МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №4 з дисципліни

«Системне програмування С і С++»

Варіант 2

|  |  |
| --- | --- |
| Перевірив:  Сокульський О.Є. | Виконав:  студент 3 курсу  групи ІП-11 ФІОТ  Прищепа В.С. |

Київ-2023

**Лабораторна робота №4**

**Технології розробки модулів ядра для операційної системи Linux.**

Мета роботи полягає у оволодінні технологією розробки модулів ядра.

**Завдання:**

1. Встановити та налаштувати середовище для розрбки модулів ядра.

2. Створити новий проект користуючись додатком. У разі появи помилок внести зміни до вихідного коду.

3. Після успішного виконання п. 1, 2 завантажити модуль ядра.

4. Переконатися у його коректній роботі.

5. Зробити висновки щодо особливостей виконання лабораторної роботи, наявності специфічних особливостей роботи програми.

**Код програми:**

**part\_hello.c:**

#include <linux/kernel.h>

#include <linux/module.h>

MODULE\_AUTHOR("Pryshchepa Vladyslav");

MODULE\_DESCRIPTION("Lab1 test kernel module");

MODULE\_LICENSE("MIT");

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

{

printk(KERN\_ERR "Hello, World!\n");

return 0;

}

static void \_\_exit hw\_exit(void)

{

printk(KERN\_ERR "Goodbye, World!\n");

}

module\_init(hw\_init);

module\_exit(hw\_exit);

**Makefile:**

ifneq ($(KERNELRELEASE),)

obj-m := part\_hello.o

else

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

default:

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

clean:

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

endif

**part\_list.c:**

#include <linux/kernel.h>

#include <linux/module.h>

#include <linux/random.h>

#include <linux/slab.h>

#include <linux/types.h>

MODULE\_AUTHOR("Pryshchepa Vladyslav");

MODULE\_DESCRIPTION("Lab 1.2, using struct list\_head");

MODULE\_LICENSE("MIT");

#define LIST\_LEN 16

#define MSG\_PREF "TEST: "

#define print\_msg(msg, ...) printk(KERN\_ERR MSG\_PREF msg, ##\_\_VA\_ARGS\_\_);

typedef struct {

struct list\_head lnode;

int32\_t val;

} int\_node\_t;

#define ilfree(list\_head) \

do { \

int\_node\_t \*ptr, \*tmp; \

list\_for\_each\_entry\_safe(ptr, tmp, (list\_head), lnode) { \

kfree(\_\_ptr); \

} \

} while(0)

#define ilprint(list\_head) \

do { \

int\_node\_t \*\_\_ptr; \

print\_msg("List: {"); \

list\_for\_each\_entry(\_\_ptr, (list\_head), lnode) { \

printk(KERN\_ERR "\t%i ", \_\_ptr->val); \

} \

printk(KERN\_ERR "}\n"); \

} while(0)

#define ilprint\_max(int32\_t) \

do { \

print\_msg("Max number is: "); \

printk(KERN\_ERR "\t%i ", (int32\_t)); \

print\_msg("End of the program"); \

} while(0)

static struct list\_head int\_list = LIST\_HEAD\_INIT(int\_list);

static void task(void) {

int\_node\_t \*ptr;

int first=0;

int32\_t max;

list\_for\_each\_entry(ptr, &int\_list, lnode) {

if(first==0){

max=ptr->val;

first=1;

}

else{

if(ptr->val>max){

max=ptr->val;

}

}

}

ilprint\_max(max);

}

static int \_\_init list\_module\_init(void) {

int i;

int ret = 0;

print\_msg("List allocation start...\n");

for (i = 0; i < LIST\_LEN; ++i) {

int\_node\_t \*ptr = (int\_node\_t \*)kmalloc(sizeof(\*ptr), GFP\_KERNEL);

if (!ptr) {

print\_msg("Can't alloc memory\n");

ret = -ENOMEM;

goto alloc\_err;

}

get\_random\_bytes(&ptr->val, sizeof(ptr->val));

list\_add\_tail(&ptr->lnode, &int\_list);

}

print\_msg("List allocation finish\n");

ilprint(&int\_list);

task();

return 0;

alloc\_err:

ilfree(&int\_list);

return ret;

}

static void \_\_exit list\_module\_exit(void) {

ilfree(&int\_list);

print\_msg("Hasta la vista, Kernel!\n");

}

module\_init(list\_module\_init);

module\_exit(list\_module\_exit);

**Makefile:**

ifneq ($(KERNELRELEASE),)

obj-m := part\_list.o

else

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

default:

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

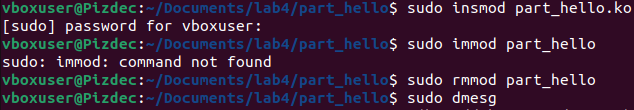
clean:

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

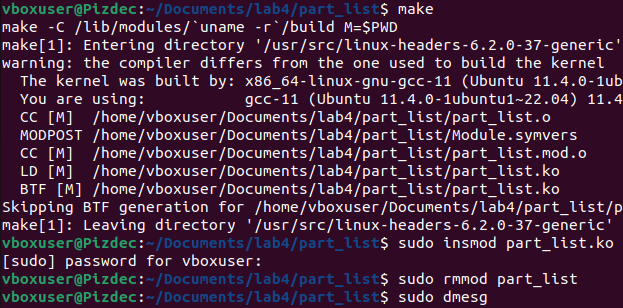
endif

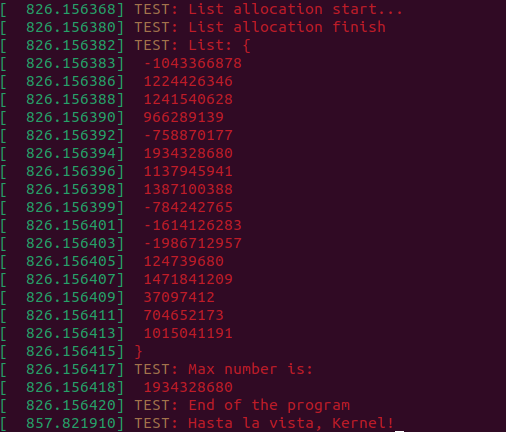
**Виконання програми:**

Частина перша:



Частина друга:





**Висновок:** під час виконання лабораторної роботи я створив два модулі ядра і завантажив їх. Код і результати виконання наведені вище.