COMP4300 – Frequently Asked Questions

Project 3 – Synchronization

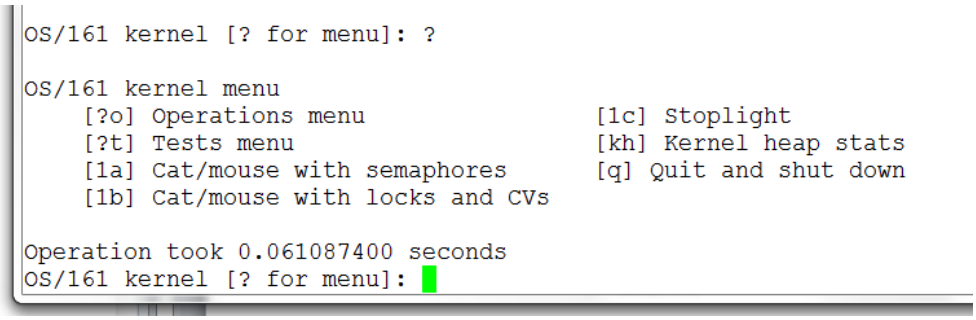
1. While trying project 3, we boot up os161, and go to the test menu, but there isn't a 1a option test for the cat and mouse.

   Answer: Please follow the following instructions to rebuild kernel for project 3.

```
%cd ~/cs161/src
%./configure
%cd ~/cs161/src/kern/conf
%./config ASST1

% cd ../compile/ASST1
% make depend
% make
```

   Now you can reboot your OS161 and you will find the following new menu.



2. How to use semaphores in OS/161?

   Answer: Please follow the sample code below to crease and use semaphores to solve the cats-mice synchronization problem.

```
struct semaphore *testsem; /* Declare a semaphore */

testsem = sem_create("testsem", 2); /* Create and init */
if (testsem == NULL) {
      panic("synchtest: sem_create failed\n");
}

P(testsem); /* Wait(testsem) */

kprintf("ok\n");

V(testsem); /* Signal(testsem) */
```

3. How to implement “play around” and “eat food” in the cats-mice algorithms?

   Answer:

   ```
   #define MAXTIME 3 /* You may change 3 to any number */

   clocksleep(random() % MAXTIME);
   ```

4. My group is having issues with implementing the catlock portion of the assignment. We tried using a setup similar to the semaphore implementation but this hasn’t worked out.

   Answer: catlock.c is use the lock and condition variable mechanisms to solve the cats-mice problem. You must implement the lock and condition variable mechanisms first before working on catlock.c

5. Does it matter where we put the play around portion of cat and mouse? Is it supposed to be immediately after they finish eating or just at the end to make it wait before it tries to eat again?

   Answer: Use the following state to make cats or mice to wait before eat again.

   ```
   clocksleep(random() % MAXTIME);
   ```