```c
/* FocalPoint LKI */
/* Lab9: Kernel Synchronization Lab */
#include < linux/module.h >
#include < linux/kernel.h >
#include < linux/init.h >

#include < linux/delay.h >
#include < linux/kthread.h >
#include < linux/semaphore.h >
#include < linux/random.h >

#define DRIVER_AUTHOR "FocalPoint"
#define DRIVER_DESC   "Lab9"

MODULE_LICENSE("GPL");           // Get rid of taint message by declaring code as GPL.

/*  Or with defines, like this: */
MODULE_AUTHOR(DRIVER_AUTHOR);    // Who wrote this module?
MODULE_DESCRIPTION(DRIVER_DESC); // What does this module do?

int init(void) ; void
cleanup(void) ;

struct task_struct *ts1, *ts2, *ts3, *ts4, *ts5;

struct semaphore sem; int gc_count = 0;

int thread(void *data)
{

        int i, lessthan1000;
        get_random_bytes(&i, sizeof( i));
        lessthan1000 = i % 100;

        while(! kthread_should_stop ())
        { gc_count++;
                printk("incrementer: thread %d started\n", gc_count);

                if(gc_count == 2)
                { set_current_state(TASK_INTERRUPTIBLE);
                        msleep(1000) ;
                        set_current_state(TASK_RUNNING);
                }

                printk("incrementer:thread %d finished\n", gc_count);
                break;
        }

        return 0;
}

int init(void)
{ printk(KERN_INFO "init_module() called\n") ;

        /* initialize the semaphore */
        sema_init(&sem, 1) ;

        ts1 = kthread_run(thread, "thread1", "thread1") ;
        ts2 = kthread_run(thread, "thread2", "thread2") ;
        ts3 = kthread_run(thread, "thread3", "thread3") ;
        ts4 = kthread_run(thread, "thread4", "thread4") ;
        ts5 = kthread_run(thread, "thread5", "thread5") ;
        return 0;
}

void cleanup(void)
{ printk(KERN_ALERT "Unloading incrementer ...\n") ;
}

module_init(init);
module_exit(cleanup);
```