Assignment Unit-3

N. Madhu Samhitha
2224IA05H0
CSE-C

1) Distinguish between pagin & Segmentation.

Paging	Segmentation.
i) In paging, program is divided into fixed or mounted size pages.	i) In segmentation, program is divided into variable size sections.
ii) OS is accountable for pagin.	ii) Compiler is accountable for segmentation.
iii) Page size determined by hardware	iii) Section size given by user.
iv) faster	iv) Slower.
v) Results in internal fragmentation	v) Results in external fragmentation
vi) Has page table that encloses page address of every page.	vi) Section table maintains section data
vii) Invisible to user	vii) Visible to user
viii) Difficult to apply protection	viii) Easier to apply protection.
ix) logical address is split into page number & page offset	ix) logical address split into section number & offset
x) Paging results in less efficient system	x) segmentation results in more efficient system.

) What are the various page replacement techniques supported by OS.

> First In First Out (FIFO)

- It is the simplest of all page replacement algorithms.
- We maintain a queue of all pages in the memory currently.
- The oldest page in memory is at the frontend of the queue & most recent page at the back or rearend of queue
- A newly requested page is added to rear end & removes oldest page from front end.

> Optimal Page Replacement

- Best page replacement algorithm
- Results in least number of page faults
- pages are replaced with the ones that wont be used for the longest duration.

> Least Recently used (LRU)

- works on basis of principle of locality of a reference.
- It says pages that have been used heavily in the past ~~have the~~ are most likely to be used heavily in future also.
- When page fault occurs, page that has not been used for longest duration is replaced by new page.

3

(b) Compute the number of page faults using FIFO, LRU for 3-frame memory.

Page Refs: 3 2 1 0 3 2 4 3 2 1 0 4 7 5

FIFO

3	2	1	0	3	2	4	3	2	1	0	4	7	5
(3)	3	3	(0)	0	0	(4)	4	4	4	4	4	4	4
	(2)	2	2	(3)	3	3	3	3	(1)	1	1	7	7
		(1)	1	1	(2)	2	2	2	2	(0)	0	0	(5)
✗	✗	✗	✗	✗	✗	✗	(H)	(H)	✗	✗	(H)	✗	✗

No. of page faults = 11

LRU

3	2	1	0	3	2	4	3	2	1	0	4	7	5
3	3	3	0	3	0	4	4	4	4	4	4	7	7
	2	2	2	3	3	3	3	3	1	1	1	1	5
		1	1	1	2	2	2	2	2	0	0	0	0
✗	✗	✗	✗	✗	✗	✗	H	H	✗	✗	H	✗	✗

page faults = 11

3) What are the various page table organizations.

→ Single level page table.
- Straightforward array where each entry directly maps a virtual page number to a physical frame number.

→ Multi level page table.
- Breaks page table into multiple levels, creating tree like hierarchy.
- Reduces memory overhead by allocating memory for required tables.

→ Inverted page table.
- Uses single table for all processes & contains one entry per physical page frame rather than one per virtual page.
- Each entry has info about virtual address & its physical add.

→ Hashed page table
- Use hash-func'n to map virtual page no's to physical page frames.

→ Clustered page tables
- Extension of hashed. But map multiple pages to a single hash table entry.

→ Segmented :- Combined segmentation & paging.

→ Hierarchial :- Organize page tables in layers, (2/3-..) Similar to multi-level but can be extended.

(1)

ASST-4

1)
(a) Explain the DISK scheduling algorithms (i) FIFO, (ii) SSTF, (iii) SCAN, (iv) CSCAN (v) CLOOK.

(i) FCFS :-
- > Simplest of all DISK scheduling algorithms.
- > In FCFS, requests are addressed in the order in they arrive the disk queue.
- > Thus every request gets a fair chance.

(ii) SSTF
- > Shortest seek time first.
- > Requests having shortest seek time are executed first.
- > So seek time of all request is calculated in advance & then scheduled
- > It decreases the avg response time & inc throughput.

(iii) SCAN
- > Disk arm moves in a particular direction & services the request coming in its path after reaching end of disk, it reverses its direction & services the request arriving in its path.
- > Works as an elevator, hence aka. Elevator algorithm.

(iv) CSCAN
- > In SCAN, same path might be scanned again it can be avoided in C-SCAN, disk arm instead of reversing goes to other end of disk & starts servicing requests there.

7

LOOK

> Similar to SCAN but it doesn't go to end of disk but just till last request & reverses direction

V. CLOOK.

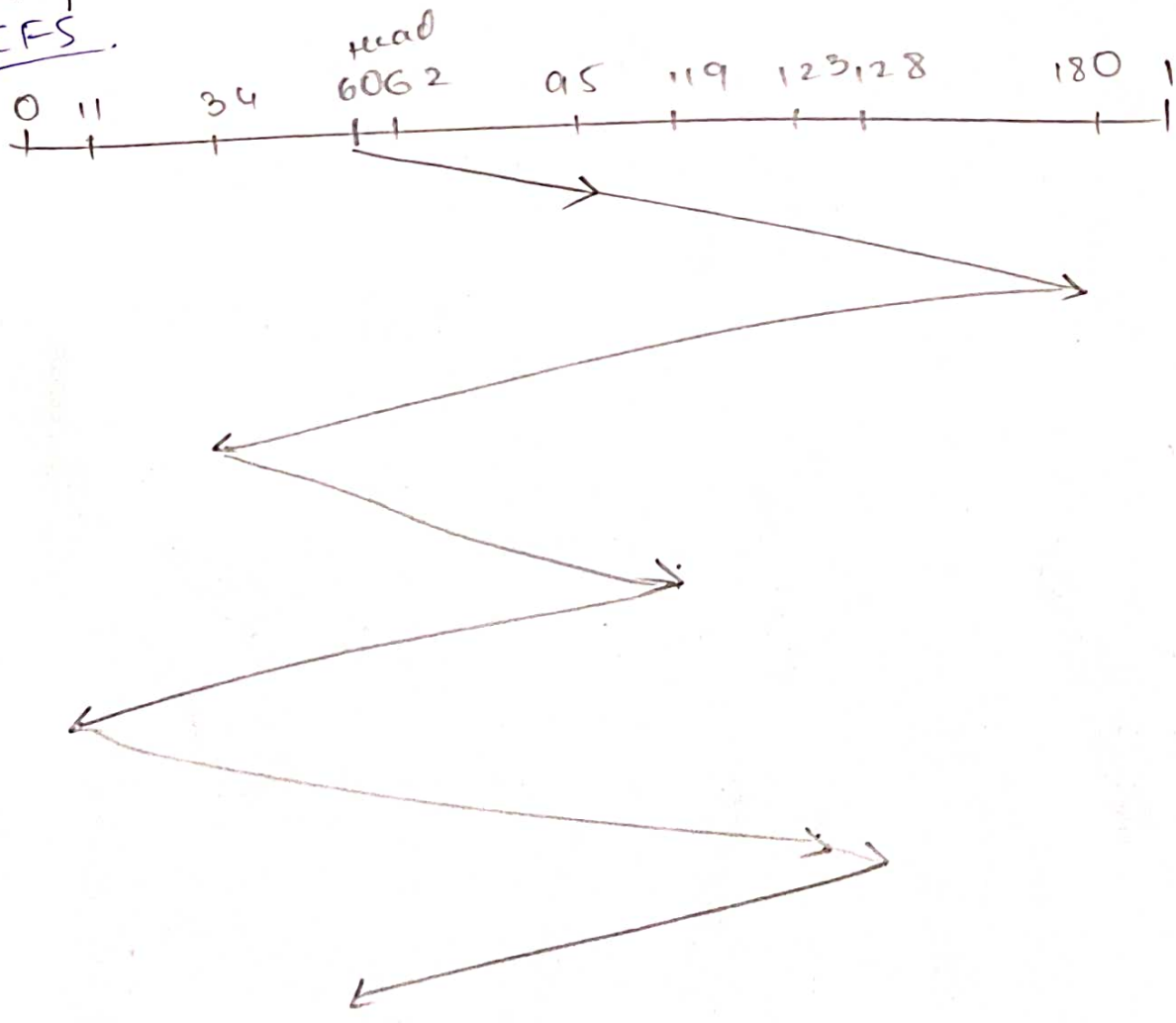
Similar to CSCAN, doesn't go till end but till last request.

1) (b) head → 60th cylinder. 11, 34, 62, 95, 119, 123, 123, 180

Requests → 95, 180, 34, 119, 11, 123, 128, 62.

Compute head movements

FCFS.

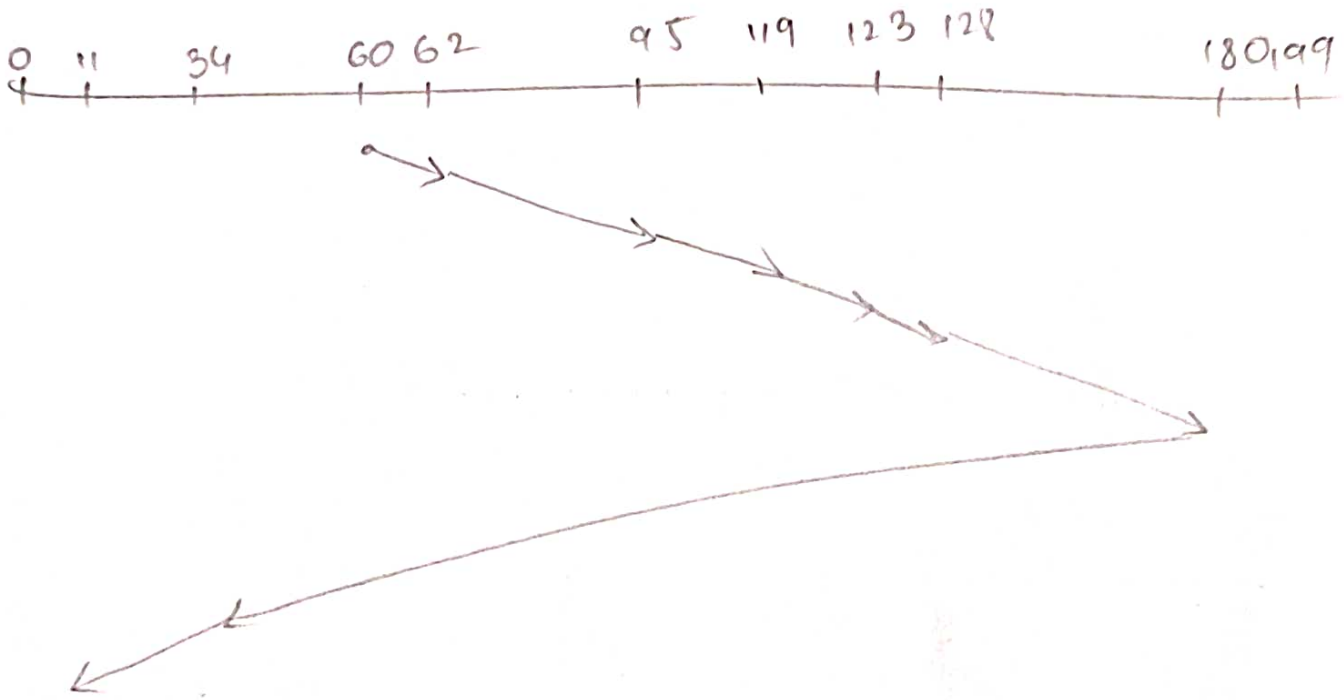


Total head movements

(95-60) + (180-95) + (180-34) + (119-34) + (119-11) +
(123-11) + (128-123) + (128-62) = 642

SSTF

Order:- 60, 62, 95, 119, 123, 128, 180, 34, 11

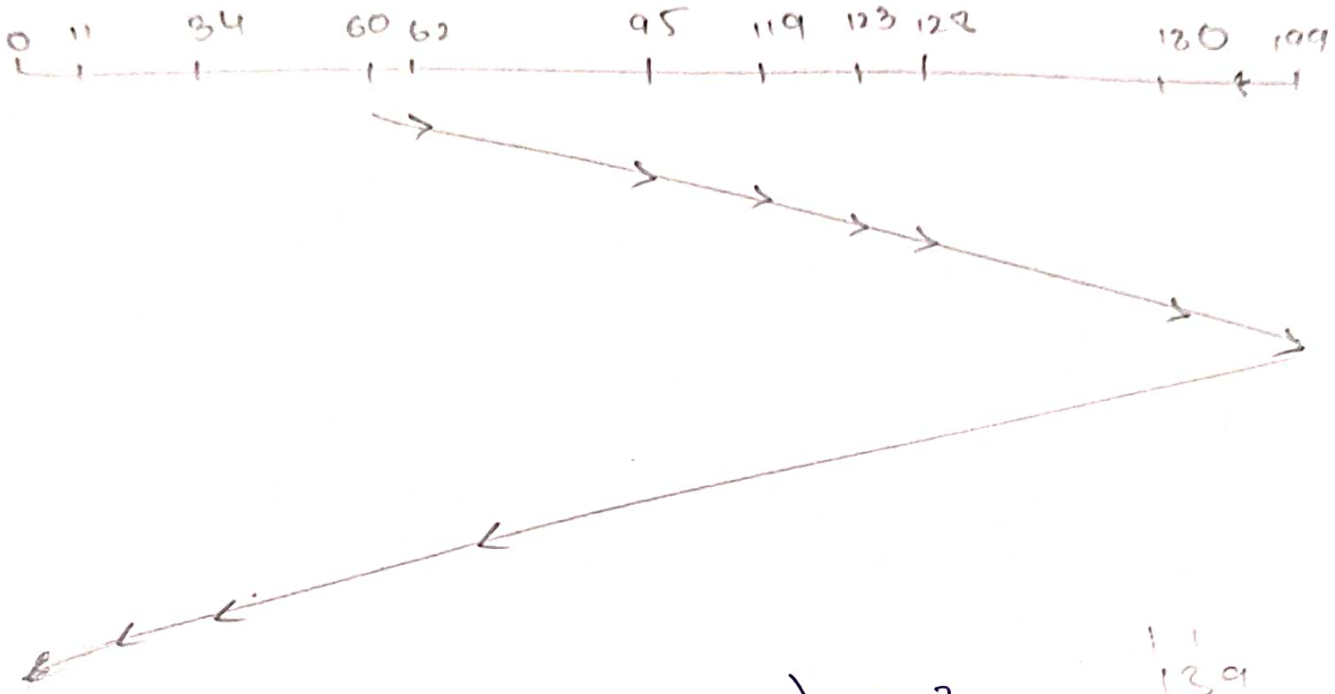


Total head movements:-

(62-60) + (95-62) + (119-95) + (123-119) + (128-123)
+ (180-128) + (180-34) + (34-11) = 289

4

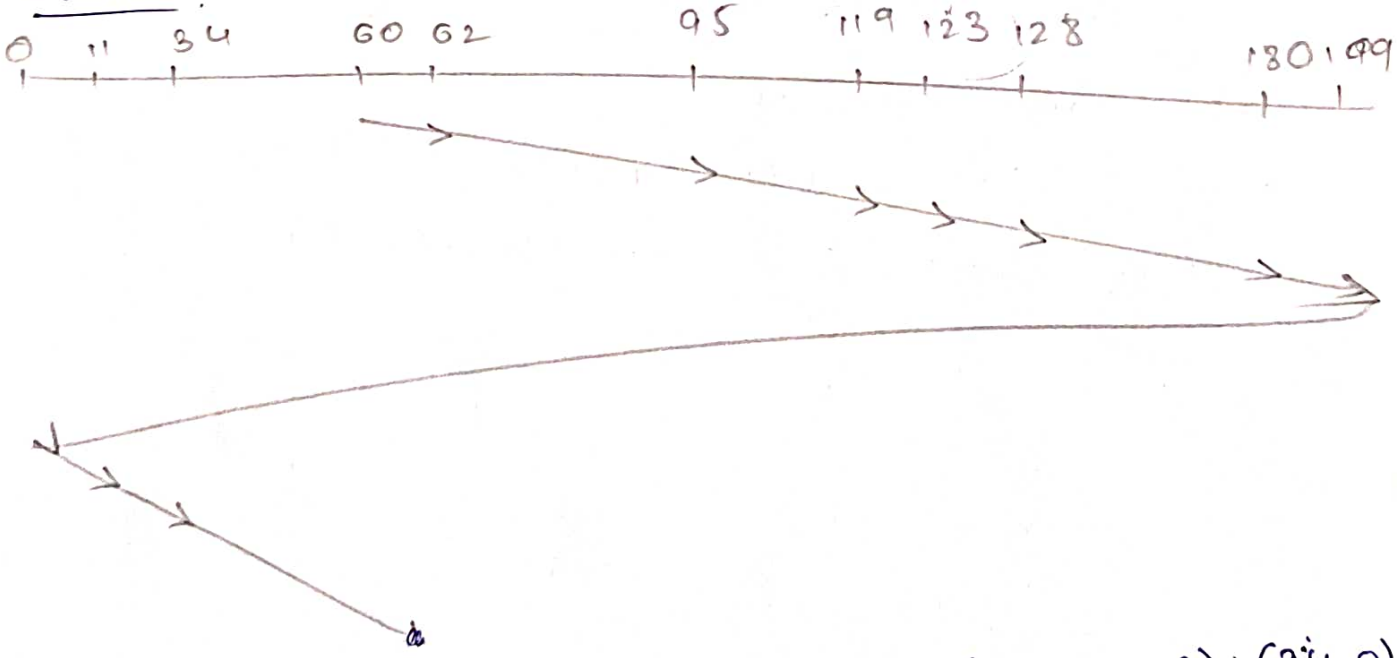
SCAN



THM = (199 - 60) + (199 - 11) = 327

11
139
188
327

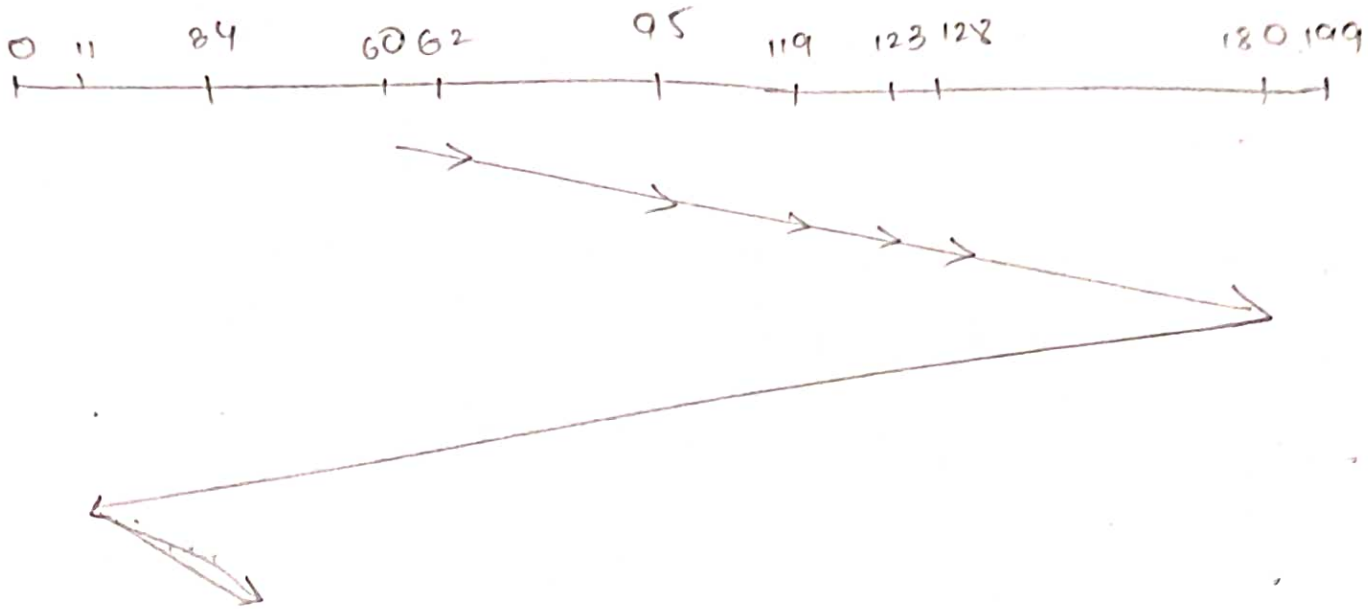
CSCAN



Total head movement = (199 - 60) + (199 - 0) + (34 - 0)

= 372

C-LOOK.



$$(180-60) + (180-11) + (34-11) = 312$$

2) Explain Swap-space management.

> Swapping is a memory-management technique used in multi-programming to increase the no. of processes sharing the CPU.

> Virtual memory uses disk space as extension of main memory.

> It is a technique of removing a process from main memory to secondary memory (swap out) & moving ~~back to me~~ from secondary memory to main memory (swap in).

> To optimize memory usage & improve system performance.

> Swap-space management allows OS to use hard disk space as virtual memory, effectively inc memory cap.

Q.

- By using Virtual M, OS can swap out less frequently used data from physical memory to disk, freeing up space for more frequently used data

3) What are the operations supported by files, File access method advantages & disadvantages.

→ Operations supported by Files

1. Creation
2. Opening.
3. Reading.
4. Writing
5. Closing.
6. Deleting.
7. Truncating.
8. Appending.

→ File Access methods.

1. Sequential access.

Adv:-

- Easy to implement & understand
- Optimal for large continguous block of data
- Minimal control information required

Disadv:-

- Not efficient where random access to data is needed.
- can be slow if req data is near end of large file.

(f)

2. Random Access

Adv:-
- Allows immediate access to any part of file.
- Reduces access time.

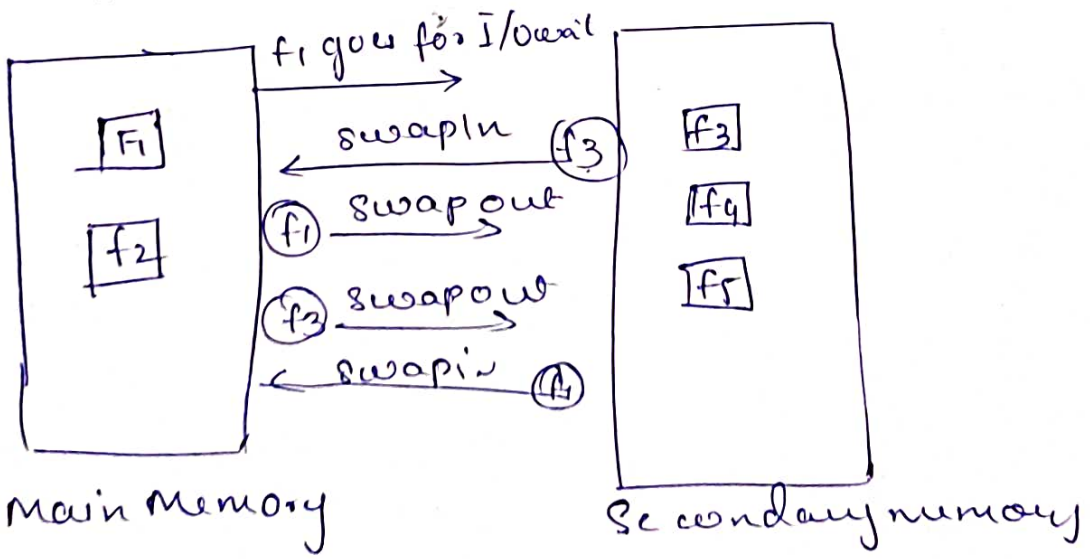
Disadv:-
- More complex to implement
- can lead to fragmentation & degrade performance

3. Indexed Access.

Adv:-
- Speeds up searches & access times.
- can handle both random & sequential access patterns

Disadv
- Requires additional storage
- More complex (update index)

4) Discuss design & implementation of swap space management with example.

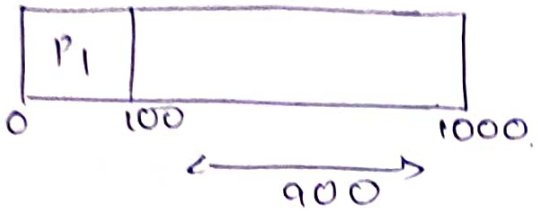


Example

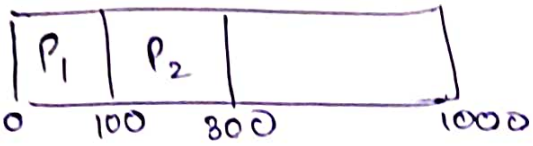
Swap Space

Memory.

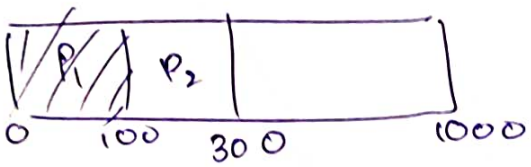
Let $P_1 = 100$

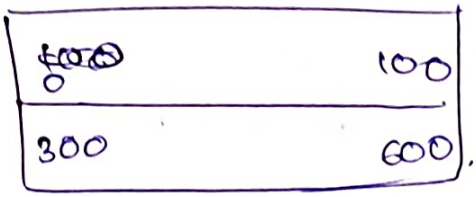


$P_2 = 200$

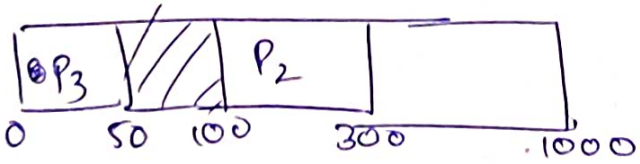


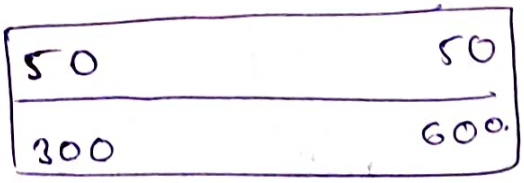
P_1 is sent back to main memory





$P_3 = 50$





(9)

5) Write short note on

I) Contiguous Allocation

II) Linked "

III) Indexed "

IV) Free-space management methods.

I) Contiguous Allocation:-

> Each file occupies set of contiguous block on disk.

> Each block is next to each other

> fast

> Easy to calculate addresses.

II) Linked Allocation.

> Stores each file as linked list of disk blocks

> Each block contains pointer to next block

> Blocks can be scattered.

> Files can easily grow & shrink.

III) Indexed Allocation:-

> Uses index block to keep track of disk blocks alloc to file.

> Each file has index block → points to actual data block

IV) free space management methods.

- Used to keep track of free sp disk space, so it can be allocated to new files.

common methods

1. Bit map → 0 - free, 1 - alloc
2. Linked list → of free blocks.
3. Grouping
4. Counting → first block & no. of free blocks.