CS241 Exam Practice (System Programming > CS241) == 1

1> Final exam date and format.

2> Scoring appeals/attendance points in by Wed 11:59pm

3> curl http://illinois.edu and press return. What happens?

4> Traversing inodes and directories.

A program executes
```
FILE *f = fopen("/etc/hosts","r");
```

The o/s first read the directory listing at `/`

Next it reads the directory listing at `/etc`

Then get the first byte of the file `/etc/hosts`?

How many disk blocks in total are read? Assume the only mounted directory is at the root directory. Assume all inode data is cached but directory listings are not cached.

5>

a. Identify the missing pieces to complete Peterson's N=2 solution to the Critical Section Problem.

```
raise my flag

WHAT IS MISSING HERE?




// Do Critical Section stuff
lower my flag
```

b. Identify the missing pieces to complete Dekker's N=2 solution to the Critical Section Problem.

```
raise my flag
while your flag is raised :
  if it's your turn to win :

    WHAT IS MISSING HERE?




// Do Critical Section stuff
set your turn to win

WHAT IS MISSING HERE?
```

6> Producer Consumer

Implement a fixed capacity, multithreaded producer consumer. Do not allow more than 100 items to be in the queue (or call `remove_raw` on an empty queue). Use two counting semaphores and a mutex. Why is the mutex necessary? State the initial values of the semaphores.

```
void add(void* value) { // Blocks if 100 items are in the
            // queue. Call add_raw(void*) to enqueue.
```

```
void* remove() { // Blocks while the queue is empty.
              // Call void* remove_raw() to dequeue.
```

7> Working with file metadata

What is the value of result.st_mode and result.st_size if the file "abc" does not exist? Fix and complete the code to only print output when "abc" is a directory.

```
    struct stat result;
    stat("abc", &result);

if(  S_ISDIR(_____________)        )
                                  puts("Is a dir!");
```

8. Fix the following code. Hint there's at least two errors.

```
struct stat result;


stat(path, &result);


if( S_ISLNK( result ) ) {
   printf("%s is a symbolic link", path);
}
```

9. If you keep calling `opendir` and never call `closedir` what will you run out of?