**Final Project Report**

This project is finished by Xianlong Zhang, and fulfilled the kernel threads part. The threads function is mainly realized by clone() and join() syscall with some functions defined in pthread library, such as lock, condition variable and semaphore.

**Modifications in xv6 are as following:**

1. In Makefile, I added pthread library and test cases.

2. In proc.c, four syscalls were added:

a) clone() behaves like fork(), it creates new thread,

b) join() waits for thread to finish, which is based on wait().

c) threadWake() wakes up thread by recognising passed argument,

d) threadSleep() sleeps current thread.

3. Created a pthread library “pthread.c”.

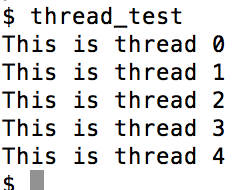
4. In user.h, full prototype of the new added system calls were included.

5. In usys.S, new macro invocations for the new system calls were added.

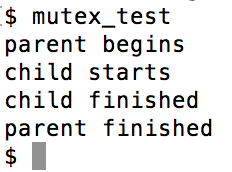
6. Besides, there were also some small modifications in defs.h, sysproc.h, fs.c, fs.h, types.h, syscall.c, syscall.h.

**Execution examples:**

1. thread\_test.c: Created 5 threads to test pthread\_create(), the output should be as following:



2. mutex\_test.c: Create a thread with using lock to test lock and conidtion varible, the output should be as following:



3. philosopher.c: The q1 in homework7 was used as an example to test thread and semphore, there would be a infinite loop without deadlock, the output should like the following graph (part).

