CS/SE 4348 Operating Systems
Exam # 2
Duration: 75 minutes

Name: **Date**:

This exam is closed notes, and closed books. Cellphones and laptops must be turned off. No calculator allowed.

There are 5 parts. Maximum points = 40.

Cheers and Best wishes.

Part 1: (8 points)

A solution based on semaphore to readers-writers problem:

do { **wait(rw_mutex);** **...** **/* writing is performed */** **...** **signal(rw_mutex);** **} while (true);**	**do {** **wait(mutex);** **read_count++;** **if (read_count == 1)** **wait(rw_mutex);** **signal(mutex);** **...** **/* reading is performed */** **...** **wait(mutex);** **read count--;** **if (read_count == 0)** **signal(rw_mutex);** **signal(mutex);** **} while (true);**

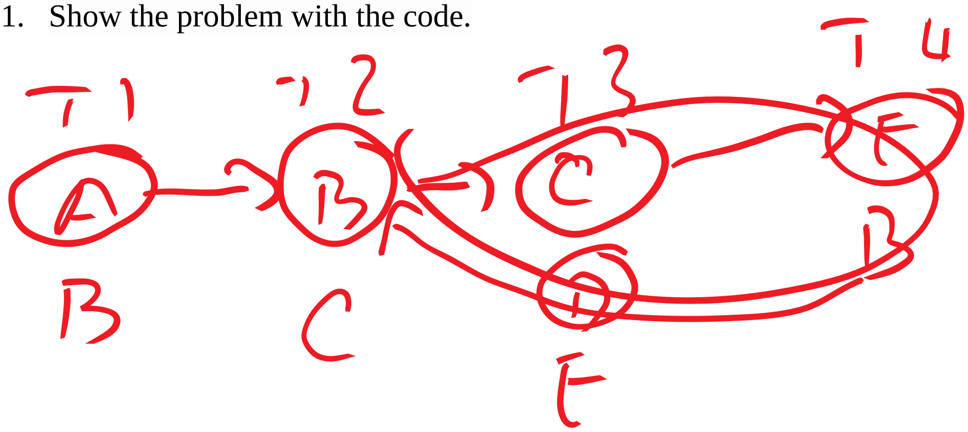
1. What is the initial value of **mutex**? 1
2. Assume that there are many readers. What is the initial value of **read_count**? 0
3. What is the initial value of **rw_mutex**? 1
4. Assume that value of **read_count** = 3. Is it possible for a reader to wait in the queue of semaphore **rw_mutex**? NO
5. Assume that value of **read_count** = 3. Is it possible for a reader to wait in the queue of semaphore **mutex**? YES
6. Assume that value of **read_count** = 0 and none of the reader process is in ready state. Is it possible for a writer to be (still) waiting in the queue of semaphore **rw_mutex**? NO
7. Assume that value of **read_count** = 1. Is it possible for a writer to be in the critical section performing some write? YES
8. Why is the above solution unfair to writers?

STARVATION

Part 2 (8 points)

A clueless CS student (not from my class☺) wrote a code using threads and locks. The code works as follows. Thread 1 tries to acquire locks A and B in an arbitrary manner. Thread 2 tries to acquire locks B and C, again, in an arbitrary manner. Thread 3 tries to acquire locks C, D, and E in some manner. Finally, Thread 4 tries to acquire locks E and B in an arbitrary manner.

1. Show the problem with the code.



2. Propose a correct solution.

$A < B < C < D < E$

a.

A	B	C	B
B	C	D	E
		E	

Part 3 (8 points)

Following questions are related to file systems, specifically to file sizes and maximum number of files supported by the files system. Assume the block size to be 4KB. Assume the size of the data region to be 4GB. Assume inode size = 256 bytes

1. Consider the file system with inodes. Assume that inode contains only direct pointers to 10 data blocks. What is the maximum file size supported by the inode?

$4 \times 10 = 40\,kB$

2^{128}

2. Assume that inode contains direct pointers to 10 data blocks and an additional single indirect pointer (to a block that contains pointers to blocks). What is the maximum file size? Assume 4 bytes are used to store a block number.

$40\,kB + \frac{\overset{1k}{4kB}}{\cancel{4}} \times 4k$

2^{10}

2^{20}

2^{30}

3. How many blocks are required to store the bitmap to find free inodes?

$2^{12} \wedge 2^{3} = 2^{15}$

$\frac{2^{32}}{2^{12}} = 2^{20}$

32

$\frac{\cancel{4GB}}{\cancel{4KB}}$

$\frac{2^{20}}{2^{15}} = 2^{5}$

a.

$\# \text{Files} = \frac{4GB}{4KB} = 2^{20} = \# \text{inodes}$

Part 4 (8 points)

Assume RAID with 4 disks. Let each disk be of size 4 GB.

1. In RAID 0 (stripe), what is the capacity available to store the data?

$4 \times 4 = 16GB$

2. In RAID 1 (mirror), what is the effective capacity available to store the data?

8

3. Describe in 3 lines how small write is handled in RAID 4 (parity).
   i. Read block of Parity block
   ii. compute parity
   iii. write data block & parity block

4. What is the major bottleneck in RAID 4? PARITY DISK

5. How is the bottleneck solved in RAID 5? ROTATE PARITY BLOCK

6. What is the latency for writing (sequentially) 16KB using RAID 0? D

D

Journaling

Virtualization