

# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

## ОТЧЕТ

### ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10

дисциплина:     *Архитектура компьютера*

Студент: Безлепкина Татьяна Игоревна

Группа: НКАбд-01-25

МОСКВА

2025 г.

## **Оглавление**

1	Цель работы.....
2	Выполнение лабораторной работы.....
3	Самостоятельные задания.....
4	Вывод.....

## **1 Цель работы**

Приобретение навыков написания программ для работы с файлами.

## 2 Порядок выполнения лабораторной работы

1)

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab09$ mkdir ~/work/arch-pc/lab10
tibezlepkina1@localhost-live:~/work/arch-pc/lab09$ cd ~/work/arch-pc/lab10
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ touch lab10-1.asm readme-1.txt readme-2.txt
```

Рисунок 2.1 - создание каталога для лабораторной работы №10, переход в него и создание файлов lab10-1.asm, readme-1.txt, readme-2.txt

2) Введу в файл lab10-1.asm текст программы из листинга 10.1 (Программа записи в файл сообщения), создам исполняемый файл и проверю его работу.

```
lab10-1.asm:1: error: unable to open include file 'in_out.asm': No such file or directory
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ cp /home/tibezlepkina1/work/arch-pc/lab09/in_out.asm ~/work/arch-pc/lab10/
```

Рисунок 2.2 - копирование файла “in\_out.asm” в нужный каталог

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ nano lab10-1.asm
```

Рисунок 2.3 - открытие текстового редактора nano

```
GNU nano 8.3 lab10-1.asm
#include 'in_out.asm'
SECTION .data
filename db 'readme-1.txt', 0h ; Имя файла
msg db 'Введите строку для записи в файл: ', 0h ; Сообщение
SECTION .bss
contents resb 255 ; переменная для вводимой строки
SECTION .text
global _start
_start:
; --- Печать сообщения msg
mov eax, msg
call sprint
; ---- Запись введенной с клавиатуры строки в contents
mov ecx, contents
mov edx, 255
call sread
; --- Открытие существующего файла (sys_open)
mov ecx, 2 ; открываем для записи (2)
mov ebx, filename
mov eax, 5
int 80h
; --- Запись дескриптора файла в esi
```

Рисунок 2.4 - записать текста программы из листинга 10.1 в редактор nano

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ nasm -f elf lab10-1.asm
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ld -m elf_i386 lab10-1.o -o lab10-1
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: Helloo!
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ cat readme-1.txt
Helloo!
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ █
```

Рисунок 2.5 - компиляция, линковка, запуск, проверка содержимого файла

3) Чтобы запретить выполнения файла используем команду `chmod`

644(буквами): `rw-r--r-- lab10-1`

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ chmod 644 lab10-1
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ./lab10-1
bash: ./lab10-1: Permission denied
```

Рисунок 2.6 - С помощью команды `chmod` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение, проверка работы.

Права 644 означают `rw-r--r--`:

6 (владелец): `rw-` - чтение и запись, но НЕТ выполнения

4 (группа): `r--` - только чтение

4 (остальные): `r--` - только чтение

Причина ошибки: В Linux файл может быть выполнен только если у него установлен бит выполнения (`x`). Команда `chmod 644` убирает этот бит у всех категорий пользователей. Даже если файл содержит корректный машинный код в формате ELF, ядро ОС отказывается его выполнять без явного разрешения.

4) Добавление прав выполнения к исходному коду. С помощью команды `chmod +x lab10-1` добавляем права для выполнения, далее проверяем текущие права, запускаем файл и проверяем его работу.

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ chmod +x lab10-1
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ls -l lab10-1
-rwxr-xr-x. 1 tibezlepkina1 tibezlepkina1 5068 Dec 13 22:33 lab10-1
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: Hello!
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ cat readme-1.txt
Hello!
```

Рисунок 2.7 - добавление прав для выполнения, проверка прав, запуск, получение детальной информации о файле, проверка содержимого файла

Объяснение изменений:

Команда `chmod +x` добавляет бит выполнения:

Для владельца: `rw-` → `rwX` (добавлен `x`)

Для группы: `r--` → `r-X` (добавлен `x`)

Для остальных: `r--` → `r-X` (добавлен `x`)

После восстановления бита выполнения командой `chmod +x lab10-1` программа снова стала работоспособной. Это подтверждает, что в Linux наличие бита выполнения является необходимым и достаточным условием для запуска исполняемых файлов корректного формата.

5) Устанавливаем права для readme-1.txt (символьный вид) и проверяем результат:

```
tibezlepkin1@localhost-live:~/work/arch-pc/lab10$ chmod u=x,g=rw,o=w readme-1.txt
tibezlepkin1@localhost-live:~/work/arch-pc/lab10$ ls -l readme-1.txt
---xrw--w-. 1 tibezlepkin1 tibezlepkin1 8 Dec 13 23:04 readme-1.txt
```

Рисунок 2.8 - установка прав и получение детальной информации о файле

Объяснение команды: u=x - для пользователя (user): только выполнение (x), g=rw - для группы (group): чтение и запись (rw), o=w - для остальных (others): только запись (w).

Устанавливаем права для readme-2.txt (из двоичного вида):

```
tibezlepkin1@localhost-live:~/work/arch-pc/lab10$ chmod 137 readme-2.txt
tibezlepkin1@localhost-live:~/work/arch-pc/lab10$ ls -l readme-2.txt
--x-wxrw-. 1 tibezlepkin1 tibezlepkin1 0 Dec 13 22:10 readme-2.txt
```

Рисунок 2.9 - установка прав, получение детальной информации о файле

Преобразование двоичного представления прав 001 011 111 в восьмеричный код 137 необходимо, поскольку команда chmod принимает аргументы в восьмеричной системе счисления. Каждая двоичная триада преобразуется по правилу: чтение(r)=4, запись(w)=2, выполнение(x)=1. Суммирование установленных битов даёт соответствующую восьмеричную цифру. Такой подход обеспечивает компактное и однозначное представление прав доступа.

Преобразуем двоичный вид в восьмеричный:  $001_2 = 1_8$  (только выполнение),  $011_2 = 3_8$  (запись и выполнение),  $111_2 = 7_8$  (чтение, запись, выполнение).

### 3 Самостоятельные задания

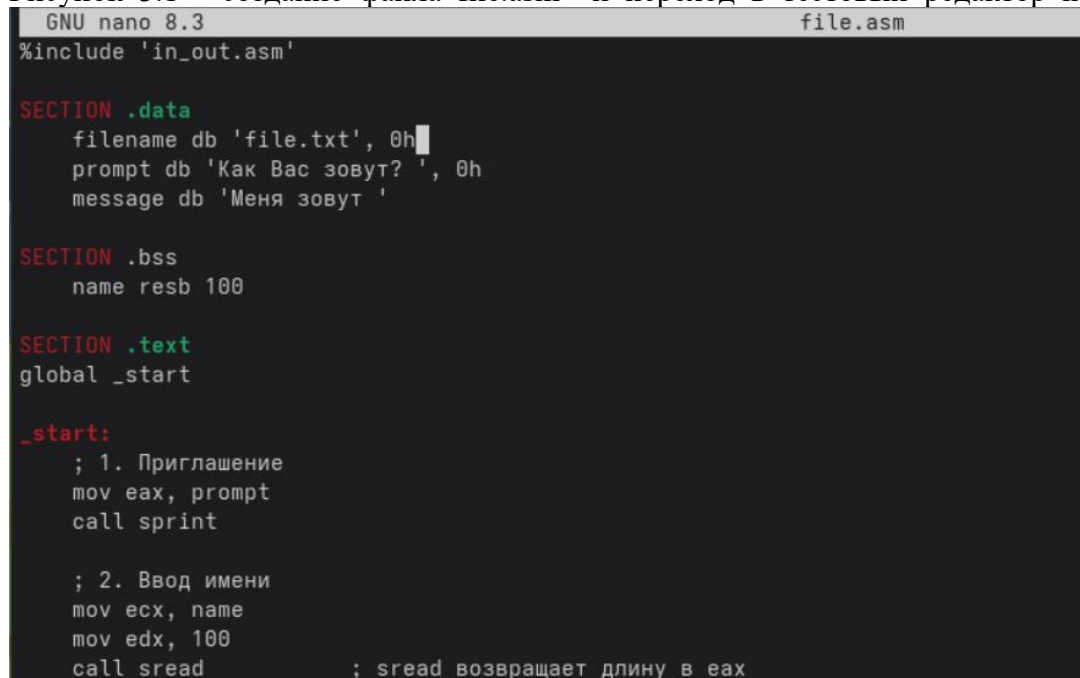
Напишу программу работающую по следующему алгоритму:

- Вывод приглашения “Как Вас зовут?”
- ввести с клавиатуры свои фамилию и имя
- создать файл с именем name.txt
- записать в файл сообщение “Меня зовут”
- дописать в файл строку введенную с клавиатуры
- закрыть файл

Создам исполняемый файл(file.asm)и проверю его работу.Создам файл (file.txt).Проверю наличие файлаи его содержимое с помощью команд ls и cat.

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ touch file.asm
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ nano file.asm
```

Рисунок 3.1 - создание файла file.asm и переход в тестовый редактор nano



```
GNU nano 8.3 file.asm
#include 'in_out.asm'

SECTION .data
    filename db 'file.txt', 0h
    prompt db 'Как Вас зовут? ', 0h
    message db 'Меня зовут '

SECTION .bss
    name resb 100

SECTION .text
global _start

_start:
    ; 1. Приглашение
    mov eax, prompt
    call sprint

    ; 2. Ввод имени
    mov ecx, name
    mov edx, 100
    call sread          ; sread возвращает длину в eax
```

Рисунок 3.2 - запись текста программы в редактор

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ touch file.txt
```

Рисунок 3.3 - создание файла file.txt

```
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ ls -l file.txt
-rw-r--r--. 1 tibezlepkina1 tibezlepkina1 22 Dec 14 18:39 file.txt
tibezlepkina1@localhost-live:~/work/arch-pc/lab10$ cat file.txt
My name is Tanya
```

Рисунок 3.4 - получение детальной информации о файле,проверка содержимого файла

#### **4 Вывод**

В ходе лабораторной работы были приобретены навыки написания программ для работы с файлами.