Name: _____

CS354 - Operating Systems
Spring 2006
Midterm Exam #2 – Take Home
April 13-20, 2006

Rules:

1. All work that you do on this take home exam must be completely and entirely your own work.

2. You may neither discuss the content of this exam nor ask questions regarding this exam of anyone other than your professor until you have turned it in on Thursday April 20.

3. You may neither discuss the content of this course nor ask questions about the content of this course of anyone other than your professor until you have turned in this exam on Thursday April 20.

4. The single exception to rule 3 is that you may discuss your presentations with your partner. However, be careful to steer well clear of anything that could be construed as violating rules 1 through 3.

Any violation of these rules constitutes cheating. All cases of cheating will be submitted to the College disciplinary system.

The point value for each question is noted beside the question.
The total number of points for the exam is 100 points.

[10 pts]
1. Consider the two threads and the shared variable x shown below. If these threads are executed concurrently, what are the possible resulting values for x? Clearly explain how each value arises. Assume that the machine executing the threads has a Load-and-Store architecture.

Shared Variables:

```
int x = 3;
```

Thread A	Thread B
`1. Section A1`	`1. Section B1`
`2. x = x + 2;`	`2. x = x * x;`
`3. Section A2`	`3. Section B2`

[10 pts]
2. Consider the three threads and the shared variables shown below. Show how semaphores can be used to ensure proper execution of these threads. For full credit, your solution must allow for maximum concurrent execution while ensuring proper execution.

Shared Variables:

```
int x = 2;
int y = 3;
int z = 5;
```

Thread A	Thread B	Thread C
`1. Section A1`	`1. Section B1`	`1. Section C1`
`2. x = x - 1;`	`2. x = x * x;`	`2. z = x + y;`
`3. y = y + z;`	`3. x = x + y;`	`3. Section C2`
`4. Section A2`	`4. Section B2`	

[15 pts]
3. Consider the two threads shown below. Show how semaphores can be used to impose the following synchronization constraints:

i. Section A1 must execute before Section B1.
ii. Sections A2 and B2 may execute concurrently, but both must complete before either Section A3 or B3 executes.

Thread A	Thread B
`1. Section A1`	`1. Section B1`
`2. Section A2`	`2. Section B2`
`3. Section A3`	`3. Section B3`

[15 pts]
4. Consider the two threads shown below. Show how semaphores and shared variables can be used to impose the following synchronization constraint: Section A2 must execute 5 times before each execution of section B2.

Thread A	Thread B
1. Section A1	1. Section B1
2. while (true) {	2. while (true) {
3. Section A2	3. Section B2
4. }	4. }

[10 pts.]
5. Consider a Unix-like system that has inodes with 10 direct entries, one single indirect entry, one double indirect entry and one triple indirect entry. Assume that the block size on the system is 1kb and 32 bits are used for disk block addresses. What is the size in MB of the largest file that could be created on this system? To receive full or partial credit you must show your work.

[10 pts.]
6. Consider the following absolute path: /users/braught/cs354s06/exam2.doc
How many disk blocks would need to be read in order to retrieve the first block of data from the file exam2.doc? You may assume that each directory file requires a single block and that the inode for the root directory is already cached by the OS. Explain your answer carefully in terms of the Unix file system discussed in class. Feel free to draw a diagram if it will help.

[15 pts.]
7. Compare and contrast the performance of contiguous, FAT and multilevel indexed allocation (e.g. inode). Be sure to include comparisons based on performance on sequential and random file access to files of different sizes and also in terms of efficiency of use of storage space due to internal and/or external fragmentation.

[15 pts.]
8. Describe in detail the changes that the OS would be required to make to both the in-memory and the on-disk file system data structures when a user:

i. Deletes a file.
ii. Opens an existing file for random access, reads a record from the middle of the file, changes the record and writes it back to disk, closes the file.