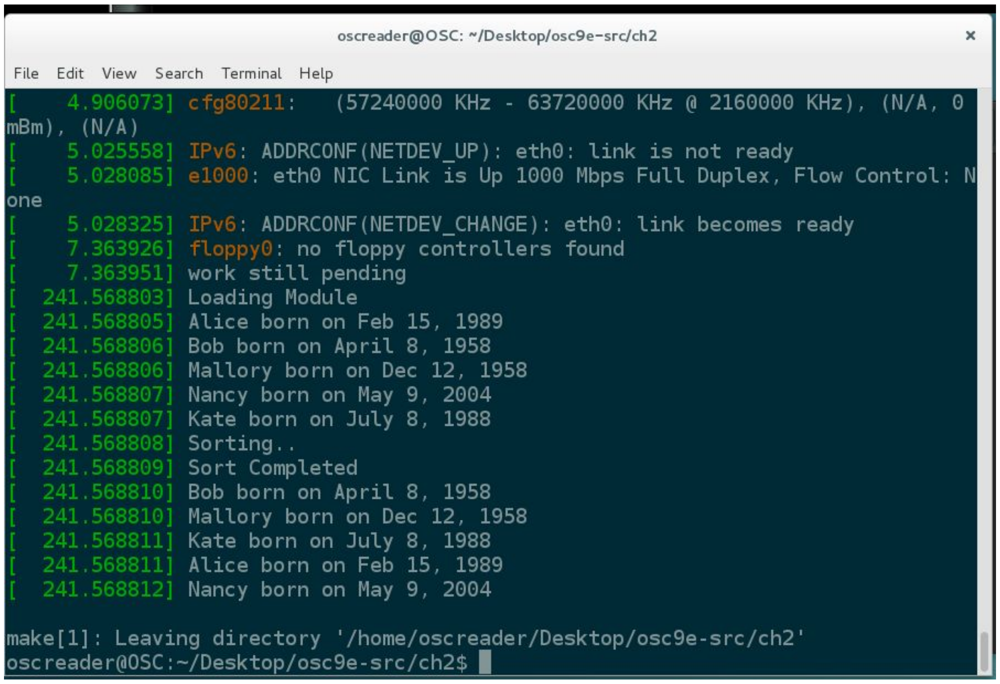
RONGUANG OU
CSC139 - HW2
Loadable Kernel Module

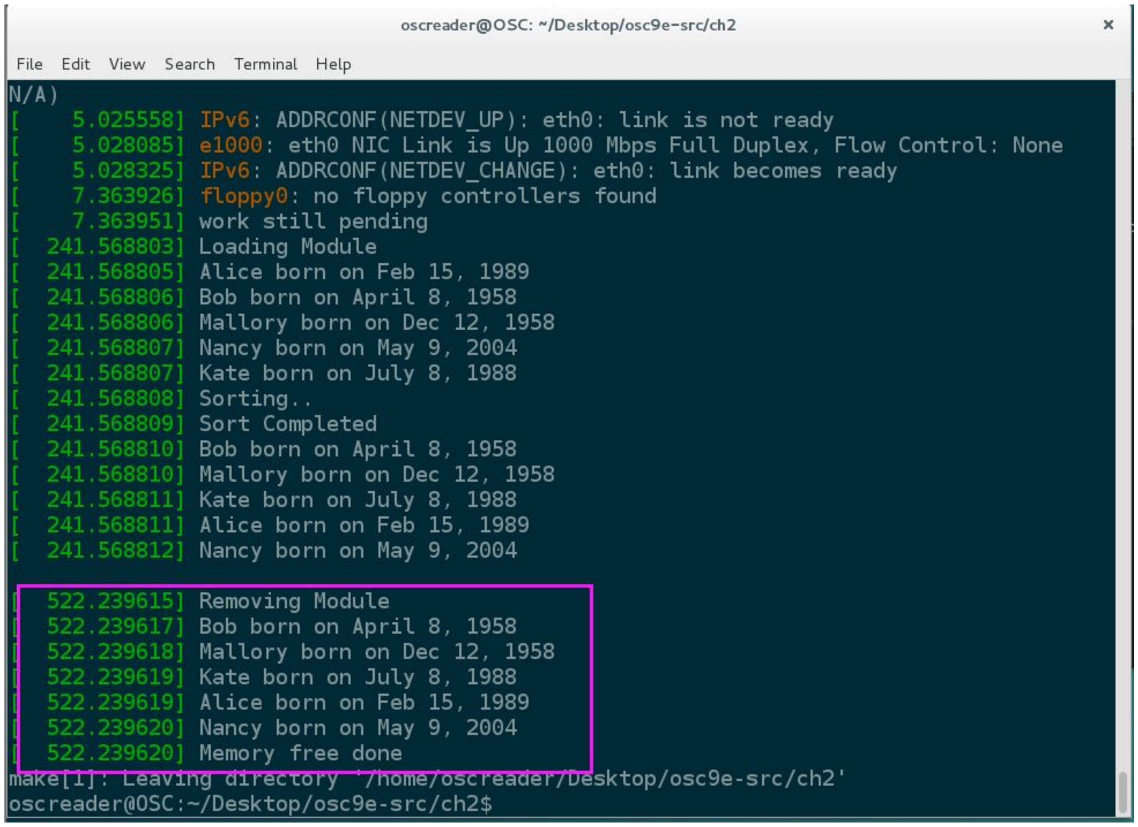
Upon compiling Simple.c. Kernel module simple is then loaded into the kernel using sudo insmod simple. Loading module should print out all the birthdays in the linked list then it will sort the birthdays according the age from oldest to youngest.


oscreader@OSC: ~/Desktop/osc9e-src/ch2
File Edit View Search Terminal Help

```
[    4.906073] cfg80211:   (57240000 KHz - 63720000 KHz @ 2160000 KHz), (N/A, 0
mBm), (N/A)
[    5.025558] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[    5.028085] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: N
one
[    5.028325] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[    7.363926] floppy0: no floppy controllers found
[    7.363951] work still pending
[  241.568803] Loading Module
[  241.568805] Alice born on Feb 15, 1989
[  241.568806] Bob born on April 8, 1958
[  241.568806] Mallory born on Dec 12, 1958
[  241.568807] Nancy born on May 9, 2004
[  241.568807] Kate born on July 8, 1988
[  241.568808] Sorting..
[  241.568809] Sort Completed
[  241.568810] Bob born on April 8, 1958
[  241.568810] Mallory born on Dec 12, 1958
[  241.568811] Kate born on July 8, 1988
[  241.568811] Alice born on Feb 15, 1989
[  241.568812] Nancy born on May 9, 2004

make[1]: Leaving directory '/home/oscreader/Desktop/osc9e-src/ch2'
oscreader@OSC:~/Desktop/osc9e-src/ch2$ 
```

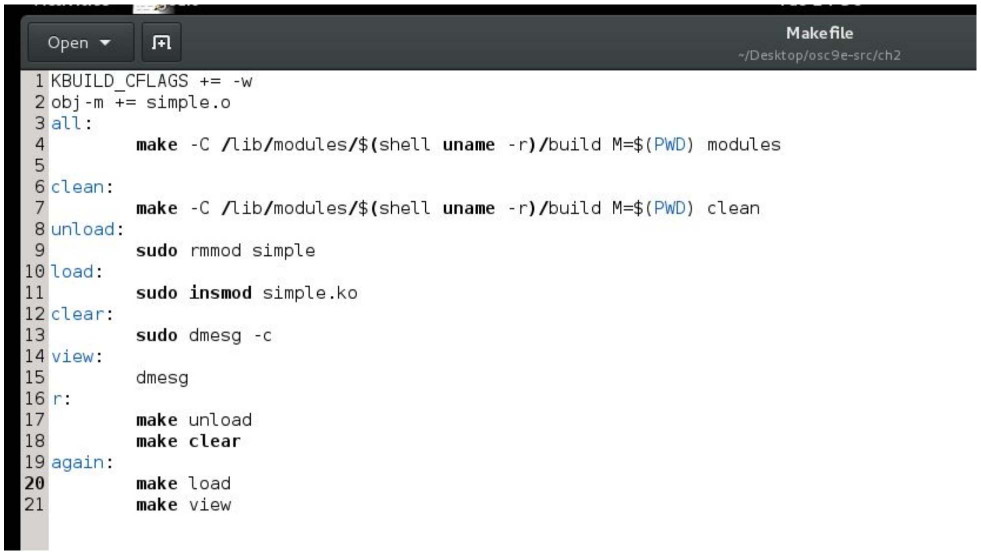
Upon removing a simple.ko from the kernel will remove the birthdays one by one and free the memory used by this module.


oscreader@OSC: ~/Desktop/osc9e-src/ch2
File Edit View Search Terminal Help

```
N/A)
[    5.025558] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[    5.028085] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: None
[    5.028325] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[    7.363926] floppy0: no floppy controllers found
[    7.363951] work still pending
[  241.568803] Loading Module
[  241.568805] Alice born on Feb 15, 1989
[  241.568806] Bob born on April 8, 1958
[  241.568806] Mallory born on Dec 12, 1958
[  241.568807] Nancy born on May 9, 2004
[  241.568807] Kate born on July 8, 1988
[  241.568808] Sorting..
[  241.568809] Sort Completed
[  241.568810] Bob born on April 8, 1958
[  241.568810] Mallory born on Dec 12, 1958
[  241.568811] Kate born on July 8, 1988
[  241.568811] Alice born on Feb 15, 1989
[  241.568812] Nancy born on May 9, 2004

[  522.239615] Removing Module
[  522.239617] Bob born on April 8, 1958
[  522.239618] Mallory born on Dec 12, 1958
[  522.239619] Kate born on July 8, 1988
[  522.239619] Alice born on Feb 15, 1989
[  522.239620] Nancy born on May 9, 2004
[  522.239620] Memory free done
make[1]: Leaving directory '/home/oscreader/Desktop/osc9e-src/ch2'
oscreader@OSC:~/Desktop/osc9e-src/ch2$
```

For the makfile, I have created few shortcut command to load module and view kernel log quicker.



```
KBUILD_CFLAGS += -w
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
unload:
	sudo rmmod simple
load:
	sudo insmod simple.ko
clear:
	sudo dmesg -c
view:
	dmesg
r:
	make unload
	make clear
again:
	make load
	make view
```

```
/*************************************************************************************
Author: Rongguang Ou
Date: 10/25/2019
Course: CSC 139 - Operating Systems    Fall 2019
Assignment 2 - Loadable kernel module

What this kernel module do?

When is loaded to kernel :  A Linked-List of 5 struct birthday will be created then list will be
sorted.
When is unloaded from kernel  : Removes each element in birthday linked list and free memory

**************************************************************************************/

#include <linux/module.h>   /* Needed by all modules  */
#include <linux/kernel.h>   /* Needed for KERN_INFO   */
#include <linux/init.h>     /* Needed for the macros  */
#include <linux/list.h>     /* Needed for linked list */
#include <linux/types.h>    /* Needed for list macros */
#include <linux/slab.h>     /* Needed for Kernel */
#include <linux/string.h>

#define DRIVER_AUTHOR       "RONGGUANG OU"
#define DRIVER_DESC         "LinuxKernelModule-V1"
#define DRIVER_LICE         "GPL" /* License Info */
#define NUM_OF_BIRTHDAYS    5

/* Birthday struct  */
typedef struct _birthday {

    int day;
    char* month;
    int year;
    char* name;
    struct list_head list;

}birthday;

/* Declare and init the head of the linked list. */
LIST_HEAD(birthday_list);

/*
```

```
  Initialize head_list  linked list then allocate memory for 5 birthday structs and add to linked list
and print content of each birthday struct to kernel log.
*/
int birthdayList_init(void) {

   printk(KERN_INFO "Loading Module\n");

   /* Allocate 5 birthdays from kernel */
   birthday *person;

        person = kmalloc(sizeof(*person), GFP_KERNEL);
        person->day = 15;
        person->month = "Feb";
        person->year = 1989;
        person->name = "Alice";
        INIT_LIST_HEAD(&person->list);
        list_add_tail(&person->list, &birthday_list);

        person = kmalloc(sizeof(*person), GFP_KERNEL);
        person->day = 8;
        person->month = "April";
        person->year = 1958;
        person->name = "Bob";
        INIT_LIST_HEAD(&person->list);
        list_add_tail(&person->list, &birthday_list);

        person = kmalloc(sizeof(*person), GFP_KERNEL);
        person->day = 12;
        person->month = "Dec";
        person->year = 1958;
        person->name = "Mallory";
        INIT_LIST_HEAD(&person->list);
        list_add_tail(&person->list, &birthday_list);

        person = kmalloc(sizeof(*person), GFP_KERNEL);
        person->day = 9;
        person->month = "May";
        person->year = 2004;
        person->name = "Nancy";
        INIT_LIST_HEAD(&person->list);
        list_add_tail(&person->list, &birthday_list);

        person = kmalloc(sizeof(*person), GFP_KERNEL);
```

```
            person->day = 8;
            person->month = "July";
            person->year = 1988;
            person->name = "Kate";
            INIT_LIST_HEAD(&person->list);
            list_add_tail(&person->list, &birthday_list);


    /* Go thru the list and print. */
    birthday *ptr;
    list_for_each_entry(ptr, &birthday_list, list) {
                    print(ptr);
    }

            printk(KERN_INFO "Sorting..\n");
            bubbleSort(&birthday_list);
            printk(KERN_INFO "Sort Completed\n");

            list_for_each_entry(ptr,&birthday_list , list){
                    print(ptr);
            }
            printk(KERN_INFO "\n");
            return 0;
}

/*
   This function is called when the module is removed.
   It prints the list of birthdays being removed, and
   then deletes the list from kernel memory
*/
void birthdayList_exit(void) {

    printk(KERN_INFO "Removing Module\n");

    /* Go thru the list and free the memory. */
    birthday *ptr, *next;
    list_for_each_entry_safe(ptr, next, &birthday_list, list) {

                    print(ptr);
        list_del(&ptr->list);
        kfree(ptr);

    }
    printk(KERN_INFO "Memory free done\n");
}

/* Helper Functions to sort() */

/*This function swaps the entire content of birthday a with birthday b */
void exchange(birthday* a, birthday* b){
            char* temp_name = a->name;
            int temp_year = a->year;
            char* temp_month = a->month;
            int temp_day = a->day;

            a->name = b->name;
            b->name = temp_name;

            a->year = b->year;
            b->year = temp_year;

            a->month = b->month;
            b->month = temp_month;

            a->day = b->day;
            b->day = temp_day;
}

/* BubbleSort the given linked-list by age from oldest to youngest */
void bubbleSort(struct list_head* birthday_list){
            birthday* entry;
            birthday* entry2;
            int i,swapped, month_int;
            int age1,age2;
            struct list_head* head = birthday_list;
            struct list_head* stop = birthday_list;
            struct list_head* ptr;


            do{
                    swapped = 0;
                    ptr = head->next;
                    while(ptr->next != stop){
                            entry = list_entry( ptr , struct _birthday , list);
                            entry2 = list_entry( ptr->next , struct _birthday, list);
                            age1 = entry->year*10000 + entry->day + getMonth(entry->month)*100;
```

```
                                age2 = entry2->year*10000 + entry2->day +
getMonth(entry2->month)*100;
                                if(age1 > age2){
                                        exchange(entry,entry2);
                                        swapped = 1;
                                }
                                ptr = ptr->next;
                        }
                        stop = ptr;
                }while(swapped);
}
/* Prints entire contents of a given struct birthday */
void print(birthday* entry){
                        printk(KERN_INFO "%s born on %s %d, %d \n",
                        entry->name,
                        entry->month,
                        entry->day,
                        entry->year);
}
/* Returns numerical representation of string month */
int getMonth(char* m){
                char months[12][10] =
{"Jan","Feb","Mar","April","May","June","July","Aug","Sept","Oct","Nov","Dec"};
                int i;
                for(i = 0 ; i < 12; i++){
                        if(strcmp(m,months[i])==0){
                                return (i+1);
                        }
                }
}

/* Macros for registering module entry and exit points. */
module_init(birthdayList_init);
module_exit(birthdayList_exit);

MODULE_LICENSE(DRIVER_LICE);
MODULE_DESCRIPTION(DRIVER_DESC);
MODULE_AUTHOR(DRIVER_AUTHOR);
```