IARE '24 DISCORD TEAM

II-II

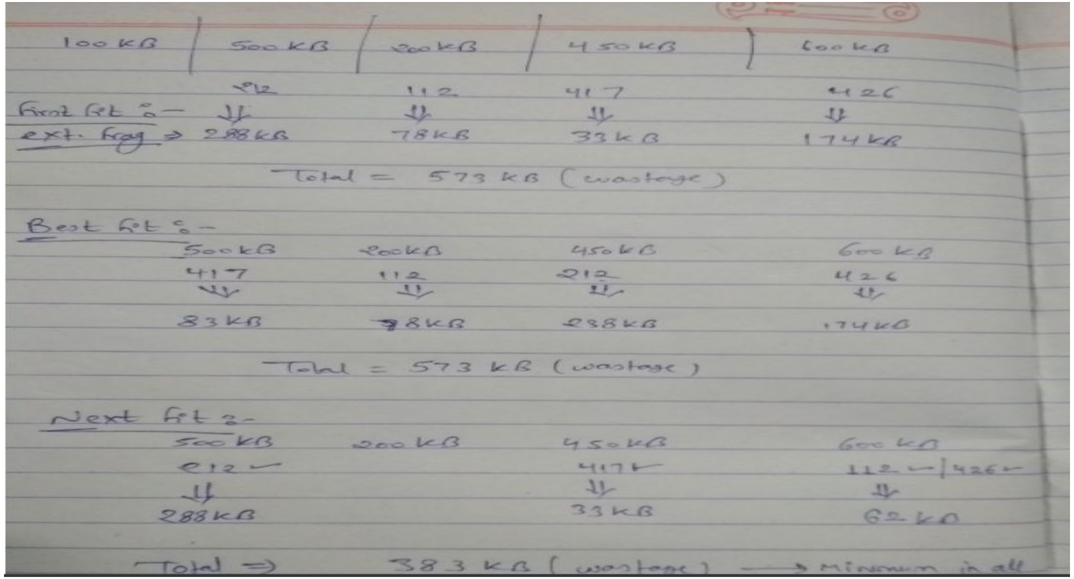
OS MODULE 3 SOLUTIONS

SUHRUTH • ANUSHKA • VISHAL • IKRAM • UJJWAL

PART A- PROBLEM SOLVING AND CRITICAL THINKING

Q1) Memory partitions of 100kb,500 kb,200 kb,300kb,600 kb are available .How would 7 best ,worst, first fit algorithms place processes 212,417,112,426 in order. Which is the best algorithm?

Ans:



100 KB	500 KB	200 KB	450 KB	600 KB

First fit :-

	212	112	417	426
	⇓	⇓	⇓	⇓
ext. frag ⇒	288KB	78KB	33KB	174KB

Total = 573 KB (wastage)

Best fit :-

500KB	200KB	450KB	600KB
417	112	212	426
⇓	⇓	⇓	⇓
83KB	88KB	238KB	174KB

Total = 573 KB (wastage)

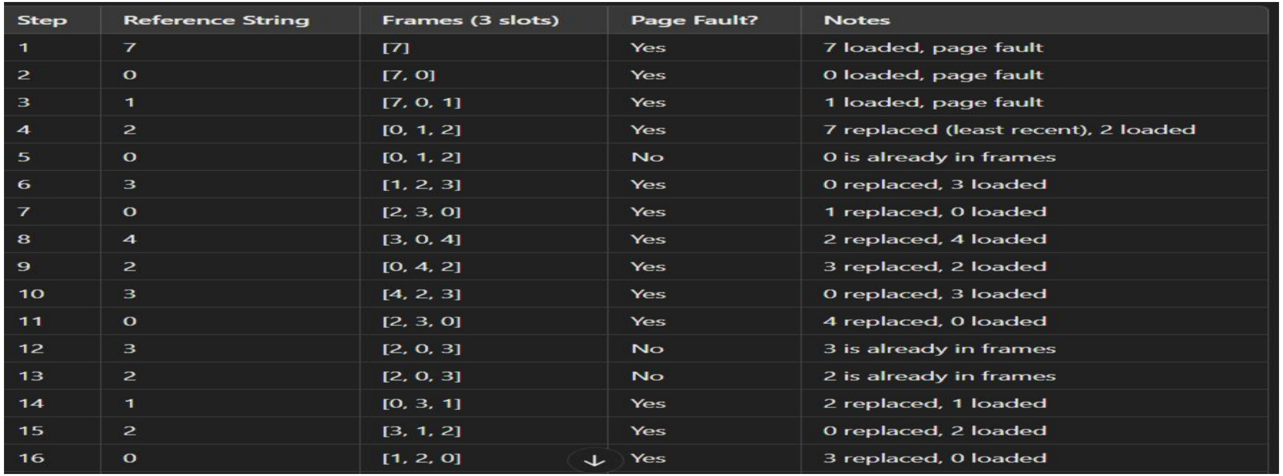
Next fit :-

500KB	200KB	450KB	600KB
212 ✓		417 ✓	112 ✓ / 426 ✓
⇓		⇓	⇓
288KB		33KB	62KB

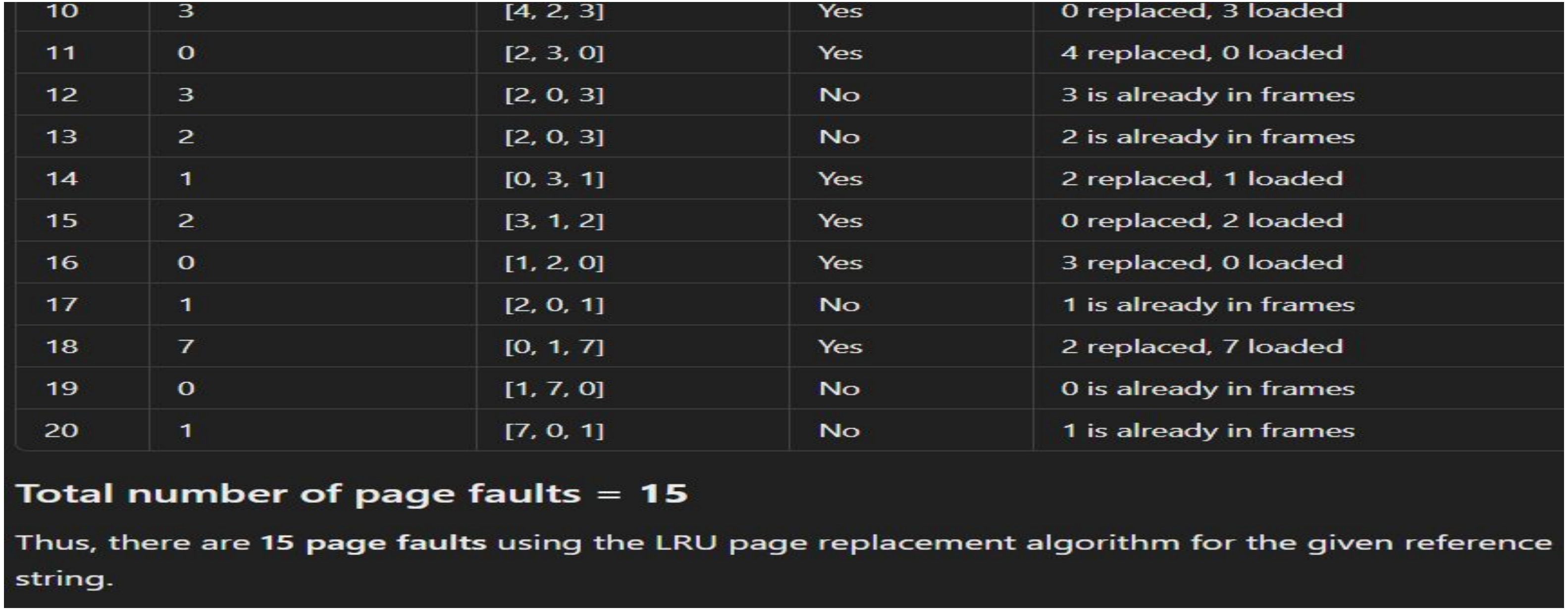
Total ⇒ 383 KB (wastage) ⟶ Minimum in all

Q2) Describe the LRU page replacement algorithm, assuming there are 3 frames and the page reference string is 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1.Find the number of page faults.

Ans:



Step	Reference String	Frames (3 slots)	Page Fault?	Notes
1	7	[7]	Yes	7 loaded, page fault
2	0	[7, 0]	Yes	0 loaded, page fault
3	1	[7, 0, 1]	Yes	1 loaded, page fault
4	2	[0, 1, 2]	Yes	7 replaced (least recent), 2 loaded
5	0	[0, 1, 2]	No	0 is already in frames
6	3	[1, 2, 3]	Yes	0 replaced, 3 loaded
7	0	[2, 3, 0]	Yes	1 replaced, 0 loaded
8	4	[3, 0, 4]	Yes	2 replaced, 4 loaded
9	2	[0, 4, 2]	Yes	3 replaced, 2 loaded
10	3	[4, 2, 3]	Yes	0 replaced, 3 loaded
11	0	[2, 3, 0]	Yes	4 replaced, 0 loaded
12	3	[2, 0, 3]	No	3 is already in frames
13	2	[2, 0, 3]	No	2 is already in frames
14	1	[0, 3, 1]	Yes	2 replaced, 1 loaded
15	2	[3, 1, 2]	Yes	0 replaced, 2 loaded
16	0	[1, 2, 0]	Yes	3 replaced, 0 loaded



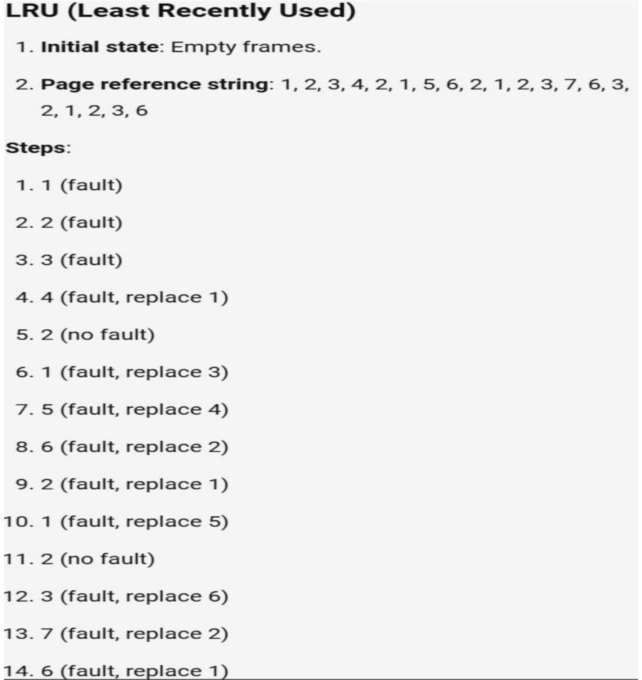
10	3	[4, 2, 3]	Yes	0 replaced, 3 loaded
11	0	[2, 3, 0]	Yes	4 replaced, 0 loaded
12	3	[2, 0, 3]	No	3 is already in frames
13	2	[2, 0, 3]	No	2 is already in frames
14	1	[0, 3, 1]	Yes	2 replaced, 1 loaded
15	2	[3, 1, 2]	Yes	0 replaced, 2 loaded
16	0	[1, 2, 0]	Yes	3 replaced, 0 loaded
17	1	[2, 0, 1]	No	1 is already in frames
18	7	[0, 1, 7]	Yes	2 replaced, 7 loaded
19	0	[1, 7, 0]	No	0 is already in frames
20	1	[7, 0, 1]	No	1 is already in frames

Total number of page faults = 15

Thus, there are **15 page faults** using the LRU page replacement algorithm for the given reference string.

Q3) Consider the following page reference string 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6 .Find out the number of page faults a) LRU b) FIFO

Ans:

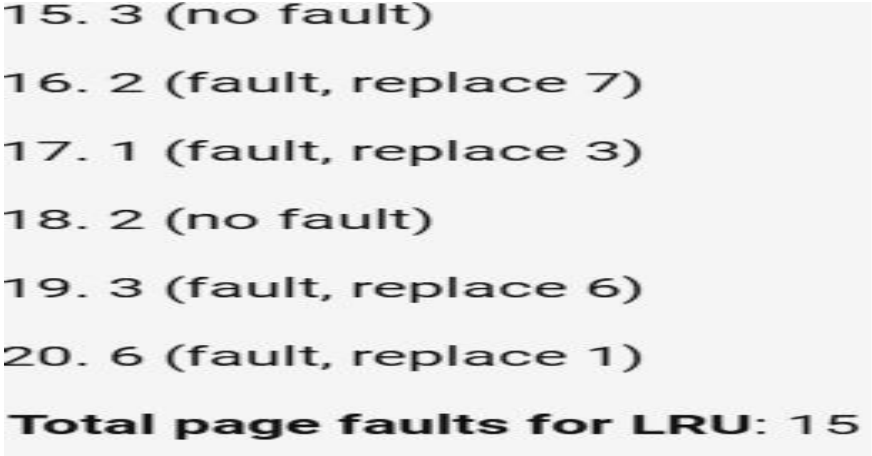


LRU (Least Recently Used)

1. **Initial state**: Empty frames.
2. **Page reference string**: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

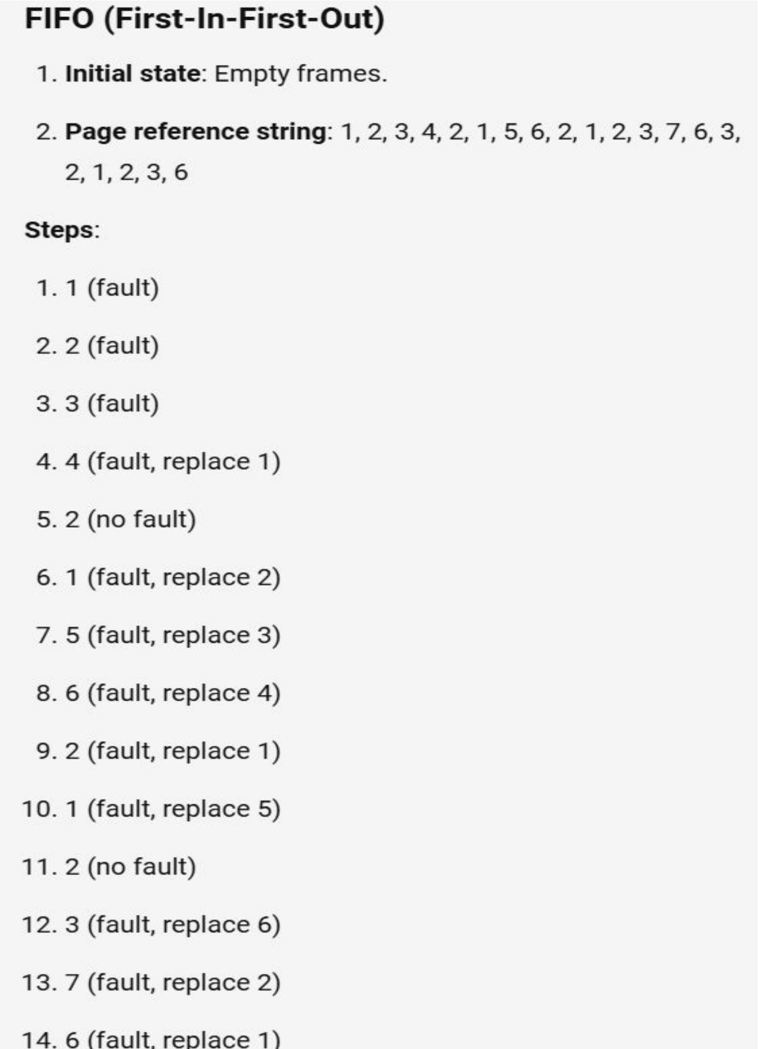
Steps:

1. 1 (fault)
2. 2 (fault)
3. 3 (fault)
4. 4 (fault, replace 1)
5. 2 (no fault)
6. 1 (fault, replace 3)
7. 5 (fault, replace 4)
8. 6 (fault, replace 2)
9. 2 (fault, replace 1)
10. 1 (fault, replace 5)
11. 2 (no fault)
12. 3 (fault, replace 6)
13. 7 (fault, replace 2)
14. 6 (fault, replace 1)



15. 3 (no fault)
16. 2 (fault, replace 7)
17. 1 (fault, replace 3)
18. 2 (no fault)
19. 3 (fault, replace 6)
20. 6 (fault, replace 1)

Total page faults for LRU: 15

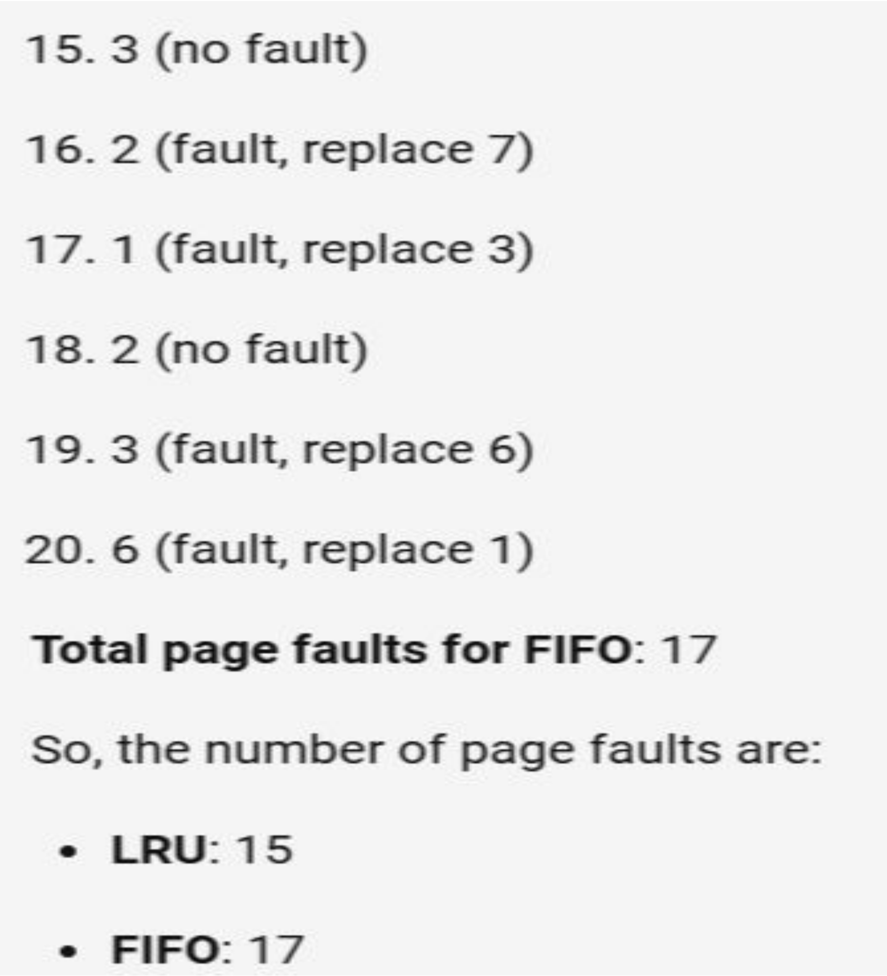


FIFO (First-In-First-Out)

1. **Initial state**: Empty frames.
2. **Page reference string**: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

Steps:

1. 1 (fault)
2. 2 (fault)
3. 3 (fault)
4. 4 (fault, replace 1)
5. 2 (no fault)
6. 1 (fault, replace 2)
7. 5 (fault, replace 3)
8. 6 (fault, replace 4)
9. 2 (fault, replace 1)
10. 1 (fault, replace 5)
11. 2 (no fault)
12. 3 (fault, replace 6)
13. 7 (fault, replace 2)
14. 6 (fault, replace 1)



15. 3 (no fault)
16. 2 (fault, replace 7)
17. 1 (fault, replace 3)
18. 2 (no fault)
19. 3 (fault, replace 6)
20. 6 (fault, replace 1)

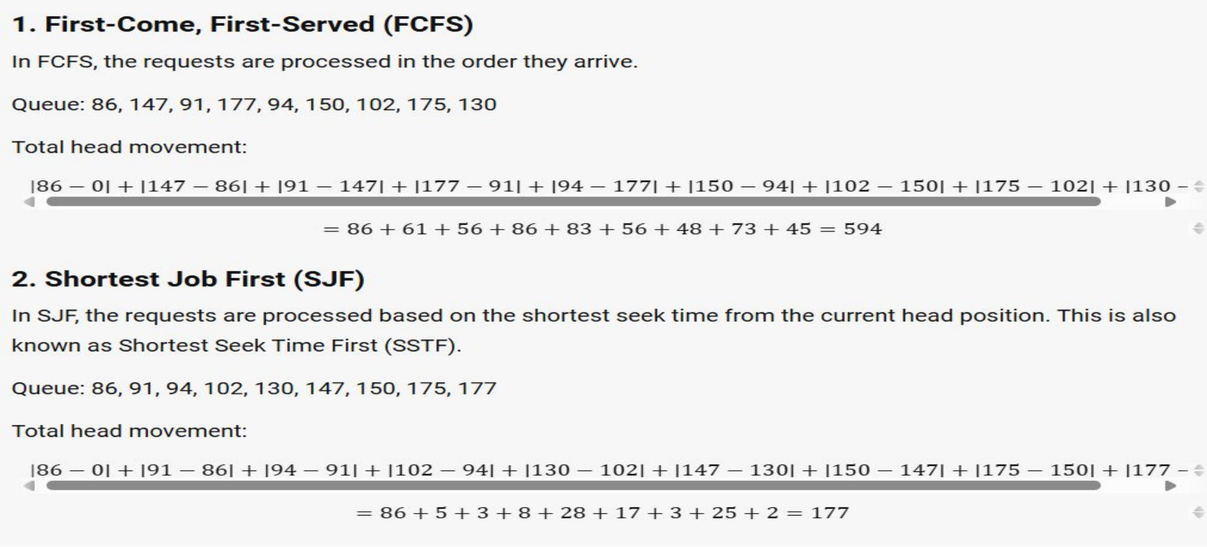
Total page faults for FIFO: 17

So, the number of page faults are:

- **LRU**: 15
- **FIFO**: 17

Q4) The queue of requests in FIFO is 86,147,91,177,94,150,102,175,130 What is the total head movement needed to satisfy the requests for the following Scheduling algorithms FCFS, SJF, SCAN

Ans:



1. First-Come, First-Served (FCFS)

In FCFS, the requests are processed in the order they arrive.

Queue: 86, 147, 91, 177, 94, 150, 102, 175, 130

Total head movement:

$$|86 - 0| + |147 - 86| + |91 - 147| + |177 - 91| + |94 - 177| + |150 - 94| + |102 - 150| + |175 - 102| + |130 -$$

$$= 86 + 61 + 56 + 86 + 83 + 56 + 48 + 73 + 45 = 594$$

2. Shortest Job First (SJF)

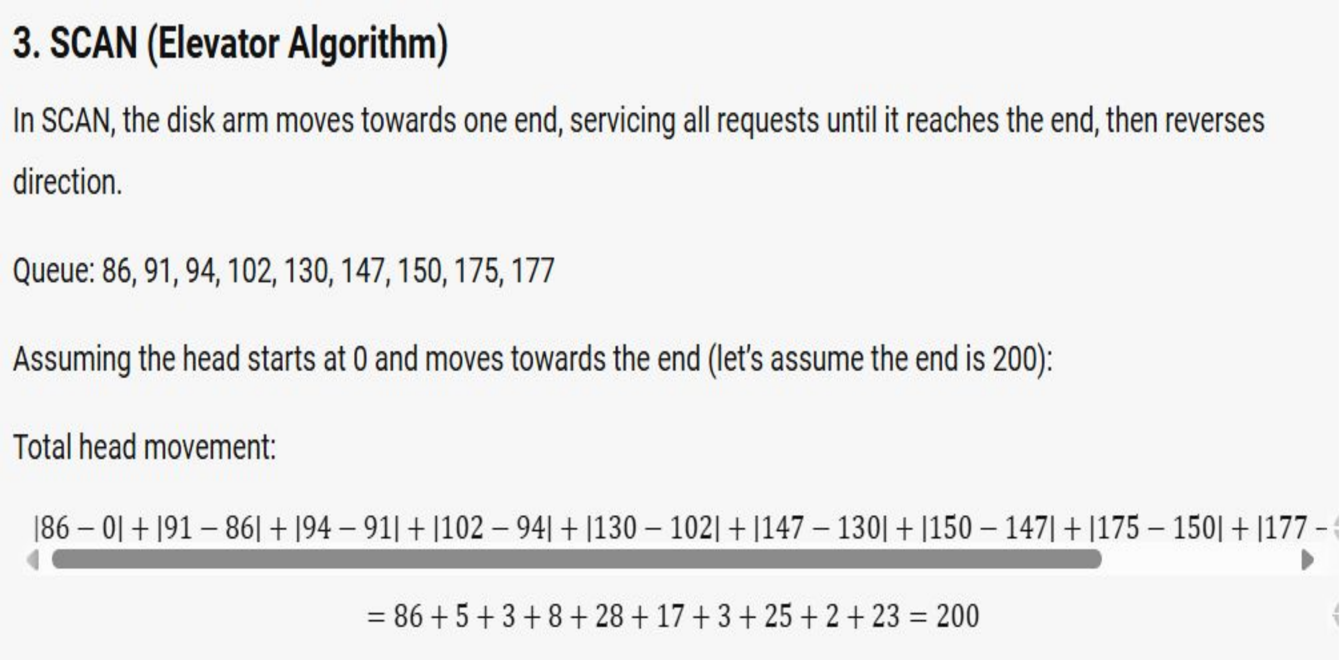
In SJF, the requests are processed based on the shortest seek time from the current head position. This is also known as Shortest Seek Time First (SSTF).

Queue: 86, 91, 94, 102, 130, 147, 150, 175, 177

Total head movement:

$$|86 - 0| + |91 - 86| + |94 - 91| + |102 - 94| + |130 - 102| + |147 - 130| + |150 - 147| + |175 - 150| + |177 -$$

$$= 86 + 5 + 3 + 8 + 28 + 17 + 3 + 25 + 2 = 177$$



3. SCAN (Elevator Algorithm)

In SCAN, the disk arm moves towards one end, servicing all requests until it reaches the end, then reverses direction.

Queue: 86, 91, 94, 102, 130, 147, 150, 175, 177

Assuming the head starts at 0 and moves towards the end (let's assume the end is 200):

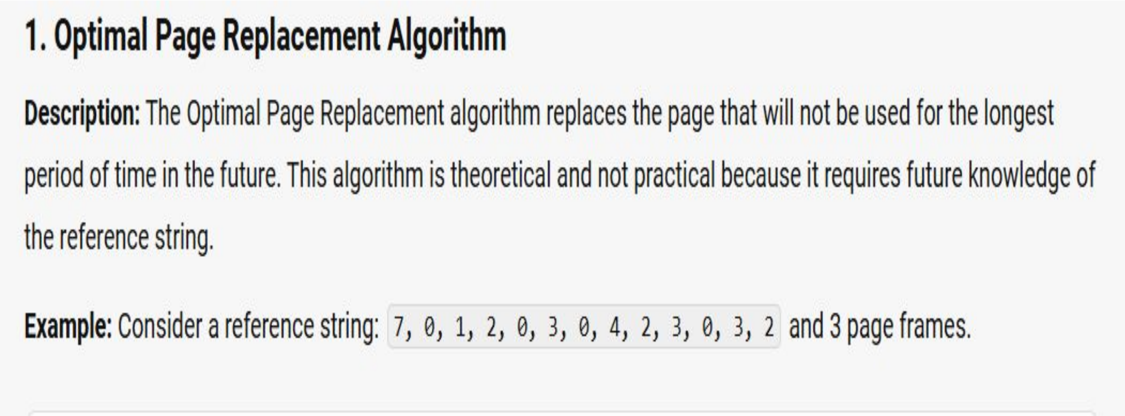
Total head movement:

$$|86-0| + |91-86| + |94-91| + |102-94| + |130-102| + |147-130| + |150-147| + |175-150| + |177 -$$

$$= 86 + 5 + 3 + 8 + 28 + 17 + 3 + 25 + 2 + 23 = 200$$

Q5) Discuss the following page replacement algorithm with an example i) Optimal ii) LRU

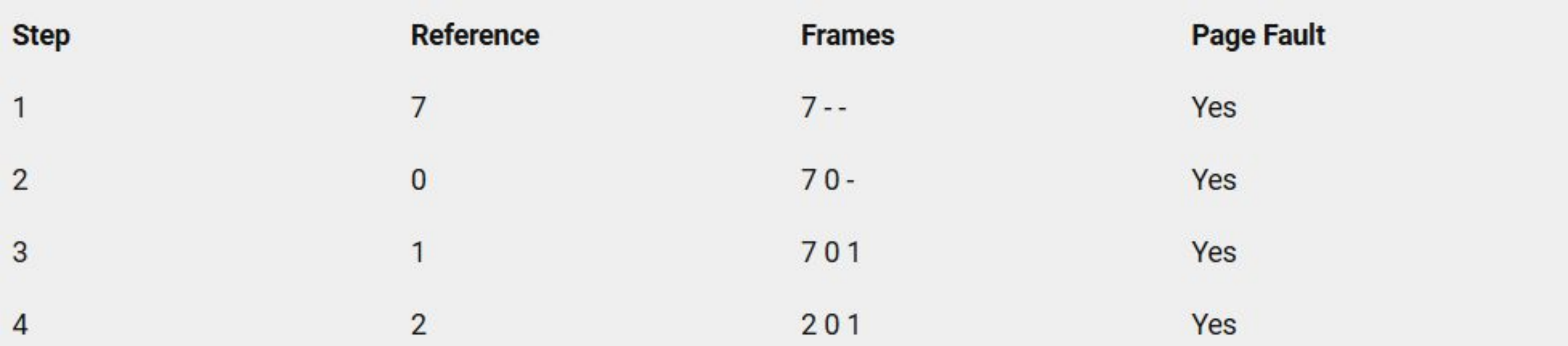
Ans:



1. Optimal Page Replacement Algorithm

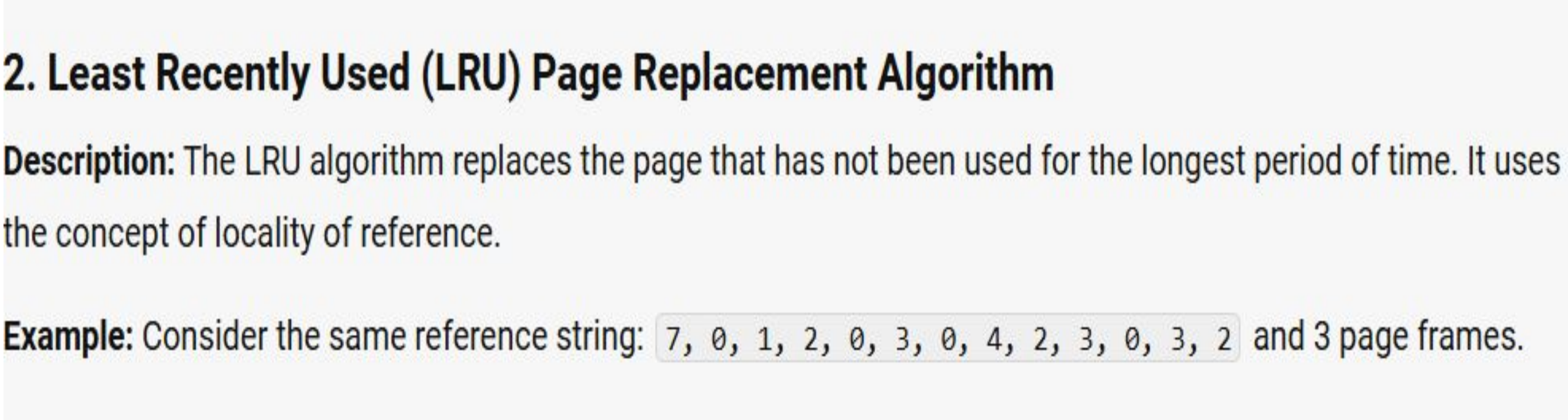
Description: The Optimal Page Replacement algorithm replaces the page that will not be used for the longest period of time in the future. This algorithm is theoretical and not practical because it requires future knowledge of the reference string.

Example: Consider a reference string: `7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2` and 3 page frames.



Step	Reference	Frames	Page Fault
1	7	7 - -	Yes
2	0	7 0 -	Yes
3	1	7 0 1	Yes
4	2	2 0 1	Yes

This example of optimal page replacement are 9 faults



2. Least Recently Used (LRU) Page Replacement Algorithm

Description: The LRU algorithm replaces the page that has not been used for the longest period of time. It uses the concept of locality of reference.

Example: Consider the same reference string: `7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2` and 3 page frames.

PART B-LONG ANSWER QUESTIONS

Q1) Describe the following. a) Virtual Memory b) Cache Memory

Ans: Virtual Memory 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. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory serves two purposes. First, it allows us to extend the use of physical memory by using disk. Second, it allows us to have memory protection, because each virtual address is translated to a physical address.

Cache Memory The data or contents of the main memory that are used again and again by CPU, are stored in the cache memory so that we can easily access that data in shorter time. Whenever the CPU needs to access memory, it first checks the cache memory. If the data is not found in cache memory then the CPU moves onto the main memory. It also transfers block of recent data into the cache and keeps on deleting the old data in cache to accommodate the new one.

Q2) What is paging and swapping?

Ans:Paging:

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non – contiguous.

▪ Logical Address or Virtual Address (represented in bits): An address generated by the CPU

▪ Logical Address Space or Virtual Address Space(represented in words or bytes): The set of all logical addresses generated by a program

▪ Physical Address (represented in bits): An address actually available on memory unit ▪ Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses
Example:

▪ If Logical Address = 31 bit, then Logical Address Space = 231 words = 2 G words (1 G = 230)

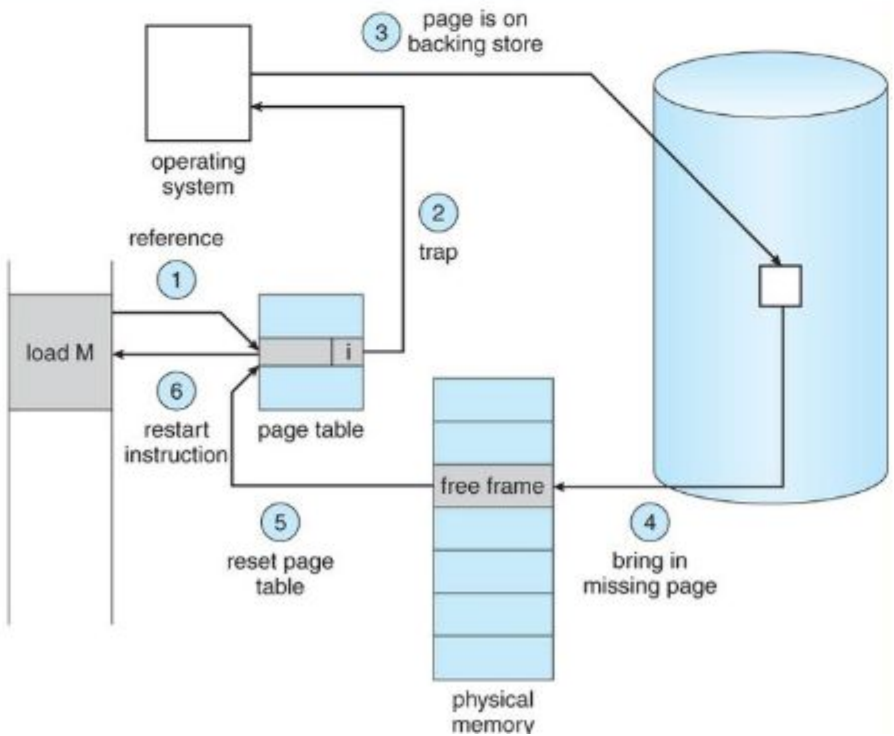
Swapping :

Swapping is a mechanism in which a process can be swapped temporarily out of main memory (or move) to secondary storage (disk) and make that memory available to other processes. At some later time, the system swaps back the process from the secondary storage to main memory. Though performance is usually affected by swapping process but it helps in running multiple and big processes in parallel and that's the reason Swapping is also known as a technique for memory compaction.

Q3) With a diagram discuss the steps involved in handling a page fault.

Ans: **Steps in Handling a Page Fault**

1. **Trap to the Operating System**: When a page fault occurs, the hardware traps to the operating system, saving the program counter and other registers.
2. **Determine the Cause**: The operating system determines whether the fault was caused by an invalid memory access or if the page simply needs to be loaded from disk.
3. **Check Validity**: If the memory access is invalid, the process is terminated. If the access is valid, the operating system proceeds to the next step.
4. **Find a Free Frame**: The operating system looks for a free frame in physical memory. If no free frame is available, it uses a page replacement algorithm to free up a frame.
5. **Schedule Disk Operation**: The operating system schedules a disk operation to read the required page into the free frame.
6. **Update Page Table**: Once the page is loaded into memory, the operating system updates the page table to reflect the new location of the page.
7. **Resume Execution**: The process is resumed from the point where the page fault occurred.


3
page is on backing store
operating system
2
trap
reference
1
load M
i
page table
6
restart instruction
free frame
5
reset page table
4
bring in missing page
physical memory

Q4) Describe a) Paging b) Page table structure

Ans: Paging: Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non – contiguous.

- Logical Address or Virtual Address (represented in bits): An address generated by the CPU

- Logical Address Space or Virtual Address Space(represented in words or bytes): The set of all logical addresses generated by a program

- Physical Address (represented in bits): An address actually available on memory unit ▪ Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses

Example:

- If Logical Address = 31 bit, then Logical Address Space = 231 words = 2 G words (1 G = 230)

Page Table:

A page table is the data structure used by a virtual memory system in a computer operating system to store the mapping between virtual addresses and physical addresses. Virtual addresses are used by the program executed by the accessing process, while physical addresses are used by the hardware, or more specifically, by the RAM subsystem.

Q5) Explain in detail the requirements that memory management technique needs to satisfy

Ans: Relocation - the user should not have to know where the program is going to be located. The hardware and OS must work together in this process of relocation. – Programmer does not know where the program will be placed in memory when it is executed – While the program is executing, it may be swapped to disk and returned to main memory at a different location (relocated) – Memory references must be translated in the code to actual physical memory address

Protection - keep the process from accessing the address space of another process – Processes should not be able to reference memory locations in another process without permission – Impossible to check absolute addresses at compile time – Must be checked at run time – Memory protection requirement must be satisfied by the processor (hardware) rather than the operating system (software) • Operating system cannot anticipate all of the memory references a program will make

Sharing - this cooperation is necessary in an OS: shared data areas, share code (DDL) - Allow several processes to access the same portion of memory – Better to allow each process access to the same copy of the program rather than have their own separate copy

Logical organization - processes are composed of modules or varying sizes, modules independently compiled, modules with different protection needs even to the degree of sharing. – Programs are written in modules – Modules can be written and compiled independently – Different degrees of protection given to modules (read-only, execute-only) – Share modules among processes

Physical organization - typically a two level organization: main memory and secondary memory. The user/programmer cannot know how a program or process will be split across the levels. – Memory available for a program plus its data may be insufficient • Overlaying allows various modules to be assigned the same region of memory - Programmer does not know how much space will be available

Q6) Describe a) Translation lookaside buffer b) Segmentation

Ans: (a)Translation lookaside buffer:

A translation lookaside buffer (TLB) is a memory cache that stores recent translations of virtual memory to physical addresses for faster retrieval. When a virtual memory address is referenced by a program, the search starts in the CPU. First, instruction caches are checked. If the required memory is not in these very fast caches, the system has to look up the memory’s physical address. At this point, TLB is checked for a quick reference to the location in physical memory.

(b)Segmentation:

A Memory Management technique in which memory is divided into variable sized chunks which can be allocated to processes. Each chunk is called a Segment. A table stores the information about all such segments and is called Segment Table

Q7) Describe contiguous memory allocation concept with advantages and disadvantages

Ans: ▪ Contiguous memory allocation is a classical memory allocation model that assigns a process consecutive memory blocks (that is, memory blocks having consecutive addresses).

▪ Contiguous memory allocation is one of the oldest memory allocation schemes. When a process needs to execute, memory is requested by the process. The size of the process is compared with the amount of contiguous main memory available to execute the process. If sufficient contiguous memory is found, the process is allocated memory to start its execution. Otherwise, it is added to a queue of waiting processes until sufficient free contiguous memory is available.

▪ The contiguous memory allocation scheme can be implemented in operating systems with the help of two registers, known as the base and limit registers.

▪ When a process is executing in main memory, its base register contains the starting address of the memory location where the process is executing, while the amount of bytes consumed by the process is stored in the limit register.

▪ The CPU generates the logical or virtual address, which is converted into an actual address with the help of the memory management unit (MMU). The base address register is used for address translation by the MMU. Thus, a physical address is calculated as follows: ▪ Physical Address = Base register address + Logical address/Virtual address

Advantages: 1) This Memory allocation provides the direct and easy access.

2) Number of disk required in type of memory allocation is reduced to minimum

3) In case of contiguous memory allocation the good performance remains a positive factor.

Disadvantages:

1) For new files it is very difficult to find the spaces here.

2) Furthermore you can't extend the file .

3)The one big disadvantage is the difficulty about fragmentation

Q8) Describe Hierarchical paging .

Ans:Hierarchical –

- A multi-tiered table which breaks up the virtual address into multiple parts

- Hierarchical reduces that memory a lot by only adding subtables that are actually in use.

- Still, every process has a root page table. And if the memory footprint of the processes is scattered, there may still be a lot of unnecessary entries in secondary tables.

- This is a far better solution regarding memory than Basic and introduces only a marginal computation increase

Q9) Describe Inverted page Table

Ans: Inverted page Tables

- An alternate approach is to use the Inverted Page Table structure that consists of one page table entry for every frame of the main memory.

- So the number of page table entries in the Inverted Page Table reduces to the number of frames in physical memory and a single page table is used to represent the paging information of all the processes.

- Through inverted page table, the overhead of storing an individual pagetable for every process gets eliminated and only a fixed portion of memory is required to store the paging information of all the processes together.

- This technique is called as inverted paging as the indexing is done with respect to the frame number instead of the logical page number. ● Each entry in the page table contains the following fields.

- Page number – It specifies the page number range of the logical address.

- Process id – An inverted page table contains the address space information of all the processes in execution. Since two different processes can have similar set of virtual addresses, it becomes necessary in Inverted Page Table to store a process Id of each process to identify it’s address space uniquely. This is done by using the combination of PId and Page Number. So this Process Id acts as an address space identifier and ensures that a virtual page for a particular process is mapped correctly to the corresponding physical frame.

- Control bits – These bits are used to store extra paging-related information. These include the valid bit, dirty bit, reference bits, protection and locking information bits.

- Chained pointer – It may be possible sometime that two or more processes share a part of main memory. In this case, two or more logical pages map to same Page Table Entry then a chaining pointer is used to map the details of these logical pages to the root page table.

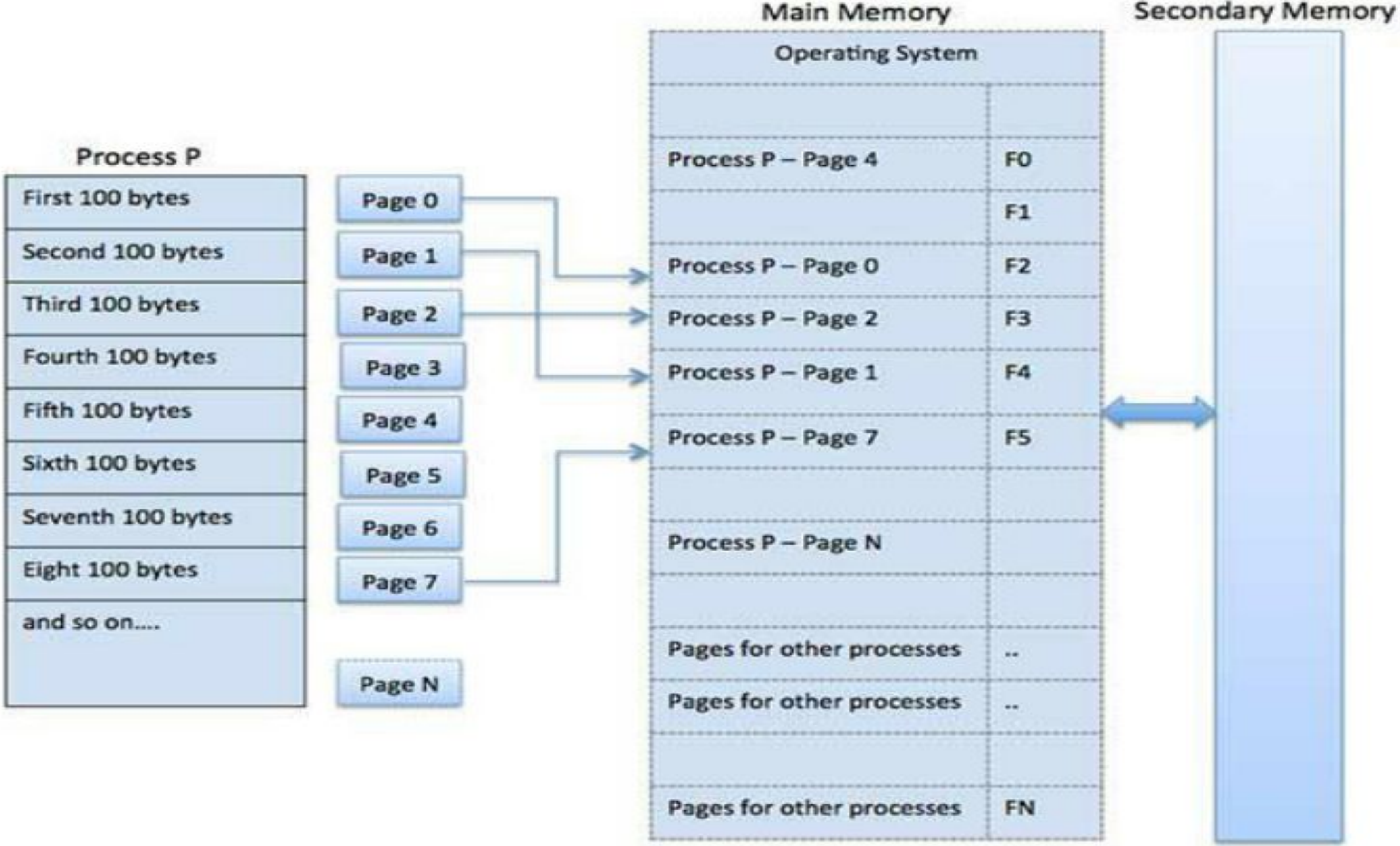
Q10) Explain briefly about paging with a neat diagram.

Ans: Paging

● Paging technique plays an important role in implementing virtual memory.

● Paging is a memory management technique in which process address space is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). The size of the process is measured in the number of pages.

● Similarly, main memory is divided into small fixed-sized blocks of (physical) memory called frames and the size of a frame is kept the same as that of a page to have optimum utilization of the main memory and to avoid external fragmentation.


Process P
First 100 bytes
Second 100 bytes
Third 100 bytes
Fourth 100 bytes
Fifth 100 bytes
Sixth 100 bytes
Seventh 100 bytes
Eight 100 bytes
and so on....
Page 0
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page N
Main Memory
Operating System
Process P – Page 4
F0
F1
Process P – Page 0
F2
Process P – Page 2
F3
Process P – Page 1
F4
Process P – Page 7
F5
Process P – Page N
Pages for other processes
..
Pages for other processes
..
Pages for other processes
FN
Secondary Memory

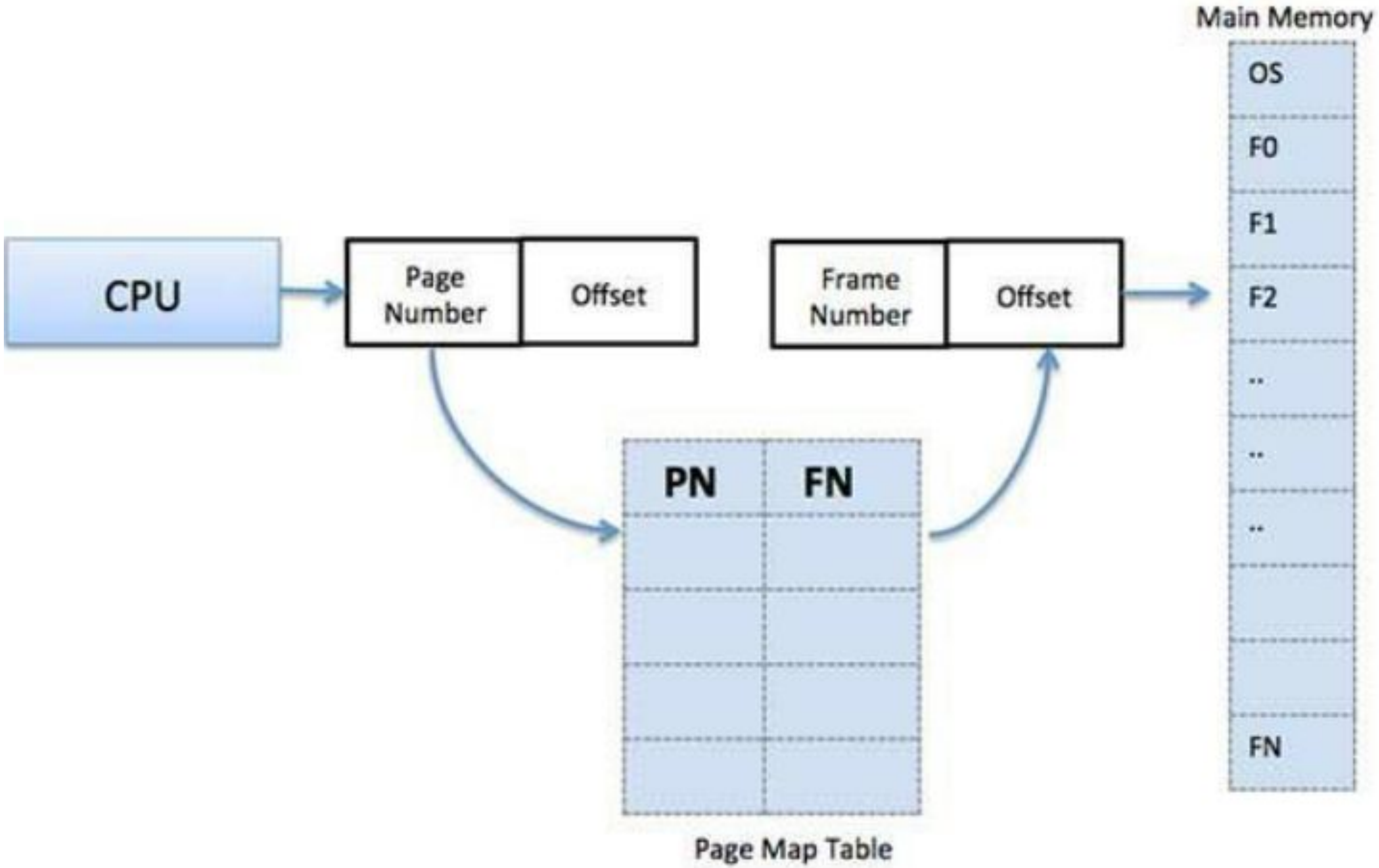
Address Translation Page address is called logical address and represented by page number and the offset.

Logical Address = Page number + page offset

Frame address is called physical address and represented by a frame number and the offset.

Physical Address = Frame number + page offset

A data structure called page map table is used to keep track of the relation between a page of a process to a frame in physical memory.


CPU
Page Number
Offset
Frame Number
Offset
Main Memory
OS
F0
F1
F2
..
..
..
FN
PN
FN
Page Map Table

When the system allocates a frame to any page, it translates this logical address into a physical address and create entry into the page table to be used throughout execution of the program. When a process is to be executed, its corresponding pages are loaded into any available memory frames. Suppose you have a program of 8Kb but your memory can accommodate only 5Kb at a given point in time, then the paging concept will come into picture. When a computer runs out of RAM, the operating system (OS) will move idle or unwanted pages of memory to secondary memory to free up RAM for other processes and brings them back when needed by the program. This process continues during the whole execution of the program where the OS keeps removing idle pages from the main memory and write them onto the secondary memory and bring them back when required by the program.

Advantages and Disadvantages of Paging Here is a list of advantages and disadvantages of paging −

- Paging reduces external fragmentation, but still suffer from internal fragmentation.
- Paging is simple to implement and assumed as an efficient memory management technique.
- Due to equal size of the pages and frames, swapping becomes very easy.
- Page table requires extra memory space, so may not be good for a system having small RAM.