OSY - Notes


V2V EDTECH LLP
Online Coaching at an Affordable Price.
OUR SERVICES:
• Diploma in All Branches, All Subjects
• Degree in All Branches, All Subjects
• BSCIT / CS
• Professional Courses
+91 93260 50669
V2V EdTech LLP
v2vedtech.com
v2vedtech

Unit - IV Memory Management

➢ 4.1 Basic Memory Management

Memory management in operating system is important because of it directly affects the execution time of a process. The execution time of a process depends on the availability of data in the main memory. Therefore, an operating system must perform the memory management in such a manner that the essential data is always present in the main memory. An effective memory management system ensures accuracy, availability and consistency of the data imported from the secondary memory to the main memory.

Advantages :

- Memory management provide protection for memory.
- Increase the performance of the system

Disadvantages :

- Memory management is complex
- It may cause high overhead
- Memory management is dependent on hardware.

• Memory Partitioning

Explain partitioning and its types. (W – 19, S - 23)

In memory partitioning, memory is divided into a number of regions or partitions. These partitions can be of different size or same size. Each region may have one program to be executed. When a region is free, a program is selected from the job queue and loaded into the free region. Memory is divided into two sections, one for user and one for resident monitor of the operating system.

Types of partitioning:

i) fixed partitioning i.e. static partitioning
ii) variable partitioning i.e. dynamic partitioning.

1) Fixed Partitioning: (Static partitioning)

Explain fixed size memory partitioning. (W - 22)

Main memory is divided into multiple partitions of fixed size at the time of System generation. A process may be loaded into a partition of equal size or greater size. Partition can be of equal size or unequal size. There are two alternatives for fixed partitioning.

a) Equal size partitioning
b) Unequal size partitioning

a) Equal size partitioning:

OS
8KB
8KB
8KB
8KB
8KB
8KB
8KB
8KB

Main memory is divided into equal size partitions. Any process with less or equal size can be loaded in any available partition.

b) Unequal size partitioning:

OS
2KB
4KB
6KB
8KB
10KB
12KB

Main memory is divided into multiple partitions of unequal size. Each process can be loaded into the smallest partition within which the process will fit.

Advantages of Static Memory Partitioning:

1. Simple to implement.
2. It requires minimal operating system software and processing overhead.
3. Partitions are fixed at the time of system generation.

Disadvantages of Static Memory partitioning

1. Memory wastage
2. Inefficient use of memory due to internal fragmentation.

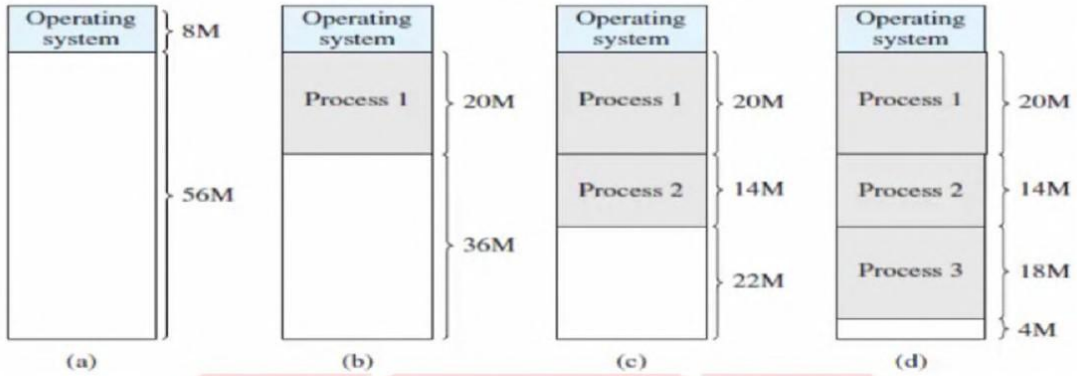
2) Variable Partitioning: (Dynamic Partitioning)

With suitable diagram, describe the concept of variable partitioning of memory. (S – 22, S - 24)

When a process enters in main memory, it is allocated exact size that is required by that process. So in this method, partitions can vary in size depending on memory space required by a process entering in main memory. Operating System maintains a table indicating which parts of memory are available and which are occupied. When new process arrives and it needs space, System searches for available memory space in main memory. If it is available, then memory is allocated to the process by creating a partition in memory.

For example:
Consider following table with process and memory space.

Process	Memory space
P1	20M
P2	14M
P3	18M



Dynamic Storage Allocation:

The set of holes is searched to determine which hole is best to allocate. The most common strategies used to select a free hole from the set of available holes are first fit, best fit and worst fit.

Types of the Dynamic Storage Allocation

1. First Fit
2. Best Fit
3. Worst Fit

- **First Fit:** Allocate the first hole (or free space) that is big enough for the new process.
- **Best Fit:** Allocate the smallest hole that is big enough. We search the entire list unless the list is kept ordered by size. This strategy produces the smallest left over hole.
- **Worst Fit:** Allocate the largest hole. We must search the entire list, unless it is sorted by size.

Advantages of Dynamic Memory Partitioning:

1. No internal fragmentation.
2. More efficient use of main memory.

Disadvantages of Dynamic Memory partitioning:

1. It suffers from external fragmentation.
2. It needs compaction.

While hole is taken for next segment request for 8KB in a swapping system for First Fit, Best Fit, and Worst Fit. (W - 23)

OS
4 KB
9 KB
20 KB
16 KB
8 KB
2 KB
6 KB

First Fit :

Allocate the first free block to the new process.

OS
4 KB
< FREE > 1 KB 8 KB
20 KB
16 KB
8 KB
2 KB
6 KB

Best Fit :

Allocate the smallest free block that is big enough to accommodate new process.

OS
4 KB
9 KB
20 KB
16 KB
< FREE> 0 KB 8 KB

2 KB
6 KB

Worst Fit :

Allocate the largest free block to the new process.

OS
4 KB
9 KB
< FREE > 12 KB 8 KB
16 KB
8 KB
2 KB
6 KB

Consider the following memory map and assume a new process P4 comes with memory requirements of 6 KB

Locate (Draw) this process in memory using

i. First Fit

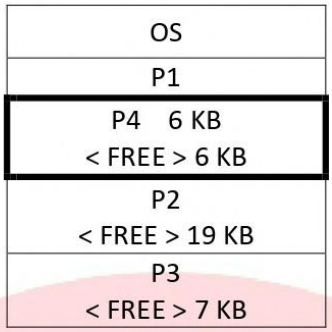
ii. Best Fit

iii. Worst Fit (W - 22)

OS
P1 < FREE > 12 KB
P2 < FREE > 19 KB
P3 < FREE > 7 KB

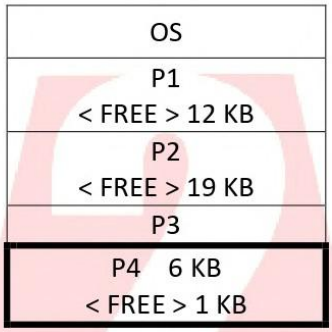
First Fit :

Allocate the first free block to the new process.



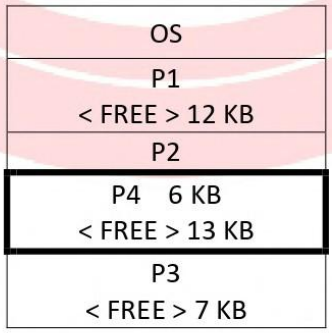
Best Fit :

Allocate the smallest free block that is big enough to accommodate new process.



Worst Fit :

Allocate the largest free block to the new process.



1. Free space management :

List free space management techniques? Describe any one in detail. (W – 19, S – 22, W – 22, W – 23, S - 24)

The process of looking at free and managing the free blocks of the disk is called free space management. Files are created and deleted frequently during the operation of a computer system. Since there is only a limited amount of disk space, it is necessary to reuse the space from deleted files for new files. To keep track of free disk space, the file system maintains a free space list. The free space list records all disk blocks, which are free. To create a file, we search the free space list for the required amount of space on and allocate it to the new file. This space is then removed from the free space list. When a file is deleted, its disk space is added to the free space list.

Following are the techniques of free space management:

1. Bit vector
2. Linked list
3. Grouping
4. Counting

1. Bit Vector :

Describe concept of free space management technique using bit map method (W - 23)

The free space list is implemented as a bit map or bit vector. Each block is represented by 1 bit. If the block is free, the bit is 0, if the block is allocated, the bit is 1.

Example :

Consider a disk where blocks 2, 3, 4, 5, 8, 9, 10, 11, 12, 13 are free and the rest of the blocks are allocated.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	0	0	0	1	1	0	0	0	0	0	0	1	1

The free space bit map would be: 1100001100000011

0 = Free Block.
1 = Allocated Block.

Advantages of Bit map / Bit vector :

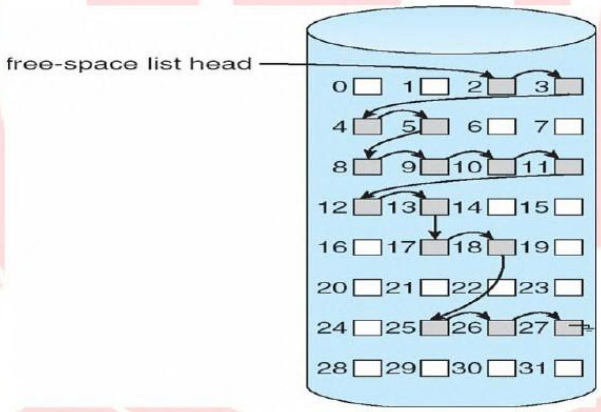
1. It is simple and efficient method to find the first free block or n-consecutive free block on the disk.
2. A bit map requires lesser space as it uses 1 bit per block.

Disadvantages of Bit map / Bit vector :

1. Extra Space required to store bit map.
2. It may require a special hardware support to do bit operation i.e. finding out which bit is 1 and which bit is 0.

2. Linked List :

Linked list is another technique for free space management in which, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block. In linked list technique, link all the disk blocks together, keeping a pointer to the first free block. This block contains a pointer to the next free disk block, and so on.



In this example we would keep a pointer to block 2, as the first free block. Block 2 would contain a pointer to block 3, which point to block 4, which would point to block 5 and so on.

Advantages of Linked Free Space List :

1. If a file is to be allocated a free block, the operating system can simply allocate the first block in the free space list and move the head pointer to the next free block in the list.

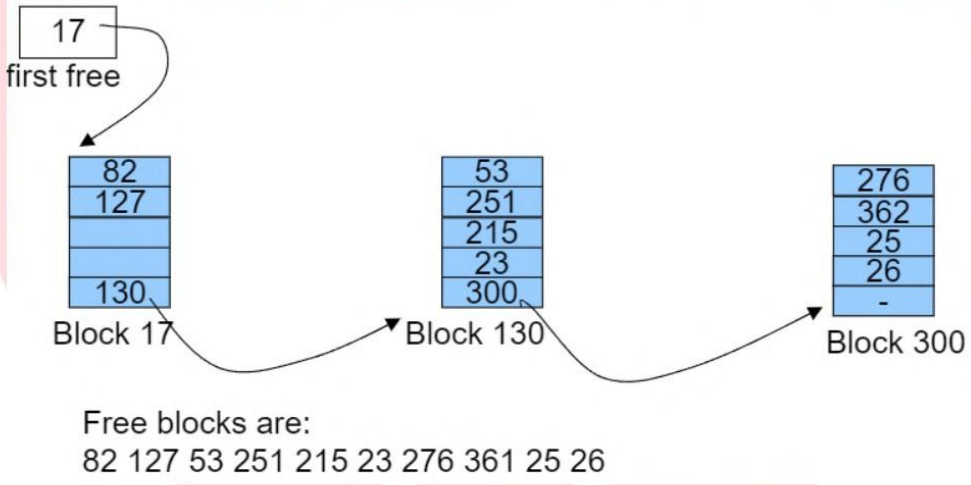
2. No disk fragmentation as every block is utilized.

Disadvantages of Linked Free Space Lists :

1. It is not very efficient scheme as to traverse the list.
2. Pointers use here will also consume disk space. Additional memory is therefore required

3. Grouping

Grouping is a free space management technique. This approach stores the address of the free blocks in the first free block. The first free block stores the address of some, say n free blocks. Out of these n blocks, the first n-1 blocks are actually free and the last block contains the address of next free n blocks. Shows an example of grouping free space management technique, in which a disk block contains addresses of many free blocks and a block containing free block pointers will get freed when those blocks are used.



Advantages :

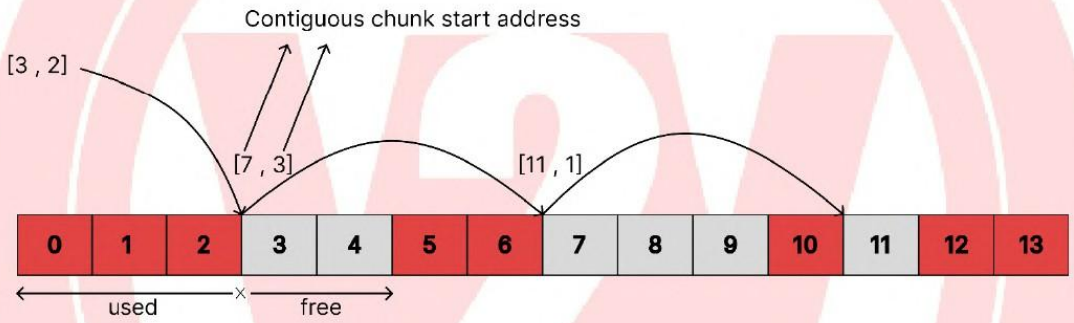
1. Finding free blocks in massive amount can be done easily using this method.
2. Reduce redundancy.

Disadvantages :

1. We need to alter the entire list, if any of the block of the list is occupied.
2. It increases the complexity.

4. Counting

Counting is also a technique for free space management. Generally several contiguous blocks may be allocated or freed simultaneously, particularly when contiguous allocation is used. Thus rather than keeping a list of 'n' free disk addresses, we keep the address of first free block and number 'n' of free contiguous blocks. Each entry in the free space list then consists of a disk address and a count. Although each entry requires more space than would a simple disk address, the overall list will be shorter as long as the count is generally greater than 1. Below fig shows an example of Conti Counting Free Space management technique. It besides the free block Pointer, keep a counter saying how many block are free contiguously after that free block.



Advantages :

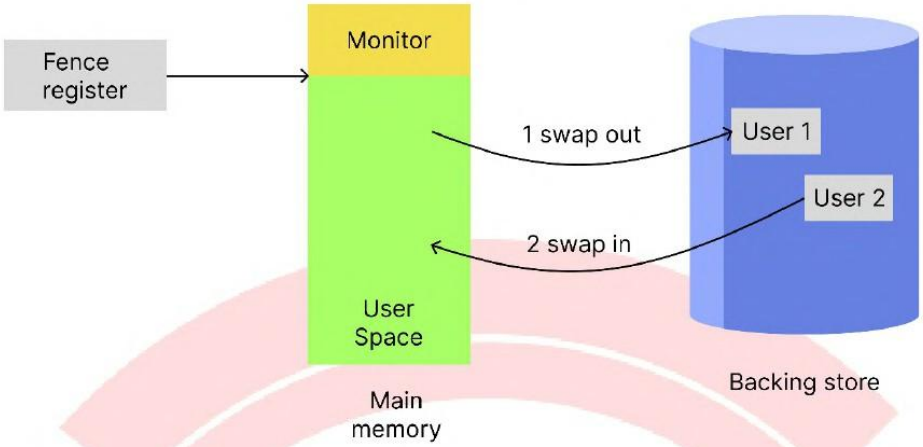
1. It provides efficient utilization of space
2. It allows fast allocation and deallocation
3. It reduces fragmentation

Disadvantages :

1. Only contiguous blocks are required
2. It may increase the overhead

➢ 4.2 Swapping

Explain the following terms with respect to memory management:
i) Dynamic relocation
ii) Swapping (S - 24)



Swapping is 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 late time, the system swaps back the process from the secondary storage to main memory. For eg. In a multiprogramming environment when number of processes is queued for execution and if the system is implemented with a round robin algorithm then when time quantum / time slice expires then the process will be swapped out and the new process will be swapped into the memory space that just has been freed.

2. Compaction :

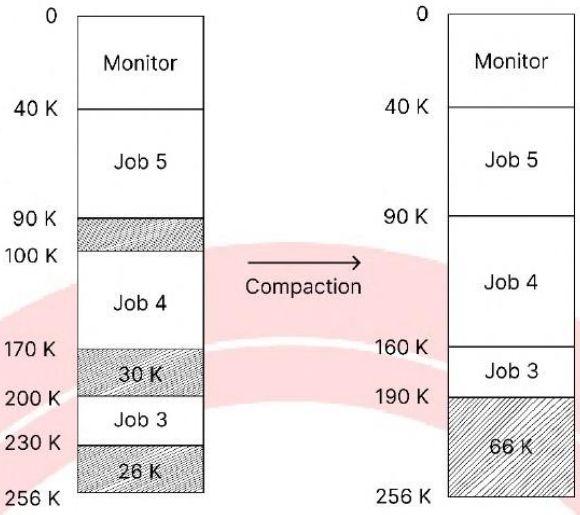
Compaction is a method used to overcome the external fragmentation problem. All free blocks are brought together as one large block of free space. The collection of free space from multiple non-contiguous blocks into one large free block in a system's memory is called compaction.

The goal compaction is to shuffle the memory contents to place all free memory together in one large block. For example, the memory map of (e) can be compacted as shown in below Fig. The three holes of sizes 10K, 30K and 26K can be compacted into one hole of 66K.

Compaction is not always possible.

In below Fig. we moved job4 and job3 for these programs to be able to work in their new locations, all internal addresses must be relocated. Compaction is possible only if relocation is dynamic, at execution time, using base and limit registers. The simplest compaction algorithm is to simply move all jobs towards one end of memory, all holes move in the other direction, producing an large hole of available memory.

Compaction can be quite expensive. Compaction changes the allocation of memory to make free space contiguous and hence useful. Compaction also consumes system resources.



Advantages:

1. **Eliminates External Fragmentation:** Consolidates free space, making it easier to allocate larger processes.
2. **Efficient Memory Usage:** Maximizes memory utilization by creating a single large free block.
3. **Improved System Performance:** Reduces allocation failures from fragmented space.
4. **Allows Large Memory Allocation:** Enables large programs to run even with scattered free memory.

Disadvantages:

1. **High CPU Overhead:** Requires significant CPU time for relocating processes and updating references.
2. **System Downtime:** Can cause other tasks to wait, leading to temporary system inefficiency.
3. **Complexity:** Demands careful management of relocation information to maintain data integrity and avoid errors.

3. Fragmentation :

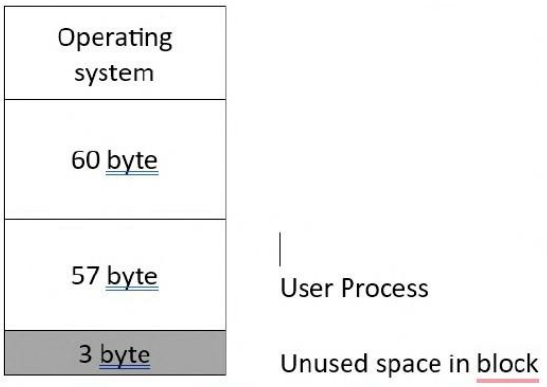
Define the term fragmentation in terms of memory. (S – 22, S - 23)

As processes are loaded and removed from memory, the free memory space is broken into little pieces. It happens after sometimes that processes cannot be allocated to memory blocks Considering their small size and memory blocks remains unused.This problem is known as Fragmentation.

Types of Fragmentation :-

1. Internal Fragmentation
2. External Fragmentation.

1. Internal Fragmentation



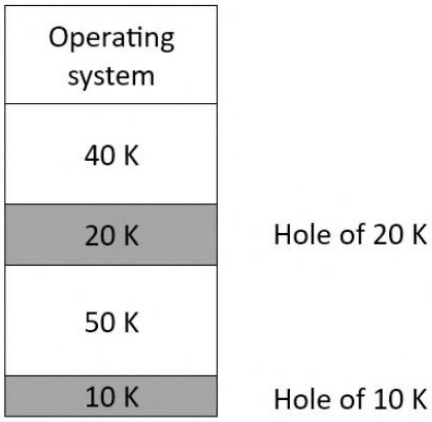
1. Wasting of memory within a partition, due to a difference in size of a partition and of the object resident within it, is called internal fragmentation.
2. Internal fragmentation occurs when the memory allocator leaves extra space empty inside block of memory that has been allocated for a client.

Example :

1. A client that needs 57 bytes of memory is given 60 bytes
2. The extra bytes that the client doesn't need go to waste,

2. External Fragmentation :

Wasting of memory between partitions, due to scattering of the free space into a number of discontinuous areas, is called external fragmentation. External fragmentation exists when enough total memory space exists to satisfy a request, but it is not contiguous, storage is fragmented into large number of small holes.



For Example :-

There is a hole of 20k and 10k is available in multiple partition allocation schemes. The next process request for 30k of memory. Actually 30k of memory is free which satisfy the request but hole is not contiguous. So there is an external fragmentation of memory.

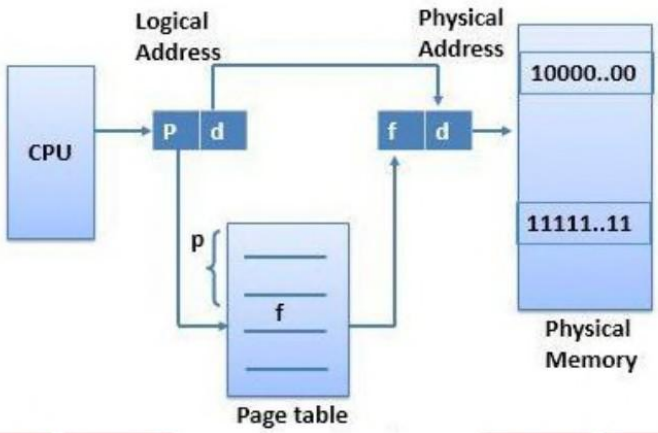
➢ 4.3 Non-contiguous Memory Management Techniques:

Define paging and segmentation (W - 23)

- **Paging:**

  Define paging. (S - 23)

Paging is a memory management technique by which a computer stores and retrieves data from secondary storage to main memory. In paging, the operating system retrieves data from secondary storage in same-size blocks called pages. The main idea behind the paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames. One page of the process is to be stored in one of the frames of the memory. Pages of the process are brought into the main memory only when they are required otherwise they reside in the secondary storage. The sizes of each frame must be equal. and page must be equal. considering the fact that Pages are mapped to the frames in Paging.



Advantages of Page Memory Allocation:

1. Allow jobs to be allocated in non-contiguous memory locations.
2. Paging eliminates fragmentation
3. Memory used more efficiently.
4. Support higher degree of multiprogramming.

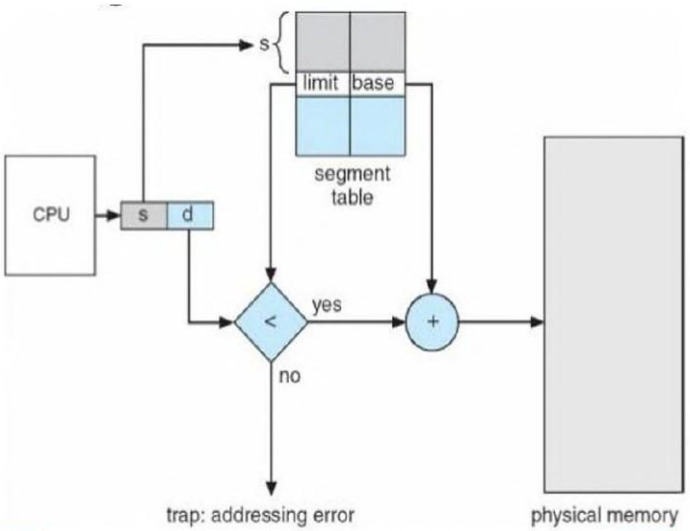
Disadvantages of Paged Memory Allocation:

1. Page Address mapping hardware usually increases the cost of the computer.
2. Internal fragmentation still exists, though in last Page.
3. Requires the entire job to be stored in memory location
4. Size of page is crucial (not too small, not too large).
5. Memory must be used to store the various tables like page table, memory map table etc.

- **Segmentation:**

In Operating System, Segmentation is a memory management technique in which the memory is divided into the variable size parts.Each part is known as a segment which can be allocated to a process. The detail about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments. Segment table contains mainly two information about segment:

1. **Base:** It is the base address of the segment.
2. **Limit:** It is the length of the segment.



Working of Segmentation:

Each entry of segment table has a segment base and a segment limit. The offset of the logical address must be between 0 and the segment limit. If it is not, we trap to the operating system. If this offset is legal, it is added to the segment base to produce the address in physical memory of the desired word.

Advantages of Segmentation:

1. Segmentation can eliminate fragmentation.
2. It provides a virtual memory.
3. It supports Dynamic linking and loading.
4. It provides a convenient way of organizing programs and data to the programmer.

Disadvantages of Segmentation:

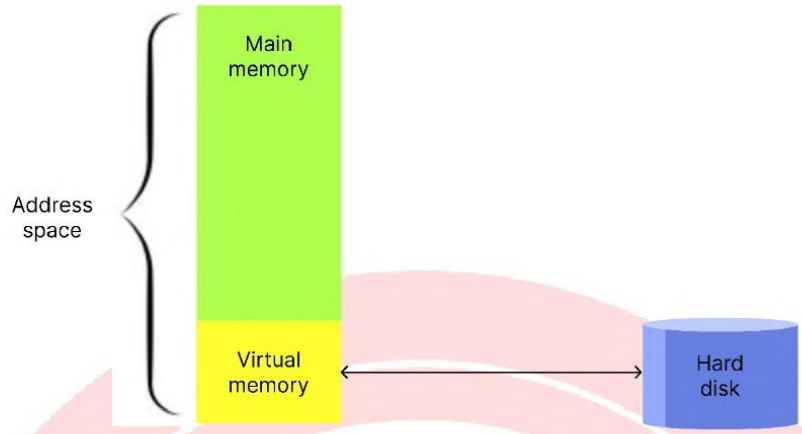
1. It suffers from external fragmentation.
2. Conversion from logical address to physical address is not a simple function.
3. It increase complexity of operating system.
4. It increase hardware cost.

Difference between paging and segmentation (W – 22, S - 23)

Parameter	Paging	Segmentation
1) Individual Memory	In paging, we break a process address space into blocks known as pages.	In the case of segmentation, we break a process address space into blocks known as sections/segments.
2) Memory Size	The pages are blocks of fixed size.	The sections/segments are blocks of varying sizes.
3) Speed	This technique is comparatively much faster in accessing memory.	This technique is comparatively much slower in accessing memory than paging.
4) Size	The available memory determines the individual page sizes.	The user determines the individual segment sizes.
5) Fragmentation	The paging technique may underutilize some of the pages, thus leading to internal fragmentation blocks at all.	The segmentation technique may not use some internal memory blocks at all. Thus, it may lead to external fragmentation.
6) Logical Address	A logical address divides into page offset and page number in the case of paging.	A logical address divides into section offset and section number in the case of segmentation.
7) Data Storage	In the case of paging, the page table leads to the storage of the page data.	In the case of segmentation, the segment table leads to the storage of the segmentation data.

➢ 4.4 Virtual Memory :

Define virtual memory (W – 19, S - 23)



Virtual memory is a memory management technique that provides an idealized abstraction of the storage resources that are actually available on a given machine which creates the illusion to users of a very large (main) memory. Virtual memory is the separation of user logical memory from physical memory. This separation allows an extremely large virtual memory to be provided for programmers when only a small physical memory is available. Virtual memory makes the task of programming much easier, because the programmer no longer needs to worry about the amount of physical memory available for execution of program. It is the process of increasing the apparent size of a computer's RAM by using a section of the hard disk storage as an extension of RAM.

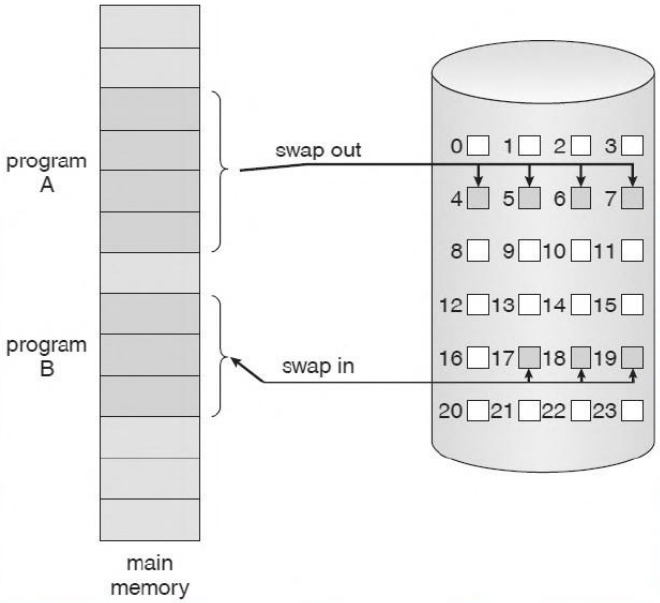
Example:

Consider, an e-mail program, a web browser and a word processor is loaded into RAM simultaneously; the 64 MB space is not enough to store all these programs. Without a virtual memory a message "You cannot load any more applications. Please close an application to load a new one." would be displayed. By using virtual memory a computer can identify data in RAM that is not actively being used by running programs and temporarily move this data to the hard disk. This process frees up RAM space for other active applications and processes.

Advantages :

1. Allow the programmer to use more memory than actual physical memory in a system.
2. Allow the swapping of processes to and from physical memory.
3. It enables multitasking.
4. Helps in keeping the system stable i.e. avoid system crash.

- **Demand paging**



A demand - Paging system is similar to a Paging system with swapping. Where processes reside in secondary memory (usually a disk). When we want to execute a process, we swap it into memory. Rather than swapping the entire process into memory, however, we use a lazy swapper called pager. A lazy swapper never swaps a page into memory unless that page will be needed. When a process is to be swapped in the pager guesses which pages will be used before the process is swapped out again. Instead of swapping in a whole process, the pager brings only those necessary pages into memory.

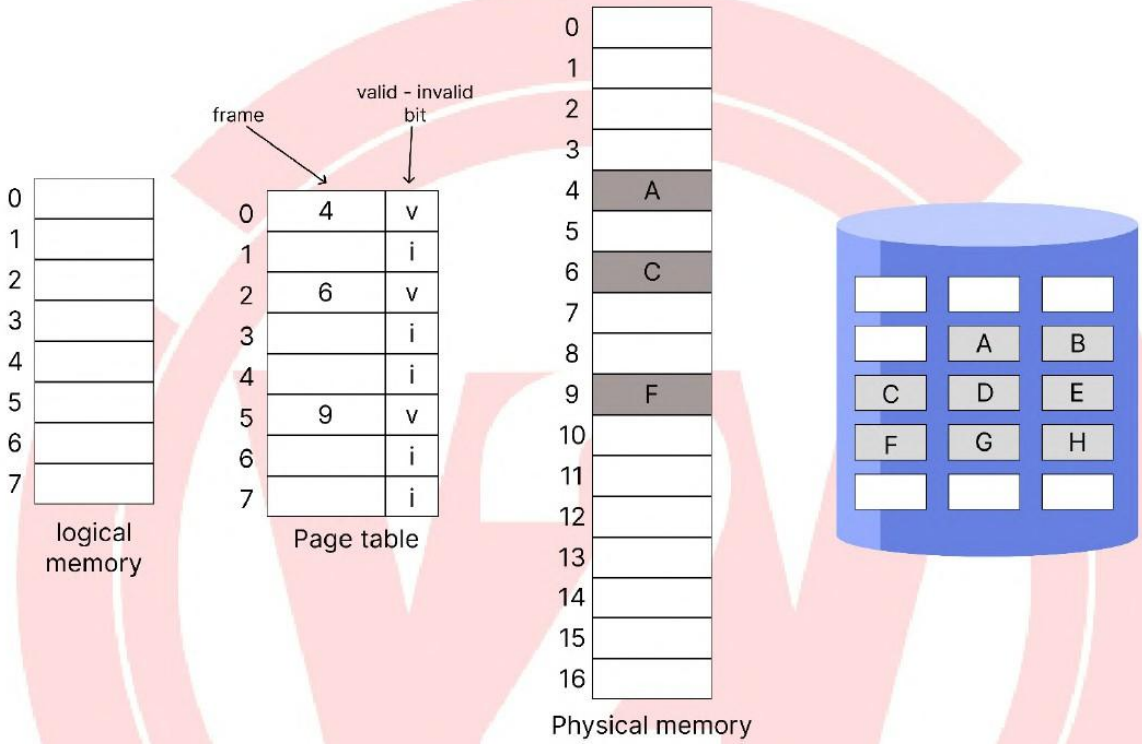
Advantages of Demand Paging :

1. Large virtual memory.
2. more efficient use of memory.
3. There is no limit on degree of multiprogramming.
4. It saves memory.

Disadvantages Demand Paging :

1. Accessing data not in memory causes delays.
2. It is complex to implement.

- **Page Fault**



A page fault occurs when a program accesses a page that has been mapped in address space, but has not been loaded in the physical memory. When the page (data) requested by a program is not available in the memory, it is called as a page fault. It is possible that not all pages of the program were brought into memory. Some pages are loaded into memory and some pages are kept on the disk. To distinguish between the pages that are in memory and the pages that are on the disk, a valid-invalid bit is provided. Pages that are not loaded into memory are marked as invalid in the page table, using the invalid bit. The bit is set to valid if the associated page is in memory. Access to a page marked invalid causes a page fault.

➢ 4.5 Page Replacement Algorithm

• First In First Out (FIFO)

The simplest page replacement algorithm is a FIFO. A FIFO replacement algorithm associates with each page the time when that page was brought into memory. When a page must be replaced, the oldest page is chosen. FIFO queue is created to hold all pages in memory. We replace the page at the head of the queue. When a page is brought into memory, we insert it at the tail of the queue.

For example, consider the following reference string: problem of FIFO Algorithm: consider the following reference string :

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Reference string Frames	1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	4	4	4	5	5	5	5	5	5
2		2	2	2	1	1	1	1	1	3	3	3
3			3	3	3	2	2	2	2	2	4	4
Page fault	F	F	F	F	F	F	F			F	F	

Fig Page fault for using three frames.

Reference string Frames	1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	1	5	5	5	5	4	4
2		2	2	2	2	2	2	1	1	1	1	5
3			3	3	3	3	3	3	2	2	2	2
4				4	4	4	4	4	4	3	3	3
Page fault	F	F	F	F			F	F	F	F	F	F

i) the number of faults for four frames (10) is greater than the number of faults for three frames (9).
ii) This most unexpected result is known as Belady's anomaly.
iii) For some page replacement algorithms, the page fault rate may increase as the number of allocated frames increases.

Advantages of FIFO Page Replacement Algorithm:

1. It is simple to implement.
2. It is easiest algorithm
3. Easy to understand and execute.

Disadvantages of FIFO Page Replacement Algorithm:

1. It is not very effective
2. System needs to track of each frame.
3. It suffers from Belady's anomaly.
4. Bad replacement choice increases the page fault rate and slow process execution.

- **Least Recently Used (LRU)**

Explain LRU page replacement algorithm for following reference string. 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Calculate the page fault. (W - 19)

The Least Recently Used (LRU) page replacement algorithm replaces the page that has not been used for the longest period of time. LRU replacement associates with each page the time of that page's last use. When a page must be replaced, LRU chooses the page that has not been used for the longest period of time. An LRU page-replacement algorithm may require substantial hardware assistance.

Following are the different approaches to implement LRU replacement algorithm:

1. Using counters for LRU:

We can implement LRU using a counter for each page frame. Whenever a page is referenced, its counter is incremented with the current timestamp. When a page fault occurs, the page with the smallest counter value is replaced.

2. Using Stack for LRU :

Another approach to implementing LRU replacement is to keep a stack of page number. Whenever a page is referenced, it is removed from the stack and put on the top. In this way, that most recently used page is always at the top of the stack and the least recently used page is always at the bottom.

Example :

Reference String:

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

The below solved sum can also be solved same as like LRU table (Only the table structure will be same logic of solving is different) in some places sums are solved in this way also

LRU:

Assume Frame Size = 3

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2		2		4	4	4	0			1		1	*	1		*
	0	0	0	*	0	*	0	0	3	3	*		3		0		0	*	
		1	1		3		3	2	2	2		*	2	*	2		7		

Page Fault = 12

Assume frame size = 4

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	7		3		3		*		*		3				7		
	0	0	0	*	0	*	0			*			0		*		0	*	
		1	1		1		4						1			*	1		*
			2		2		2	*				*	2	*			2		

Page fault = 08

Advantages of LRU Page Replacement Algorithm :

- LRU is actually quite a good algorithm.
- It never suffers from Belady's anomaly.
- LRU algorithm is very feasible to implement.

Disadvantages of LRU Page Replacement Algorithm :

- LRU algorithm is that it requires additional data structure and hardware support.
- Its implementation is not very easy.

- **Optimal**

An optimal page replacement algorithm has the lowest page fault rate of all algorithms and would never suffer from Belady's anomaly. Optimal replacement algorithm states replace that page which will not be used for the longest period of time.

For example, Consider the following reference string.
7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1

- The first three reference cause faults which fill the three empty frames.
- The reference to page 2 replaces page 7, because 7 will be used until reference 18, whereas page 0 will be used at 5 and page 1 at 14.
- The reference to page 3 replaces page 1, as page 1 will be the last of the three pages in memory to be reference referenced again
- with only nice Page fault, optimal replacement is much better than a FIFO algorithm, which had 15 faults.
- The optimal page replacement algorithm is difficult to implement because it requires future knowledge of the reference string.

Frame	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
F2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F3			1	1	1	3	3	4	4	3	3	3	3	1	1	1	1	1	1	1
Page fault	F	F	F	F		F		F		F				F				F		

Total Page Fault = 9
Total page hits = 11

Advantages of optimal page Replacement Algorithm

1. It give the smallest number of Page faults.
2. It never suffers from Belady's anomaly.
3. Twice as good as FIFO Page Replacement Algorithm.

Disadvantage of optimal page Replacement Algorithm:

1. This algorithm is difficult to implement.

2. It is only use as a theoretical part of page replacement.
3. It requires future knowledge of reference string.

Find out the total number of page faults using:

- **Least Recently Used page Replacement**
- **Optimal page Replacement**

Page Replacement algorithms of memory management, if the pages are coming in the order:

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1 (W - 23)

i) LRU

considering frame size is 3:

Ref	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
F2		0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
F3			1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
Fault	F	F	F	F		F		F	F	F	F			F		F		F		

Total page faults using LRU page Replacement / Fault = 12

ii) optimal:

Considering Frame size is 3:

Ref	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
F2		0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
F3			1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
Fault	F	F	F	F		F		F			F			F				F		

Total page faults using optimal page Replacement / page Fault = 09

Given a page reference string with three (3) page frames. Calculate the page fault with 'optimal' and 'LRU' page replacement algorithm respectively. (W - 22)

Page Reference String :
7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1

i. Optimal Page Replacement Algorithm

REF	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
F2		0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
F3			1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
Fault	F	F	F	F		F		F			F			F				F		

Total page Fault = 9

ii. LRU

REF	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
F2		0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
F3			1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
Fault	F	F	F	F		F		F	F	F	F			F		F		F		

Total page Fault = 12

For the page reference string 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1 Calculate the page faults applying: i) optimal ii) LRU iii) FIFO page Replacement algorithms for a memory with three frames. (S - 23)

i) optimal

Ref	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
F2		0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
F3			1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
Fault	F	F	F	F		F		F			F			F				F		

Total Page faults - 9

ii) LRU

Ref	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
F2		0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
F3			1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
Fault	F	F	F	F		F		F	F	F	F			F		F		F		

Total page faults - 12

iii) FIFO

REF	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
F1	7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
F2		0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
F3			1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1
Fault	F	F	F	F		F	F	F	F	F	F			F	F			F	F	F

Total page faults – 15

Consider the following page reference string arrival with three page frames:- 5, 6, 7, 8, 9, 8, 7, 5, 8, 9, 8, 7, 9, 6, 5, 6.
Calculate the number of page faults with optimal and FIFO (First In First Out) page replacement algorithms. (S - 22)

Optimal :

REF	5	6	7	8	9	7	8	5	9	7	8	7	9	6	5	6
F1	5	5	5	5	9	9	9	9	9	9	9	9	9	9	9	9
F2		6	6	8	8	8	8	5	5	5	8	8	8	8	5	5
F3			7	7	7	7	7	7	7	7	7	7	7	6	6	6
Fault	F	F	F	F	F			F			F			F	F	

Number of Hits: 7
Page Faults: 9

FIFO :

REF	5	6	7	8	9	7	8	5	9	7	8	7	9	6	5	6
F1	5	5	5	8	8	8	8	8	8	7	7	7	7	6	6	6
F2		6	6	6	9	9	9	9	9	9	8	8	8	8	5	5
F3			7	7	7	7	7	5	5	5	5	5	9	9	9	9
Fault	F	F	F	F	F			F		F	F		F	F	F	

Number of Hits: 5

Page Faults: 11

Consider the string: 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 4, 5, 6, 7 with Frame Size 3 and calculate page Fault in both the cases using FIFO algorithm. (S - 24)

FIFO

i. Frame Size = 3

Ref	0	1	2	3	0	1	2	3	0	1	2	3	4	5	6	7
F1	0	0	0	3	3	3	2	2	2	1	1	1	4	4	4	7
F2		1	1	1	0	0	0	3	3	3	2	2	2	5	5	5
F3			2	2	2	1	1	1	0	0	0	3	3	3	6	6
Fault	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F

Total Page Fault = 16

ii. Frame size : 4

Ref	0	1	2	3	0	1	2	3	0	1	2	3	4	5	6	7
F1	0	0	0	0	0	0	0	0	0	0	0	0	4	4	4	4
F2		1	1	1	1	1	1	1	1	1	1	1	1	5	5	5
F3			2	2	2	2	2	2	2	2	2	2	2	2	6	6
F4				3	3	3	3	3	3	3	3	3	3	3	3	7
Fault	F	F	F	F									F	F	F	F

Total page fault = 8