

CSE 120 Principles of Computer Operating Systems
Winter Quarter, 2007
Final Exam

Instructors: Keith Marzullo and Geoffrey M. Voelker

One Lock to rule them all,
One Lock to find them
One Lock to bring them all...
and in the Project bind them.

Three Locks for the Mailboxes under the sky,
Seven for the Readers/Writers in their halls of stone
Nine for Whales doomed to mate,
One for the Professor on his throne

In the Land of Nachos where Concurrency lies,
One Lock to rule them all, One Lock to find them
One Lock to bring them all and in the Project bind them.
In the Land of Nachos where Concurrency lies.

Name _____

Student ID _____

1	/12
2	/15
3	/15
4	/16
5	/10
6	/20
7	/24
8	/10
9	10/10
Total	/132

Attention: This exam has ten questions worth a total of 140 points (and the last one is a freebie). You have three hours to complete the questions. As with any exam, you should read through the questions first and start with those that you are most comfortable with. If you believe that you cannot answer a question without making some assumptions, state those assumptions in your answer.

Best of luck, and have a great spring break!

1. (12 pts) Potpourri: Answer yes or no, or with a single term or short answer, as appropriate. You should go through these quickly, answering with the first answer that comes to mind — it probably is the correct one. Only dwell on ones you are unsure of after finishing the other questions.

- Is a password a capability or an access control?
- In Nachos Project 3, are the # of PageIns always/sometimes/never more than the # of PageFaults?
- What bit in the PTE is used to determine whether a page is not in physical memory?
- For any sequence of page references, an LRU page replacement policy always/sometimes/never generates no more page faults than a FIFO page replacement policy.
- RPC strives to be as much like local procedure calls as possible to programs. Identify one way in which this transparency can break down.
- Belady's Algorithm is an optimal page replacement algorithm. What does optimal mean in this context?
- True or False: a reference to a memory mapped file will never page fault.
- A file can always/sometimes/never have more than one inode associated with it?
- Does the file buffer cache exploit temporal, spatial, or both kinds of locality?
- When using an LRU page fault replacement policy, for any reference string, increasing the number of page frames will always/sometimes/never increase the number of page faults.
- In general, are there much fewer/nearly the same/many more TLB misses than page faults?
- True or False: using demand paging removes all forms of physical memory fragmentation.

2. (15 pts) Identify the following and state their purpose in a couple of sentences (do more than just expand the acronym, though).

(a) Internal fragmentation

(b) RPC client stub

(c) Cylinder group

(d) Global page replacement

(e) Software managed TLB

3. (15 points) Consider the following fragment of a page table:

<i>page number</i>	<i>frame</i>	<i>valid</i>	<i>permissions</i>	<i>referenced</i>	<i>dirty</i>
12	40	1	r	1	0
13	32	0	rx	0	0
14	1	1	rw	0	0
15	11	1	rw	0	0
16	17	1	rx	0	0
17	6	1	r	0	0

The valid bit indicates whether or not the page frame listed in the page table entry is, in fact, associated with the virtual page number.

The virtual memory system uses Clock with the clock hand scanning the pages in increasing virtual page number. The hand is currently pointing to page 14, and this is the page that Clock will consider next.

Assume that the page size (and page frame size) is 100 bytes. Consider the operations given below. They are executed in order. For each operation, indicate what happens. For example, if the operation results in a fault, say what kind of fault it is. If it is a page fault, then say which page frame is used to satisfy the page fault and how the clock hand and the page table are updated. If the operation does not result in a fault, then say which physical address results from the virtual address translation and say how the page table is updated.

(a) read virtual address 1280.

(b) write virtual address 1444.

(c) write virtual address 1745.

(d) read virtual address 1310.

(e) jump to virtual address 1660.

4. (16 pts) The Java 2 Platform Standard Edition v1.5.0 API documentation for the `Thread::stop()` method states: “**Deprecated.** This method is inherently unsafe.” `Thread::stop()` ends a thread, and is analogous to `Finish()` in Nachos. When `stop` is called, it causes the thread to unlock all of the monitor locks it has acquired (in Java, a thread acquires a monitor lock by, e.g., executing a synchronized method on an object).

(a) When a thread has its `stop` method invoked, why should it release all of its monitor locks?

(b) Given that a thread will release all of its locks, why is `stop` inherently unsafe and deprecated?

(c) If a thread cannot be explicitly stopped, what do you think the Java designers recommend instead?

(d) The Java API also deprecates `Thread::suspend`, which is equivalent to the Nachos `Sleep()` method. Unlike `stop`, `suspend` does not release monitor locks. If Java programmers use `suspend`, what are the Java developers afraid can easily happen to programs that use it? Why can this happen?

5. (10 points) Suppose that there are three programs A , B and C that generate the following reference strings:

- $A : (0, 1, 2)^{100}$
- $B : (3, 4, 5)^{100}$
- $C : (6, 7, 8)^{100}$.

where the notation $(0, 1, 2)^{100}$ means the sequence 0, 1, 2 repeated 100 times. These programs are to be run concurrently: thus, the resulting reference string will be some merge of these three strings. There are eight page frames that are initially empty.

(a) What is the largest number of page faults that can occur running these programs using demand-paged LRU page replacement algorithm? Explain.

(b) What is the largest number of page faults that can occur running these programs using a working set algorithm with $\tau = 3$? Explain.

6. (20 points) The shell line

```
echo hello > foo
```

creates a file foo (assuming that it doesn't already exist), opens the file, writes the string "hello" to the file, and closes the file.

(a) How many file data blocks are directly written as a result of this execution? Which blocks are they?

(b) What metadata is modified by the execution of this shell line?

(c) Give an instance of a reliability-induced synchronous write that will be generated during the execution of this shell line.

(d) Explain what benefit, if any, an inode cache would during the execution of this shell line.

(e) If you were designing a file system, you would need to decide whether the close operation would flush cached data blocks of that file. What are the benefits and drawbacks of doing so?

7. (24 pts) In Nachos Project 3, you implemented demand paged virtual memory using a single-level page table. In this problem, you will outline an implementation of demand paged virtual memory using a Translation Lookaside Buffer (TLB).

In Nachos, the TLB is simply a small array of `TranslationEntries` (the default size is 4 entries). When translating virtual addresses in `Machine::Translate`, Nachos searches through the `TranslationEntries` in the TLB array to find an entry that matches the address being translated. Otherwise, it raises a `PageFaultException`. Answer the questions below descriptively, do not use pseudo-code.

- (a) How does `Machine::Translate` know that it found the correct `TranslationEntry` to translate a virtual address?

- (b) The semantics of a `PageFaultException` will change. What does a `PageFaultException` indicate now?

- (c) What, if anything, needs to be done to the TLB as part of paging in from `BackingStore`? Why?

(d) What, if anything, needs to be done to the TLB as part of page eviction? Why?

(e) TLBs support a flush operation. Why? When will you invoke it?

(f) Will the TLB ever hold a `TranslationEntry` that is invalid (the `valid` bit is `FALSE`)? Why?

8. (10 pts) Briefly describe the most challenging bug that your group encountered in project 2 or 3, how you found it, and how long it took you to find it. Which of your group members do you think will give the best answer to this question?

9. (10 pts) You get full credit for the problems below whether you answer them or not, or what your answer is. In other words, these questions will have no impact on your grade. **Do not** take time to write answers to these questions until you are finished with the rest of the exam, if you choose to do so at all.

(a) What topic in this course struck you as the least interesting or least relevant?

(b) What topic in this course was the most interesting to you?

(c) What operating system topic were you hoping to learn about, but we didn't cover?