Problem 1

Consider a simple paging system with the following parameters: 232 bytes of physical memory; page size of 210 bytes; 216 pages of logical address space. Answer the following questions: [10 pts]

1. How many bits are in a logical address?

**216 + 210 = 226**

**26 bits are in a logical address**

1. How many bytes are in frame?

**The page size is the same size as the frame, meaning the frame is 210 bytes**

1. How many bits in the physical address specify a frame?

**232/210 = 222**

**To specify a frame its 222 bits.**

1. How many entries in the page table

**There are 216 bits within the logical address space, meaning there is 216 number of entries in the page table.**

1. How many bits in each page table entry? Assume each page table entry includes a valid/invalid bit.

**23 bits are in each page table due to the number of bits required to specify the frame location in physical address is 22, meaning bits on each page is 23.**

Problem 2

Consider a paging system with page table stored in memory, answer the following [3 pts each]:

1. If a memory reference takes 1.2 microseconds, how long does a paged memory reference take?

**Length of time = time to check lookup table + accessing memory for instruction**

**To reference the memory taking 1.2 microseconds, then to reference instruction would also take 1.2, meaning total time taken to complete this process would be 2.4 microseconds.**

1. If we add 8 associative registers and 75% of all page table references are found in the associative registers, what is the effective memory reference time? Assume that checking a page table entry in the associative registers takes 0.1 microsecond.

**With 1.2 microseconds to access memory and 0.1 microseconds to search the page table we get the following equation**

**75% ~ Beginning Portion**

**0.75 \* (1.2 + 0.1) = 9.75 microseconds**

**then**

**25% ~ Remaining portion**

**0.25 \* (2.4) = 0.6 microseconds**

**Total time is going to be the sum of the two values to get the effective memory reference time of 10.35 microseconds**

Problem 3

Consider the following sequence of page references (each element represents a page number in a virtual memory system):

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

Show how many page faults would occur under each of the following policies:   
Assume only 3 frames are available and that they were initially empty.

1. FIFO. [3 pts]

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 2 | 1 | 3 | 3 | 2 | 3 | 4 | 5 | 4 | 5 | 1 | 2 | 3 | 5 |
| 1 | 1 | 1 | 4 | 4 | 4 | 1 | 1 | 1 | 1 | 1 | 4 | 4 | 4 | 4 | 4 | 2 | 2 | 2 |
|  | 2 | 2 | 2 | 5 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 5 |
|  |  | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 5 | 3 | 3 |
| F | F | F | F | F | F | F | f |  |  |  | F | f |  |  | F | F | F | F |

**TOTAL OF 14 FAULTS**

1. LRU. [3 pts]

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 2 | 1 | 3 | 3 | 2 | 3 | 4 | 5 | 4 | 5 | 1 | 2 | 3 | 5 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
|  | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 5 |
|  |  | 3 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 1 | 1 | 1 | 1 |
| F | F | F | F | F |  |  | F |  |  |  | F |  |  |  | F |  | F | F |

**TOTAL OF 11 FAULTS**

1. Optimal. [3 pts]

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 1 | 1 | 3 | 3 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 1 | 1 | 3 | 3 |
|  | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
|  |  | 3 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| F | F | F | F | F |  |  | F |  |  |  | F |  |  |  | F |  | F |  |

**TOTAL OF 9 FAUTLS**

Problem 4

Consider a page reference string for a process with a working set of M frames initially all empty. The page reference string is of length P with N distinct page numbers in it. For any page replacement algorithm, answer the following [3 pts each]:

1. What is the lower bound on the number of page faults?

**N is the lower bound number in this situation.**

1. What is the upper bound on the number of page faults?

**P is the upper bound case using the algorithm as reference.**

Problem 5

Consider the following requests for tracks that arrived to a hard disk in that order: 98, 183, 37, 122, 14, 124, 65, 67. The hard disk has 200 tracks total and the head is currently positioned on track 53. Answer the following [3 pts each]:

1. What is the average seek length under FIFO scheduling?

**45+85+146+85+108+ 110+59+2 = 640 tracks**

**To determine the average, divide total by 8 to determine the seek length in which results in**

**640/8 = 80 tracks per access.**

1. What is the average seek length under SSTF scheduling?

**12 + 2 + 30 + 23 + 84 + 24 + 2 + 59 = 236**

**divide by 8**

**then you get 29.5 - 30 tracks per access as the average seek length**

1. What is the average seek length under SCAN scheduling? Assume SCAN goes towards track 0 first and does not use LOOK.

**16 + 23 + 14+ 65 + 2 + 31 + 24 + 2 + 59 = 236**

**Then again divide by 8**

**Then you get around 29.5 - 30 also like SSTF in this case**