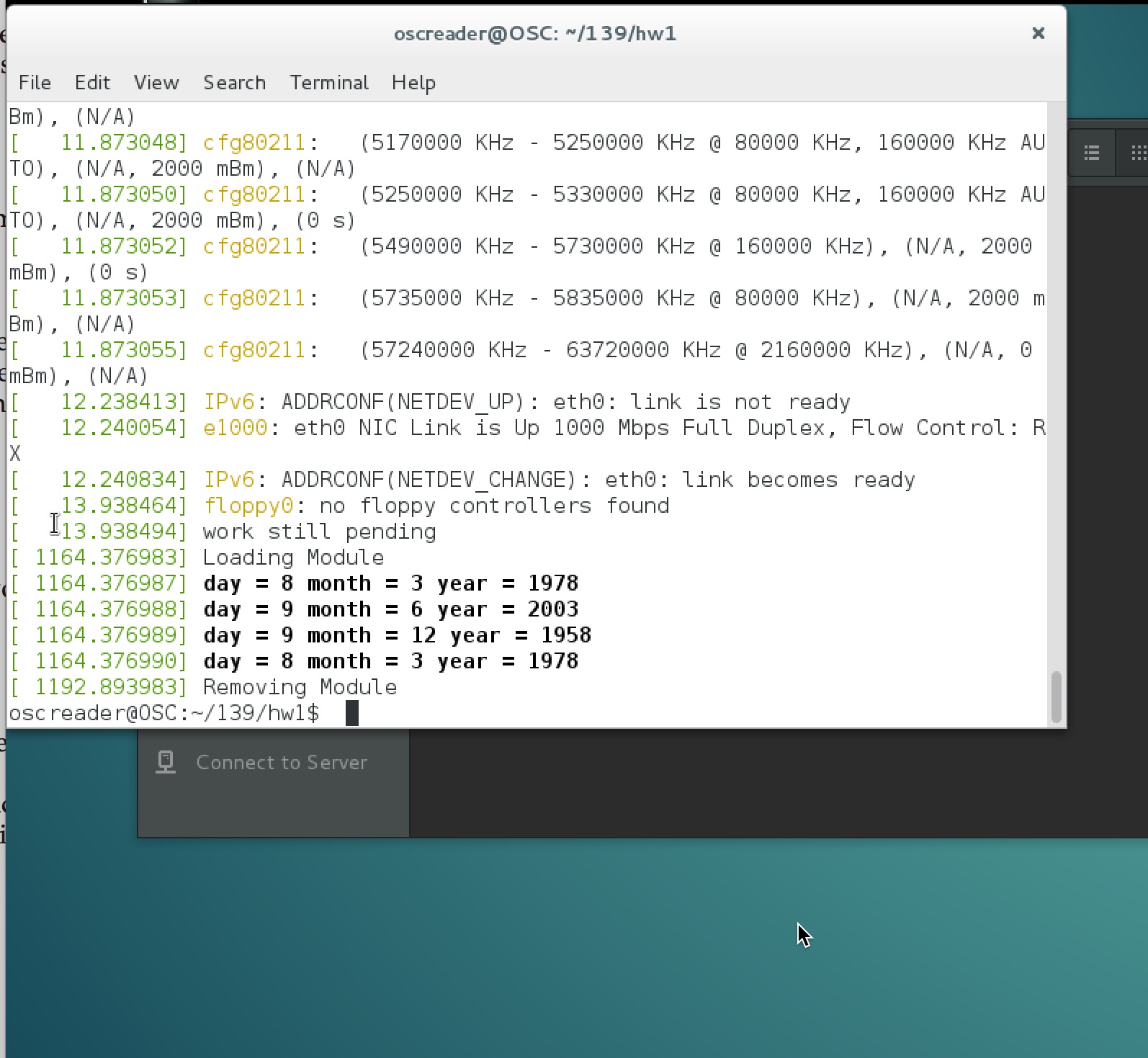
Varun Ved

CSC 139-04 Assignment 1

Screenshot



Source Code

#include <linux/init.h>

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/types.h>

#include <linux/slab.h>

/\*

\*1) In the module entry point, create a linked list containing five struct birthday elements

(respectively for the following people, from head to tail: Alice, Bob, Mallory, Nancy and

Kate). Traverse the linked list and output its contents to the kernel log buffer. Invoke the

dmesg command to ensure the list is properly constructed once the kernel module has been

loaded. Please take the screen shots.

Alice born on Jan 9, 1999;

Bob born on March 8, 1978;

Mallory born on Dec 9, 1958;

Nancy born on Jun 9, 2003;

Kate born on March 8, 1978;

2) Please sort the five struct birthday elements in the list by this sequence: from head to tail, the people are from old to young. You may need to come up with new functions if you need. Traverse the linked list again and output its contents to the kernel log buffer. Invoke

the dmesg command to ensure the list is properly rearranged. Please take the screen shots.

3) In the module exit point, delete the elements from the linked list and return the free memory

back to the kernel. Again, invoke the dmesg command to check that the list has been

removed once the kernel module has been unloaded. Please take the screen shots.

\*/

struct birthday {

int day;

int month;

int year;

struct list\_head list;

};

struct birthday \*person, \*tmp, \*ptr1, \*ptr2, \*ptr3, \*ptr4, \*ptr5;

static LIST\_HEAD(birthday\_list);

struct list\_head \*pos;

/\* This function is called when the module is loaded. \*/

int simple\_init(void)

{

//Alice first

person = kmalloc(sizeof(\*person),GFP\_KERNEL);

person->day = 1;

person->month = 9;

person->year = 1999;

int bob\_day = 8;

int bob\_month = 3;

int bob\_year = 1978;

int mallory\_day = 9;

int mallory\_month = 12;

int mallory\_year = 1958;

int nancy\_day = 9;

int nancy\_month = 6;

int nancy\_year = 2003;

int kate\_day = 8;

int kate\_month = 3;

int kate\_year = 1978;

///set to head

INIT\_LIST\_HEAD(&person->list);

printk(KERN\_INFO "Loading Module\n");

ptr1 = kmalloc(sizeof(\*ptr1), GFP\_KERNEL);

ptr1->day = bob\_day;

ptr1->month = bob\_month;

ptr1->year = bob\_year;

list\_add(&(ptr1->list), &(person->list));

ptr2 = kmalloc(sizeof(\*ptr2), GFP\_KERNEL);

ptr2->day = mallory\_day;

ptr2->month = mallory\_month;

ptr2->year = mallory\_year;

list\_add(&(ptr2->list), &(person->list));

ptr3 = kmalloc(sizeof(\*ptr3), GFP\_KERNEL);

ptr3->day = nancy\_day;

ptr3->month = nancy\_month;

ptr3->year = nancy\_year;

list\_add(&(ptr3->list), &(person->list));

ptr4 = kmalloc(sizeof(\*ptr4), GFP\_KERNEL);

ptr4->day = kate\_day;

ptr4->month = kate\_month;

ptr4->year = kate\_year;

list\_add(&(ptr4->list), &(person->list));

/\*

ptr5 = kmalloc(sizeof(\*ptr5), GFP\_KERNEL);

ptr5->day = nancy\_day;

ptr5->month = nancy\_month;

ptr5->year = nancy\_year;

list\_add(&(ptr5->list), &(person->list));

\*/

list\_for\_each(pos, &(person->list)){

tmp = list\_entry(pos, struct birthday, list);

printk("day = %d month = %d year = %d\n", tmp->day, tmp->month, tmp->year);

}

return 0;

}

/\* This function is called when the module is removed. \*/

void simple\_exit(void) {

printk(KERN\_INFO "Removing Module\n");

}

/\* Macros for registering module entry and exit points. \*/

module\_init( simple\_init );

module\_exit( simple\_exit );

MODULE\_LICENSE("GPL");

MODULE\_DESCRIPTION("Simple Module");

MODULE\_AUTHOR("SGG");

Makefile

obj-m += simple.o

all:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean