Syllabus of operating system:-

1) Basic Introduction :- types, process diagram, System call

2) Process scheduling :- FIFO, SJF, Round Robin

3) Process Synchronization :- Semaphore

4) Deadlock & threads :- Banker.

5) memory managment :- virtual memory, paging, segmentation, fragmentation

6) Disk scheduling :- SCAN, CSCAN, FCFS

7) UNIX commands :- ls, mb, dir, cal, =

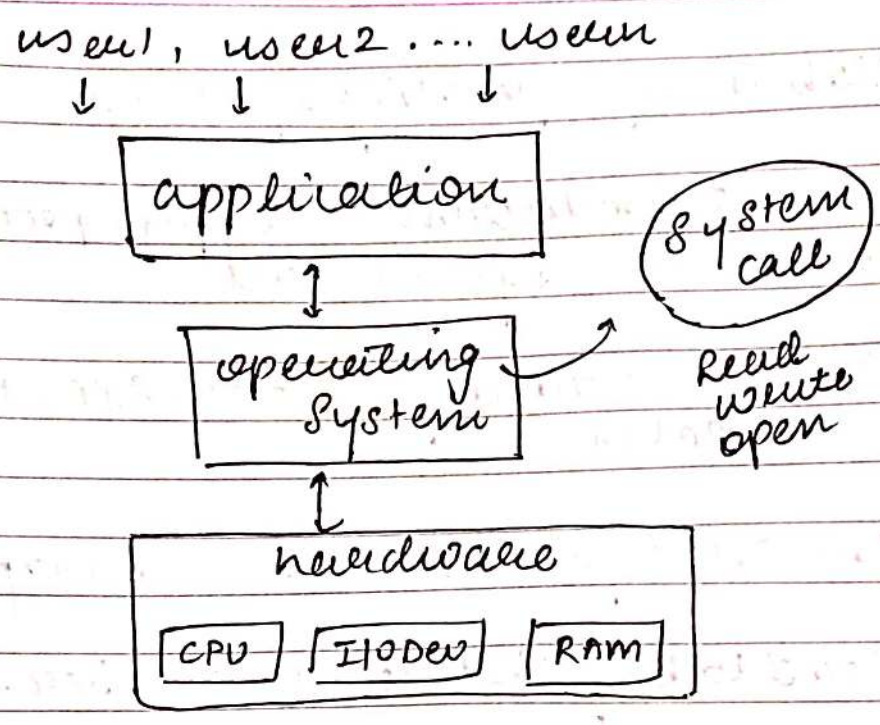
8) File managment & security :- sequential access, random access, linked.

* operating system :- is a system software that works as a interface between users and hardware.

→ need of OS? → If any user wants to access the application without OS then it have to write a program to access the devices.

This document is available free of charge on studocu

Downloaded by Anil Begar 14 (begaranil88@gmail.com)



→ Definition:
Primary goal :- to provide convinience to the user.
throughput (linux)

* Functionalities of OS :-

1) Resource management :- to divide the resources to particular apps, devices & what level of load is to be managed.

2) ~~Storage~~ Process management :- to execute and multiple processes and CPU Scheduling is used to manage the process for executing of cuess.

3) Storage management. (Secondary)

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

:- is done through file system. how data is to be stored in harddisk.

4) memory management (RAM):- limitation of size & all processes which execute first come to RAM & then are given to CPU so the allocation/deallocation after the execution because size is limited.

5) Security and privacy :- to authenticate the proper users.

* Types of operating system :-

1) Batch :- A batch of job is given to the ~~2) multiprogrammed~~ system.
~~3)~~

paper tape → operator
mag. tape ↓
batches

B1 B2 B3
2 3 4

Random I/O
CPU ← Fortran
lot of Idle time.
IBSYS 709X
1900

IDLENESS

2) multiprogrammed OS :- we will take more & more multiple processes. It is generally non preemptive.

This document is available free of charge on studocu
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

CPU ← | P1 P2 P3 | P4 P5 P6 |

It will complete P1 (no idleness) & then only go to P2

higher response time.

→ Processes :- set of instructions

3) * multitasking / time sharing OS :- we have firstly only decided a time for the processes & if it executes in that time then well n' good otherwise it will be scheduled for future execution. (less response time)

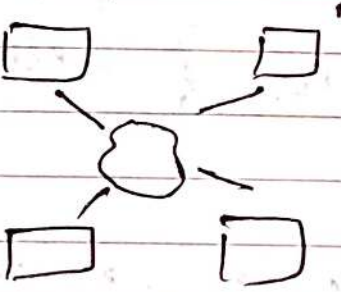
Advants :- responsiveness of premptive

4) * Real time OS :- immediately. i.e time matters & no delay.

hard → no excuses, no delay

soft → bit delay is allowed.

5) * Distributed OS :- Actually environment.

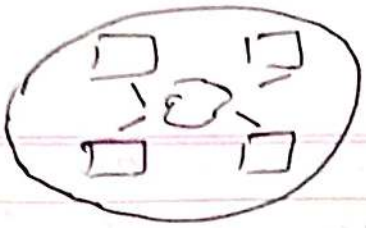


loosely coupled environment.

working independently

6) * clustered OS :- multiple devices are worked as one unit

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

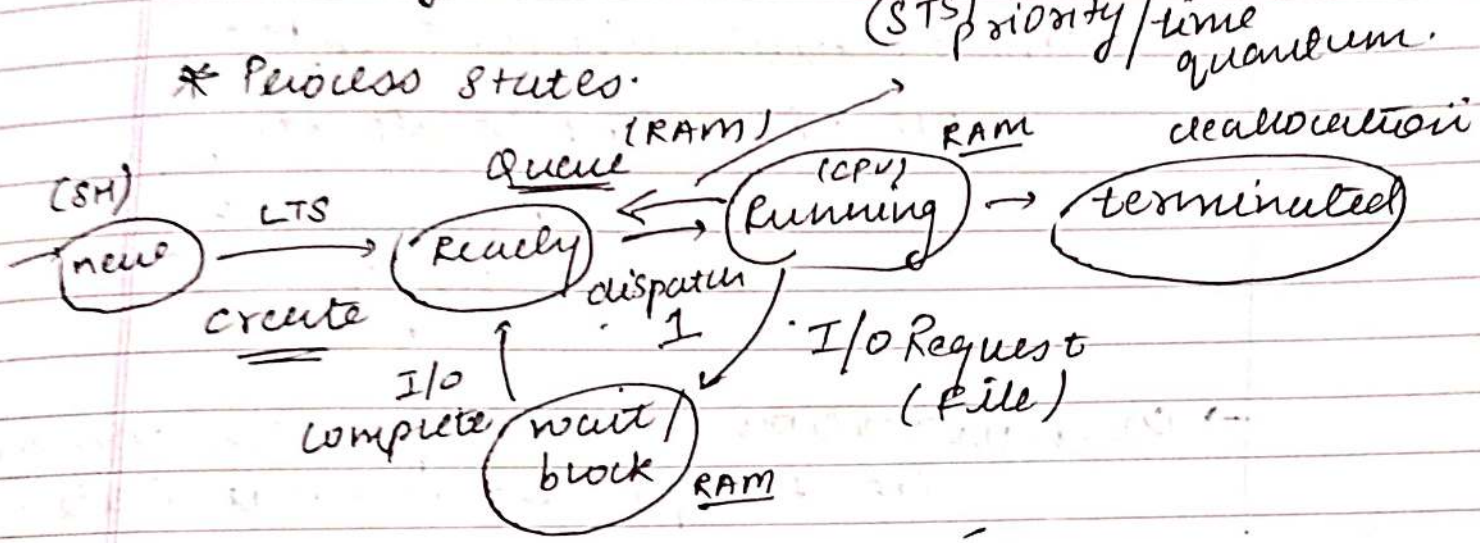


one computer system

Advant:- availability.
fault tolerance
load balance.
scalability.

7) * Embedded:- they work on a fixed functionality.

* Process states.

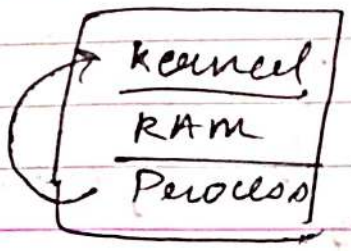


→ long term scheduler:- to take more & more in Ready queue.

Short term scheduler:- Ready state se ek process utao and running state to dedo i.e CPU.

* System call:- to get into kernel mode

→ File Related:- open(), read(), write(), close(), create file, etc.



This document is available free of charge on studocu

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

→ Device Related :- Read, write, reposition, ioctl, fcntl. (to take privilige)

→ Information :- to get information getpid, attributes, getSystemtime & -date

→ Process control :- to load any process (load, execute, abort, Fork, wait, signal, allocate etc).

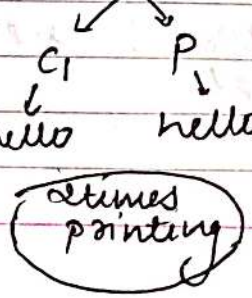
Process creates a child process.

→ communication :- interprocess comm" (ie two process communicate)

Pipe(), create/delete connections, Shmget()

* Fork System call() :- is used to create a child process.

Fork() → 0 child ✓
→ +1 +ve → Parent ✓
→ -1 → child ✗ ✗



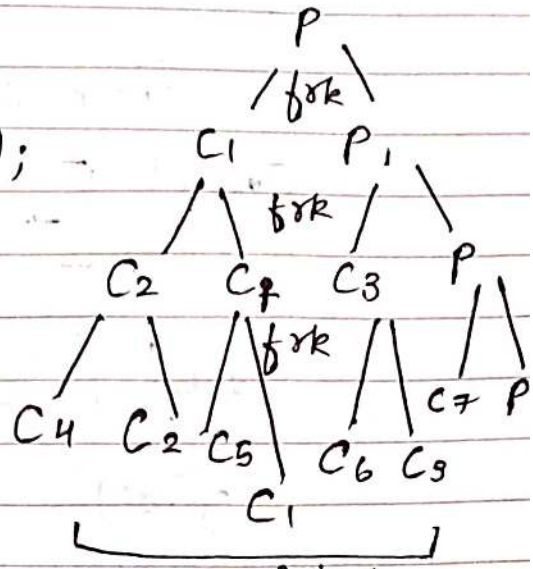

```
main() {
fork();
printf("hello");
}
```

API

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

eg] main() {
fork();
fork();
printf("hello");
}

fork
fork
fork
pf("hello").

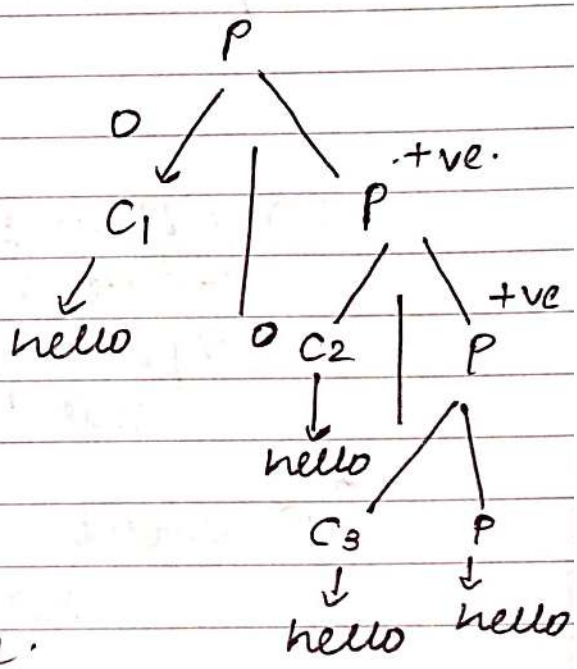


7 - child.
1 - Parent process.

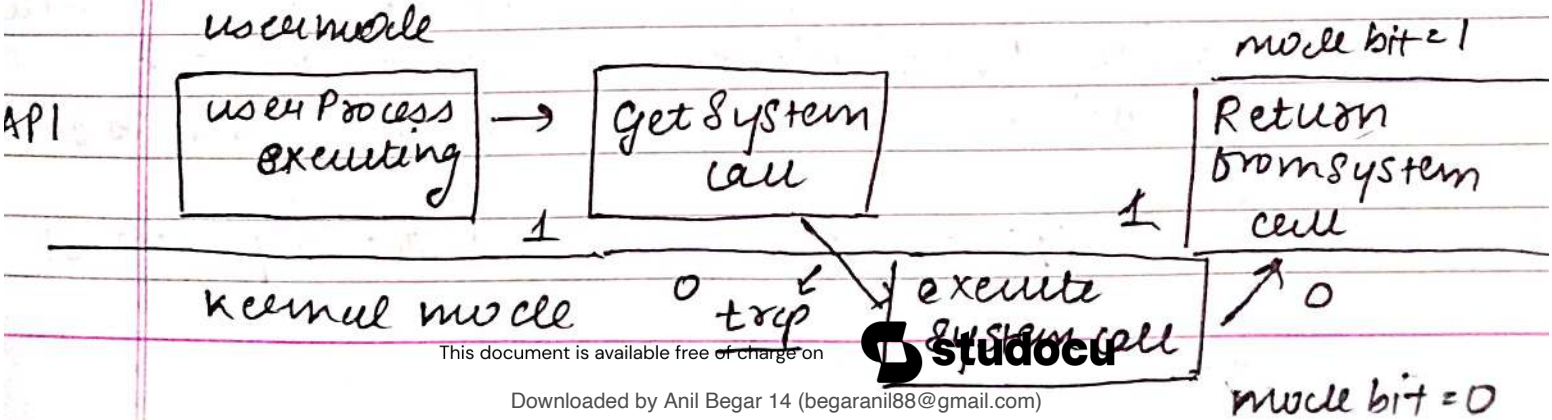
2^n → print hello

$2^n - 1$ → total child process

Q] #in —
#—
int main() {
if (fork() && fork())
fork();
pf("hello");
return 0;
}



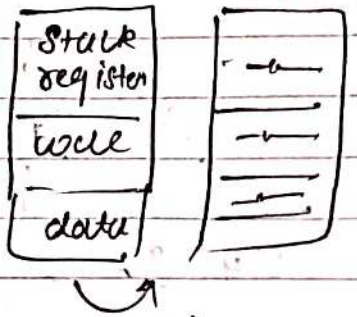
* user mode vs kernel mode.



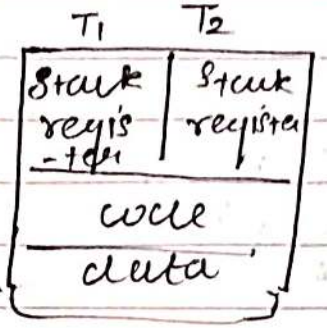
This document is available free of charge on studocu
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

* Process VS threads

Process



threads



① system called are involved in Process	There is no system call involved (user level).
② OS treats different processes differently	All user level threads treated as single task for OS.
③ different processes ~~create~~ have diff copies of data, files, code	shares same copy of code & data.
④ context switching is slower	context switching is faster
⑤ Blocking a process will not block another	Blocking a thread will block entire process.
⑥ independent	interdependent

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

L-13

* User level & kernel level thread.

	User level thread	Kernel level thread
1)	managed by user level Library	managed by the OS. (system call)
2)	are typically fast	are slower than user level.
3)	context switching is faster	context switching is slower.
4)	If one user level thread perform blocking opt then entire process get blocked	If one kernel level thread blocked, no affect on others.

Process > KLT > ULT.
CT

* Various services by OS:-

① program execution:- to execute a prgm, several tasks needed to be ~~program~~ performed. both instruction & data must be loaded into main memory & I/O devices should be initialized.

② control I/O devices:- each I/O devices calls for its own peculiar set of instructions.

③ Program creation:- structures & tools are used to create & modify prgm.

This document is available free of charge on studocu
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

(4) Error Detection & Response :- may cause malfunction of entire device. This may include hardware & software error such as device failure, memory error, etc.

(5) Accounting :- An operating device collects utilization records for numerous assets & keeps records of it overall to improve performance.

(6) Security & Protection :- It affords safety to statistics & packages a person and protects any interference from unauthorised user.

(7) Filemanagement :- computers keep data & information on secondary storage like magnetic tape, etc. each has its speed, capacity, transfer Rate, etc.

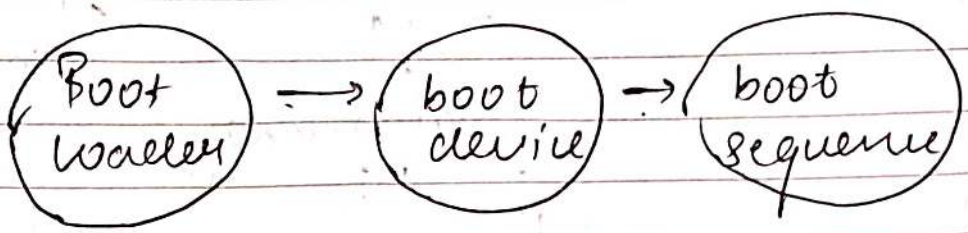
(8) Communication :- managing the exchange of data among the diff computers connected over a network

* System boot :- booting is process of starting a computer. it can be initiated by hardware ~~sth~~. After its switched on, a CPU has no software in its main memory, so

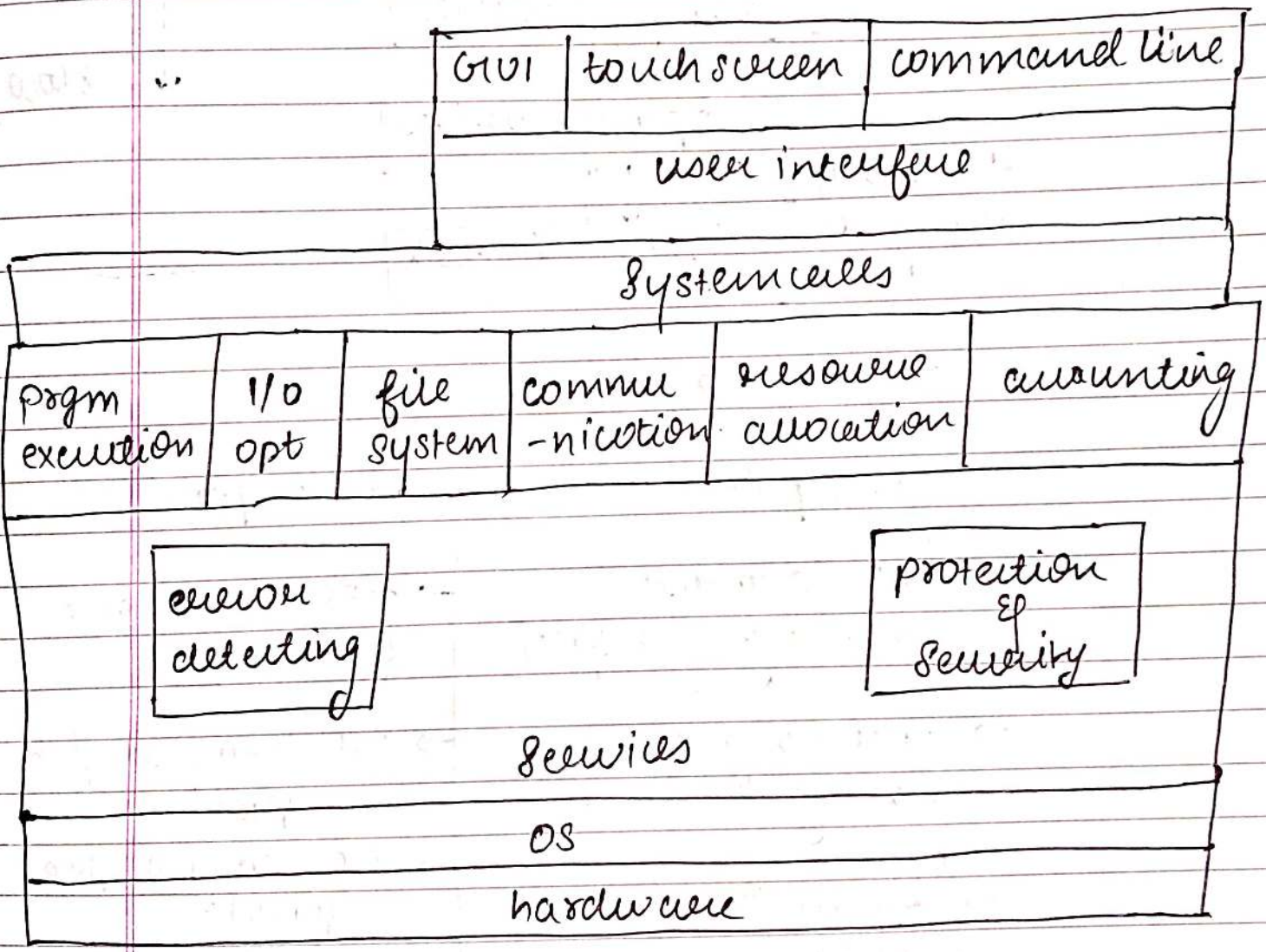
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

Some processes must be loaded into memory before execution.

→ Sequence of booting

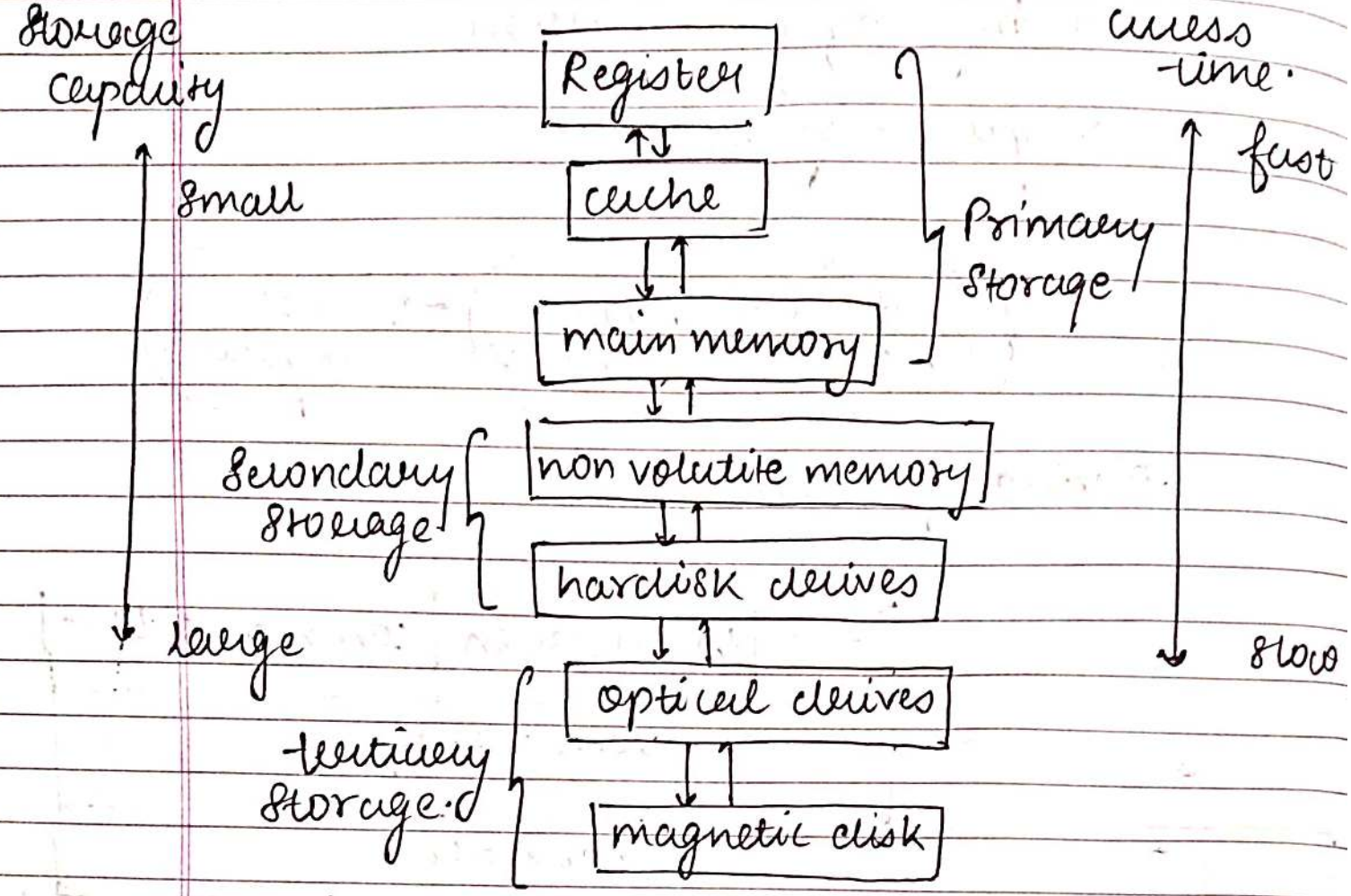


* view of OS services.



This document is available free of charge on studocu

Downloaded by Anil Begar 14 (begaranil88@gmail.com)



* Scheduling algorithm:

Preemptive

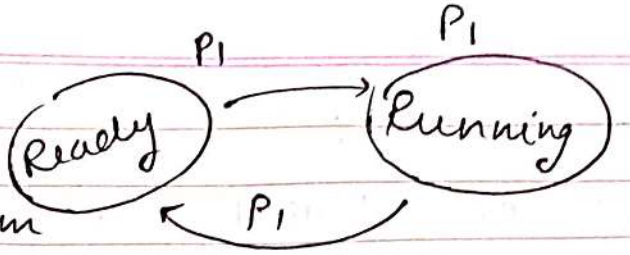
- *** SRFT (Shortest remaining time first)
- LRTF (longest remaining time first)
- * Round Robin.
- Priority based.

Non-pre-emptive.

- FCFS (First come first serve)
- SJF (shortest job first)
- LJF (longest job first)
- HRRN (Highest response Ratio next).
- multilevel queue.
- priority based.

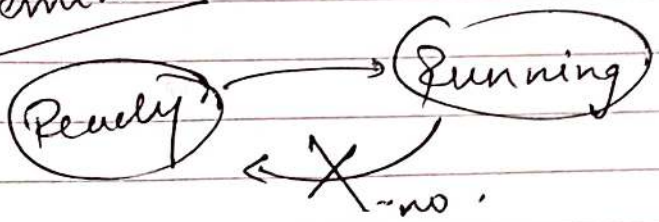
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

Preemptive



Reason :-
1) time quantum
2) priority.

non premitive.



* CPU scheduling

→ Arrival time :- the time at which process enter the Ready queue or state

→ Burst time :- time duration req by a process to get execute on CPU

→ Completion time :- The time at which process gets executed.

→ ~~turnaround~~ turn around time :- 12 11 → 1 hour.
{ completion time - Arrival time }
the whole time. of the difference betwn these two 45 min. ← 60 min − 15 min

→ waiting time :- { turn Around time − burst time }

→ Response time :- { (the time at which a process get CPU first time) − (Arrival time) }

This document is available free of charge on studocu

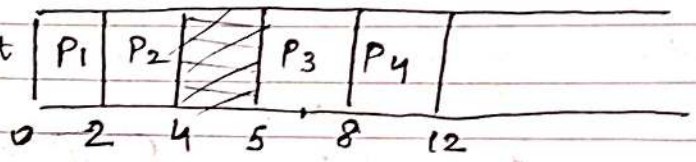
Downloaded by Anil Begar 14 (begaranil88@gmail.com)

* FCFS (first come first serve scheduling Algo)

Process no	Arrival time	Burst time	Comp^n time	TAT	WT	RT
P_1	0	2	2	2	0	0
P_2	1	2	4	3	2	1
P_3	5	3	8	3	0	0
P_4	6	4	12	6	2	2

Criteria "Arrival time"
mode :- Non-Premptive

Gantt chart



At time 12 all processes got executed.

time →

→ what is Avg TAT
$= \frac{14}{4} = 3.5$.

→ Avg WT $= \frac{3}{4} = 0.75$

* Shortest job First Algo (SJF)

Process no	Arival time	Burst time	Completion time	TAT	WT	RT
P1	1	3	6	5	2	
P2	2	4	10	8	4	
P3	1	2	3	2	0	0
P4	4	4	14	10	6	

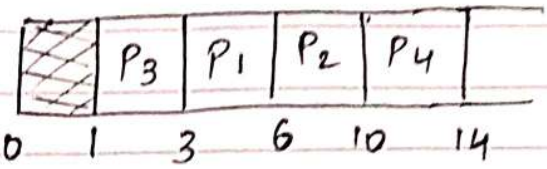
Criteria :- "Burst time"
mode "non premptive".

Downloaded by Anil Begar 14 (begaranil88@gmail.com)

TAT = CT - AT

WT = TAT - BT

Gantt chart



Avg TAT = $\frac{25}{4}$ = 6.25

Avg WT = $\frac{12}{4}$ = 3

$P_1 P_3$ ⟶ time.

$P_1 P_2$

$P_2 P_4$ ↘ burst time same. then check Arrival time.

* SRTF (Shortest remaining time first).

Process no	Arrival time	burst time	complen time	TAT	WT	RT
P_1	0	~~5~~ 3	9	9	4	0
P_2	1	~~3~~ 2 1 0	4	3	0	0
P_3	2	4	13	11	7	7
P_4	4	1	5	1	0	0

Criteria :- Burst time

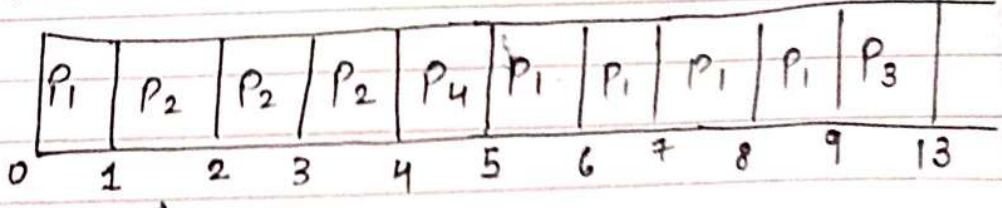
mode :- premptive

TAT = CT - AT

WT = TAT - BT

RT = {CPU first time - AT}.

Gannett chart :-



$P_1 P_2$.

$P_1 P_2 P_3$

$P_1 P_2 P_3$

$P_1 P_2 P_3 P_4$

Avg TAT = 24/4 = 6

Avg WT = 11/4 = 2.75

Avg RT = 7/4 = 1.75.

This document is available free of charge on studocu

Downloaded by Anil Begar 14 (begaranil88@gmail.com)