Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №5

з дисципліни «Архітектура комп’ютерів-3»

на тему : «Модулі»

Роботу виконали:

студенти 3 курсу ФІОТ

групи ІВ-71

Молчанова В.С. (ЗК №7110)

Музика О.А. (ЗК №7111)

Роботу перевірив:

Каплунов А.В.

Київ – 2020

**Мета:** навчитися працювати зі зв’язними списками ядра, правильно робити

алокацію пам’яті на рівні ядра.

## Лістинг

Makefile

ifneq ($(KERNELRELEASE),)

# kbuild part of makefile

obj-m := hello.o

else

# normal makefile

KDIR ?= /lib/modules/`uname -r`/build

default:

$(MAKE) -C $(KDIR) M=$$PWD

clean:

$(MAKE) -C $(KDIR) M=$$PWD clean

endif

hello.c

#include <linux/init.h>

#include <linux/module.h>

#include <linux/printk.h>

#include <linux/ktime.h>

#include <linux/list.h>

#include <linux/slab.h>

MODULE\_AUTHOR("Varvara Molchanova and Oleksandr Musyka");

MODULE\_DESCRIPTION("Hello, world in Linux Kernel Training");

MODULE\_LICENSE("Dual BSD/GPL");

struct linked\_list {

struct list\_head node;

ktime\_t time;

};

static LIST\_HEAD(linked\_head);

static uint count = 1;

module\_param(count, uint, S\_IRUGO);

MODULE\_PARM\_DESC(count, "uint: number of lines to print");

static int \_\_init hello\_init(void)

{

int err = -EINVAL;

uint i;

ktime\_t time;

struct linked\_list \*ptr, \*tmp;

if (count > 10) {

printk(KERN\_EMERG "Error! Number is greater than 10!\n");

return err;

}

if (count == 0 || count >= 5) {

printk(KERN\_EMERG "Warning! Number is 0 or betwee 5 and 10.\n");

}

for (i = 0; i < count; i++) {

time = ktime\_get();

printk(KERN\_EMERG "Hello, world!\n");

ptr = kmalloc(sizeof(struct linked\_list), GFP\_KERNEL);

if (ZERO\_OR\_NULL\_PTR(ptr)) {

list\_for\_each\_entry\_safe(ptr, tmp, &linked\_head, node) {

list\_del(&ptr->node);

kfree(ptr);

}

printk(KERN\_ERR "Error! Something wrong with memory\n");

return -1;

}

\*ptr = (struct linked\_list) {

(struct list\_head) {NULL, NULL}, time,

};

list\_add\_tail(&ptr->node, &linked\_head);

}

return 0;

}

static void \_\_exit hello\_exit(void)

{

struct linked\_list \*data, \*tmp;

list\_for\_each\_entry\_safe(data, tmp, &linked\_head, node) {

printk(KERN\_INFO "%lld\n", data->time);

list\_del(&data->node);

kfree(data);

}

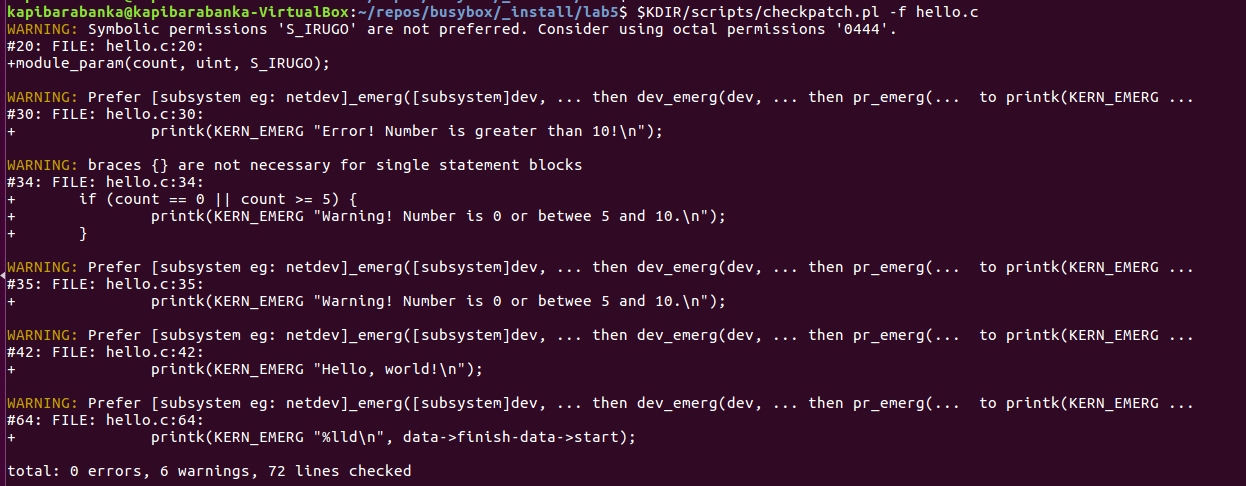
}

module\_init(hello\_init);

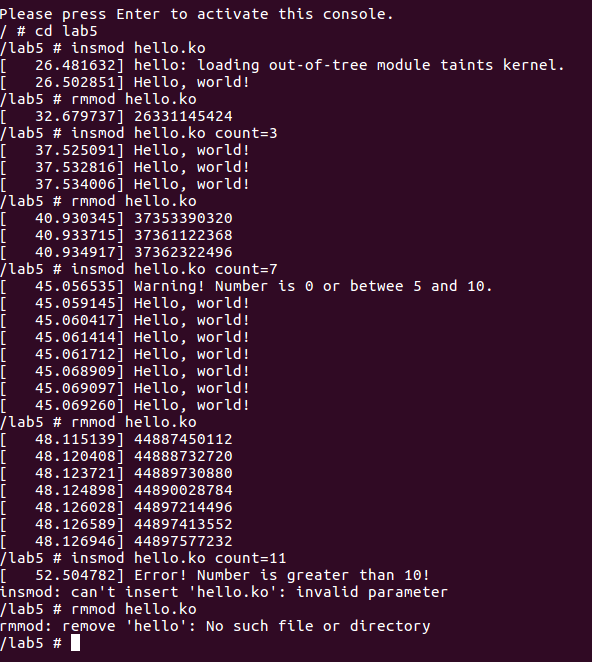
module\_exit(hello\_exit);

## Виконання

Перевірка відповідності Linux Kernel Coding Style:



Тестування при різних значеннях параметру:



## Висновок

Під час даної лабораторної роботи навчилися працювати зі зв’язними

списками ядра та правильно робити алокацію пам’яті на рівні ядра.