1. (30%, 3% each) Answer each of the following questions concisely but precisely.

- What is a system call?

- Why is the busy-waiting in the following implementation of semaphore wait() on multiprocessors not considered a problem?

```c
void wait(semaphore s)
{
    disable interrupts;
    while (ldl(s->lock) != 0 || !stc(s->lock, 1));
    if (s->count > 0) {
        s->count--; s->lock = 0; enable interrupts;
        return;
    }
    add(s->q, current_thread); s->lock = 0;
    enable interrupts; sleep();
}
```

- Why do we still disable interrupts to prevent context switch in addition to using the lock with TAS in the following implementation of semaphore wait() for multiprocessors?

```c
void wait(semaphore s)
{
    disable interrupts;
    while (TAS(s->lock, 1) == 1);
    if (s->count > 0) {
        s->count--; s->lock = 0; enable interrupts;
        return;
    }
    add(s->q, current_thread); s->lock = 0;
    enable interrupts; sleep();
}
```

- Consider CPU-only jobs. Assume all jobs have already arrived. Can Round-Robin ever be the worst possible preemptive CPU scheduling algorithm in terms of average turnaround time? If so, under what circumstances? If not, explain why not.
• In a generic storage allocation problem, what are the fundamental causes for external fragmentation?

• What are the tradeoffs in choosing the page size in a paged main memory management system?

• What is the challenge/downside in using an inverted page table?

• In a paging system, when malloc(16385) invoked by user process $P$ successfully returns, how many physical pages have been allocated to process $P$, assuming a page size of 4096 bytes? Are they consecutive in the physical memory?

• Why are TLBs often implemented as fully set-associative, in contrast to an L1 cache that is generally two-way set-associative or direct-mapped?

• What is a page fault?
2. (True-or-False - 20%, 2% each) For each of the statements below, indicate in one sentence whether or not the statement is true or false, and why.

- The use of a log stored in fast, non-volatile RAM increases the performance of a file system.

- When a file is being accessed by a process, i.e., read or written, the current position within the file where it is being accessed is stored in the in-memory copy of its inode.

- Virtual memory are virtual address are synonymous.

- A multi-level indexed file descriptor permits faster random access than a file descriptor with a single level of index.

- In the Unix File System, the name of a (non-directory) file is stored in its inode.

- In the Unix File System, the name of a directory is stored in its inode.

- In the UNIX file system, after file system initialization, there is a fixed upper limit on how large files can be.

- In the Unix File System, the operation "ls -l" is potentially more expensive and hence slower than "ls".
- In File Systems, the buffer cache is implemented purely in software, unlike demand paging, which is implemented jointly in software and with hardware support (i.e. MMU).

- In disk scheduling, the SSTF (shortest seek time first) scheduling algorithm tends to favor middle cylinders over the innermost/outmost cylinders.
3. (CPU Scheduling - 15%) Consider a preemptive priority scheduling algorithm based on dynamically changing priorities. Larger priority numbers imply higher priority. When a process is waiting for the CPU (in the ready queue, but not running), its priority changes at a rate \( \alpha \); when it is running, its priority changes at a rate \( \beta \). All processes are given a priority of 0 when they enter the ready queue. The parameters \( \alpha \) and \( \beta \) can be set to give many different scheduling algorithms.

(a) (7%) What is the algorithm that results from \( \beta > \alpha > 0 \)?

(b) (8%) What is the algorithm that results from \( \alpha < \beta < 0 \)?
4. (Directories and Links - 15%)

(a) (6%) Given an absolute pathname, "/homes/user1/QE.pdf", what are the disk reads performed by UFS in order to read the first data block of the file? Assume no data or metadata blocks are cached, and that each directory file contains only 1 data block.

(b) (4%) If "/hardlink" is a hard link that points to "/home/user1/QE.pdf", using "/hardlink", what are the disk reads performed by UFS in order to read the first data block of the file? Assume no data or metadata blocks are cached, and that each directory file contains only 1 data block.

(c) (5%) If "/softlink" is a soft link that points to "/home/user1/QE.pdf", using "/softlink", what are the disk reads performed by UFS in order to read the first data block of the file? Assume (1) the inode for "/" is cached in memory; (2) no other data or inode blocks are cached, but once loaded, they are cached; (3) each directory file has only one data block, which contains information for all the files in it; (4) the file name pointed to by a softlink is stored in its inode.
5. (Redundant Arrays of Inexpensive Disks - 20%)
   
   (a) (4%) Name two possible advantages of RAID over single disks.

   (b) Consider a RAID Level 5 that performs block-level striping over 8 disks for data and uses 1 (additional) disk for parity blocks.
   
   i. (8%) How many block reads and writes happen in writing a large file with 8 blocks (assuming they are perfectly striped over the 8 disks for data). Explain which reads/writes can happen in parallel.

   ii. (8%) How many block reads and writes happen in writing a small file with 1 single block. Explain which reads/writes can happen in parallel.