# Project: Linux Kernel Modules

## Description:

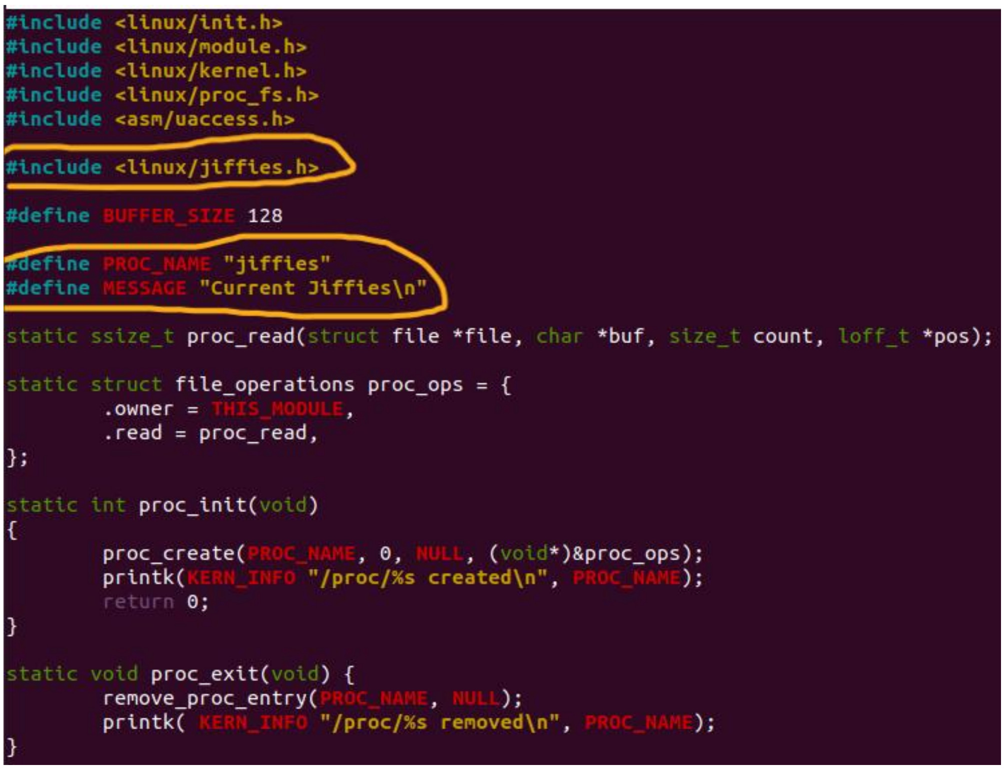
This programming project is composed of 4 files:
<1> "jiffies.c", "seconds.c" which I edited from the original file hello.c.
<2> "Makefile" that can make these 2 files together.
<3> "run.sh" that can display all necessary procedures and show the results concurrently.

To check my programs:
<1> You can use "make" to compile them and use "insmod", "rmmod", and "dmesg" to see what happened step by step.
<2> Or you can simply run "run.sh" to see all the necessary procedures by the outputs of the command line, and see the results of the 2 problems. (Note that "run.sh" may require your passcode for the "sudo" procedures I put in it, so don't be afraid of it. ^^)

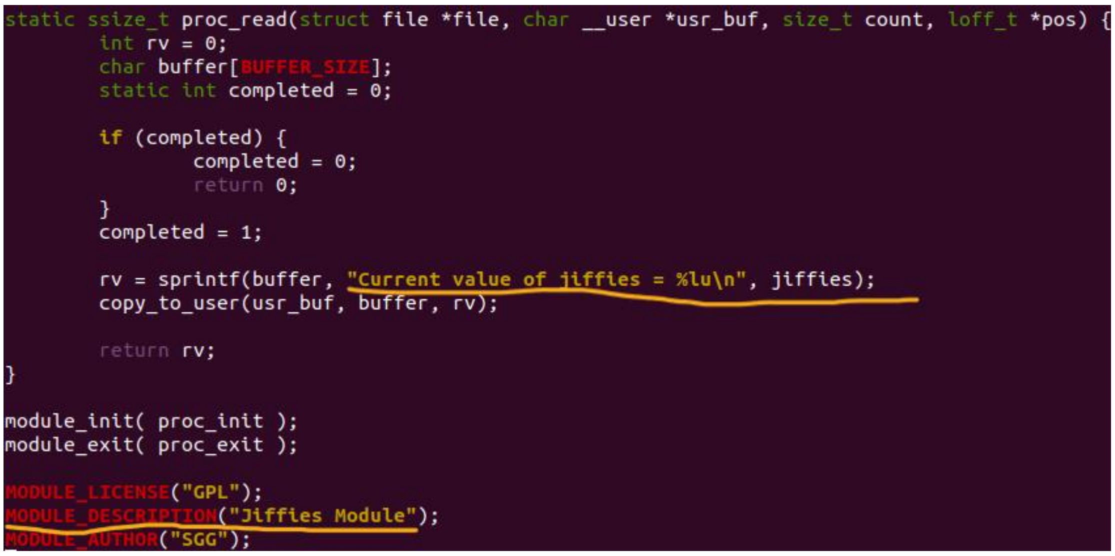
For this report, I will show you the code of "jiffies.c" and "seconds.c" first, and point out where I have made changes based on hello.c and explain why. Then, I will show you the running results of the 2 programs. And finally, I will show you the code and how I use "run.sh" to show you all procedures at one time.

## Problem 1: Report the current value of jiffies.

## Code:



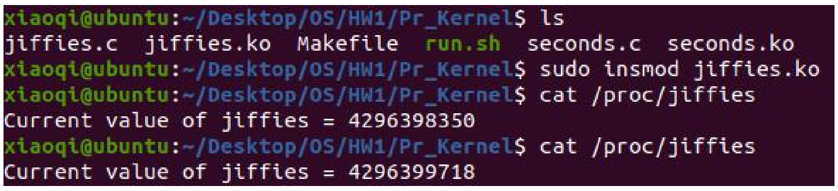
```c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/proc_fs.h>
#include <asm/uaccess.h>

#include <linux/jiffies.h>

#define BUFFER_SIZE 128

#define PROC_NAME "jiffies"
#define MESSAGE "Current Jiffies\n"

static ssize_t proc_read(struct file *file, char *buf, size_t count, loff_t *pos);

static struct file_operations proc_ops = {
        .owner = THIS_MODULE,
        .read = proc_read,
};

static int proc_init(void)
{
        proc_create(PROC_NAME, 0, NULL, (void*)&proc_ops);
        printk(KERN_INFO "/proc/%s created\n", PROC_NAME);
        return 0;
}

static void proc_exit(void) {
        remove_proc_entry(PROC_NAME, NULL);
        printk( KERN_INFO "/proc/%s removed\n", PROC_NAME);
}
```



```
static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos) {
        int rv = 0;
        char buffer[BUFFER_SIZE];
        static int completed = 0;

        if (completed) {
                completed = 0;
                return 0;
        }
        completed = 1;

        rv = sprintf(buffer, "Current value of jiffies = %lu\n", jiffies);
        copy_to_user(usr_buf, buffer, rv);

        return rv;
}

module_init( proc_init );
module_exit( proc_exit );

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Jiffies Module");
MODULE_AUTHOR("SGG");
```

The places with mark are only the changes I made based on hello.c.
After reading the tutorials of jiffies in the official website of Linux, I know that "jiffies" is a global variable from <linux/jiffies.h> that represents the current value of jiffies. So I just include this library, and print out the global variable "jiffies" to show the current value in the function "proc_read", which will work when we read the "/proc/jiffies".

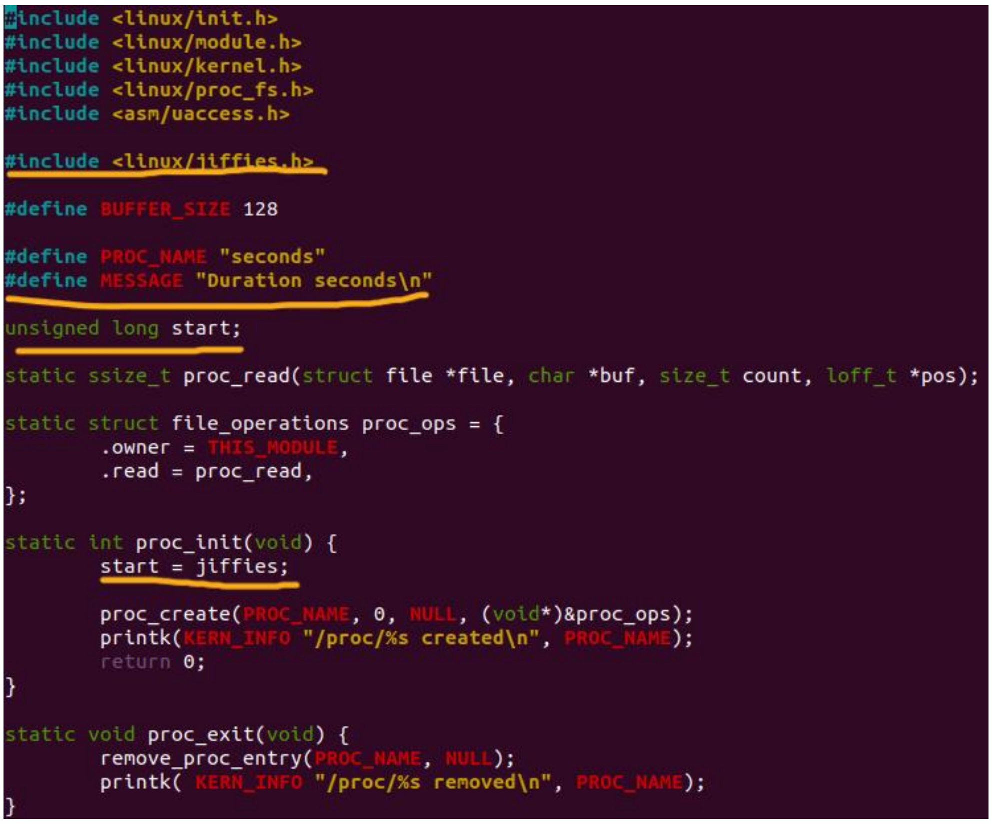
## Outputs:



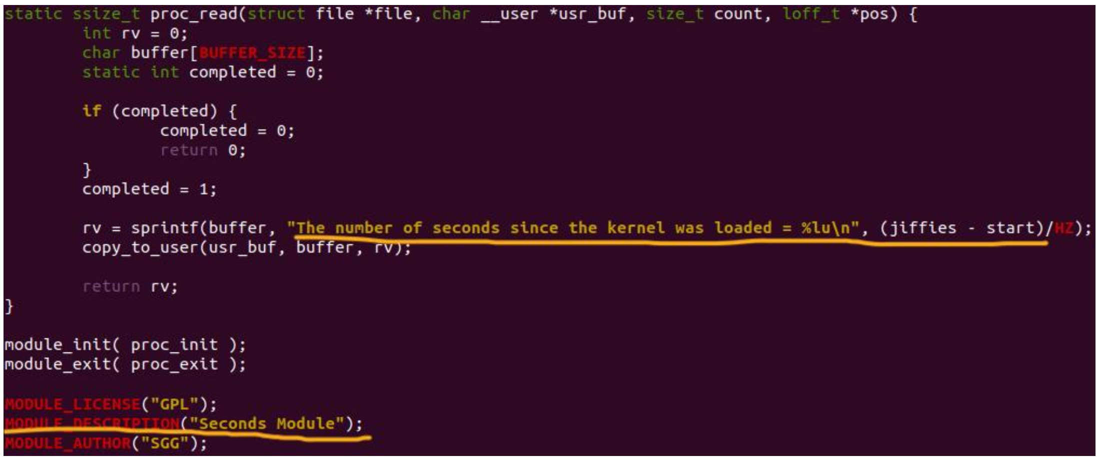
```
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ ls
jiffies.c  jiffies.ko  Makefile  run.sh  seconds.c  seconds.ko
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ sudo insmod jiffies.ko
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ cat /proc/jiffies
Current value of jiffies = 4296398350
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ cat /proc/jiffies
Current value of jiffies = 4296399718
```

## Problem2: Report the number of seconds since the kernel module was loaded.

## Code:



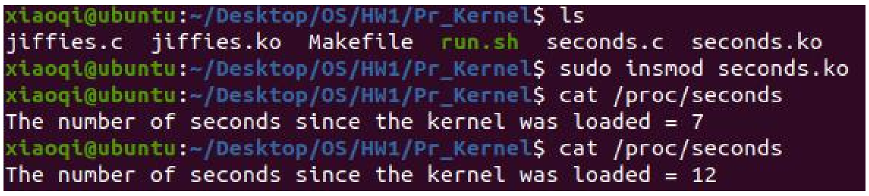
```c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/proc_fs.h>
#include <asm/uaccess.h>

#include <linux/jiffies.h>

#define BUFFER_SIZE 128

#define PROC_NAME "seconds"
#define MESSAGE "Duration seconds\n"

unsigned long start;

static ssize_t proc_read(struct file *file, char *buf, size_t count, loff_t *pos);

static struct file_operations proc_ops = {
        .owner = THIS_MODULE,
        .read = proc_read,
};

static int proc_init(void) {
        start = jiffies;

        proc_create(PROC_NAME, 0, NULL, (void*)&proc_ops);
        printk(KERN_INFO "/proc/%s created\n", PROC_NAME);
        return 0;
}

static void proc_exit(void) {
        remove_proc_entry(PROC_NAME, NULL);
        printk( KERN_INFO "/proc/%s removed\n", PROC_NAME);
}
```



```c
static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count, loff_t *pos) {
        int rv = 0;
        char buffer[BUFFER_SIZE];
        static int completed = 0;

        if (completed) {
                completed = 0;
                return 0;
        }
        completed = 1;

        rv = sprintf(buffer, "The number of seconds since the kernel was loaded = %lu\n", (jiffies - start)/HZ);
        copy_to_user(usr_buf, buffer, rv);

        return rv;
}

module_init( proc_init );
module_exit( proc_exit );

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Seconds Module");
MODULE_AUTHOR("SGG");
```

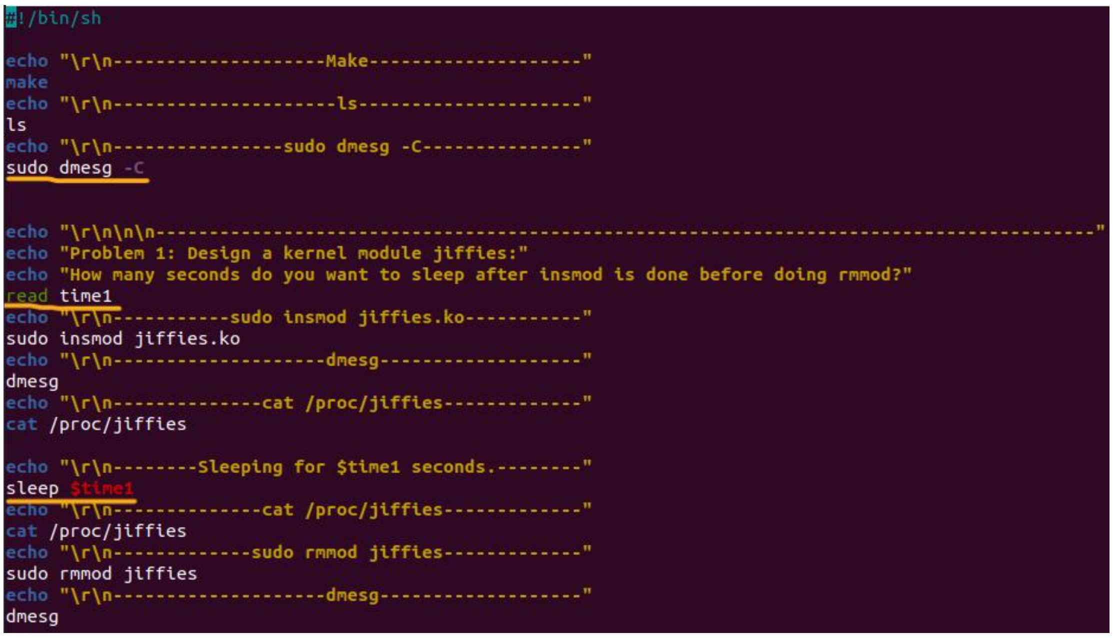
The places with mark are only the changes I made based on hello.c.
I know jiffies/HZ = elapsed time in seconds, in order to get the current elapsed time, we need to know the jiffies during the progress. So I first add a variable "unsigned long start" to store the the value of jiffies at the start in function "proc_init", which will work one we load this kernel module. The I divided the subtraction of start jiffies and current jiffies by HZ (a global variable from <linux/jiffies.h>) in function "proc_read", which will work when we read "/proc/seconds", and get the elapsed time since the kernel module was loaded in seconds.

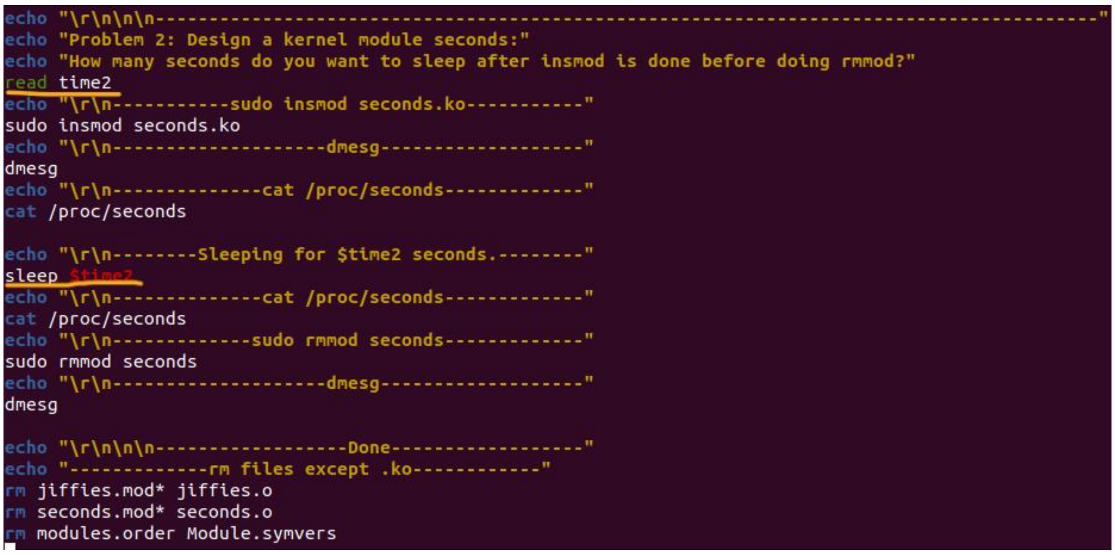
## Outputs:


```
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ ls
jiffies.c  jiffies.ko  Makefile  run.sh  seconds.c  seconds.ko
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ sudo insmod seconds.ko
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ cat /proc/seconds
The number of seconds since the kernel was loaded = 7
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ cat /proc/seconds
The number of seconds since the kernel was loaded = 12
```

## To show you the results easier: run and display all necessary procedures at one time: run.sh

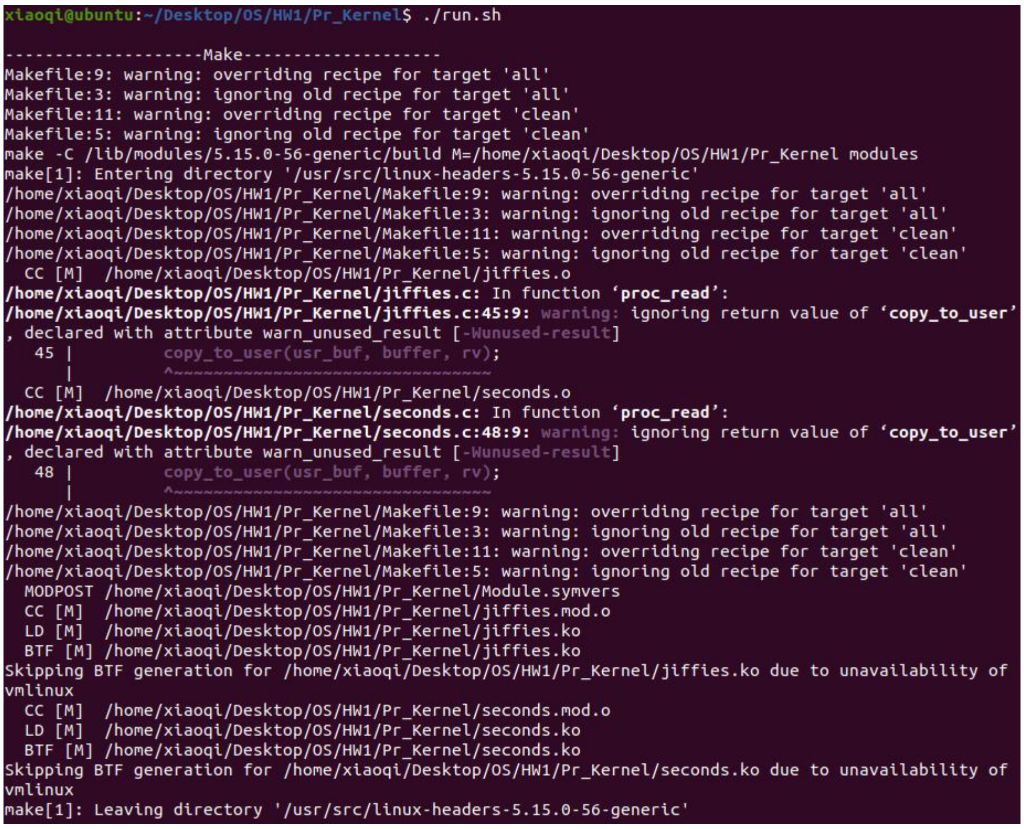
## Code:


```
#!/bin/sh

echo "\r\n-------------------Make-------------------"
make
echo "\r\n--------------------ls--------------------"
ls
echo "\r\n---------------sudo dmesg -C--------------"
sudo dmesg -C


echo "\r\n\n\n--------------------------------------------------------------------------------------------"
echo "Problem 1: Design a kernel module jiffies:"
echo "How many seconds do you want to sleep after insmod is done before doing rmmod?"
read time1
echo "\r\n-----------sudo insmod jiffies.ko-----------"
sudo insmod jiffies.ko
echo "\r\n-------------------dmesg------------------"
dmesg
echo "\r\n-------------cat /proc/jiffies------------"
cat /proc/jiffies

echo "\r\n--------Sleeping for $time1 seconds.--------"
sleep $time1
echo "\r\n-------------cat /proc/jiffies------------"
cat /proc/jiffies
echo "\r\n-------------sudo rmmod jiffies------------"
sudo rmmod jiffies
echo "\r\n-------------------dmesg------------------"
dmesg
```

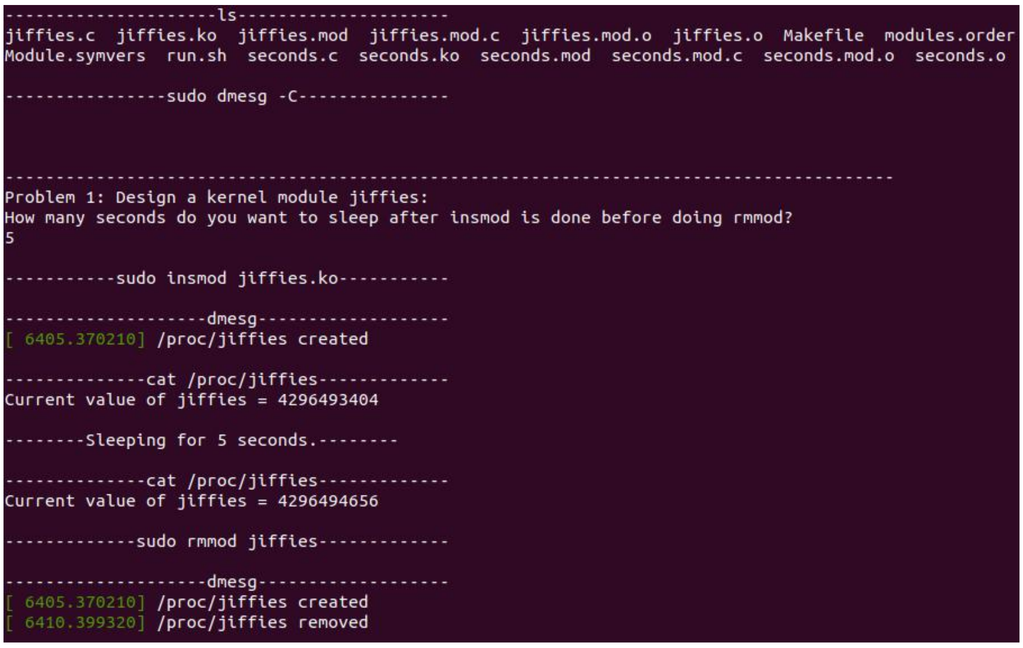

```
echo "\r\n\n\n--------------------------------------------------------------------------------------------"
echo "Problem 2: Design a kernel module seconds:"
echo "How many seconds do you want to sleep after insmod is done before doing rmmod?"
read time2
echo "\r\n-----------sudo insmod seconds.ko-----------"
sudo insmod seconds.ko
echo "\r\n-------------------dmesg------------------"
dmesg
echo "\r\n-------------cat /proc/seconds------------"
cat /proc/seconds

echo "\r\n--------Sleeping for $time2 seconds.--------"
sleep $time2
echo "\r\n-------------cat /proc/seconds------------"
cat /proc/seconds
echo "\r\n-------------sudo rmmod seconds------------"
sudo rmmod seconds
echo "\r\n-------------------dmesg------------------"
dmesg

echo "\r\n\n\n------------------Done------------------"
echo "-------------rm files except .ko------------"
rm jiffies.mod* jiffies.o
rm seconds.mod* seconds.o
rm modules.order Module.symvers
```

I used “make” first to compile them first. I used “echo” to display every procedures I will do in the output of the command line before I do it in program. I used “dmesg -C” to clean all the information of the kernel to let you see the changes made by insmod and rmmod easier. I used “read” and “sleep” to let you control the time between doing insmod and rmmod, or you can’t see the changes of the “cat /proc/seconds”, because it will be finished very fast of the two “cat”, and the result of the 2 “cat /proc/...” will be almost the same. Finally, I used “rm” to remove all the resulting files compiled by Makefile, thus it makes it tidy of these programs.

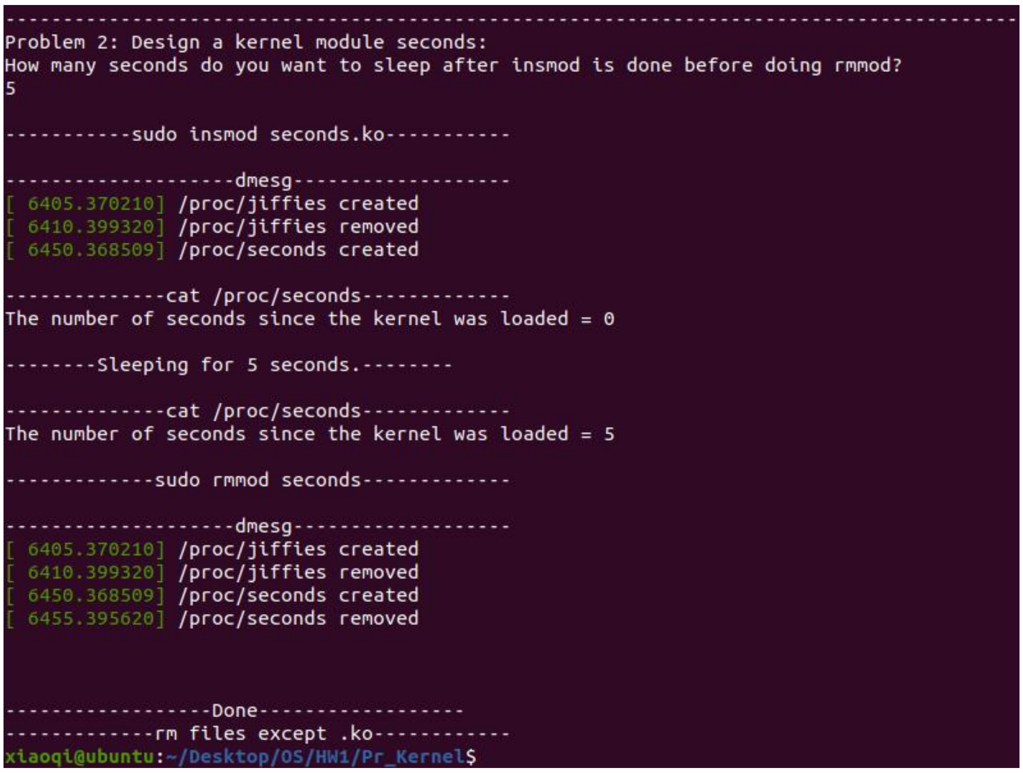
## Outputs:



```
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$ ./run.sh

-------------------Make-------------------
Makefile:9: warning: overriding recipe for target 'all'
Makefile:3: warning: ignoring old recipe for target 'all'
Makefile:11: warning: overriding recipe for target 'clean'
Makefile:5: warning: ignoring old recipe for target 'clean'
make -C /lib/modules/5.15.0-56-generic/build M=/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.0-56-generic'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:9: warning: overriding recipe for target 'all'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:3: warning: ignoring old recipe for target 'all'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:11: warning: overriding recipe for target 'clean'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:5: warning: ignoring old recipe for target 'clean'
  CC [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.o
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.c: In function ‘proc_read’:
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.c:45:9: warning: ignoring return value of ‘copy_to_user’
, declared with attribute warn_unused_result [-Wunused-result]
   45 |         copy_to_user(usr_buf, buffer, rv);
      |         ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  CC [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.o
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.c: In function ‘proc_read’:
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.c:48:9: warning: ignoring return value of ‘copy_to_user’
, declared with attribute warn_unused_result [-Wunused-result]
   48 |         copy_to_user(usr_buf, buffer, rv);
      |         ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:9: warning: overriding recipe for target 'all'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:3: warning: ignoring old recipe for target 'all'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:11: warning: overriding recipe for target 'clean'
/home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Makefile:5: warning: ignoring old recipe for target 'clean'
  MODPOST /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/Module.symvers
  CC [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.mod.o
  LD [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.ko
  BTF [M] /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.ko
Skipping BTF generation for /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/jiffies.ko due to unavailability of
vmlinux
  CC [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.mod.o
  LD [M]  /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.ko
  BTF [M] /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.ko
Skipping BTF generation for /home/xiaoqi/Desktop/OS/HW1/Pr_Kernel/seconds.ko due to unavailability of
vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-5.15.0-56-generic'
```



```
--------------------ls--------------------
jiffies.c  jiffies.ko  jiffies.mod  jiffies.mod.c  jiffies.mod.o  jiffies.o  Makefile  modules.order
Module.symvers  run.sh  seconds.c  seconds.ko  seconds.mod  seconds.mod.c  seconds.mod.o  seconds.o

---------------sudo dmesg -C---------------



-----------------------------------------------------------------------------------------
Problem 1: Design a kernel module jiffies:
How many seconds do you want to sleep after insmod is done before doing rmmod?
5

----------sudo insmod jiffies.ko----------

-------------------dmesg-------------------
[ 6405.370210] /proc/jiffies created

-------------cat /proc/jiffies-------------
Current value of jiffies = 4296493404

--------Sleeping for 5 seconds.--------

-------------cat /proc/jiffies-------------
Current value of jiffies = 4296494656

------------sudo rmmod jiffies------------

-------------------dmesg-------------------
[ 6405.370210] /proc/jiffies created
[ 6410.399320] /proc/jiffies removed
```



```
----------------------------------------------------------------------------------------
Problem 2: Design a kernel module seconds:
How many seconds do you want to sleep after insmod is done before doing rmmod?
5

-----------sudo insmod seconds.ko-----------

--------------------dmesg-------------------
[ 6405.370210] /proc/jiffies created
[ 6410.399320] /proc/jiffies removed
[ 6450.368509] /proc/seconds created

-------------cat /proc/seconds-------------
The number of seconds since the kernel was loaded = 0

--------Sleeping for 5 seconds.--------

-------------cat /proc/seconds-------------
The number of seconds since the kernel was loaded = 5

-------------sudo rmmod seconds-------------

--------------------dmesg-------------------
[ 6405.370210] /proc/jiffies created
[ 6410.399320] /proc/jiffies removed
[ 6450.368509] /proc/seconds created
[ 6455.395620] /proc/seconds removed



------------------Done------------------
-------------rm files except .ko------------
xiaoqi@ubuntu:~/Desktop/OS/HW1/Pr_Kernel$
```

As you can see, the output are three main parts: Preparation part, Problem 1, and Problem 2. Every necessary procedures and results are displayed here, and you can see everything of how the program works.

That's the end of this report, thank you very much for your attention!
Author: Xiaoqi LIU 999009335