Name	**UBID**	**Seat**

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

CSE 421/521 Midterm Exam

23 Mar 2016

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.

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 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.

There are **5** 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) What is GWA's special talent?
   ○ French ○ C++ coding ○ Haircut detection ○ Haruspication

   (b) Which of the following is a requirement of a critical section?
   ○ progress ○ concurrency ○ mutual inclusion ○ idleness

   (c) Intra-process (within) communication is easier than interprocess (between) communication.
   ○ True ○ False

   (d) Which of the following requires communication with the operating system?
   ○ Switching between two threads ○ Inverting a matrix ○ Recursion ○ Creating a new process

   (e) Which of the following is *not* an example of an operating system mechanism?
   ○ A context switch ○ Using timer interrupts to stop a running thread ○ Maintaining the running, ready and waiting queues ○ Choosing a thread to run at random

   (f) The Rotating Staircase Deadline Scheduler is most similar to which other scheduling algorithm?
   ○ Lottery scheduling ○ Multi-level feedback queues ○ Round-robin ○ Random

   (g) What would probably be stored in a page table entry?
   ○ the physical memory address ○ the virtual memory address ○ the process ID ○ the file name

   (h) Address translation allows the kernel to implement what abstraction?
   ○ Files ○ Threads ○ Processes ○ Address spaces

   (i) Con Kolivas was particularly interested in improving what aspect of Linux scheduling?
   ○ Overhead ○ Throughput ○ Interactive performance ○ Awesomeness

   (j) Which is probably *not* a privileged operation?
   ○ Changing the interrupt mask ○ Loading an entry into the TLB ○ Modifying the exception handlers ○ Adding two registers and placing the result in a third register

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 (3 points). Second, for each describe a scenario in which that form of waiting is more efficient and why (1 point each).

UB ID: □ □ □ □ □ □ □ □

3. (5 points) Consider a 32-bit system with 4K pages uses multi-level page tables as described in class:

   1. 10 bits for the first-level index,
   2. 10 bits for the second-level index,
   3. and a 12-bit offset.

   (Assume that page table entries (PTEs) are 32-bits and so can be stored directly in the second-level table.)

   If a process has 10 contiguous code pages (including global variables), 4 contiguous heap pages, and a single 1 page stack. What is the minimum number of pages required for the process's page table (1 point)? What is the maximum (2 points)? In both cases, briefly explain your answer (1 point each).

UB ID: □□□□□□□□

Index	SLT
0x0	0x800
0x12	0x10400
0x34	0x32000
0x45	0x10000
0x7f	0x100000

(a) **TLT**

VPN	PPN
0x0	0x3200
0x23	0x1040
0xfd	0x320
0xff	0x1

(b) **SLT at 0x800**

Index	PPN
0x0	0x1048
0xcf	0x7888
0xdf	0x9000

(c) **SLT at 0x100000**

Index	PPN
0x0	0x1048
0x10	0x7888
0x11	0x7888
0x17	0x7888
0xfe	0x9000

(d) **SLT at 0x10040**

Table 1: **Process Page Tables**

4. (5 points) The HappyShawn architecture has byte-addressable memory with 64K virtual pages. The HappyShawnOS operating system uses two-level pages tables to support 2GB virtual address spaces by dividing the virtual address into a 64K offset and two 8-bit first- and second-level page table indices.

   First, given the virtual address 0x7f001000, identify the virtual page number, the offset, and the first and second level page table indices (1 point).

   Second, given the top-level page table (TLT) and second-level page tables (SLT) above for the currently-running process provided above, indicate the result of the following four virtual to physical page translations (1 point each).

   1. NULL
   2. 0x7fffabcd
   3. 0x22ff0020
   4. 0xff000000

5. (5 points) Identify one separation between OS mechanism and policy that we have discussed this semester (1 point). Describe the mechanism (2 points) and one policy (2 points).

UB ID: □ □ □ □ □ □ □ □

6. (5 points) Identify three system calls that allocate new virtual addresses (i.e., allow the process to use them) and provide a brief description for each.

7. (5 points) Given a simple MLFQ approach that rewards any thread that sleeps or yields before its quantum expires, first describe a way that a computationally-intensive thread can take advantage of a weakness in this approach to remain in one of the top queues (2 points). Second, propose a modification to MLFQ that addresses this problem (3 points).

Long Answer

Choose 1 of the following questions to answer. **Do not answer both questions.** If you do, we will only read the shorter one. If you need additional space, continue and clearly label your answer on other exam sheets.

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

    1. **A New Synchronization Primitive.** We have introduced semaphores, spin and sleep locks, condition variables, and reader-writer locks. However, many other useful synchronization primitives exist. First, describe one additional synchronization primitive (4 points). Provide a complete interface for it in C pseudo-code (2 points) and describe how to implement it (4 points).

       Second, provide two different use cases for your new synchronization primitive (5 points each). Feel free to use pseudo-code as well as English to describe why your new primitive is useful.

    ---

    2. **Scheduling Core Count and Frequency.** We framed the scheduling problem as a question of what threads to run on which cores. However, the modern scheduling problem is much more complicated—particularly on energy-constrained battery-powered devices like laptops and smartphones.

       These devices introduce many new wrinkles. First, cores can be powered off to save energy, since even a completely idle core consumes some amount of power. Second, the speed of each core can be throttled up and down at runtime. Generally, when cores run faster they are *less efficient*—consuming more energy to perform the same set of instructions more quickly. (Third, both these transitions take a non-trivial amount of time, but you can ignore that fact for now.)

       First, describe how these capabilities complicate the original scheduling problem. Identify two new decisions that need to be made (2 points each) and two new tradeoffs that result (3 points each).

       Second, describe a scheduling algorithm for this type of common multicore system (10 points). It can be a variant of one of the algorithms we described in class, or something completely new. However, it should do something intelligent to manage the new capabilities described above. (Put another way, random earns you no credit.)

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.**

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

UB ID: □ □ □ □ □ □ □ □

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

BLANK