CMPS 111 – Spring 2013
Prof. Scott A. Brandt
Midterm Examination
May 6, 2013

Name:____ ID:________________

This is a closed note, closed book exam. There are 23 multiple choice questions, 6 short answer questions. Plan your time accordingly.

Part I: Multiple Choice [2 points each].
Write the letter of the correct answer to the left of the question. If none of the answers are correct, write "None." If more than one answer is correct, write the letter of all correct answers.

1. The two main purposes of an operating system are to manage:
   a. Hardware utilization and overhead minimization
   b. Resource utilization and deadlock avoidance
   c. Resource sharing and hardware abstraction

2. System calls:
   a. Protect important kernel data structures from user code
   b. Increase the performance of the operating system
   c. Provide a rich, flexible API for software developers to use

3. Which operation requires a system call:
   a. Accessing a variable
   b. Accessing a file
   c. Calling a function

4. In the process state transition diagram, the transition from the Running to the Ready state indicates that the running process has:
   a. Blocked on an I/O operation
   b. Finished waiting on an I/O operation
   c. Stopped running so another process can execute

5. The three types of information describing a process in the process table are:
   a. Process state, address space information, and file information
   b. Register contents, program counter, and stack pointer
   c. Priority, context switch overhead, and burstiness

6. Two threads in the same process share the same:
   a. Program counter
   b. Address space
   c. Stack

7. Semaphores provide what two operations:
   a. Mutual exclusion and synchronization
   b. Mutual synchronization and utilization
   c. Locks and condition variables

8. Which is true:
   a. Round-robin scheduling is non-preemptive
   b. First-come-first-serve scheduling produces the minimum average response time
   c. Lottery scheduling always runs a randomly selected process

9. Starvation is when:
   a. A set of processes is waiting for an event that only another process in the set can cause
   b. A process could run but doesn't get to run because other processes keep getting to run ahead of it
   c. A set of processes are prevented from simultaneously accessing a shared data structure

10. Preemption is when:
   a. A process terminates because of an error
   b. A process blocks waiting for an I/O operation
   c. A process blocks to another process can run

11. Which is false:
   a. The memory usage of bitmapped free memory management depends on the amount of memory in the system
   b. Linked list free memory management requires extra hardware
   c. Linked list free memory management makes it efficient to merge adjacent free regions

12. The main advantage of multi-level page tables is that they:
   a. Use less memory
   b. Speed up page table references
   c. Reduce the complexity of the paging system

13. TLBs are used to reduce the amount of time it takes to:
   a. Read data from disk
   b. Perform an address translation
   c. Load a page table into memory

14. The best practical page replacement algorithm is:
   a. Optimal
   b. Least Recently Used
   c. Not Recently Used

15. Which suffers the least internal fragmentation

a. Large pages
b. Medium pages
c. Small pages

16. Suppose you have a memory access stream that references pages 0, 1, and 2 over and over again, in that order, and you have only two pages of physical memory for the process. Which page replacement algorithm will generate the fewest page faults:
    a. Least Recently Used
    b. Most Recently Used
    c. Random

17. A shell:
    a. Is a user program
    b. Is part of the kernal
    c. Is a system call

18. Lottery scheduling randomly choose a ready process to run every time quantum.
    a. That can't possibly work
    b. That could starve an unlucky process for a while
    c. That would be very hard to implement

19. File systems:
    a. Store data long-term
    b. Organize large amounts of data
    c. Serve as backing store for memory
    d. Protect data from unauthorized access

20. Disk space management: Bitmaps vs. free block lists
    a. Bitmaps are faster
    b. Bitmaps are easier to implement
    c. Bitmaps are smaller

21. Unix directory entries do not contain
    a. Filename
    b. Inode numbers
    c. Permission information

22. Unix inodes
    a. Efficiently use memory
    b. Always cause two extra disk accesses per file
    c. Can be shared by two different files

23. File Systems
    a. Are only accessed when data or programs are first loaded into memory
    b. Aren't necessary when a computer has flash-based storage
    c. Can significantly affect the performance of a computer

Part II: Short answers [9 points each]

1. Explain what happens when a program makes a system call.

```
1. The program usually calls a library function that does the actual
system call.
2. The program (or the library function) executes a trap instruction.
3. The CPU enters kernel mode
4. The kernel saves the program state
5. The kernel figures out which system call was requested and calls
the appropriate function.
6. The request is processed (which may involve permission checks,
blocking the process, etc.)
7. The system decides who to run next, possibly the process that made
the system call.
8. The state of the chosen process is loaded.
9. The system switches back to user mode and begins executing the
selected process.
```

2. Suppose you have monitors and condition variables. Show how you could implement
a (binary) semaphore. In other words, implement two functions in a monitor that
behave like the two semaphore operations get() and put().

```
monitor {
  int sem_val = 1;
  condition sem_cond;

  get() {

    while(sem_val == 0)
         sem_cond.wait();
    sem_val = 0;

  }

  put() {
    if(sem_val == 0) {
         sem_val = 1;
         sem_cond.signal();
    }
  }
```

3. Given four (physical) page frames and the memory reference string 04551440223, show the page faults that will occur with each page replacement algorithm by circling the memory references that cause a page fault. Show your work.

a) Least Recently Used

	0	4	5	5	1	4	4	0	2	2	3
0	0	4	5	5	1	4	4	0	2	2	3
1		0	4	4	5	1	1	4	0	0	2
2			0	0	4	5	5	1	4	4	0
3					0	0	0	5	1	1	4

b) First-In-First-Out

	0	4	5	5	1	4	4	0	2	2	3
0	0	4	5	5	1	1	1	1	2	2	3
1		0	4	4	5	5	5	5	1	1	2
2			0	0	4	4	4	4	5	5	1
3					0	0	0	0	4	4	5

c) Most Recently Used

	0	4	5	5	1	4	4	0	2	2	3
0	0	4	5	5	1	4	4	0	2	2	3
1		0	4	4	5	1	1	4	4	4	4
2			0	0	4	5	5	1	1	1	1
3					0	0	0	5	5	5	5

4. We have discussed several situations in which hardware features have been developed to support operating system needs. Give two examples of this and describe how they are used.

Interrupt hardware -- allows devices to notify the CPU when they need attention, so that the CPU does not have to poll. Done by sending a signal over specific wires on the bus.

Memory Management Hardware -- allows the kernel to protect memory from user processes and create process-specific memory mappings. Works by intercepting and translating memory addresses.

TLB -- similar to the MMU, but smaller and faster. Caches memory address translations.

DMA -- allows devices to directly access memory so they can access and copy data directly rather than requiring the CPU to do it.

Timer -- Hardware clock keeps track of time. Timers allow the system and user processes to receive a signal at a specified time. Works via a hardware oscillator and a countdown register.

5. Given the following processes:

Process	Arrival Time	Run Time	Priority (lower number = higher priority)
0	0	10	3
1	3	4	2
2	4	5	1
3	7	3	0

a) Show the schedule that would occur with non-preemptive Shortest Job First scheduling

0, 3, 1, 2

b) Show the schedule that would occur with Round Robin (time quantum = 2)

0, 0, 1, 2, 0, 1, 3, 2, 0, 3, 2, 0 OR 0, 0, 1, 2, 0, 1, 2, 3, 0, 2, 3, 0

c) Show the schedule that would occur with Preemptive Priority scheduling

0 (time 0 to 3), 1 (time 3 to 4), 2 (time 4 to 7), 3 (time 7 to 10), 2 (time 10 to 12), 1 (time 12 to 15), 0 (time 15 to 22)

6. Describe the sequence of disk accesses (reads and writes) that will occur if I run a process that opens “/usr/sbrandt/foo”, writes two blocks to it, then closes it. Assume the process is running on a unix system with the kind of inodes that we have discussed in class.

1. Read root inode
2. Read root directory blocks to find "usr"
3. Read "usr" inode
4. Read "usr" blocks to find "sbrandt"
5. Read "sbrandt" inode
6. Read "sbrandt" blocks to find "foo"
7. Read "foo" inode
8. Write "foo" blocks
9. Update "foo" inode to reflect new size, access time, modification time

Each directory and file access above may involve reading multiple inode indirect blocks or data blocks