QUIZ 2
PRACTICE QUESTIONS

March 4, 2026

QUESTION 1 – BASE AND BOUNDS

- A computer system uses a simple base/bounds register pair to virtualize address spaces. For each traces fill in the missing values of virtual addresses, physical addresses, base, and/or bounds registers. In some cases, it is not possible to provide an exact value. If so, specify a range (e.g. greater than 100), or value that is not a single number.

Scenario 1

Virtual Address	Physical Address		
100	600		
300	800	Base?	500
699	1199		
700	[fault]	Bounds?	399

Q1 - 2

Scenario 2

Virtual Address	Physical Address		
300	1500	Base?	1200
1600	2800		
1801	3001 ?	Bounds?	> 2801
2801	4001		

Scenario 3

Virtual Address	Physical Address		
0	1000	Base?	1000
100	1100		
1999	2999	Bounds?	2000
2000	[fault]		

QUESTION 2 – SINGLE-LEVEL PAGE TABLE

- Consider a computer with 4 GB (2^{32}) of physical memory, where the page size is 4 KB (2^{12}). For simplicity assume than 1GB=1000MB, 1MB=1000KB, 1KB=1000 bytes
- (a) How many pages must be tracked by a single-level page table if the computer has 4GB (2^{32}) of physical memory and the page size is 4 KB (2^{12})?

$2^{32} \div 2^{12} = 2^{20} \sim 1$ million PFs

- (b) How many bits are required for the virtual page number (VPN) to address any page within this 4GB (2^{32}) memory space?
- (c) Assuming that the smallest addressable unit of memory within a page is a byte (8-bits), how many bits are required for the offset to refer to any byte in the 4 KB page?

20

- (d) Assuming each page table entry (PTE) requires 4 bytes of memory, how much memory is required to store the page table for one process (in MB)?

12

2^{20} PFs $\times$ $2^2 = 2^{22} = 4$ MB

Q2 - 2

- (e) Using this memory requirement, how many processes would fill the memory with page table data on a 4GB computer?

1000

QUESTION 3 – SINGLE-LEVEL PAGE TABLE EXAMPLE 2

- Consider a computer with 1 GB (2^{30}) of physical memory, where the page size is 1 KB (2^{10}). For simplicity assume than 1GB=1000MB, 1MB=1000KB, 1KB=1000 bytes
- (a) How many pages must be tracked by a single-level page table if the computer has 1GB (2^{30}) of physical memory and the page size is 1 KB (2^{10})?

$2^{30} \div 2^{10} \Rightarrow 2^{20}$ ~1 million pgs

- (b) How many bits are required for the virtual page number (VPN) to address any page within this 1GB (2^{30}) memory space?
- (c) Assuming that the smallest addressable unit of memory within a page is a byte (8-bits), how many bits are required for the offset to refer to any byte in the 1 KB page?

20

- (d) Assuming each page table entry (PTE) requires 4 bytes of memory, how much memory is required to store the page table for one process (in MB)?

10

$2^{20} \times 2^{2} = 2^{22} = 4$ MB

Q3 - 2

- (e) Using this memory requirement, how many processes would fill the memory with page table data on a 1GB computer?

$2^{30} \div 2^{22} = 2^8 = 256$ Processes

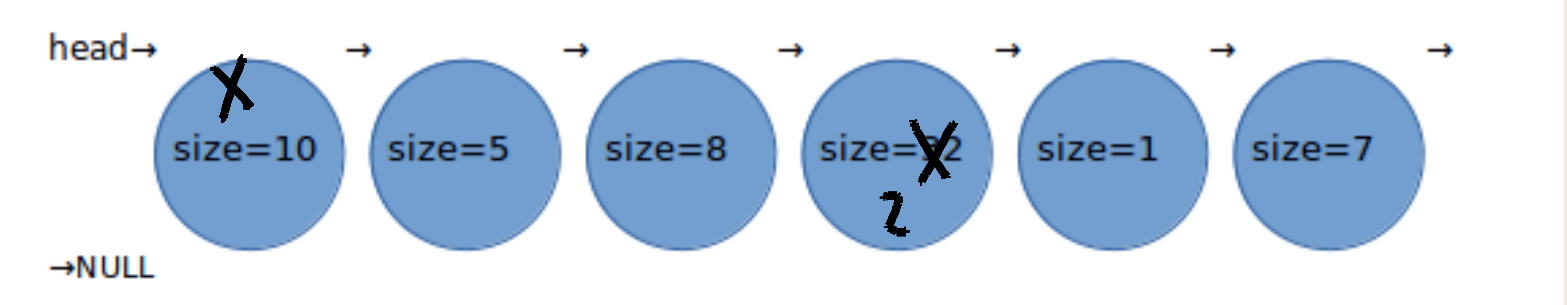
QUESTION 4 – FREE SPACE MANAGEMENT

- Free space management involves capturing a description of the computer's free memory using a data structure, storing this data structure in memory, and OS support to rapidly use this structure to determine an appropriate location for new memory allocations. An efficient implementation is very important when scaling up the number of operations the OS is required to perform.
- Consider the use of a linked list for a free space list where each node is represented by placing the following structure in the header of the memory chunk:

```
typedef struct __node_t
{
    int size;
    struct __node_t *next;
} node_t;
```

Q4 - 2

- Consider the following free space list:



- (a) Consider the **next fit** allocation strategy. For this free list above, how many comparison operations must be performed to identify a free chunk of **30-bytes** ?

- (b) After the last free space identification, the chunk is split and the remaining free space is returned to the free space list. Now, consider the **next fit** allocation strategy. After finding a free space for the previous request, how many comparisons are required to identify a free chunk of 10-bytes?

3 IF PTR ADVANCES OTHERWISE 4

Q4 - 3

- Now, after the last free space identification the chunk is split and the remaining free space is returned to the free space list. Now consider each of the following free space allocation strategies. How many comparisons are required on the updated free space list to find a free chunk of 2 bytes using:
- (c) best fit?
- (d) worst fit?
- (e) first fit?

5 OR (3)

5 (4)

1

Q5 – SLOPPY (APPROXIMATE) COUNTER

Below is a tradeoff space graph similar to those we've shown in class. For the sloppy (approximate) counter from Chapter 29, for the sloppy threshold (S), add numbers on the **left** or **right** side of the graph for each of the following tradeoffs:

1. High number of Global Updates
2. High Performance
3. High Overhead
4. High Accuracy
5. Low number of Global Updates
6. Low Performance
7. Low Overhead
8. Low Accuracy

Low sloppy threshold (S) 1 3 4 6

High sloppy threshold (S) 2 5 7 8

Q6 – CONCURRENCY BUGS

- (A) 97% of non-dead lock bugs with multi-threaded programming were found to be caused by which two problems?

Atomicity Violation
Order Violation

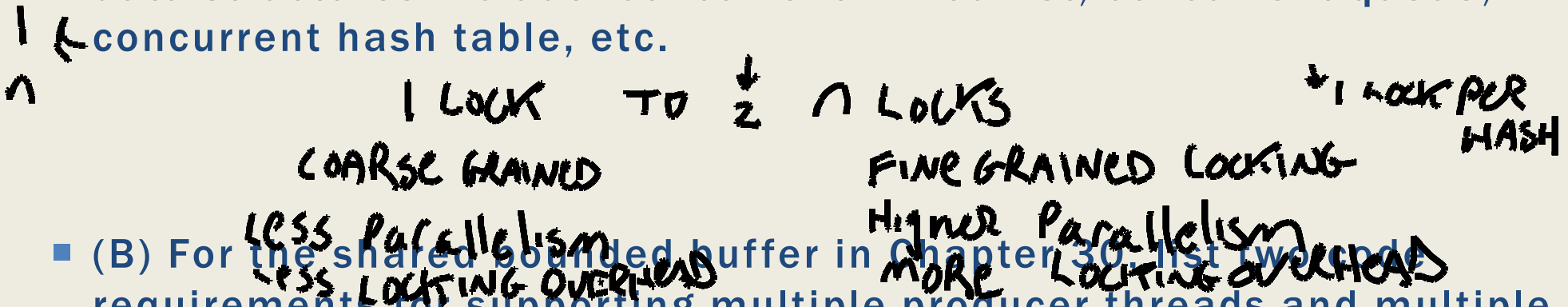
- (B) Briefly state the solution for each of the two problems

Atomicity – ADD locks TO enforce mutual exclusion for critical sections of code

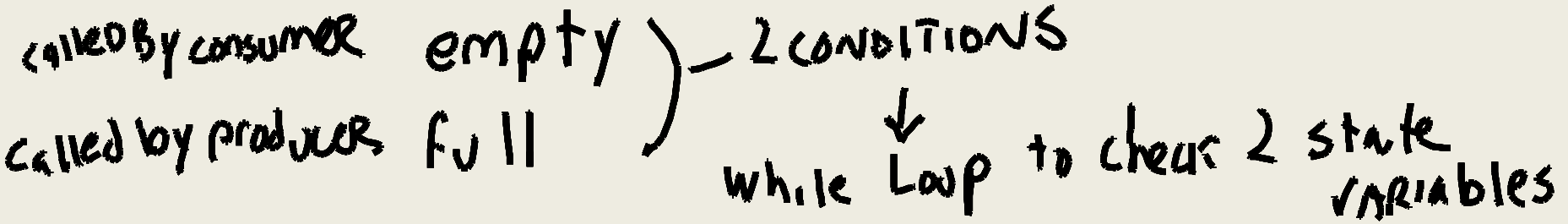
ORDER violation – ADD A COND VARIABLE + state VARIABLE to enforce a strict ORDERING of events in the code

Q7 – CONCURRENCY

- (A) In the context of concurrent data structures, describe the broad concept of the lock granularity trade-off. Examples of concurrent data structures include: concurrent linked-list, concurrent queue, concurrent hash table, etc.



- (B) For the shared bounded buffer in Chapter 30, list two code requirements for supporting multiple producer threads and multiple consumer threads?



Q8 – MEMORY APIS

- (A) For allocating dynamic memory on the heap, which API helps security by zero-ing any existing data that may be contained for the newly allocated memory returned to the programmer ?

calloc()

- (B) A programmer calls malloc() to allocate a huge array, and then later drastically reduces the size of the array by calling realloc(). What happens to the overall memory used by the program?

memory used by the program is drastically reduced

Q9 – FREE SPACE

- (A) From the perspective of the operating system, list one advantage of internal memory fragmentation

  Faster + easier for OS to return fixed size pieces that can be preallocated

- (B) From the perspective of the programmer, list one advantage of external memory fragmentation

  correctable w/ compaction

- (C) From the perspective of the programmer, list one disadvantage of internal memory fragmentation

  Not correctable w/ compaction memory is LOST!

QUESTIONS