

Operating Systems — CSCI 4630

Robert Hochberg, Instructor
Spring Semester, 2002
Office: Austin 325B
Phone: 328-0126
Email: hochberg@cs.ecu.edu
Website: www.cs.ecu.edu/~hochberg

| | | |
|--------------------------------------|------------|-------------|
| Official Office Hours | MWF | 9-10am |
| | MW | 11:15-12:15 |

But also by appt., of course. Also, stop by any time for help, but call first to make sure I'm in.

“IF YOU GET STUCK, DON'T STAY STUCK.”

Text: *Modern Operating Systems* by Tannenbaum, 2nd Edition.

Homework: In a class at this level, I expect you to keep up with the material by reading the appropriate sections in the text (a reading guide will be provided on the course website). Therefore written homework will not be given. Instead, there will be quizzes to make sure you are doing the reading, and programming assignments to give you the hands-on experience.

Quizzes: There will be 10 unannounced 10-point quizzes during the semester. If you are absent for a quiz, your grade will be 0.

Programs: There will be 5 programming assignments, each worth 30 points. Late work will absolutely not be accepted.

Exams: You will have two 50-minute in-class exams, worth 100 points each, and one cumulative final exam, during finals week, worth 150 points.

Final Grade: With quizzes, programs, regular exams and the final, you will have $100 + 150 + 200 + 150 = 600$ points available. You will receive:

- at least an A if you get > 539
- at least a B if you get > 479
- at least a C if you get > 419
- at least a D if you get > 359

Notes: There are no make-up exams or quizzes available after the fact. Absences may be excused in advance on a per-case basis. It is the responsibility of the student to stay on top of the material, by doing the reading and attending the classes. If you miss a class, see me or another student to find out about announcements, material covered, etc... DO NOT FALL BEHIND! This is the kiss of death in any computer science course, especially one like this, where the material is at times rather abstract, new material builds on old material and there are frequent definitions. Remember, you learn a new thing by doing it, not by watching it. You must work in this class in order to succeed; and often, time spent on the programming assignments makes the difference.

East Carolina University seeks to fully comply with the Americans with Disabilities Act (ADA). Students requesting accommodations based on a covered disability must go to the Department for Disability Support Services, located in Brewster A-114, to verify the disability before any accommodations can occur. The telephone number is 252-328-6799.

1. We said that operating systems can be viewed in two different ways. (For a hint, the initials of these two ways are “R.M.” and “E.M.”) List these two ways, and give a very *brief, one-sentence* description of each.

Resource Manager: The OS deals with the hardware components of the computer.

Extended Machine: The OS presents the computer to the user in a usable format.

2. Draw a diagram showing the three states that a process can be in and the possible transitions between those states.

3. What is the “process table,” and what is the most important information stored therein?

5. For each of the following, answer either “User Threads” or “Kernel Threads.” [8]
 - a. Which implementation requires more overhead on a context switch?
 - b. Which implementation is more allows for quicker thread-switching?
 - c. Which implementation would be better for a process doing lots of I/O?
 - d. Which implementation could be provided by a library on top of a non-threading OS?

6. Barnes and Noble maintains a database containing the account balance for each B&N CafeCard holder. Unfortunately, this database does not implement any method to guarantee protected access to its data. Let B be the variable which contains the balance (initially \$30) in some account, and suppose that two people use their cards at the same time, to purchase drinks costing \$5 and \$3 respectively:
 - a. Give an example of a sequence of events which could result in the wrong value being saved back to the database. [5]

 - b. What are all possible final values that could occur following these two transactions? [6]

7. We described two operations $P(s)$ and $V(s)$ on a semaphore s . [6]
 - a. What does P do?

 - b. What does V do?

 - c. What is special about the way P and V are performed that enable them to solve the “race conditions” problem?

8. Give a bit of code (or pseudocode) for the P and V operations: [10]

| | |
|---|---|
| <pre>void P(semaphore &s){ }</pre> | <pre>void V(semaphore &s){ }</pre> |
|---|---|

9. What is meant by a “binary semaphore?” [4]

10. We discussed monitors and how they can be used for interprocess synchronization [6]

a. Briefly describe what a monitor looks like in a program

b. Which of the following is most responsible for making sure monitors guarantee mutual exclusion: The programmer, the OS, the kernel, the compiler or the user?

11. What is meant by a “test and set lock” (TSL) instruction? In particular, mention what the return value of a TSL instruction is, and mention what is special about the way TSL instructions are executed. [5]

12. Below is a bit of code which attempts to solve the producer-consumer problem with semaphores. It doesn't work. Make appropriate corrections in the initialization, Producer and Consumer code. [10]

| Initialization: | |
|---|---|
| <pre>full = 0; //initialize semaphore empty = 0; //initialize semaphore mutex = 0; //initialize a mutual exclusion semaphore</pre> | |
| Producer: | Consumer: |
| <pre>while(1){ Produce_item(thing); P(empty); V(mutex); Insert_item(thing); P(mutex); V(empty) }</pre> | <pre>while(1){ P(full); V(mutex); Remove_item(thing); P(mutex); V(full) Consume_item(thing); }</pre> |

13. The solution below attempts to solve the producer/consumer problem using message passing. It almost works. All that is missing is some sort of initialization to “get the ball rolling.”
- a. What would happen if the code was run as written? [4]

b. Add a snippet of code to fix this problem. [6]

| Producer: | Consumer: |
|--|---|
| <pre>while(1){ message m; while (TRUE) { make_item(thing); receive(consumer, &m); build_message(&m, thing); send(consumer, &m); } }</pre> | <pre>while(1){ message m; while (TRUE) { receive(producer, &m) extract_item(&m, thing); send(producer, &m); consume_item(thing); } }</pre> |

14. Define each of the following terms in one succinct sentence: [6]

a. Context switch

b. Scheduler

c. quantum

15. Suppose that a quantum is 80ms and that a context switch takes 20ms. What fraction of the time is the CPU doing work on processes? [5]

16. What are meant by the terms *CPU-bound* and *I/O-bound*? [4]

1. How long is a “jiffy” in Linux scheduling?

2. What is the arithmetical relationship between *goodness*, *priority* and *quantum* in the Linux scheduling algorithm?

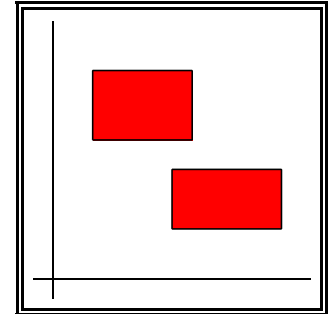
3. A certain operating system maintains 4 priority levels for its processes: *I/O*, *short quantum*, *Terminal*, and *long quantum*.
 - a. Arrange them in order from highest to lowest priority

 - b. Suppose I tell you that a process had spent several minutes classified as a long quantum process, and then suddenly switched to another priority level, gradually working its way back to the long quantum priority level, passing through the other two levels as well, spending a brief time in each. Do you suppose that this process was I/O-bound or CPU-bound? And what event could have happened to cause this priority-hopping to take place?

4. A certain operating system uses “ageing” to estimate how long a process will spend on the CPU on its next timeslice. Let t be the estimated value and let d be the amount of time the process last spent on the CPU. The formula it uses to update its estimate is: $t \leftarrow \frac{1}{2} t + \frac{1}{2} d$. If t has the value “10” and the next three times it spends on the CPU are 8, 7 and 12, what will the final value of t be?

1. Draw the trajectory diagram for the two processes shown below, which request the resources in the order shown. (As a polite reminder, I've drawn a sample trajectory diagram to the right.)

| Process 1 | Process 2 |
|-----------|-----------|
| Request A | Request B |
| Request B | Request A |
| Release A | Release B |
| Release B | Request B |
| | Release B |
| | Release A |



2. For the processes and resource requests given above, is it possible for deadlock to occur? Explain your answer, referring to your diagram.
3. Suppose that each of k processes that run on a certain system will be using the CPU at any given time with probability p , independently of one another. What is the probability that at any given time, all k processes will be blocked — that is, will *not* be able to use the CPU.

Operating Systems
Exam II

Name _____
April 8, 2002

This exam has 11 questions. Please make sure that you have them all. There are a total of 110 points on the exam, which includes 10 “extra credit” points. Each problem is worth 10 points.

1. Let us begin with some questions about Linux scheduling:
 - a. How long is a *jiffy* in Linux scheduling

 - b. What is the arithmetical relationship between *goodness*, *priority* and *quantum* in the Linux scheduling algorithm?

 - c. Process A has a priority of 32 and a quantum of 16. Suppose it uses none of its quantum for each of the next two “epochs,” that is, each of the next two times that quantum values are re-computed. What will its final priority and quantum be?

 - d. Process B has priority 18. What is the largest its quantum could become? Explain.

2. What are the 4 conditions necessary for deadlocks to occur? Give a brief description of each.
 - a.

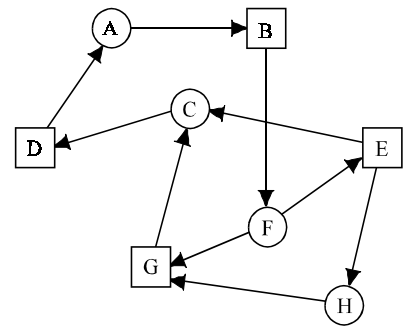
 - b.

 - c.

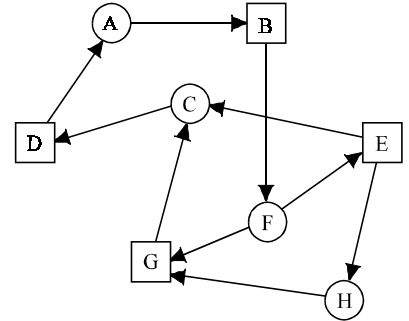
 - d.

3. One way of avoiding deadlocks is to linearly order the resources. Explain in one sentence what this means.

4. To the right is a resource / process diagram showing processes that have locked, or are waiting for, various resources. (A second copy is provided beneath it in case you want to do some scrap work.)
- Has deadlock occurred?



- Suppose an algorithm does depth-first-search from vertex A:
 - List an order in which vertices might be visited for the first time. That is, list the vertices in the order in which they become “gray”, as described in class.
 - Say at what point in the algorithm, following your list above, we would have discovered whether or not there was a deadlock



5. Suppose a bitmap is used to keep track of which blocks of memory are used and which are free. How large, in bytes, would such a bitmap be if a computer had 64MB of memory and used blocks of size 32 bytes?

6. Here is a linked list showing memory usage. Process P was just placed into the queue, and now process Q, of size 10, enters the scene. Into which hole would this process be placed if we use:



- best fit
- first fit
- next fit
- worst fit

7. One of the advantages of having many processes in memory at the same time is that we can increase CPU utilization.
- What is meant by CPU utilization?
 - How does having more processes in memory increase CPU utilization?
 - Suppose n processes are in memory, and each is ready to run (not blocked) at any given time independently with probability p . What will the CPU utilization be, on average?
8. Suppose a certain computer uses 24-bit addresses and virtual memory with pages of size 2048.
- How many pages comprise a process's virtual address space?
 - How many entries would be in that process's page table?
 - How large would the page frames be?
 - If each page table entry was 8 bytes, how much memory would be required to store the page tables for 20 processes?
 - If the computer has 256MB of RAM, then how many frames would it have?
 - How many frames would be needed to hold one process's page table?

9. We continue the example above: 24-bit addresses and virtual memory with pages of size 2048.

a. Here is a portion of the page table for a certain process: Translate the virtual address 8197 into a real address, using this table.

| Page | Frame |
|------|-------|
| 0 | 21 |
| 1 | 9 |
| 2 | 18 |
| 3 | 12 |
| 4 | 1 |
| 5 | 17 |
| 6 | 11 |
| ... | ... |

b. Show how that calculation would happen inside the TLB within the MMU, using bits.

10. The Not Recently Used page-replacement algorithm uses an “R” bit and an “M” bit.

a. What do “R” and “M” stand for?

b. When is the “R” bit set?

c. When is the “R” bit reset?

d. When is the “M” bit set?

e. When is the “M” bit reset?

11. The NRU algorithm divides pages into 4 classes: Class 0 through Class 3. When a decision is made to evict some page, a page from the lowest possible class is selected. What characterizes pages in each of these classes:

a. Class 0

b. Class 1

c. Class 2

d. Class 3

1. We saw 3 different implementations of the “Least Recently Used” algorithm. Fill in the blanks below to describe each of them:
 - a. In one implementation we kept a ___-bit counter which was incremented on every _____. Each page’s entry in the page table was stamped with this counter each time that page was referenced.

 - b. In another implementation we kept an $n \times n$ matrix, where n is the number of pages. This matrix started out as all 0s, and each time page i was referenced, we every bit in _____ i to ___ and then every bit in _____ i to ___. Then when we needed to evict a page, we selected the page whose _____, interpreted as a binary number, was _____.

2. Finish this description of the second chance algorithm: In the second-chance algorithm, we keep a queue of pages together with their “R” bits. When we have to decide which page to evict, we first consider the page at the front of the queue. If that page’s “R” bit is 0, then...

But if the “R” bit is 1, then...

3. What is the basic difference between the “second chance” and “clock” algorithms?

For all three problems below, assume you are in a directory with exactly one entry, "file1," (aside from the usual "." and "..").

1. What will be printed out when the following code is executed (output on the right):

```
void main(){
    cout << "Place 1\n";
    execlp("ls", "ls");
    cout << "Place 2\n";
    execlp("ls", "ls", "-i");
    cout << "Place 3\n";
}
```

| |
|--|
| |
| |
| |
| |

Explain your answer briefly:

2. What will be printed out when the following code is executed (output on the right):

```
void main(){
    if(fork() == 0);
        execlp("ls", "ls");
    else{
        fork();
        cout << "Greetings\n";
    }
}
```

| |
|--|
| |
| |
| |
| |