CS 3013 Operating Systems WPI, A Term 2006
Craig E. Wills Final Exam (100 pts)
Given: Thursday, October 12, 2006

NAME:____________________________________

This is a closed book (and notes) examination. Answer all questions on the exam itself. Take the number of points assigned to each problem and the amount of space provided for your answer as a measure of the length and difficulty of the expected solution. The exam totals 100 points.

SCORE:____________________________________

1. (8 points) What happens when a process receives a software interrupt? What software interrupt is used as part of routines like *sleep()* and *usleep()*, which was employed in Project 4?

2. (15 points) The generic organization of a device driver contains three parts: code in the upper half, code in the lower half, and shared data between.

   (a) When is the upper-half code of the device driver called?

   (b) When is the lower-half code of the device driver called?

   (c) What kind of data is stored in the shared buffer?

   (d) When does the upper-half code block? Give a specific example or indicate why this code cannot block.

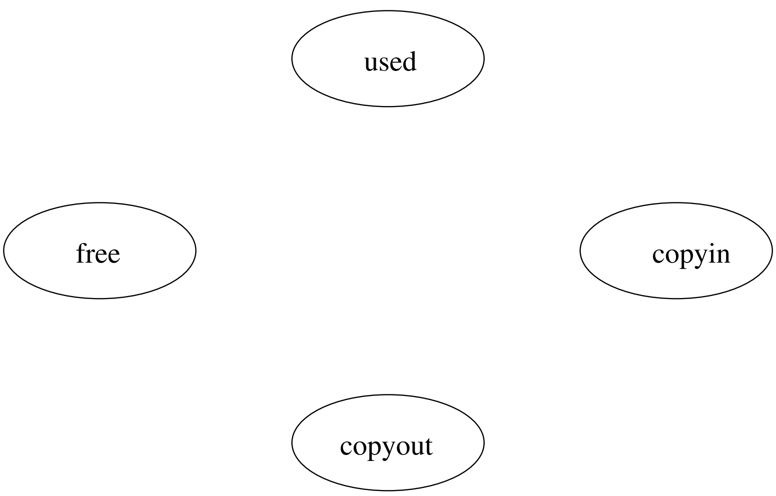
   (e) When does the lower-half code block? Give a specific example or indicate why this code cannot block.

3. (12 points) Project 4.

   (a) What synchronization primitives can be used to correctly control access to the request queue accessed by the generator and worker threads. Be sure to indicate how many primitives and the initial counts of any semaphores you use.

   (b) The project you implemented used only a single thread to generate requests. Indicate how your answer to the previous portion of this question would change if *two* threads were used to generate requests.

4. (12 points) In a paging system a *frame* in memory can have four possible states: 1) free, available for use; 2) used, being used to store a page; 3) copyin, currently a page is being copied into the frame from disk; and 4) copyout, currently the page in the frame is being copied out to disk. These states are shown in the following diagram. Show the possible transitions between these states and the reason for each transition.



5. (15 points) Consider the following piece of code that accesses an allocated array where the lines have been added for reference.

```
1 int i;
2 main()
3 {
4     char *buf;
5     int cnt;
6     buf = new char[128];
7     for (i = 0; i < 128; i++)
8         buf[i] = 'x';
9 }
```

For each of the following situations indicate whether a *major page fault*, a *minor page fault* or *no page fault* occurs assuming that all pages are brought into memory via demand paging.

(a) When the compiled instructions for this code are accessed.

(b) When the pointer variable `buf` is set on line 6.

(c) When the variable `i` is initialized and used on line 7.

(d) When the array referenced by `buf` is accessed on line 8.

(e) Not part of this code, but when the contents of a memory mapped file are accessed for the first time as was done in Project 5.

6. (9 points) Consider a disk with 40 cylinders. Assume that at a given instance the read/write head is sitting at cylinder 10 after having just been at cylinder 3. Assume the following requests to or from the given cylinders have been made. The requests are listed in the order in which they arrived.

17, 35, 13, 9, and 22

Assume that no new requests arrive while we service this set. Give the order in which the requests are handled under each of the following disk head scheduling policies:

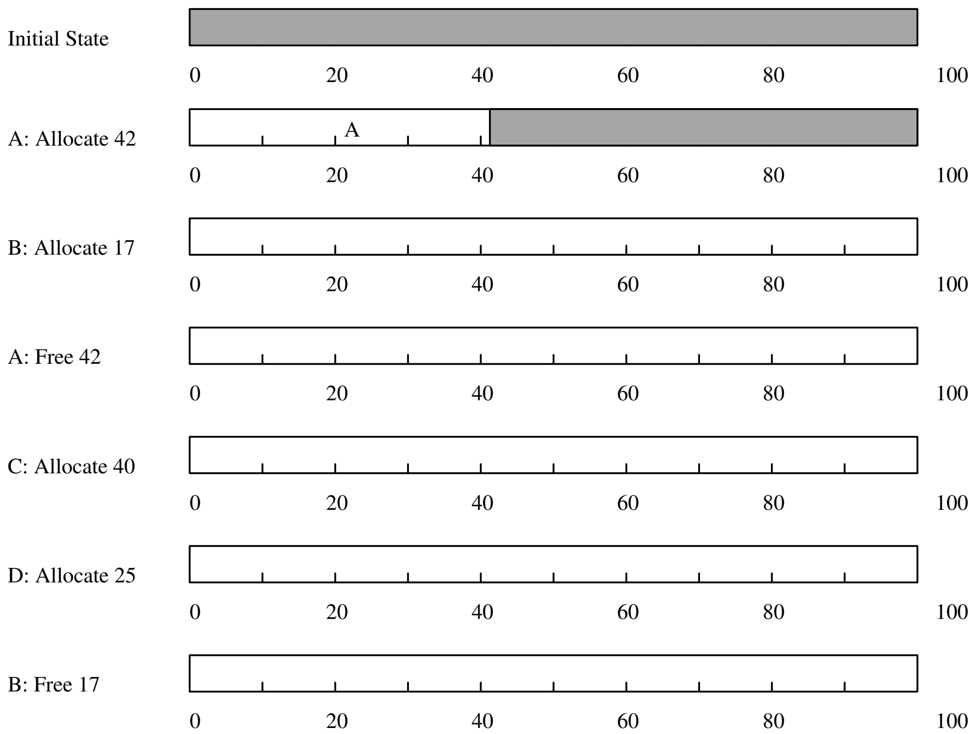
(a) First Come First Served (FCFS)

(b) Shortest Seek-latency First (SSF)

(c) Elevator

7. (12 points) Assume you begin with a block of 100 bytes in which you are using a boundary tag memory management scheme with a best fit policy for locating free space (ignore any overhead bytes for this question).

(a) Show the allocation of memory at each step given the following sequence of requests for memory. The initial state and first request are done. The shaded areas indicate memory not in use.



(b) Will this scheme lead to external or internal fragmentation? Why?

8. (7 points) What is thrashing in the context of virtual memory management?

9. (10 points) A goal of a page replacement algorithm is to keep frequently used pages in memory. One policy that implements this goal is the *Least Frequently Used* (LFU) policy, which always replaces the page in memory that has the fewest *number* of references. What are the advantages and disadvantages of this policy?

   The NRU and Clock page replacement policies use a reference bit to approximate the Least Recently Used policy. Is it possible to approximate the LFU policy in a similar manner? If not then describe why not. If so then describe how, including use of the reference bit if appropriate.