Worked out Examples

4.1 Paging suffers from ___________ _______________ and Segmentation suffers from __________ ____________ .

Ans: Internal Fragmentation, External Fragmentation

4.2 Which of the following is/are fastest memory allocation policy?

- **a. First Fit**
- **b. Best Fit**
- **c. Worst Fit**
- **d. Next Fit**
- **e. All of the above**
- **f. None of the above**

Ans: a and d

4.3 What replacement policy is used by Windows NT

- **a.** LRU
- **b.** NFU
- **c.** FIFO
- **d.** Clock Replacement
- **e.** None of the above

Ans: c

4.4 Consider a movie player application that supports functions like play movie, skip forward x frames and skip backward x frames. Suggest a memory management policy that will be best suited for this application.

Ans: A paging based system with a look ahead mechanism that loads the pages even before they are actually asked for would be most suitable. Also as a page replacement policy, all the frames that are closest to the current frame are not replaced (Based on locality of reference concept). This will help when the forward skip or backward skip functions are used.

4.5 What do you think are the flaws in following page replacement policies.

- **a. FIFO**
- **b. LRU**
- **c. NFU**

State the factors which should be considered for a good replacement policy and Hence design one of your own, giving reasons for each one you assume. Also prioritize these factors in accordance of significance, and justify your ranking.

Ans: Drawbacks in each Scheduling Policy

FIFO replacement policy

1 A page which is being accessed quite often may also get replaced because it arrived earlier than those present

2 Ignores locality of reference. A page which was referenced last may also get replaced, although there is high probability that the same page may be needed again.

LRU replacement policy

1 A good scheme because focuses on replacing dormant pages and takes care of locality of reference, but

2 A page which is accessed quite often may get replaced because of not being accessed for some time, although being a significant page may be needed in the very next access.

NFU replacement policy

1 Again doesn't take care of locality of reference, and reacts slowly to changes in it. If a program changes the set of pages it is currently using, the frequency counts will tend to cause the pages in the new location to be replaced even though they are being used currently.

Pages like page directory are very important and should not at all be replaced, thus some factor indicating the context of the page should be considered in every replacement policy, which is not done in all three of above.

Factors for a good replacement policy:

1 Context

2 Access count

3 Time of last access

4 Time of arrival

Priority

1 Context – To avoid replacing most significant pages
2 Access count along with time of arrival – Frequently accessed page w.r.t the time when the page was brought was brought into memory
3 Time of last access – Since access count along with time of arrival takes care of locality of reference (as a page which has come some time back is accessed quite frequently in **proportion**, it is high priority that the same page may be called again), thus last time of access is now not that important, although maintain its significance as such and thus should be considered.

Scheme

The best replacement policy would be one which takes up all the 3 factors listed above in the same order.

4.6 Implement following memory allocation policies with given data

FCFS

First Fit

Best Fit

Available memory: 35 units

OS : 10 units

User process : 25 units

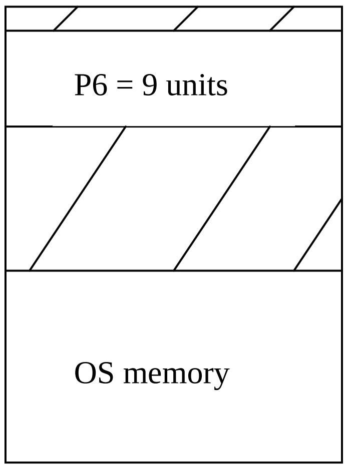
Time of arrival (units)	**0**	**3**	**7**	**12**	**18**	**25**	**29**
Processing Time (units)	**5**	**3**	**9**	**10**	**16**	**2**	**8**
Memory Required (units)	**3**	**5**	**8**	**12**	**2**	**6**	**9**

Ans : Start time = 0 (units)

End time = 37 (units)

Finally process P7 with memory of 9 units, is left before time 37 (units)

Status: time=35 to 37 (after process P5 completes)



4.7 If the hit ratio to a TLB is 80%, and it takes 15 nanoseconds to search the TLB, and 150 nanoseconds to access the main memory, then what must be the effective memory access time in nanoseconds?

a. 185

b. 195

c. 205

d. 175

Ans: b.

4.8 If the no of pages in a 32 bit machine is 8kB then what is the size of the page table?

a. 8kb

b. 16kB

c. 4 KB

d. **Cant say**

Ans: a

4.9 In a 64 bit machine, with 256 MB RAM, and a 4KB page size, how many entries will there be in the page table if its inverted?

a. 2^16

b. 2^50

c. 2^14

d. **None of the above**

Ans: a

4.10 If the total number of available frames is 50, and there are 2 processes one of 10 pages and the other of 5 pages then how much of memory would be proportionally allocated to each of these processes?

a. Depends on the process requirements

b. 33 and 16 frames respectively

c. Memory is allocated equally for both

d. 5 and 10 respectively

Ans: b

4.11 Consider a system with 80% hit ratio, 50 nano-seconds time to search the associative registers , 750 nano-seconds time to access memory.

Find the time to access a page

a. When the page number is in associative memory.

b. When the time to access a page when not in associative memory.

c. Find the effective memory access time.

Ans:

a. The time required is

50 nano seconds to get the page number from associative memory and 750 nano-seconds to read the desired word from memory.

Time = 50+750= 800 nano seconds.

b. Now the time when not in associative memory is

Time = 50+750+750= 1550 nano seconds

One memory access extra is required to read the page table from memory.

c. Effective access time = Page number in associative memory + Page number not in associated memory.

Page number in associative memory = 0.8 * 800.

Page number not in associated memory = 0.2 * 1550.

Effective access time = 0.8 * 800 + 0.2 * 1550 = 950 nano seconds

4.12 Consider a logical address space of 8 pages of 1024 words mapped into memory of 32 frames.

a. How many bits are there in the logical address ?

b. How many bits are there in physical address ?

Ans:

a. Logical address will have

3 bits to specify the page number (for 8 pages) .

10 bits to specify the offset into each page (2^{10} =1024 words) = 13 bits.

b. For (2^5) 11 32 frames of 1024 words each (Page size = Frame size)

We have 5 + 10 = 15 bits.

4.13 Consider the segment table:

What are the physical address for the following logical addresses :

a. 0,430

b. 1,10

c. 1,11

d. 2,500

Segment	***Base***	***Length***
0	**219**	**600**
1	**2300**	**14**
2	**90**	**100**
3	**1327**	**580**
4	**1952**	**96**

Ans:

a. 219+430 = 649.

b. 2300+10=2310.

c. 2300+11=2311

d. Illegal address since size of segment 2 is 100 and the offset in logical address is 500.

4.14 Consider a demand-paging system with the following time-measured utilizations: CPU utilization 20%, Paging disk 97.7%, Other I/O devices 5%. Which (if any) of the following will (probably) improve CPU utilization?

a. **Install a faster CPU.**
b. **Install a bigger paging disk.**
c. **Increase the degree of multiprogramming.**
d. **Decrease the degree of multiprogramming.**
e. **Install more main memory.**
f. **Install a faster hard disk or multiple controllers with multiple hard disks.**
g. **Add prepaging to the page fetch algorithms. (h) Increase the page size.**

Ans: f

4.15 On a system with paging, a process cannot access memory that it does not own; why? How could the operating system allow access to other memory? Why should it or should it not?

Ans: An address on a paging system is a logical page number and an offset. The physical page is found by searching a table based on the logical page number to produce a physical page number. Because the operating system controls the contents of this table, it can limit a process to accessing only those physical pages allocated to the process. There is no way for a process to refer to a page it does not own because the page will not be in the page table. To allow such access, an operating system simply needs to allow entries for non-process memory to be added to the process's page table. This is useful when two or more processes need to exchange data—they just read and write to the same physical addresses (which may be at varying logical addresses). This makes for very efficient interprocess communication.

4.16 Consider a paging system with the page table stored in memory. If a memory reference takes 200 nanoseconds, how long does a paged memory reference take?

a. **600 nanoseconds**
b. **200 naboseconds**
c. **400 nanoseconds**
d. **can't say**

Ans: c

4.17 Assume a page reference string for a process with *m* frames (initially all empty). The page reference string has length *p* with *n* distinct page numbers occurring in it. For any page-replacement algorithms, what is a lower bound & an upper bound on the number of page faults?

a. **n/2,p**
b. **p,n**
c. **n,p/2**
d. **n,p**

Ans: d

4.18 What is magic number?

Ans: Magic number is the special octal number which was originally used as identification for the executable files. This number was in fact a machine executable jump instruction which would simply set the program counter to fetch the first executable instruction in the file. It was previously used on **pdp-11 (UNIX) machine**, but in modern systems it is used to identify which application created or will execute a certain file.