1. (20 pts, 4 pts each) For each of the statements below, indicate whether the statement is true or false. Explanations may allow partial credit for wrong answers.

- A system call is triggered by hardware.

- The printf statement will get executed after "/bin/ls" is finished.

```c
void main() {
    int pid = fork();
    if (pid == 0) {
        execlp("/bin/ls", "ls", NULL);
        printf("The child process at after exec !");
    } else {
        wait(NULL);
        exit(0);
    }
}
```

- Assume synchronization primitives are correctly supported by the OS. If a multithreaded user program using the synchronization primitives is known to execute correctly on a time-shared (i.e. preemptively scheduled) uniprocessor, then it will execute correctly on a multiprocessor.

- A binary semaphore can only be used by two processes/threads for synchronization between them.

- Condition variables alone can be used to solve synchronization problems, for example, the producer-consumer problem.
2. (CPU Scheduling - 10 pts) On planet Mars, the scheduler knows how many instructions are left till completion. Also assume that the algorithm must run a thread if there is one ready, and that a context switch has zero overhead.

Under these assumptions, what is the worst possible CPU scheduling algorithm (by worst we mean the one that maximizes the average turn around time)?
3. (Deadlock - 20 pts)

(a) (4 pts) List the four conditions required for deadlock to occur. Briefly explain each.

(b) (5 pts)
Consider the three processes below, (proc1 through proc3,) each of which competes for six shared resources, (A through F):

```c
proc1()
{
    while (1)
    {
        lock(&D);
        lock(&E);
        lock(&B);
        // Use D, E, and B
        unlock(&D);
        unlock(&E);
        unlock(&B);
    }
}
```

```c
proc2()
{
    while (1)
    {
        lock(&C);
        lock(&F);
        lock(&D);
        // Use C, F, and D
        unlock(&C);
        unlock(&F);
        unlock(&D);
    }
}
```

```c
proc3()
{
    while (1)
    {
        lock(&A);
        lock(&B);
        lock(&C);
        // Use A, B, and C
        unlock(&A);
        unlock(&B);
        unlock(&C);
    }
}
```

Could the three processes enter into deadlock? If so draw the resource allocation graph with the cycle that represents this deadlock. (Remember: Processes are circles, resources rectangles.)

Write in Exam Book Only
(c) (5 pts) (Deadlock prevention) Can you think of a way of reordering the statements in the above processes to prevent deadlock from happening? Note that (1) the resource usage of each process should not be changed – each process still needs to use its corresponding TWO or THREE resources simultaneously in each of its critical section; and (2) each process should only hold the locks for the resources used in each critical section before enter that critical section.

(d) (6 pts) (Deadlock avoidance) Consider the version of the dining-philosophers problem in which the chopsticks are placed at the enter of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers. Assume there are \( N \) philosophers and a total of \( N \) chopsticks. Note you solution should not only allow one philosopher to eat at a time.
4. (Page Replacement - 20 pts)

(a) (10 pts) Consider the following virtual page reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5, 1. Assume the physical memory is initially empty. Under the FIFO page replacement policy, if the physical memory has 4 frames, how many page faults will there be? Under the FIFO page replacement policy, if the physical memory has 3 frames, how many page faults will there be?

(b) (10 pts) Can you come up with a page reference string for which under the FIFO page replacement policy, there will be more page faults if the physical memory has 16 frames than if the physical memory has 12 frames?
5. (Directories and Links - 20 pts)

(a) (8 pts) Given an absolute pathname, "/homes/ychu/exam2.txt", 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 pts) If "/hardlink" is a hard link that points to "/home/ychu/exam2.txt", 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) (8 pts) If "/softlink" is a soft link that points to "/home/ychu/exam2.txt", 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.
6. (File System Interface - 10 pts)

(a) (5 pts) Compare the read/write model of file access, and the memory-mapped file model of file access. What potential drawbacks of the read/write model are addressed by the memory-mapped model?

(b) (5 pts) The generic version of the system call used to set up a memory-mapped file looks like this:
\[
\text{mmap(void *start address, size_t length, int protection, int flags, int fd, off_t offset)}
\]
After an invocation of mmap, with length = 16384 bytes, successfully returns, how many physical pages have been allocated to the virtual address segment being mapped?