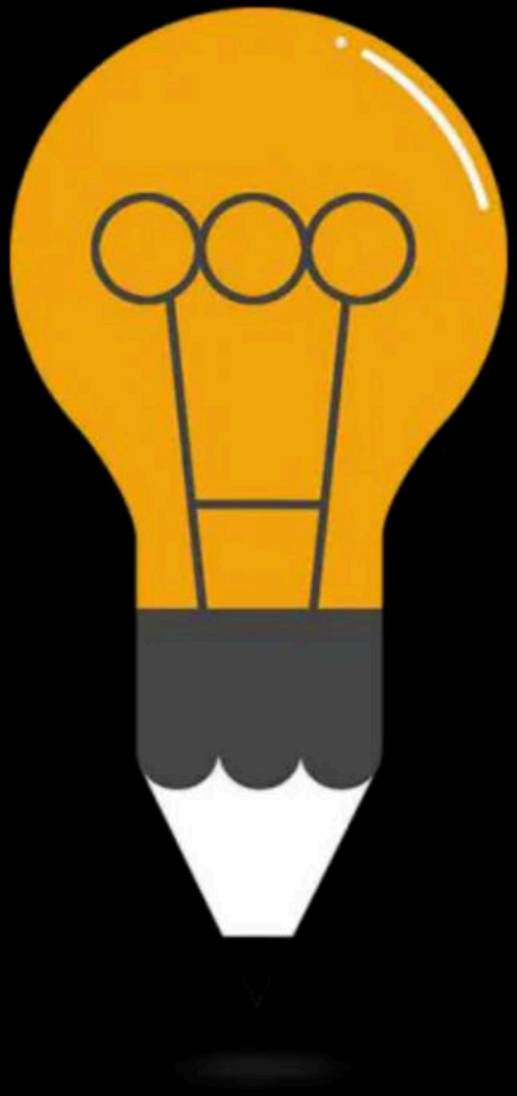
unacademy

Multilevel Paging

Comprehensive Course on Operating System for GATE - 2024/25

Vishvadeep Gothi • Lesson 35 • May 29, 2023

unacademy



Operating System
Page Replacement & Multilevel Paging

By: Vishvadeep Gothi

Question GATE-2016

Consider a computer system with ten physical page frames. The system is provided with an access sequence a_1, a_2, …, a_{20}, a_1, a_2, …, a_{20} where each a_i number. The difference in the number of page faults between the last-in-first-out page replacement policy and the optimal page replacement policy is _____

Question GATE-2014

A computer has twenty physical page frames which contain pages numbered 101 through 120. Now a program accesses the pages numbered 1, 2, …, 100 in that order, and repeats the access sequence THRICE. Which one of the following page replacement policies experiences the same number of page faults as the optimal page replacement policy for this program?

(A) Least-recently-used
(B) First-in-first-out
(C) Last-in-first-out
(D) Most-recently-used

Question GATE-2007

The address sequence generated by tracing a particular program executing in a pure demand paging system with 100 bytes per page is

$$0100, 0200, 0430, 0499, 0510, 0530, 0560, 0120, 0220, 0240, 0260, 0320, 0410.$$

Suppose that the memory can store only one page and if x is the address which causes a page fault then the bytes from addresses x to $x + 99$ are loaded on to the memory.

How many page faults will occur?

A. 0
B. 4
C. 7
D. 8

Page size = 100 Bytes

0100 - 0199

0200 - 0299

0430 - 0529

0530 - 0629

0120 - 0219

0220 - 0319

0320 - 0419

Making Page Reference Sequence

Assume CPU generates logical addresses in following order

	p	d
1	3	16
2	4	8
3	2	0
4	4	9
5	3	18
6	1	26

Page Reference Seq:-

3, 4, 2, 4, 3, 1

L.A.

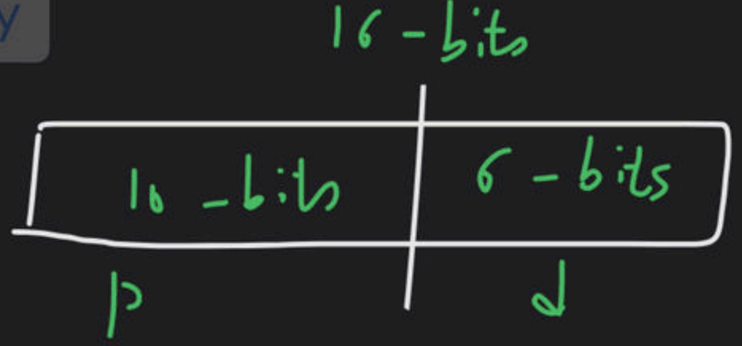
→	↓
3	8
3	2
2	5
5	3
8	15
4	9
5	6
5	8

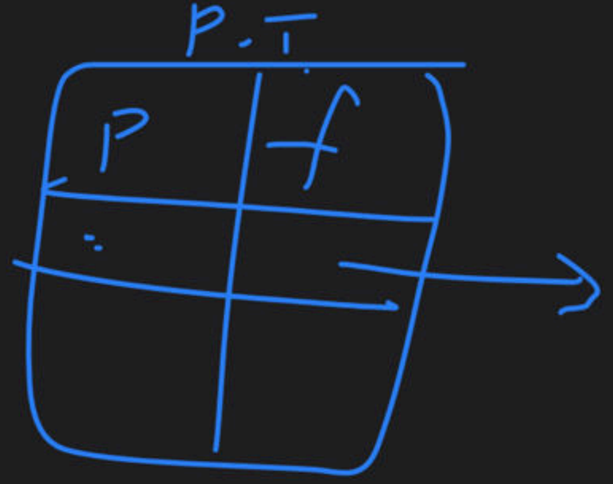
Page reference sequence:-

3, 2, 5, 8, 4, 5

If a page referred consecutively then in page reference sequence, it is mentioned only once.

L.A:-





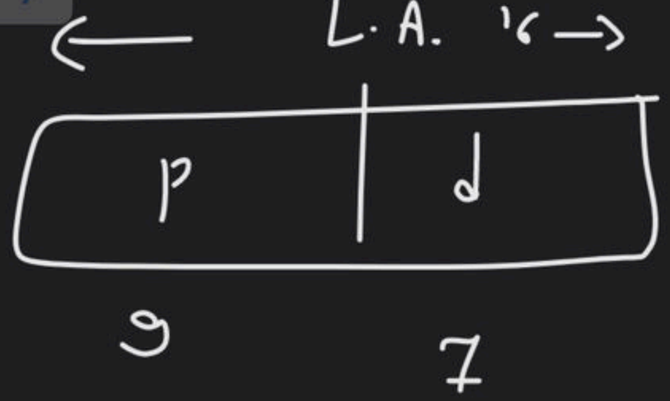
1010101011100000

P d

P = 1010101011 = $(683)_{10}$

d = 100000 = $(32)_{10}$

Ex:



$$LA = (05A4)_{16}$$

$$\underbrace{0000\ 0101}_{p}\ \underbrace{1010\ 0100}_{d}$$

$$p = 000000101 = (11)_{10}$$

$$d = 0100100 = (36)_{10}$$

Ques) Consider,

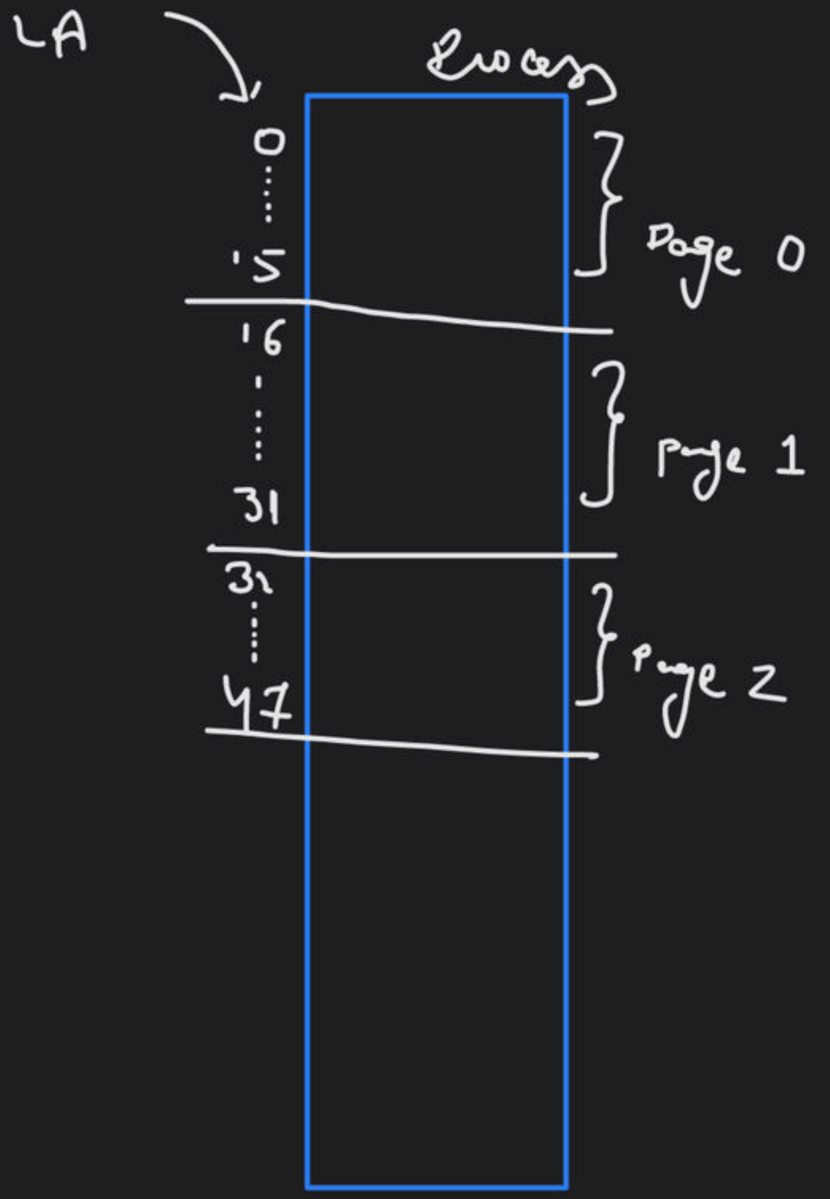
Page size = 16 bytes

L.A. $= (135)_{10}$

L.A. belongs to which page?

Page no. $= \left\lfloor \frac{135}{16} \right\rfloor = 8$

$d = 135 \% 16 = 7$

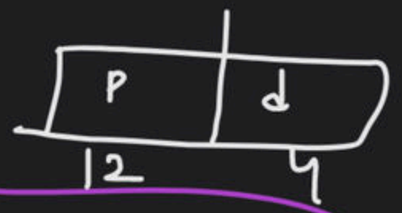


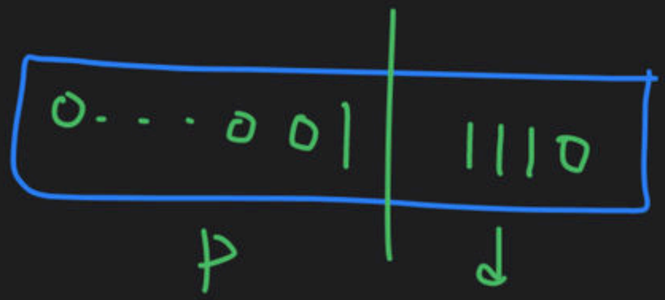
unacademy

$$P = \left\lfloor \frac{(L.A.)_{10}}{(page\ size)_{10}} \right\rfloor$$

$$d = L.A.\ \%\ page\ size$$

L.A. => $(30)_{10} = (11110)_2$





$p = (1)_{10}$

$d = (14)_{10}$

$\left\lfloor \frac{30}{16} \right\rfloor = 1$

$30\ \%\ 16 = 14.$

Frame Allocation

Frame Allocation

How many frames do we allocate per process?

Frame Allocation

How many frames do we allocate per process?

- If it is a single-user, single-tasking system, it's simple – all the frames belong to the user's process

Frame Allocation

2 Questions

- What is the minimum number of frames that a process needs?
- Is page replacement global or local?

Minimum Number of Frames

Every process must have enough pages to complete an instruction.

Frame Allocation

1. Equal Allocation → all processes get equal no. of frames
2. Proportional Allocation
   ↳ a process gets no. of frames allocated based on it's size.

Example:-

no. of frames = 8

	Equal allocation	Proportional
Process P1 ⇒ 12 pages	4 frames	$\frac{12}{(12+4)} * 8 = 6$ frames.
Process P2 ⇒ 4 pages	4 frames	$(4/16) * 8 = 2$ fra

Local Vs Global Allocation

P1 ⇒ 4 frames

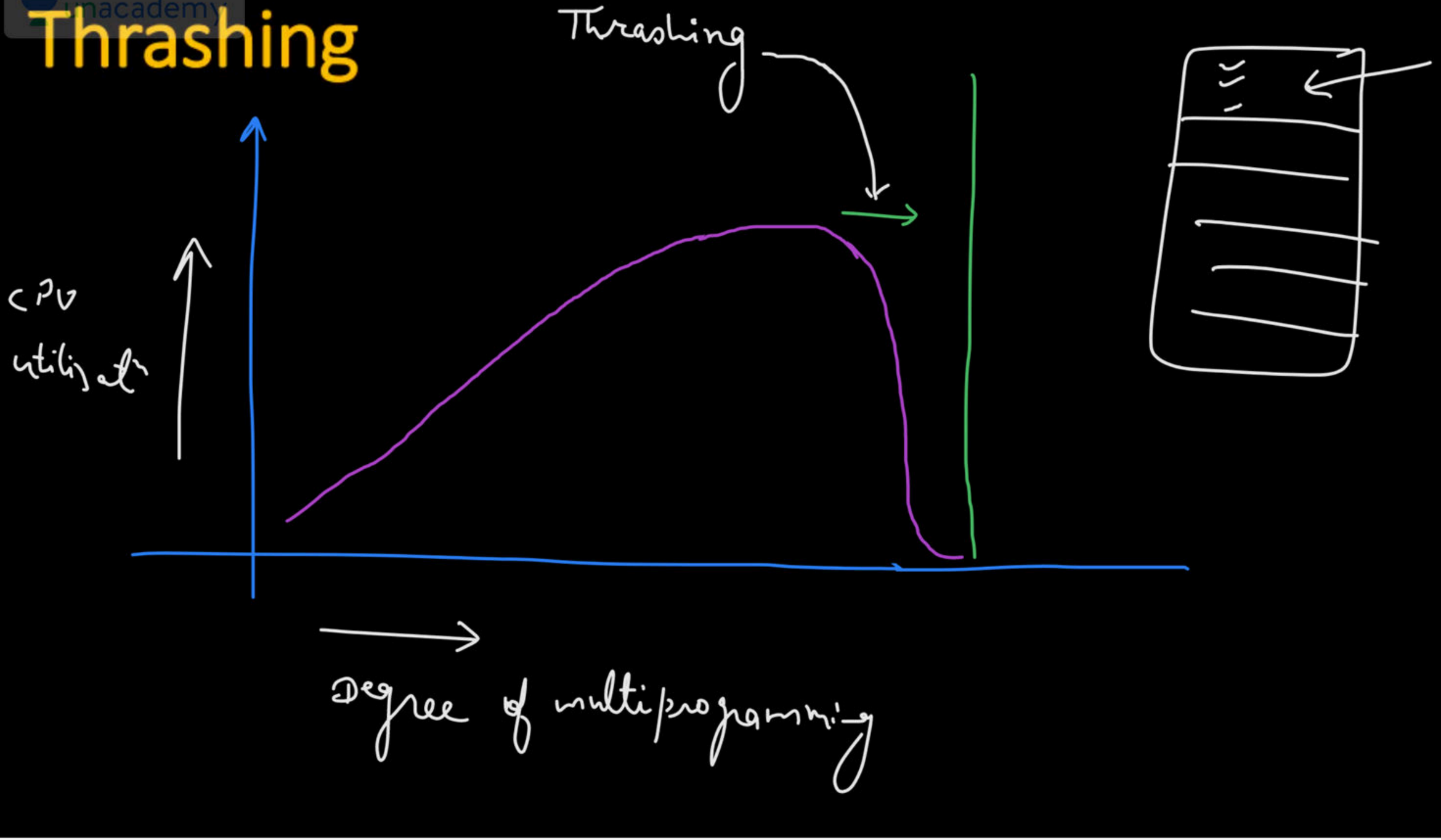
Local Allocation

1. Local replacement requires that the page being replaced be in a frame belonging to the same process
2. The number of frames belonging to the process will not change
3. This allows processes to control their own page fault rate

Global Allocation

1. The process can replace a page from a set that includes all the frames allocated to user processes
2. High-priority processes can increase their allocation at the expense of lower-priority processes
3. Global allocation makes for more efficient use of frames and their better throughput

Thrashing



→ High-level paging activity is called as Thrashing

⟶ when CPU spends more time on page fault service as compared to process execution.

How to Handle Thrashing

Locality Model

1. Working Set Model
2. Page Fault Frequency

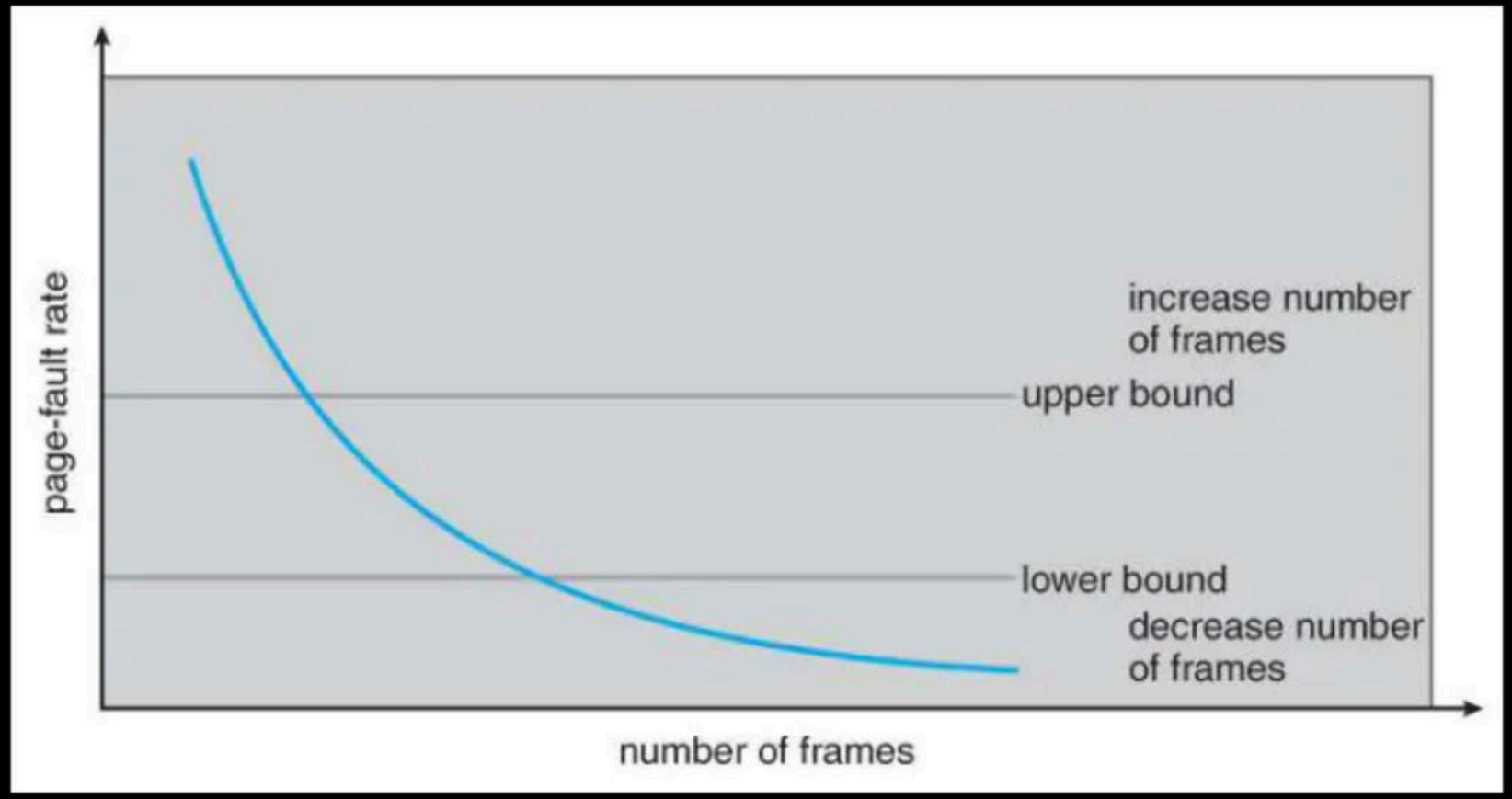
Working Set

↳ keeps min. no. of required pages of a process in mm

↳ working set

such that process experiences min. no. of faults.

Page Fault Frequency



Assume

Process Size = 16MB
Page size = 1KB
Page table entry size = 4Bytes

Page Table Size = ?

$$\text{no. of pages} = \frac{16\,MB}{1\,KB}$$

$$= 16K$$

↳ no. of pages * P.T.E. size

$= 16K * 4B$

$= 64K$ bytes

Now What?

Can you keep entire page table in single page? no

Page size = 1KB
P. T. size = 64KB

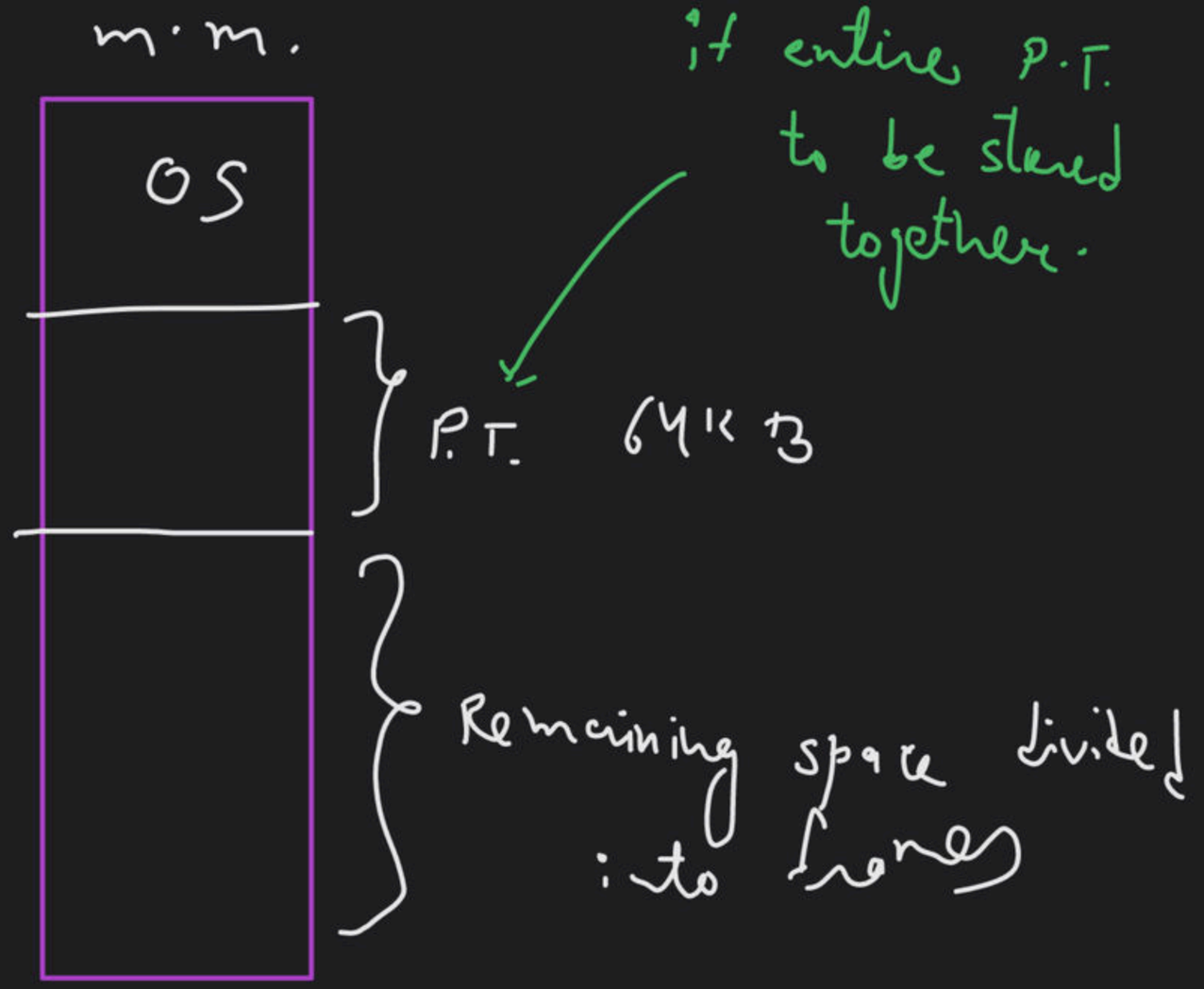
If not then how to access the specific entry?

unacademy

Prev. ques:-

Page size = 1 KB

P.T. size = 64 K bytes



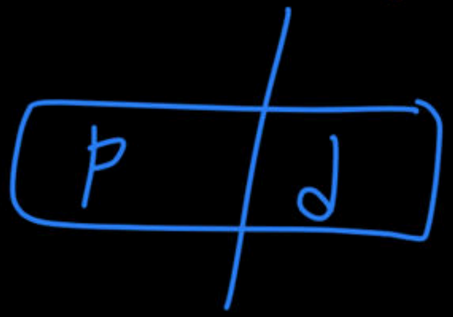
Let's Take a Simple Example

Process Size = 32B
Page size = 4B
Page table entry size = 1B

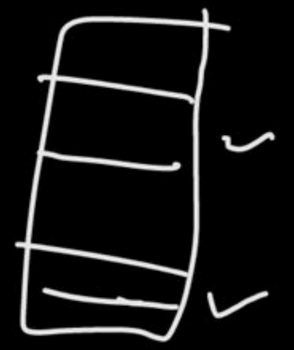
Page Table Size = ?

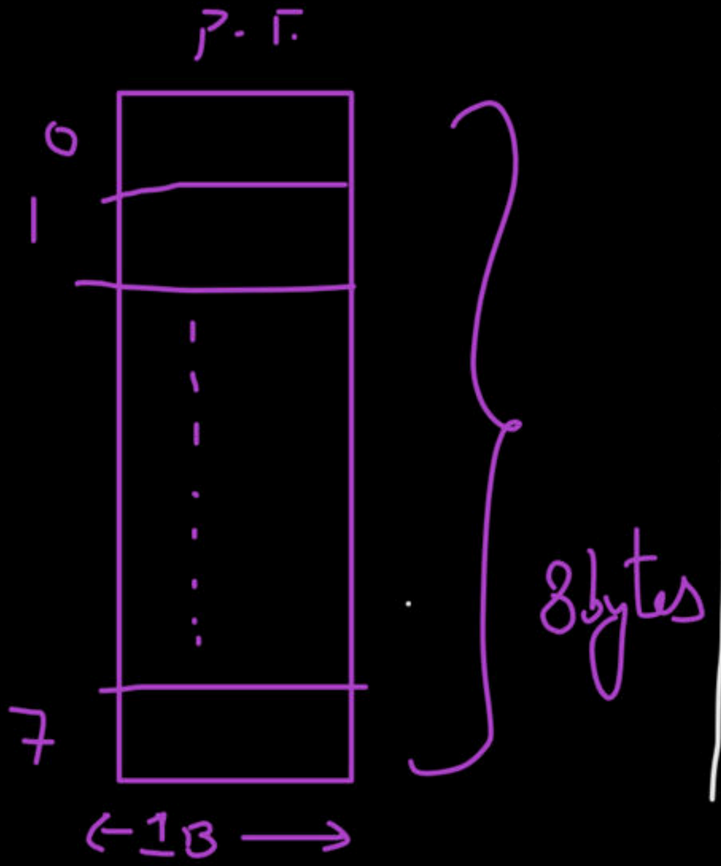
= 8 * 1B

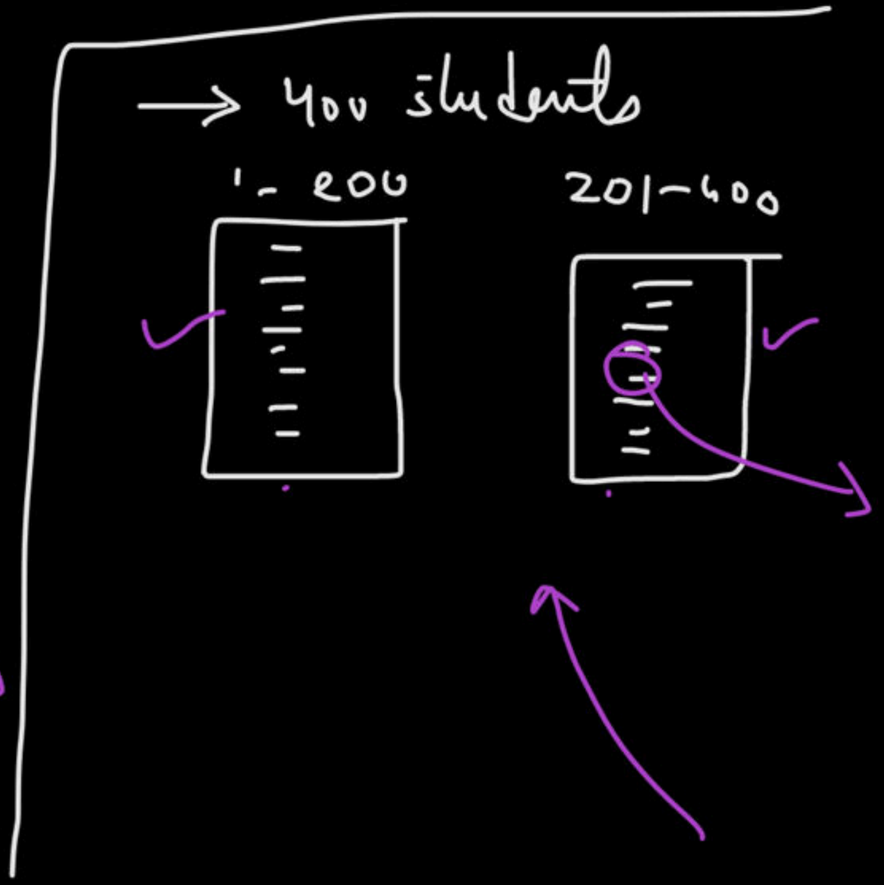
= 8 bytes



no. of pages = $\frac{32B}{4B}$ = 8



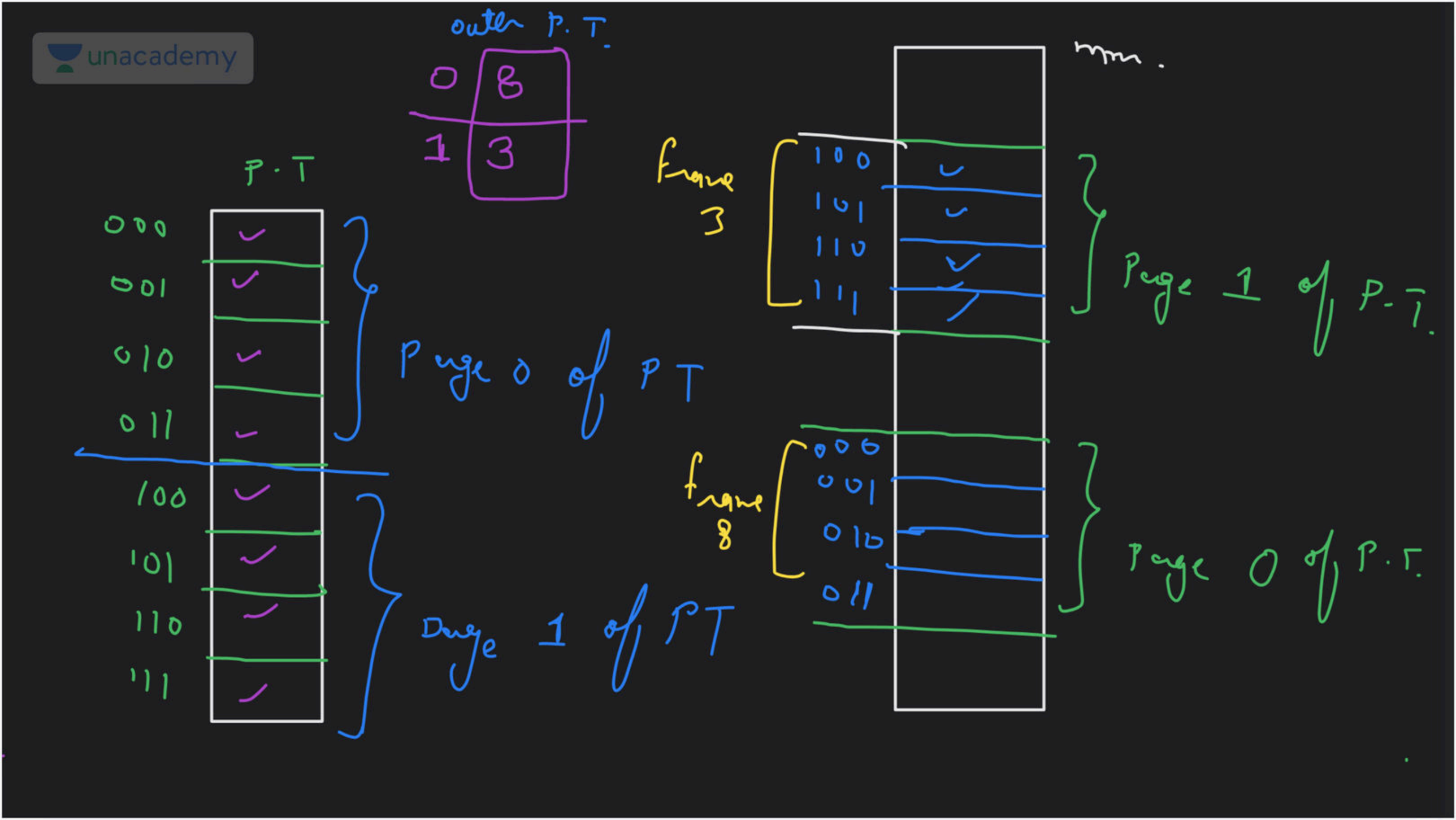




no. of pages of 4 bytes required to store P.T. of 8 Bytes

$$= \frac{8B}{4B} = 2$$

no. of entries per page $= \frac{4B}{1B} = 4$


outer P.T.
0 8
1 3
P.T
000
001
010
011
100
101
110
111
Page 0 of PT
Page 1 of PT
frame 3
100
101
110
111
Page 1 of P.T.
frame 8
000
001
010
011
Page 0 of P.T.

Page Table in Memory

CPU L.A. ⇒ p = 011

⇓

Page no. 011 P.T. entry is in which P.T. page?

⇓

P.T. page 0

⇓

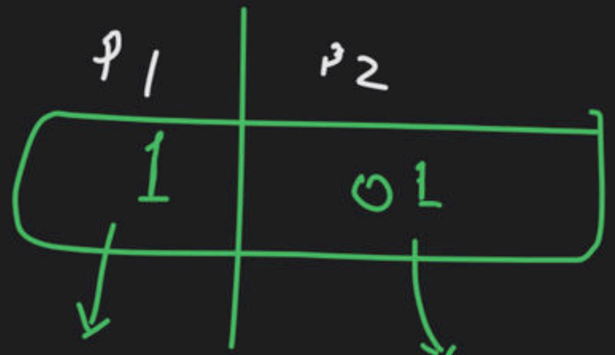
search in P.T. of P.T.

⇓

frame no. 8

P.T. entry for page 011

CPU L.A. $\Rightarrow$ P = 101



P.T. Page entry 01

1

⇓

Search P.T. of P.T.

⇓

frame 3

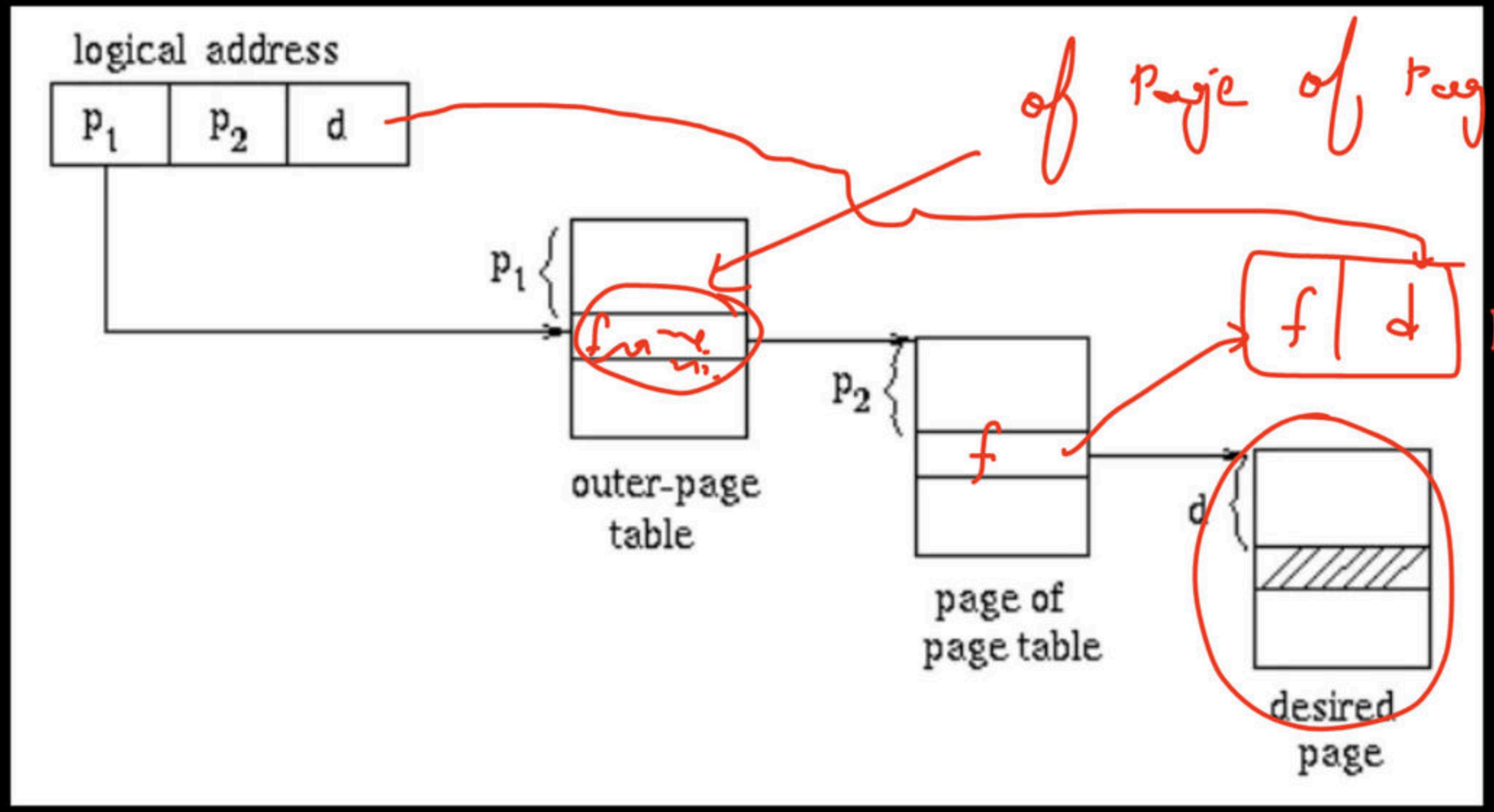
goto frame 31

access entry 01

Multilevel Paging

Multilevel Paging

Multilevel Paging



of page of page table to check for entry.

f | d

P.A.

Multilevel Paging

Question

Consider a virtual memory system with physical memory of 8GB, a page size of 8KB and 46-bit virtual address. Assume every page table exactly fits into a single page. If page table entry size is 4B then how many levels of page tables would be required.

Question

A processor uses 2–level page tables for virtual to physical address translation. Page tables for both levels are stored in the main memory. Virtual and physical addresses are both 32 bits wide. The memory is byte addressable. For virtual to physical address translation, the 10 most significant bits of the virtual address are used as index into the first level page table while the next 10 bits are used as index into the second level page table. The 12 least significant bits of the virtual address are used as offset within the page. Assume that the page table entries in both levels of page tables are 4 bytes wide. Further, the processor has a translation look-aside buffer (TLB), with a hit rate of 96%. The TLB caches recently used virtual page numbers and the corresponding physical page numbers. The processor also has a physically addressed cache with a hit rate of 90%. Main memory access time is 10 ns, cache access time is 1 ns, and TLB access time is also 1 ns.
Assuming that no page faults occur, the average time taken to access a virtual address is approximately (to the nearest 0.5 ns)

unacademy

Happy Learning.!