Name:

Operating Systems
Fall 2013
Final Exam
December 11, 2013

Closed books, notes, cell phones, PDAs, iPods, laptops, etc. No headphones, please. You may use a simple calculator.

You have 120 minutes to solve 9 problems, each worth 10 points. You get 10 points for writing your name on the top of this page. 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.

Partial credit will be offered only for meaningful progress towards solving the problems.

Please read and sign below if you agree with the following statement:

In recognition of and in the spirit of the academic code of honor, I certify that I will neither give nor receive unpermitted aid on this exam.

Signature:_____

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
Total	

1. **Short attention span:** Answer the following questions on general concepts in operating systems.

    a. List and briefly explain three main responsibilities of a typical OS.

    b. What is multithreading?

    c. What is the kernel of an OS?

    d. What is usually the critical performance requirement in an interactive operating systems?

2. Below is the listing of a short assembly language program for a computer with 512-byte pages. The program is located at address 1020, and its stack pointer is at 8192 (the stack grows toward 0). Give the page reference string generated by this program. Each instruction occupies 4 bytes (1 word), and both instruction and data references count in the reference string.

```
Load word 6144 into register 0

Push register 0 onto the stack

Call a procedure at 5120, stacking the return address

Subtract the immediate constant 16 from the stack pointer

Compare the actual parameter to the immediate constant 4

Jump if equal to 5152
```

3. Consider the following program.

```
#define N 128
int A[N, N], B[N, N], C[N, N];
int i, j;

for (j = 0; j < N; j ++)
     for (i = 0; i < N; i++)
          C[i, j] = A[i, j] + B[i, j];
```

Assume that the program is running on a system using demand paging and the page size is 4KB. Each integer is 4 bytes long. It is clear that each array requires 16-page space. As an example, the following elements from matrix A will fit in one page: A[0,0]-A[0,127], A[1,0]-A[1,127], A[2,0]-A[2,127], … to A[7,0]-A[7,127]. A similar storage pattern can be derived for the rest of array A and for arrays B and C.

Assume that the system allocates a 4-page working set for this process. One of the pages will be used by the program and three pages can be used for the data. Also, assume that two index registers are assigned for i and j (so, no memory accesses are needed for references to these two variables).

a. Discuss how frequently page faults would occur.
b. Can you modify the program to minimize the page fault frequency?
c. What will be the frequency of page faults after your modification?

4. What file organization on secondary storage would you choose in order to maximize efficiency in terms of speed of access, use of storage space and ease of updating (adding/deleting/modifying) when the data are:
    a. Updated infrequently and accessed frequently in random order?
    b. Updated frequently and accessed in its entirety relatively frequently?
    c. Updated frequently and accessed frequently in random order?

5. In UNIX System V, the size of a block is 1KB, and each block can hold a total of 256 block addresses. Assume that there are 12 direct block pointers and a singly, doubly and triply indirect pointer in each inode.
    a. Using the inode scheme, what is the maximum size of a file?
    b. Assuming no information other than the file inode is already in main memory, how many disk accesses are required to access the byte in position 13,423,956? (Hint: If you don't have a calculator with you, this information might save you time: $2^{23} < 13{,}423{,}956 < 2^{24}$)

6. Two computer scientists, Carolyn and Elinor, are having a discussion about i-nodes. Carolyn maintains that memories have gotten so large and so cheap that when a file is opened, it is simpler and faster just to fetch a new copy of the i-node into the i-node table held in memory, rather than search the entire table to see if it is already there. Elinor disagrees. Who is right and why? (Support your answer clearly.)

7. Sketch rough graphs ("back of napkin") illustrating the relationships among the following quantities. Note any important assumptions underlying your analysis.
    a. Disk access latency as a function of rotation speed.
    b. CPU utilization as a function of the degree of multiprogramming. Consider a uni-processor system.
    c. Memory fragmentation as a function of page size. Is that internal or external?

8. Storage management:
    a. A computer has 512 GB of secondary storage and a block size of 512 bytes. How many KB are needed if a bitmap is used to keep track of free disk?

    b. Should the bitmap be kept in memory or on the disk? **Why**?

    c. For what performance objectives would you choose each of the following disk scheduling algorithms:

        FIFO
        SSTF
        CSCAN

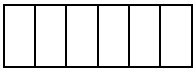
9. Two threads are sharing a buffer as shown below. The buffer will be accessed by the threads from left to right in a **circular** fashion. Thread 2 reads characters from a file and places them in the buffer. Thread 1 reads characters from the buffer and prints them to the screen. The buffer is initialized with "blank" characters in each position. To protect the buffer from overwriting by thread 2 or overreading by thread 1, a confused programmer creates two semaphores, A and B and initializes A to 2 and B to 10 and places them in the threads as shown below. The content of the file is as follows:

abcdefghijklmnop

Thread 2 is assigned the CPU first and runs until it is blocked by the semaphore. Then Thread 1 runs until it is blocked by the semaphore.

Thread 1	**Thread 2**
Repeat down(A) read item from buffer print item to screen up(B) until false	Repeat down(B) read character from file place item in buffer up(A) until false

Describe the content of the buffer when Thread 2 was blocked and explain why the buffer contains that.



Describe what Thread 1 printed to the screen and explain why it printed that.