CSci 402 - Operating Systems

Final Exam (DEN Section)

Spring 2024

[9:00:00am-9:40:00am), Monday, May 6)

Instructor: Bill Cheng

Teaching Assistant: Zhuojin Li

(This exam is open book and open notes.
Remember what you have promised when you signed your
Academic Integrity Honor Code Pledge.)

Time: 40 minutes

_____-
Name (please print)

Total: 38 points

_____-
Signature

Instructions

1. This is the first page of your exam. The previous page is a title page and does not have a page number. Since this is a take-home exam, no need to sign above since you won't submit this file.
2. Read problem descriptions carefully. You may not receive any credit if you answer the wrong question. Furthermore, if a problem says *"in N words or less"*, use that as a hint that N words or less are expected in the answer (your answer can be longer if you want). Please note that points may get *deducted* if you put in wrong stuff in your answer.
3. If a question doesn't say `weenix`, please do not give `weenix`-specific answers.
4. Write answers to all problems in the **answers text file**.
5. For non-multiple-choice and non-fill-in-the blank questions, please show all work (if applicable and appropriate). If you cannot finish a problem, your written work may help us to give you partial credit. We may not give full credit for answers only (i.e., for answers that do not show any work). Grading can only be based on what you wrote and cannot be based on what's on your mind when you wrote your answers.
6. Please do *not* just draw pictures to answer questions (unless you are specifically asked to draw pictures). Pictures will not be considered for grading unless they are clearly explained with words, equations, and/or formulas. It's very difficult to draw pictures in a text file and you are not permitted to submit additional files other than the answers text file.
7. For problems that have multiple parts, please clearly *label* which part you are providing answers for.
8. Please ignore minor spelling and grammatical errors. They do not make an answer invalid or incorrect.
9. During the exam, please only ask questions to *clarify* problems. Questions such as "would it be okay if I answer it this way" will not be answered (unless it can be answered to the whole class). Also, you are suppose to know the definitions and abbreviations/acronyms of *all technical terms*. We cannot "clarify" them for you. We also will **not** answer any clarification-type question for multiple choice problems since that would often give answers away.
10. Unless otherwise specified and stated explicitly, multiple choice questions have one or more correct answers. You will get points for selecting correct ones and you will lose points for selecting wrong ones.
11. When we grade your exam, we must assume that you wrote what you meant and you meant what you wrote. So, please write your answers accordingly.

(Q1) (3 points) Let's say that you have four threads A, B, C, and D and you are using **stride scheduling**. You have decided to give thread A 8 ticket, thread B 9 tickets, thread C 10 tickets, and thread D 12 tickets. The initial pass values that **you must used** for the four threads are shown below along with the "winner" of the iteration 1. Please run **stride scheduling** to fill out all the entries (pass values) in the table and keep track of the "winner" in each round. For **iterations 2 through 7**, please write on your answer sheet the "winner" and the winning pass value of that iteration. (For example, you would write "B:9" for iteration 1 since B is the "winner" of iteration 1 and the winning pass value is 9.) You must use the **smallest possible integer stride values** when calculating all the pass values. If you get the stride values wrong, you will not get any partial credit for this problem.

itr	A	B	C	D
1	14	(9)	15	18
2				
3				
4				
5				
6				
7				

(Q2) (2 points) which of the following statements are correct about the **N x 1 (two-level)** thread implementation model?

(1) in this model, a user thread does not need to trap into the kernel to lock or unlock a mutex
(2) in this model, when one user thread wants to give up the processor to switch to another user thread in the same process, it must make a system call
(3) this model is used in the old days when a user space program didn't know that the kernel was multithreaded
(4) in this model, when a user thread makes a system call and gets blocked inside the kernel, other threads in the same process can still run as long as they don't make system calls
(5) in this model, thread creation and destruction are implemented as user space library function calls

Answer (just give numbers): _____

(Q3) (2 points) For a terminal device, it is possible that characters can arrive from the keyboard even though there isn't a waiting read request from an application. It is also possible that characters generation are too fast for the terminal to display them. What **kernel data structures** are used to **solve these two problems**?

(1) page table
(2) B+ tree and hash map
(3) shadow objects
(4) memory management unit (MMU)
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q4) (2 points) Which of the following are maintained in a **S5FS superblock**?

(1) checkpoint files
(2) disk map
(3) inode cache
(4) first node of the free disk block list
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q5) (2 points) Let's say that you are using a **rate-monotonic scheduler** to schedule 4 periodic tasks with $T_1 = 0.5$, $P_1 = 5.5$, $T_2 = 1$, $P_2 = 6$, $T_3 = 0.5$, $P_3 = 6.5$, and $T_4 = 1$, $P_4 = 7$ (all in seconds). If you were to simulate the **rate-monotonic scheduler** to see if it can schedule these 4 periodic tasks without any of them missing a deadline, assuming that you schedule all 4 periodic tasks to start a time = 0, how many seconds into the simulation would be the first time all 4 jobs would be scheduled to start executing at exactly the same time again? Please just give a numeric answer (no partial credit for this problem).

(Q6) (2 points) Consider the follow implementation of mutex_lock() and mutex_unlock() for a single CPU system (this is from **straight-threads synchronization**):

```
void mutex_lock(mutex_t *m)
{
  if (m->locked) {
    enqueue(m->queue,
        CurrentThread);
    thread_switch();
  } else
    m->locked = 1;
}
```

```
void mutex_unlock(mutex_t *m)
{
  if (queue_empty(m->queue))
    m->locked = 0;
  else
    enqueue(runqueue,
        dequeue(m->queue));
}
```

Let's say that currently, thread X owns mutex **m1** and thread Y owns a different mutex **m2**. Let's say that we have a non-preemtive kernel with interrupt enabled and thread X is currently waiting for I/O and interrupt won't occur until much later. Which of the following statements are correct about what would happen if thread Y now calls **mutex_lock(m1)**?

(1) thread X will enter **mutex_unlock()** immediately and thread Y will return from mutex_lock() immediately with mutex **m1** locked
(2) thread X will enter **thread_switch()** and as a result, thread Y will return from **thread_switch()** immediately
(3) thread Y will go to sleep in **m1**'s mutex queue
(4) thread Y will enter **thread_switch()** and as a result, thread X will return from **thread_switch()** immediately
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q7) (2 points) Which of the following statements are correct about **crash resiliency** in file systems?

(1) to provide crash resiliency in Linux file systems, "shadow paging" is preferred over transaction-based approaches
(2) journaling is not available to file systems that uses "extents"
(3) to provide crash resiliency in Linux file systems, ext3 uses journaling (which improved on ext2 which does not use journaling)
(4) a major problem with "soft update" is hierarchical dependency among modified disk blocks
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q8) (2 points) Which of the following statements are correct about a **B+ tree of order** $m = 9$?

(1) since m is 9, it's okay for an intermediate node (i.e., neither a root node nor a leaf node) to have 5 child nodes
(2) since m is 9, the root node must have at least 5 child nodes
(3) since m is 9, it's okay for the root node to have 9 child nodes
(4) since m is 9, it's okay for an intermediate node (i.e., neither a root node nor a leaf node) to have 4 child nodes
(5) since m is 9, the height of the B+ tree must be strictly less than 9

Answer (just give numbers): _____

(Q9) (2 points) Which of the following statements are correct about thread implementation strategies?

(1) N x 1 model is preferred over 1 x 1 model because N x 1 model can achieve higher parallelism
(2) one main problem with the 1 x 1 model is that it's slow because locking and unlocking mutex are slow for this model
(3) one problem with the M x N model is priority inversion
(4) one main problem with the N x 1 model is that it's slow because system calls are slow for this model
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q10) (2 points) Which of the following statements are correct about **terminal driver** and **pseudo-terminal driver**?

(1) typically, pseudo-terminal driver and terminal driver are both user space drivers
(2) for a pseudo-terminal, the input and output (on the device end) comes from and goes to the window manager, which is a kernel-only process
(3) a terminal driver typically runs in kernel while a pseudo-terminal driver typically runs in user space
(4) for a pseudo-terminal, the output (on the device end) goes indirectly to the window manager, which is a user space process
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q11) (2 points) Which of the following statements are correct about **executing sensitive instructions** in an IBM 360 **virtual machine**?

(1) when a sensitive instruction is executed in the **virtual user mode** inside the **virtual machine**, the sensitive instruction will get delivered to the guest OS
(2) when a sensitive instruction is executed in the **privileged mode** of the **real machine**, it should cause a trap into the VMM
(3) when a sensitive instruction is executed in the **virtual privileged mode** inside the **virtual machine**, it should get delivered to the guest OS
(4) when a sensitive instruction is executed in the **VMM**, the sensitive instruction will get emulated by the VMM
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q12) (2 points) Which of the following statements are correct about the **round-robin (RR)** scheduler?

(1) with this scheduler, it's still possible for a thread to "starve" at the scheduler
(2) time-slice values should be as large as possible
(3) time-slice values should be as small as possible
(4) it typically has a larger average waiting time than SJF scheduling
(5) it typically has a smaller variance in waiting time than SJF scheduling

Answer (just give numbers): _____

(Q13) (3 points) Let's say that you have four threads A, B, C, and D and you are using the basic **round-robin (RR) / time-slicing** scheduler with a very small time slice. At time zero, all four threads are in the run queue and their processing times are shown in the table below. Assuming that there are no future arrivals into the run queue, please complete the table below with the "waiting time" of all four threads and the "average waiting time" (AWT) of these four threads and write the results on your answer sheet. Please make it very clear which waiting time is for which thread and which one is the AWT. For non-integer answers, you can use fractions or decimals with two digits after the decimal point. Your answer must not contain plus or multiplication symbols. You must use the definition of "waiting time" given in lectures.

	A	B	C	D	AWT (1 pt)
T (hrs)	15	13	13	12	-
wt (hrs)					

(Q14) (2 points) Let's say that the address space of a user space process in **weenix** looks like the following:

```
    VADDR RANGE       PROT FLAGS    MMOBJ    OFFSET    VFN RANGE
0x08039000-0x08048000 rw-  PRIVATE 0xcfe0c034 0x00018 0x08039-0x08048
0x08048000-0x0805e000 r-x  PRIVATE 0xcfe0c004 0x00019 0x08048-0x0805e
0x0805e000-0x0806f000 rw-  PRIVATE 0xcfe0c064 0x0001a 0x0805e-0x0806f
```

If you get a page fault with vaddr = 0x08065786, what **pagenum** would you use to lookup a page frame when you are handling a page fault? Please just give an integer value answer (no partial credit for this problem).

(Q15) (2 points) For a terminal, input characters may need to be processed/edited in some way before they reach the application. Which of the following **data structures** are used to solve this problem?

(1) a translation lookaside buffer
(2) a hash table that uses extensible hashing
(3) a B tree or a B+ tree
(4) a memory map and a page table
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q16) (2 points) Which of the following statements are correct about **microkernel**?

(1) in the design of the microkernel architecture, device drivers cannot be moved into user space
(2) almost all microkernel implementations have good performance
(3) access control in a microkernel system typically is based on user IDs and group IDs just like in a traditional Unix system
(4) one main differences between a message port and a Unix pipe is that you can assign names to Unix pipes
(5) none of the above is a correct answer

Answer (just give numbers): _____

(Q17) (2 points) The naive **spin lock** looks like the following:

```
void spin_lock(int *mutex) {
  while(CAS(mutex, 0, 1))
    ;
}
```

Which of the following statements are correct about a thread calling the above function to lock a mutex in a **multiple CPU** system?

(1) if the mutex is currently held by another thread, this code can be very inefficient
(2) if the mutex is not held by any thread and if only one thread is calling this function, this function can execute efficiently
(3) this code is inefficient because **CAS()** is a system call
(4) **CAS()** is only inefficient if this code is executing in the user space (but much more efficient if this code is executing in kernel mode)
(5) the code is every inefficient no matter if another thread is holding the mutex or not

Answer (just give numbers): _____

(Q18) (2 points) Which of the following statements are correct about **futex**?

(1) futex is designed to be used inside the kernel in multiple CPU systems
(2) futex is considered "fast" because if the futex is available, a user thread can lock it quickly in user space without making a system call
(3) if a futex is currently **locked** and not being released any time soon, a thread calling `futex_lock()` must enter the kernel to wait for the lock to be released
(4) futex is designed to work only in multiple CPU systems and will not work in single-CPU systems
(5) none of the above is a correct answer

Answer (just give numbers): _____