$\boxed{}$: Answer

Problem 1

Consider a simple paging system with the following parameters: 2^{32} bytes of physical memory; page size of 2^{12} bytes; 2^{16} pages of logical address space. Answer the following questions: **[10 pts]**

(a) How many bits are in a logical address?

(b) How many bytes are in frame?

(c) How many bits in the physical address specify a frame?

(d) How many entries in the page table

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

a)

m=16 : bits in page number

n=12 : bits to position byte data / page offset

m-n = Page num bits   n = offset

page number + page offset = logical addr

$\boxed{\text{logical addr has 28 bits}}$

b)

Page size == Frame size

$\boxed{2^{12} = \text{Frame Size} = 4096 \text{ bytes}}$

c)

Page Size = 2^{12} = Frame Size

bits in physical addr = $\log_2$(num frames) num Frames = $\frac{\text{tot Phys mem}}{\text{Frame Size}}$

$= \log_2\left(\frac{2^{32}}{2^{12}}\right) = \log_2\left(2^{20}\right)$

$\boxed{= 20 \text{ bits in phys addr}}$

d)

entries = $\frac{\text{Virtual addr Space}}{\text{page size}}$

$= \frac{2^{16}(2^{12})}{2^{12}} = 2^{16}$

$\boxed{= 2^{16} \text{ entries}}$

Problem 2

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

(a) If a memory reference takes 0.2 microseconds, how long does a paged memory reference take?

(b) If we add 64 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 10 nanosecond.

a)

- When referencing paged memory:

1. retrieve frame # from page table (mem ref)
2. retrieve actual data from mem addr

So, it takes...

2 mem refs
=
2(0.2 μS) = 0.4 μS

b) mem ref time 25% of time is 0.4 μS

mem ref time 75% of time is 0.2 μS + 10 nS

$$\boxed{\begin{aligned}\text{mem ref time} &= 0.25(0.4E\text{-}6) + 0.75(0.2E\text{-}6 + 10E\text{-}9) \\ &= 2.575E\text{-}7 \text{ sec or } 257.5 \text{ nS}\end{aligned}}$$

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:

(a) FIFO. **[3 pts]**

(b) LRU. **[3 pts]**

(c) Optimal. **[3 pts]**

Assume only 3 frames are available and that they were initially empty.

a)

	1	2	3	4	5	2	1	3	3	2	3	4	5	4	5	1	2	3	5
	F	F	F	F	F	F	F	F				F	F			F	F	F	F
a	1	1	1	4	4	4	1	1	1	1	1	1	5	5	5	5	5	3	3
b		2	2	2	5	5	5	3	3	3	3	3	3	3	6	1	1	1	5
c			3	3	3	2	2	2	2	2	2	4	4	4	4	4	2	2	2
	a	b	c	a	b	c	a	b				c	a			b	c	a	b

$\boxed{\text{14 FAULTS}}$

b)

	1	2	3	4	5	2	1	3	3	2	3	4	5	4	5	1	2	3	5
	F	F	F	F	F	F	F	F				F	F			F	F	F	F
a	1	1	1	4	4	4	1	1	1	1	1	4	4	4	4	4	2	2	2
b		2	2	2	5	5	5	3	3	3	3	3	3	3	3	1	1	1	5
c			3	3	3	2	2	2	2	2	2	2	5	5	5	5	5	3	3
Least Rec			1	2	3	4	5	2	2	1	1	2	3	3	3	4	5	1	
...		1	2	3	4	5	2	1	1	3	2	3	4	5	4	5	1	2	
↑ Most	1	2	3	4	5	2	1	3	3	2	3	4	5	4	5	1	2	3	

$\boxed{\text{14 FAULTS}}$

c) OPTIMAL

y

replace LRU

	1	2	3	4	5	2	1	3	3	2	3	4	5	4	5	1	2	3	5
	F	F	F	F	F			F				F				F		F	
a	1	1	1	1	1	1	1	3	3	3	3	4	4	4	4	1	1	3	3
b		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
c			3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	6	5

nextRefs in ?

4t	3t	9t	6t	6t	Nev 2t	NC
3t	2t	5t	~~3t~~	5t	4t	NC
5t	8t	6t		1		~~1~~

$\boxed{\text{8 FAULTS}}$

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**]:

(a) What is the lower bound on the number of page faults?

(b) What is the upper bound on the number of page faults?

a) The lower bound on the number of page faults for any policy in this case would be N distinct page numbers
- This would imply faults only occûr when page is referenced for the first time and can be referenced again w out fault

b) The Upper bound on the number of page faults for any policy in this case would be P references
- This implies that every reference is a page fault which is the worst case of all time