Name: _____

UB ID Number: | | | | | | | | |

Question:	1	2	3	4	5	6	7	8	Total
Points:	10	5	5	5	5	5	5	20	50
Score:									

CSE421 Alternate Midterm Exam

06 Mar 2013

This midterm exam consists of three types of questions:

1. **10 multiple choice** questions worth 1 point each. These are drawn directly from lecture slides and intended to be very easy.
2. **6 short answer** questions worth 5 points each. You can answer as many as you want, but we will give you credit for your best four answers for a total of up to 20 points. You should be able to answer the short answer questions in four or five sentences.
3. **2 long answer** questions worth 20 points each. **Please answer only one long answer question.** If you answer both, we will only grade one. Your answer to the long answer should span a page or two.

Please answer each question as **clearly** and **succinctly** as possible. Feel free to draw pictures or diagrams if they help you to do so. **No aids of any kind are permitted.**

The point value assigned to each question is intended to suggest how to allocate your time. So you should work on a 5 point question for roughly 5 minutes.

Please fill out your name and UB ID number above. Also write your UB ID number at the bottom of each page of the exam in case the pages become separated.

There are **three** scratch pages at the end of the exam if you need them. If you use them, please clearly indicate which question you are answering.

I have neither given nor received help on this exam.

Sign and Date: _____

Multiple Choice

1. (10 points) Answer all **ten** of the following questions. Each is worth **one** point.

   (a) Why wouldn't the diva sing?

   ◯ She needed vocal surgery. ◯ Her dressing room contained inappropriately-colored M&Ms. ◯ She was hungry. ◯ The lip-sync track was broken.

   (b) Which of the following operating system abstractions is not tied to a hardware component?

   ◯ Threads. ◯ Processes. ◯ Address spaces. ◯ Files.

   (c) What does the following system call do?

   ```
   exec("/bin/true");
   ```

   ◯ Fail. ◯ Load and execute "/bin/true". ◯ Return the exit code of process "/bin/true". ◯ Wait for process "/bin/true" to exit.

   (d) What does `exec()` do to the process file table?

   ◯ Copies it. ◯ Nothing. ◯ Opens `STDIN`, `STDOUT` and `STDERR`. ◯ Clears it.

   (e) Which is probably a privileged instruction?

   ◯ Load a word from memory. ◯ Change the interrupt mask. ◯ Add two registers and place the result in a third register. ◯ Rotate the value of a register right by the value of a second register.

   (f) True or false: the following code ensures that variable `foo` is protected? (Assume `foo_lock` exists and was properly initialized; do not make any other assumptions.)

   ```
   lock_acquire(foo_lock);
   // modify foo
   lock_release(foo_lock);
   ```

   ◯ True. ◯ False.

   (g) Which of the following is *not* an example of an operating system mechanism?

   ◯ Virtual to physical address translation. ◯ Using priorities to choose the next thread to run. ◯ An interrupt handler. ◯ Identifying interactive threads by observing their wait patterns.

   (h) Normal users are not aware of laptop

   ◯ responsiveness. ◯ screensavers. ◯ resource allocation. ◯ weight.

   (i) Who has to approve patches to mainline Linux?

   ◯ Linus Torvalds. ◯ God. ◯ Con Kolivas. ◯ Erasmus B. Dragon.

   (j) A virtual address might point to

   ◯ virtual memory. ◯ grapes. ◯ 0xdeadbeef. ◯ a register on the CPU.

 UB ID: | | | | | | | | |

Short Answer

Choose **4 of the following 6** questions to answer. You may choose to answer additional questions, in which case you will receive credit for your best four answers.

2. (5 points) We've presented synchronization primitives that use both active (or busy) and inactive (or blocking) waiting. First, explain the difference. Second, for each describe a scenario in which that form of waiting is more efficient and why.

UB ID: | | | | | | | | |

3. (5 points) Explain how to use the `fork()`, `pipe()`, and any other required system calls to establish the interprocess communication (IPC) between a parent and child process started by the following shell command:

   ```
   $ cat exam.tex | wc
   ```

   Drawing a diagram or set of diagrams may be helpful. You may also write pseudo-code as long as it is sufficiently clear. (It does not have to be able to compile.)

UB ID: | | | | | | | | |

4. (5 points) List and describe the three thread states we discussed in class. Next, describe four transitions between them including when and how they occur.

UB ID: | | | | | | | | |

Starting Virtual Address	Bound	Base Physical Address	Permissions
98	100	450	R,W,E
2	40	100	R
1040	1000	10,000	R,W
2200	500	1000	R,W,E

Table 1: **Segmentation Table.**

5. (5 points) Segmentation translation using table.

   Given the segment table above, indicate the result of the following five load, stores, and fetches (load and execute.) **Note: to make things easier on everyone the question uses base-10 arithmetic.**

   1. load 1200
   2. store 10
   3. load 2080
   4. fetch 143
   5. store 1050

UB ID: | | | | | | | | |

6. (5 points) Differentiate between time and space multiplexing, and explain how each concept applies to processor and memory sharing.

UB ID: | | | | | | | | |

7. (5 points) Your boss at Engitech is an radical egalitarian. You've written some clever code using two threads performing separate tasks that together accomplish your objectives of meeting the customer's every need, but he's concerned about fairness. He's worried that if one of your threads ends up doing too much work, the other thread will feel bad and become discouraged. So he's instructed you to implement a fairness policy so that your two threads must periodically meet at a fixed place in your code. If one thread arrives first, it should wait for the other, and this should work regardless of which thread arrives first. Once both threads meet together, both can continue with their separate tasks.

   Explain how to implement this egalitarian dream world using two semaphores.

   Statistics:

   - **5 out of 6** students answered this question.
   - **5** was the median score.
   - **4.4** was the average score.
   - **0.8** was the standard deviation of the scores.

UB ID: | | | | | | | | |

Long Answer

Choose 1 of the following 2 questions to answer. **Please do not answer both questions.** If you do, we will only read one.

If you need additional space, continue and clearly label your answer on the back of this or other exam sheets.

8. (20 points) Choose one of the following two questions to answer:

   1. **Starvation v. Utilization in Reader-Writer Locks.** Describe the tension between improving utilization and preventing starvation when implementing reader-writer locks. Outline two *conceptual approaches* (do not write code) to implementing reader-writer locks. For each, explain how it avoids starvation and the effect this has on the utilization of the resource the lock is protecting.

---

   2. **Deadlock.** List the four requirements for deadlock. In class, we solved the dining philosophers problem by eliminating one of the deadlock conditions. Describe two separate approaches to eliminating deadlock for this problem that eliminate other deadlock requirements, as well as any additional kernel support each solution requires over-and-above the primitives that exist or we have asked you to implement for OS/161.

 UB ID: | | | | | | | | |

Scratch. **Please indicate what question you are answering.**

UB ID: | | | | | | | | |

Scratch. **Please indicate what question you are answering.**

UB ID: | | | | | | | | |

Scratch. **Please indicate what question you are answering.**

UB ID: | | | | | | | | |