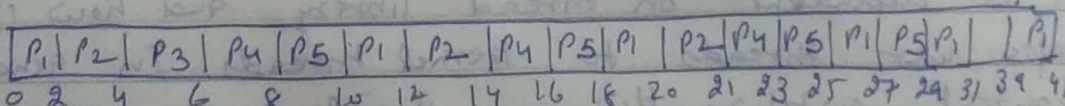
ii) Shortest Job First



iii) Non-premptive priority (small priority to high priority)



iv) Round Robin (quantum = 2)



b) What is turn around time of each process for each algo.

c) What is waiting time of each process for each algo.

Ans Turn around time = Completion Time − Arrival time.

Process	FCFS	SJF	Priority	Round Robin
P_1	20	41	33	41
P_2	25	7	5	21
P_3	27	9	35	6
P_4	33	13	41	23
P_5	41	21	13	29

Ans (c) waiting time = Turn around time − Burst time

Process	FCFS	SJF	Priority	Round Robin
P_1	0	21	13	21
P_2	20	2	0	16
P_3	25	0	33	4
P_4	27	7	35	17
P_5	33	13	5	21

d) Which of the following has the minimum average time.

Ans Shortes Job first = $\frac{21+2+0+7+13}{5}$

= 8.6 millisecond

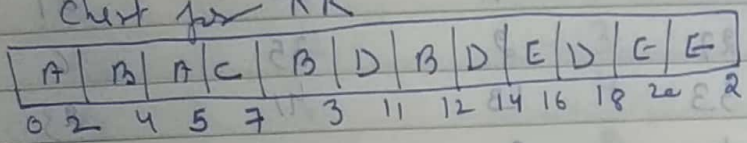
Q. Consider the following set of process with length of CPU burst time given.

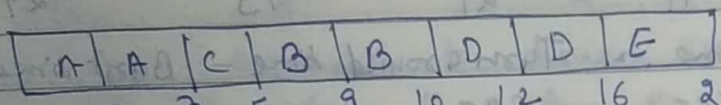
Process Name	Arrival time	CPU Burst time
A	0	3
B	1	5
C	3	2
D	9	6
E	12	5

predict the waiting
critical section.

Calculate avg. waiting time and turnaround time (for each process) for the RR (quantum 2ms) & SJF (pre emptive)

Ans. Chart for RR





New table (RR)

Process	AT	CT	BT	TAT	WT
A	0	3	3	3	0
B	1	10	5	9	4
C	3	5	2	2	0
D	9	16	6	7	1
E	12	21	5	9	4

Avg TAT = $\frac{3+9+2+7+9}{5}$ = 6 ms

Avg WT = $\frac{0+4+0+1+4}{5}$ = 1.8 ms

Tut-4

Q. Define the critical section problem and explain the necessary characteristics of a correct solution?

Ans :- Critical section is the part of a program which tries to access the shared resources. That resource may be any resource in a computer like a memory location, data structure etc.

→ Critical section problem is used to design a set of protocols which can ensure that the race condition among all the process will never arise.

Necessary characteristics :-

Primary

1) Mutual exclusion :-
If one process is executing inside critical section then the other process must not enter in the section.

2) Progress :-
If one process doesn't need to execute into critical section then it should not stop other process to get into it.

Secondary

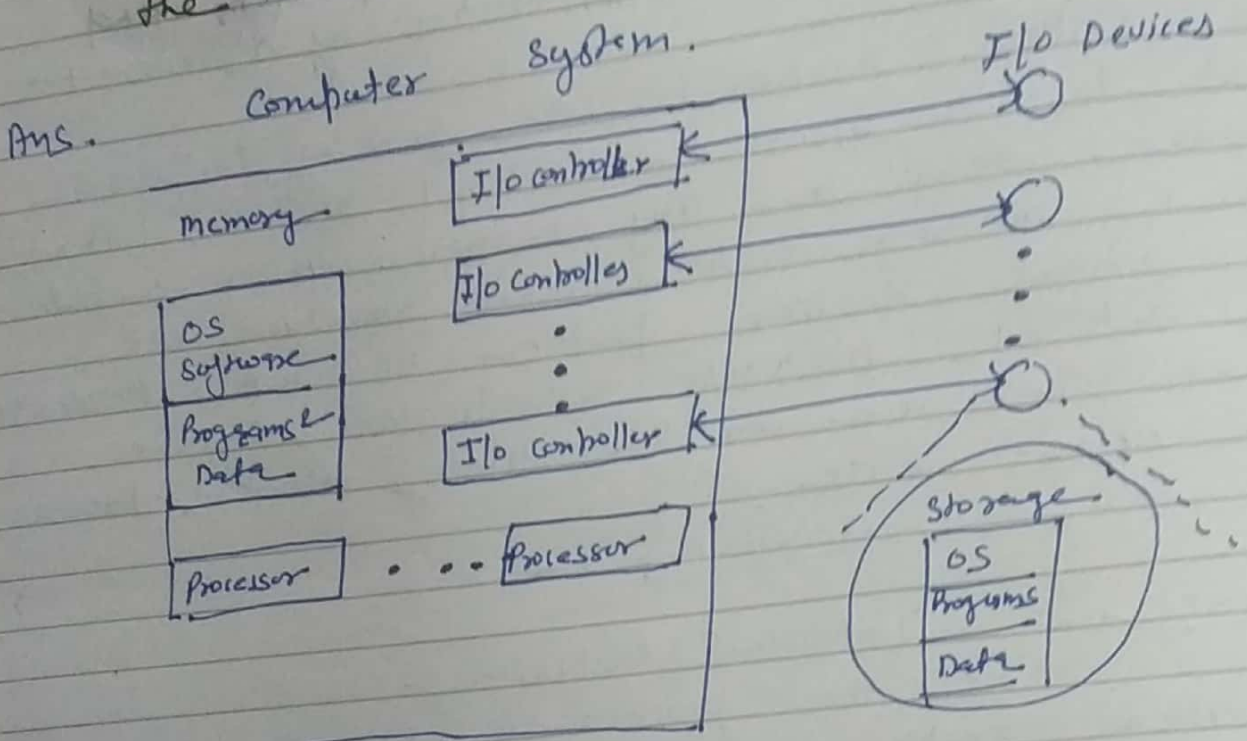
1. Bounded waiting :-
We should be able to predict the waiting time for every process to get into critical section.

2) Architectural Neutrality :-
Our mechanism must be architecturally neutral. It means that soln should be working on each architecture.

Q with the help of model resource management explain the tasks and goals of

the resource manager.

Ans. Computer System.



OS as a Resource Manager

Resource manager
- Tasks and Goals.
- Process manager - Next program to be executed?
  → Time to be given to each program?
- Memory manager - Best use of memory to run as programs as possible.
- I/O Device manager - which program should use a particular I/O device?
- Network manager - which computer should execute a particular program.?

Q. Explain critical region and mutual exclusion with respect to producer consumer problem.

Ans. Critical region - - Producer and consumer share a common fixed sized buffer used as a

queue. This buffer is known as critical section.

Mutual exclusion — Either producer or consumer will be allowed to use or access the shared buffer (critical region) at a particular time. This is known as mutual exclusion semaphore S is used for this.

Q. What are semaphores? What are binary semaphores?

Ans. Semaphores - They are integer variables that are used to solve the critical section problem by using 2 atomic operations, wait and signal that are used for process synchronization.

Binary Semaphores - These are like counting semaphores but their value is restricted to 0 & 1. The wait operation works when the value is 1 and signal operation when value is 0.

Q. When do we need semaphores with structure like a semaphore array? In which application do we need to use structured semaphores.

Ans. A semaphore is a structured data type consisting of variables:

- S - value (integer)
- Queue

and operations

- wait ()
- signal ()

to contain all this, semaphore with structure like array are used.

Applications.

- Process synchronization
- Producer consumer problem.
- Can be used as atomic counter.

Q. Write the algorithm for wait() and signal() functions. Explain their usage in an example.

Ans. S → Semaphore

wait command ⇒ P(s) or wait(s)

if S > 0 then

set S to S-1

else

Block the calling process (ie wait on S)

Signal operation :- V(s) or signal(s)

If any proces are waiting on S

start one of these process

Else

set S to S+1

Q. Describe how semaphore can be used to block wake up Synchronization b/w processes.

Ans Semaphores can be used to synchronize certain operations between processes. For example suppose it is important that process P_1 execute

→ first we create a semaphore named synch that is shared by 2 processes and initialize it to zero.

→ Then in process put insert code:

S1;

signal(synch)

→ And in process P_2 we insert:

wait(synch)

S2;

→ Because synch was initialized to 0, process P_2 will block on the wait until after P_1 execute the call to signal.

Alternate

```
type def struct { int value
                  struct process * list
                } semaphore
wait op :   wait (semaphore * s) { s -> value --;
                if (s -> value < 0) { add this process to
                                      s -> list
                                      block ()
                                    }
                }
signal op:
    signal (semaphore * s) { s -> value ++;
                             if (s -> value <= 0) {
                                 remove a process P for s -> list;
                                 wakeup (P);
                             }
                           }
```

using wake up function.

Q. Explain reader / writter problem and protocol.

Ans The reader writter problem relates to an object such as file that is shared b/w multiple processes. Some of these are readers ie they only want to read the data from object and some of process are writters ie they want to write into the object.

→ This problem is used to manage synchronization so that there are no problems with the object data.

→ It can be implemented using semaphores.

Reader Process

```
wait (mutex);
rc++
if (rc==1)
  wait (wrt)
signal (mutex)
Read the OBJECT
wait (mutex)
rc--;
if (rc==0)
signal (wrt);
Signal (mutex);
```

writter Process

```
wait (wrt);
```

WRITE INTO THE OBJECT

```
signal (wrt);
```

Q.

a)

Tut-5

Aparav Garg

Q. Consider the following snapshot of a system.

Process	Allocation	max	Available
A B C D	A B C D	A B C D	A B C D
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

a) What is the content of matrix need? Is the system in a safe state?

Need [i, j] = max [i, j] − Allocation [i, j]

Process	Need A	B	C	D
P_0	0	0	0	0
P_1	0	7	5	0
P_2	1	0	0	2
P_3	0	0	2	0
P_4	0	6	4	2

work = Available

work = | 1 | 5 | 2 | 0 |

finish =

0	1	2	3	4
false	false	false	false	false

For i = 0

Need = 0 0 0 0

finish[0] is false and $Need_0 < work$ ✓

so P_0 must be kept in safe sequence.

work = work + $Allocation_0$ (0 0 1 2)

(1 5 2 0)

work = | 1 | 5 | 3 | 2 |

finish =

0	1	2	3	4
True	false	false	false	false

For i = 1

$Need_1$ = 0 7 5 0

Finish[1] is false, and $need_1 < work$ ✓

so P_1 must be kept in safe sequence.

Work = Work + Allocation$_1$ (1 0 0)

= | 2 | 5 | 3 | 2 |

Finish = | true | true | false | false | false |

for i = 2

Need$_2$ = 1 0 0 2

Finish[2] is false and Need$_2$ < Work ✓

so P_2 must be kept in safe sequence.

Work = Work + Allocation$_2$ (1 3 5 4)

= | 3 | 8 | 8 | 6 |

Finish = | true | true | true | false | false |

for i = 3

need$_3$ = 0 0 2 0

finish [3] is false and Need$_3$ < Work ✓

so P_3 must be kept in safe sequence.

Work = Work + Allocation$_3$ (0 6 3 2)

= | 3 | 14 | 11 | 8 |

Finish = | true | true | true | true | false |

For i = 4

Need$_4$ = 0 6 4 2

Finish [4] is false and need$_4$ < work ✓

so P_5 must be kept in safe sequence.

Work = Work + Allocation$_4$ (0 0 1 4)

= | 3 | 14 | 12 | 12 |

finish = | true | true | true | true | true |

∴ Finish[i] = true for $0 \le i \le 4$

Hence the system is in safe state with sequence P_0, P_1, P_2, P_3, P_4.

(b) If a request from process P_1 arrives for (0,4,20) can the request be granted immediately?

Ans. P_1 Request = A B C D
0 4 2 0

we use Resource - Request algo.

Request$_1$ < Need$_1$ ✓

0 4 2 0 < 0 7 5 0

Request$_1$ < Available$_1$ ✓

0 4 2 0 < 1 5 2 0

Available = Available − Request$_1$

Allocation$_1$ = Allocation$_1$ + Request$_1$

Need$_1$ = Need$_1$ − Request$_1$

Process	Allocation	Need	Available
P_0	0 0 1 2	0 0 0 0	1 1 0 0
P_1	1 4 2 0	0 3 3 0	
P_2	1 3 5 4	1 0 0 2	
P_3	0 6 3 2	0 0 2 0	
P_4	0 0 1 4	0 6 4 2	

Yes it can be granted.

Q. 4. Consider the following processes snapshot of a system. Execute Bankers algo answer.

Process	Allocation (A B C)	max (A B C)	Available (A B C)
P_0	0 0 1	0 0 1	1 5 2
P_1	1 0 0	1 7 5	
P_2	1 3 5	2 3 5	
P_3	0 6 3	1 6 5	
P_4	0 0 1	3 6 5	

Need = Max − Allocation

Process	Need		
	A	B	C
P_0	0	0	0
P_1	0	7	5
P_2	1	0	0
P_3	1	0	2
P_4	5	6	4

a) If a request from Process P_1 arrives (0, 4, 2) can the request be granted? If granted, write the sequence of processes

b) Is the system in a safe state?

Ans. P_1 Request 1 = A 0, B 4, C 2

we use Resource - Request algo

Request$_1$ < Need$_1$ ✓

0 4 2 < 0 7 5

Request$_1$ < Available ✓

0 4 2 < 1 5 2

Available = Available - Request$_1$

Allocation$_1$ = Allocation$_1$ + Request$_1$

Need$_1$ = Need$_1$ - Request$_1$

Process	Allocation			Need			Available		
P_0	0	0	1	0	0	0	1	1	0
P_1	1	4	2	0	3	3			
P_2	1	3	5	1	0	0			
P_3	0	6	3	1	0	2			
P_4	0	0	1	5	6	4			

Now for checking safe state

work = Available = | 1 | 1 | 0 |

finish =

0	1	2	3	4
false	false	false	false	false

For i = 0

Need$_0$ = 0 0 0

finish[0] = is false and Need$_0$ ≤ work ✓

So P_0 must be in safe sequence

work = work^{0} + allocation$_0$ (0 0 1)

= | 1 | 1 | 1 |

finish = | 0: true | 1: false | 2: false | 3: false | 4: false |

for i = 1

Need$_1$ = 0 3 3

finish[1] is false and Need$_1$ < work ✓

So P_1 must be kept in safe sequence

work = work^{1} + Allocation$_1$ (1 4 2)

= | 2 | 5 | 3 |

finish = | 0: true | 1: true | 2: false | 3: false | 4: false |

for i = 2

Need$_2$ = 1 0 0

finish[2] is false Need$_2$ < work ✓

So P_2 must be kept in safe sequence.

work = work^{2} + Allocation$_2$ (1 3 5)

= | 3 | 8 | 8 |

finish = | true | true | true | false | false |

for i = 3

Need$_3$ = 1 0 2

finish[3] is false and Need$_3$ < work ✓

So P_3 must be kept in safe sequence.

work = work^{3} + Allocation$_3$ (0 6 3)

= | 3 | 14 | 11 |

finish = | true | true | true | true | false |

for i = 4

Need 4 = 5 6 4

Finish[4] is false and Need 4 < work

So P_4 must be in safe sequence

work = 3 14 " + Allocation 4 (0 0 1)

= 3 | 14 | 12

0	1	2	3	4
true	true	true	true	true

Finish =

∴ finish [i] = true for $0 \le i \le 4$

Hence the system is in safe state

Also P_1 request is granted and sequence is P_0, P_1, P_2, P_3, P_4.

Q. What is the sequence in which resource may be utilized?

Ans: → Request – If the request cannot be granted immediately then the requesting process must wait until it can acquire the resource.

→ Use – The process can operate on the resource.

→ Release – The process releases the resource.

Q. What are conditions under which a deadlock situation may arise?

Ans. Following are the conditions.

- mutual exclusion.
- Hold and wait
- No pre-emption.

Q. What is resource allocation Graph?

Ans – Deadlocks can be described more precisely in

in terms of a directed graph called a system resource allocation graph. It consists of a set of ~~variables~~ vertices V and a set of edges E. The set of vertices V is partitioned into 2 different types of nodes. P is the set of consisting of all active processes in the system. R is the set of consisting of all resource types in the system.

6.

Tut-6

Apoorv Garg

Q. Suppose that a disk drive has 5000 cylinders, numbered 0 to 9999. This drive is currently serving a request at 143, and previous request at 125. The queue of pending requests in FIFO order is

86, 1470, 913, 1779, 948, 1509, 1022, 1750, 130

Starting from current head pos what is total dis. that the disk arm moves to satisfy the requests. use following disk scheduling.

(a) FCFS

The FCFS schedule is - 143, 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

⇒ (143-86) + (1470-86) + (1470-913) + (9704-948) + (1509-948) + (1509-1022) + (1750-1022) + (1750-130)

= 7081

(b) SSTF

The SSTF schedule is - 143, 130, 66, 913, 948, 1022, 1470, 1509, 1750, 1774

⇒ (143-130) + (130-86) + (313-86) + (948-913) + (1022-948) + (1470-1022) + (1509-1470) + (1750-1509) + (1779-1750)

Total seek dis = 1745.

(c) SCAN

The Scan schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 4999, 130, 86

Total seek dis = 9769.

d) LOOK

The Look schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1770, 130, 86

Total seek dis = 3319

e) C-SCAN

The C-Scan schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 4999, 0, 86, 130

Total seek dis = 9985

Q What is RAID? Explain its various levels.

Ans: RAID is Redundant Array of Independent Disk". is is a technique which makes use of a combination of multiple disks instead of using a single disk for increased performance, data redundancy or both.

Different Levels of RAID

- RAID - 0 (striping)
  - Blocks are striped across disks.
- RAID - 1 (MIRRORING)
  - More than one copy of each block is stored in a seperate disk.
- RAID - 0+1
  - It is a mirrored set in a striped set.
- RAID - 3
  - It is byte level striping with a dedicated parity disk.
- RAID - 4
  - It is block level striping with a dedicated parity disc.

⊙ RAID - 5
→ It is striping with distributed parity.

⊙ RAID - 6
- It is striping with dual distributed parity

Q. Define spooling.

Ans. Spooling is a process in which data is temporarily held to be used and executed by a device, program on the system. Data is sent to and stored in memory or other volatile storage until the program or computer requests it for execution.
Spool is acronym for "~~disim~~ Simultaneous Peripheral operations online.

Q. What is low level formatting?

Ans. It is a formatting method that creates the tracks and sector on a hard disk. It creates the physical format that dictates where data is stored on the disk.

Q. What is the use of boot block?

Ans. For a computer to start running, it need to have an initial program to run which is known as boot strap.
Boot block is a dedicated block usually at the beginning & first block on first track of a storage medium that holds special data. (bootstrap)

used to start a system.

Tut-7

Apoorv Garg

7)

Q.1. For the file extensions given below indicate the corresponding file type and the usually associated purpose.

a) i) .bat => Batch file (DOS)
- It executes batch files.

ii) .exe => Directly executable program (DOS)
- An executable file contains program that is capable of being execut or run as a program in computer.

iii) .zip = ZIP compressed file archive
- It supports lossless compression.

iv) .acc => sound (audio) file (SUN microsystem)
- It supports may audio encoding formats.

b)

i) .bin = Binary file
- The content of BIN files consists of data in binary form that was copied from a disk onto this o/p file.

ii) .lib = liber library file
- It contains the actual executable code that does work as specified in header file.

iii) .tex = Latex source Document file
- It includes features designed for production of technical and scientific documentation.

iv) .gif => Graphics Interchange Format bitmap graphics (Compu show)
- common format files for web graphics and are saved in bassing format.

v). .ar – AR File (Archive file)
- It maintains the indexed librarian used by linkage editor.

Q. Interpret the following instructions.

a) ls -a

⇒ ls –a ⟶ do not ignore entries starting with :)
| list directory contents

b) rm a* ⟶ Remove (unlink) all the a files
↓
Remove files or directories.

c) cp ? aa* ? ab*
| ↑ Source ↑ Destination
copy files & directories — all 3 lettered files having name ending with aa — 3 lettered all files having name ending with ab

Q. 3 Describe the interpretation of the following under the command chmd

a) 4 4 4

Binary :- 100 100 100
r-- r-- r--

⇒ only read for user class
⇒ only read for Group class
⇒ only read for other class.

b) 1111 001 101
rwx --x r-x

=> read, write and executes for the user class
=> only executes for group class
=> read and executes for other class.

Q. Describe an encryption method provided in Unix to secure files. How does one retrieve an encrypted file using an encryption command? explain the basic principle.

Ans. 1) Navigate to the directory containing all files you which to encrypt.

2) Enter the following command into cmd replacing "myfile.txt" with the name of file you wish to encrypt. An encryption command other than "crypt" may be installed on your system and can be used instead.

Cat myfile.txt | crypt > myfile.cpy

3) Enter the key that will be used to encrypt the file when prompted.

4) Verify that encryption was successful by entering the following command and verifying that output is not human-readable.

Cat myfile.cpy.

5) Delete the unencrypted version of your file.

6) We can retrieve or decrypt file by entering the following comand as well as your encryption key when prompted.

Cat myfile.cpy | crypt > myUnencrypted File.txt.

Q. What is the role of an inode?

Ans Inodes are data structures that contain information about files in UNIX file systems that are created when a file system is created.

→ They provide information on file such as user and groups ownership, access mode (r, w, e) and type.

Q. What is root file system?

Ans. The root file system is the file system contained on the same disk partition on which the root directory is located. It is the file system on top of which all other file systems are mounted and the system boots up.

Q. What are different types of file? What are the tasks of the file management system? List some file system related command in UNIX? How does OS ensure security in file system?

Ans Different file types
- Ordinary files
- Directory files
- Special files.

Tasks of file management system -
- To provide I/O support for a variety of storage device types.
- To minimise the chance of lost or destroyed data.
- To help OS to standardise I/O interface routines for user processes.

- To provide I/O support for multiple users in a multi user-system environment

(c) File System related commands in UNIX —

- touch : Creates a new file or update the timestamps

  syntax → touch [OPTION] ... [FILE]

  ex :- $ touch file1 file2.

- Cat : concatenate files and prints to stdout

  syntax :- cat [OPTION] ... [FILE]

  Ex :- $ cat > file1
  hello

- rm : Remove file or directories

  syntax :- rm [OPTION] ... [FILE]

  Ex :- $ rm file1

Security in File System :

- Main method of control is through access control.
- Accessing files systems operations (ex. modify or deleting a file) are controlled through access control lists or capabilities.
- Capabilities are more secure so they tend to be used by OS on file system like NTFS or ext3.
- Secondary method of protection is through the use of back up system on recovery system.

Q. List the various file attributes.

Ans. Various file attributes.

- Name
- Identifire
- Type
- Location
- Size
- Protection
- Time & Date.

Tut - 8

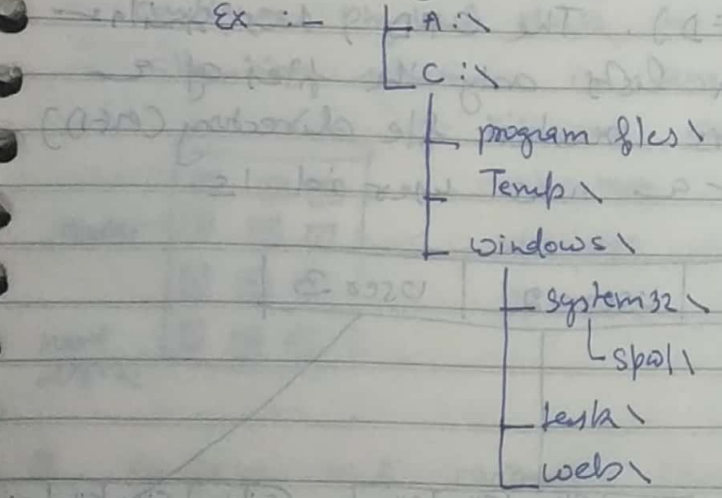
Apoorv Garg

Q. Describe the file system organization. Describe how file hierarchy is managed.

Ans – The file organization module knows about files and their logical blocks, and how they map on physical blocks on the disk.

- In addition to translating from logical to physical blocks, the file organization module also maintains the list of free blocks, and allocates free blocks to files as needed.
- A hierarchical file system is how drives, files are displayed on OS. They are displayed in groups, which allows the user to see only the files, they are interested in seeing.

Ex :-



In this windows\ folder hierarchy contains System 32, tasks and webs folders. Each of these could have hundreds of their own files, but unless they are opened the files are not displayed.

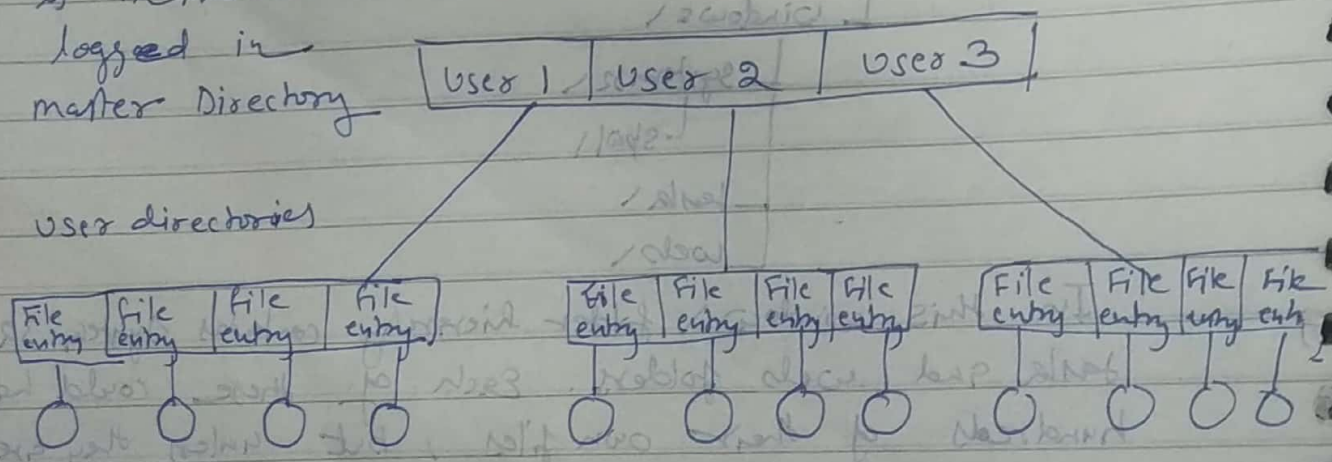
Q Describe briefly :-

a) The methods of file accessing

- Sequential access - Info in the file is processed in order, one record after the other.
- Direct access - A file length logical record that allow the program to read and write record rapidly.
- Index Sequential Method - To find a record in the file, we first search the index and then by the help of pointer we access the file directly.

b) Two level directory structure

- In this, each user has their own user file directory (UFD). The UFD, has similar structures, but each lists only the files of a single user. Systems master file directory (MFD) is searched whenever a new user id is logged in.



Two level directory.

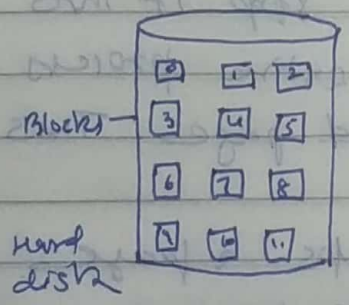
Q. List the various disk space allocation strategies. Explain clearly the contiguous allocation technique.

Ans Various disk space allocation strategies :—

- Contiguous allocation
- Extreme
- Linked allocation
- Clustering
- Indexed
- FAT
- Inode.

Contiguous Allocation.

If the blocks are allocated to the file in such a way that all the logical blocks of the file get the contiguous physical block in the hard disk then such allocation scheme is known as contiguous allocation.



Filename	start	length	Allocation Block
abc.text	0	3	0, 1, 2
video.mp4	4	2	4, 5
jlp.docs	9	3	9, 10, 11

Q. Compare and contrast chained allocation with indexed technique of file allocation.

Chained Allocation	Indexed Allocation
- It does not support direct or random access	- It supports direct & random access
- pointer overhead is lesser than indexed Allocation	- Pointer overhead is greater than chained method.

Chained Allocation
- It is slower
- Better in terms of memory utilization

Indexed Allocation
- It is faster
- Inefficient in terms of memory utilization

Q. Define external and internal fragmentation

Ans. External fragmentation :
- When there are small and non contiguous memory blocks which cannot be assigned to any process the problem is termed as external fragmentation.

Internal Fragmentation -
When there is a difference between required memory space vs alloted memory space, problem is termed as internal fragmentation.

Q. Explain lazy swapper.

Ans - Generally process reside on secondary memory. When we want to execute a process we swap it into memory. Rather than swapping the entire process into memory, it swaps the required page. This can be done by lazy swapper.

A lazy swapper never swaps a page into memory unless that page will be needed.
A swapper manipulates entire process.

Q. What is the information associated with an open file?

Ans Several pieces of information are associated with an open file may be
- file pointer
- File open count

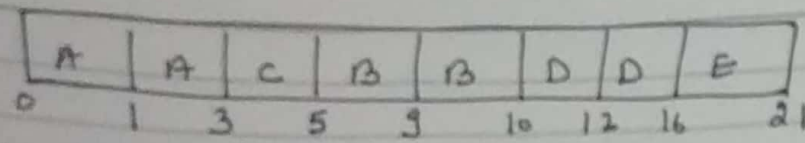
- Disk location of the file
- Access right

Q. Describe the difference among FAT16/32, NTFS & ext file system.

	FAT16	FAT32	NTFS	ext
maximum file length	255 UCS-2	255 UCS-2	255	255 bytes
max file size	4 GiB	4GiB	16 EiB	2GiB
stores file owner	NO	NO	yes	yes
Host OS	MS DOS-3.0	DOS/window	window	Linux
max volume size	1-32 MiB	512 MiB to 16 TiB	16 EiB	2 GiB

Tut 3-4

Apoorv Garg



New table (R, R)

Process	Arrival time	Burst time	Turnaround Time ($C_i - A_i$)	Waiting time (TAT − BT)
A	0	3	5	2
B	1	5	11	6
C	3	2	4	2
D	9	6	9	3
E	12	5	9	4

Avg TAT = $\frac{5+11+4+9+9}{5}$ = 7.6 ms

Avg WT = $\frac{2+6+2+3+4}{5}$ = 3.4 ms

New table (SJF)

Process	AT	CT	BT	TAT	WT
A	0	3	3	3	0
B	1	10	5	9	4
C	3	5	2	2	0
D	9	16	6	7	1
E	12	21	5	9	4

Avg TAT = $\frac{3+9+2+7+9}{5}$ = 6 ms

Avg WT = $\frac{0+4+0+1+4}{5}$ = 1.8 ms.