МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

**Факультет программной инженерии и компьютерной техники**

# Лабораторная работа №2

по дисциплине

“Системы Ввода-Вывода”

## Основы написания драйверов устройств с использованием операционной системы

Вариант 4

**Студент:**

Румский А.

Группа 1.1

**Преподаватель:**

Табунщик Сергей Михайлович

Санкт-Петербург, 2025 год

## Задание

Написать драйвер символьного устройства, удовлетворяющий требованиям:

• должен создавать символьное устройство /dev/varN, где N – это номер варианта

• должен обрабатывать операции записи и чтения в соответствии с вариантом задания

При записи текста в файл символьного устройства должен осуществляться подсчет введенных цифр. Последовательность полученных результатов (количество цифр) с момента загрузки модуля ядра должна выводиться при чтении файла.

Код

Чтение

```
static ssize_t dev_read(struct file *file, char __user *buf,
size_t count, loff_t *ppos) {
    if (*ppos >= results_len) return 0;
    if (count > results_len - *ppos) count = results_len -
*ppos;
    if (copy_to_user(buf, results + *ppos, count)) return -
EFAULT;
    *ppos += count;
    return count;
}
```

Запись

```
static ssize_t dev_write(struct file *file, const char __user
*buf, size_t count, loff_t *ppos) {
    char *kbuf;
    int i;
    int local_digit_count = 0;

    kbuf = kmalloc(count + 1, GFP_KERNEL); // память
    if (!kbuf) return -ENOMEM;

    if (copy_from_user(kbuf, buf, count)) { // данные в ядро
        kfree(kbuf);
        return -EFAULT;
    }
    kbuf[count] = '\0';

    for (i = 0; i < count; i++) { // считаем цифры
        if (kbuf[i] >= '0' && kbuf[i] <= '9') {
            local_digit_count++;
        }
    }

    digit_count += local_digit_count;
    results_len += snprintf(results + results_len,
sizeof(results) - results_len, "%d\n", local_digit_count); //
сохранение

    kfree(kbuf);
    return count;
}
```

Загрузка

```
static int __init dev_init(void) {
    if (alloc_chrdev_region(&dev_number, 0, 1, DEVICE_NAME) < 0)
{
        return -1;
    }

    cdev_init(&cdev, &fops);
    if (cdev_add(&cdev, dev_number, 1) < 0) {
        unregister_chrdev_region(dev_number, 1);
        return -1;
    }

    cls = class_create(THIS_MODULE, CLASS_NAME);
    if (IS_ERR(cls)) {
        cdev_del(&cdev);
        unregister_chrdev_region(dev_number, 1);
        return PTR_ERR(cls);
    }

    device_create(cls, NULL, dev_number, NULL, DEVICE_NAME);
    printk(KERN_INFO "var4 device driver loaded\n");
    return 0;
}
```

Выгрузка

```
static void __exit dev_exit(void) {
    device_destroy(cls, dev_number);
    class_destroy(cls);
    cdev_del(&cdev);
    unregister_chrdev_region(dev_number, 1);
    printk(KERN_INFO "var4 device driver unloaded\n");
}
```

Работа


root@DESKTOP-OKUTDB3:/# insmod io_lab2/ch_drv.ko
root@DESKTOP-OKUTDB3:/# echo "hello123" > /dev/var4
root@DESKTOP-OKUTDB3:/# echo "test4567" > /dev/var4
root@DESKTOP-OKUTDB3:/# echo "no_digits_here" > /dev/var4
root@DESKTOP-OKUTDB3:/# cat /dev/var4
3
4
0
root@DESKTOP-OKUTDB3:/#root@DESKTOP-OKUTDB3:/#