

# **Отчёт по лабораторной работе №10**

**Дисциплина: Архитектура Компьютера**

Курилко-Рюмин Е.М

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>16</b>
	<b>Список литературы</b>	<b>17</b>

## Список иллюстраций

4.1	Создание исполняемого файла и его работа . . . . .	9
4.2	Запрет на исполнение файла . . . . .	10
4.3	Добавление прав на исполнение . . . . .	11
4.4	Предоставление прав доступа в символьном виде . . . . .	12
4.5	Предоставление прав доступа в числовом виде . . . . .	12
4.6	Создание и редактирование файла . . . . .	13
4.7	Редактирование файла . . . . .	13

# 1 Цель работы

Целью данной работы является приобретение практического опыта в написании программ для работы с файлами.

## 2 Задание

1. Общее ознакомление с программами для работы с файлами.
2. Работа с файлами средствами NASM.
3. Выполнение заданий для самостоятельной работы

### 3 Теоретическое введение

ОС GNU/Linux является многопользовательской операционной системой. И для обеспечения защиты данных одного пользователя от действий других пользователей существуют специальные механизмы разграничения доступа к файлам. Кроме ограничения доступа, данный механизм позволяет разрешить другим пользователям доступ данным для совместной работы. Права доступа определяют набор действий (чтение, запись, выполнение), разрешённых для выполнения пользователям системы над файлами. Для каждого файла пользователь может входить в одну из трех групп:

владелец, член группы владельца, все остальные.

Для каждой из этих групп может быть установлен свой набор прав доступа. Владелец файла является его создатель. Набор прав доступа задается тройками битов и состоит из прав на чтение, запись и исполнение файла. В символьном представлении он имеет вид строк `gwx`, где вместо любого символа может стоять дефис. Буква означает наличие права (установлен в единицу второй бит триады `g` — чтение, первый бит `w` — запись, нулевой бит `x` — исполнение), а дефис означает отсутствие права (нулевое значение соответствующего бита). Также права доступа могут быть представлены как восьмеричное число. Так, права доступа `rw` (чтение и запись, без исполнения) понимаются как три двоичные цифры `110` или как восьмеричная цифра `6`. Тип файла определяется первой позицией, это может быть: каталог — `d`, обычный файл — дефис (`-`) или символьная ссылка на другой файл — `l`. Следующие 3 набора по 3 символа определяют конкретные права для конкретных групп: `r` — разрешено чтение файла, `w` — разрешена запись в файл; `x`

— разрешено исполнение файл и дефис (-) — право не дано.

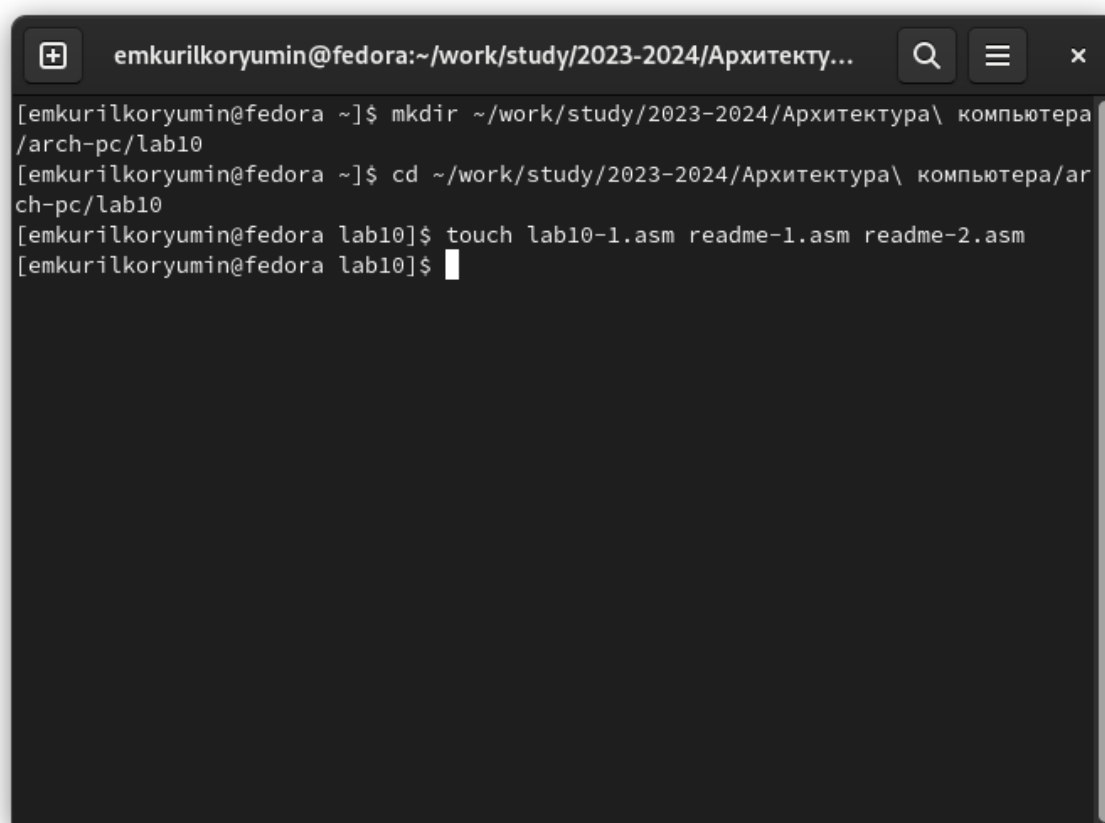
Для изменения прав доступа служит команда `chmod`, которая понимает как символьное, так и числовое указание прав. Для того чтобы назначить файлу `/home/debugger/README` права `rw-r`, то есть разрешить владельцу чтение и запись, группе только чтение, остальным пользователям — ничего. В символьном представлении есть возможность явно указывать какой группе какие права необходимо добавить, отнять или присвоить. В операционной системе Linux существуют различные методы управления файлами, например, такие как создание и открытие файла, только для чтения или для чтения и записи, добавления в существующий файл, закрытия и удаления файла, предоставление прав доступа. Обработка файлов в операционной системе Linux осуществляется за счет использования определенных системных вызовов. Для корректной работы и доступа к файлу при его открытии или создании, файлу присваивается уникальный номер (16-битное целое число) – дескриптор файла. Для создания и открытия файла служит системный вызов `sys_creat`, который использует следующие аргументы: права доступа к файлу в регистре `ECX`, имя файла в `EBX` и номер системного вызова `sys_creat` (8) в `EAX`. Для открытия существующего файла служит системный вызов `sys_open`, который использует следующие аргументы: права доступа к файлу в регистре `EDX`, режим доступа к файлу в регистр `ECX`, имя файла в `EBX` и номер системного вызова `sys_open` (5) в `EAX`. Для записи в файл служит системный вызов `sys_write`, который использует следующие аргументы: количество байтов для записи в регистре `EDX`, строку содержимого для записи `ECX`, файловый дескриптор в `EBX` и номер системного вызова `sys_write` (4) в `EAX`.

Системный вызов возвращает фактическое количество записанных байтов в регистр `EAX`. В случае ошибки, код ошибки также будет находиться в регистре `EAX`. прежде чем записывать в файл, его необходимо создать или открыть, что позволит получить дескриптор файла.

## 4 Выполнение лабораторной работы

### 4.1) Работа с файлами средствами NASM.

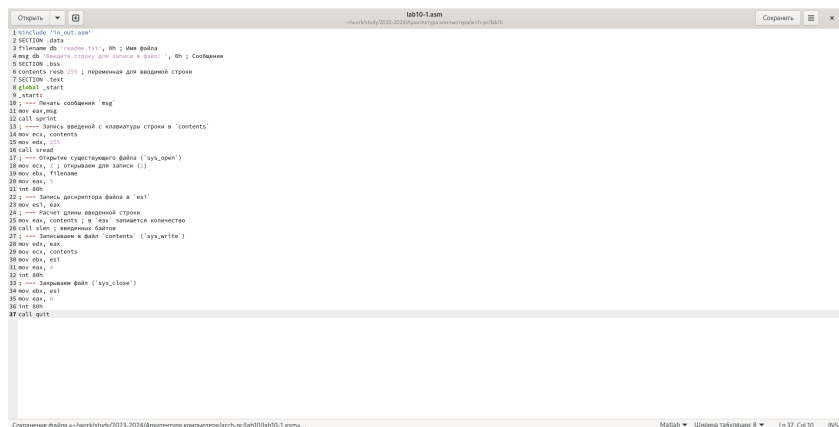
С помощью утилиты `mkdir` создаю директорию `lab10` для выполнения соответствующей лабораторной работы. Перехожу в созданный каталог с помощью утилиты `cd`. С помощью `touch` создаю файл `lab10-1.asm`. Открываю созданный файл `lab10-1.asm`, вставляю в него следующую программу: (рис.1).



```
emkurilkoryumin@fedora:~/work/study/2023-2024/Архитектура\ компьютера
[emkurilkoryumin@fedora ~]$ mkdir ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc/lab10
[emkurilkoryumin@fedora ~]$ cd ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc/lab10
[emkurilkoryumin@fedora lab10]$ touch lab10-1.asm readme-1.asm readme-2.asm
[emkurilkoryumin@fedora lab10]$
```

(image/1.pn

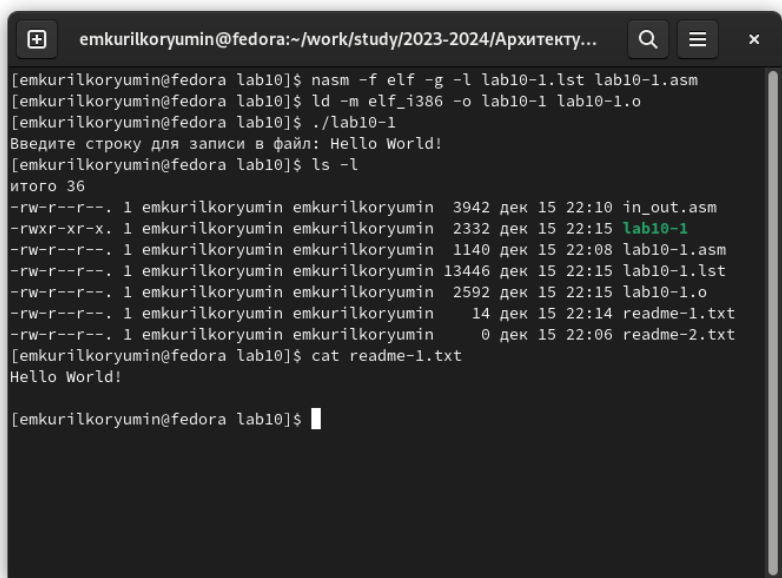




```
1 include "in_out.asm"
2 section .data
3 filename db "readme.txt", 0h ; имя файла
4 msg db "Введите строку для записи в файл: ", 0h ; Сообщение
5 section .bss
6 contents resb 255 ; переменная для входной строки
7 section .text
8 global _start
9 _start:
10 ; -- Чтение сообщения "msg"
11 mov esi, msg
12 call get_int
13 ; -- Запись введенной с клавиатуры строки в "contents"
14 mov ecx, contents
15 mov edi, 0
16 call read
17 ; -- Открытие существующего файла ("sys_open")
18 mov ecx, 0 ; открытие для записи (2)
19 mov ebx, filename
20 mov esi, 0
21 int 0x80
22 ; -- Запись дескриптора файла в "eax"
23 mov esi, eax
24 ; -- Расчет длины введенной строки
25 mov ecx, contents ; в "ecx" находится количество
26 call strlen ; возвращаемый байтов
27 ; -- Записываем в файл "contents" ("sys_write")
28 mov ebx, eax
29 mov esi, contents
30 mov edi, 0
31 mov ecx, 0
32 int 0x80
33 ; -- Закрываем файл ("sys_close")
34 mov ebx, esi
35 mov ecx, 0
36 int 0x80
37 call quit
```

width=70%}

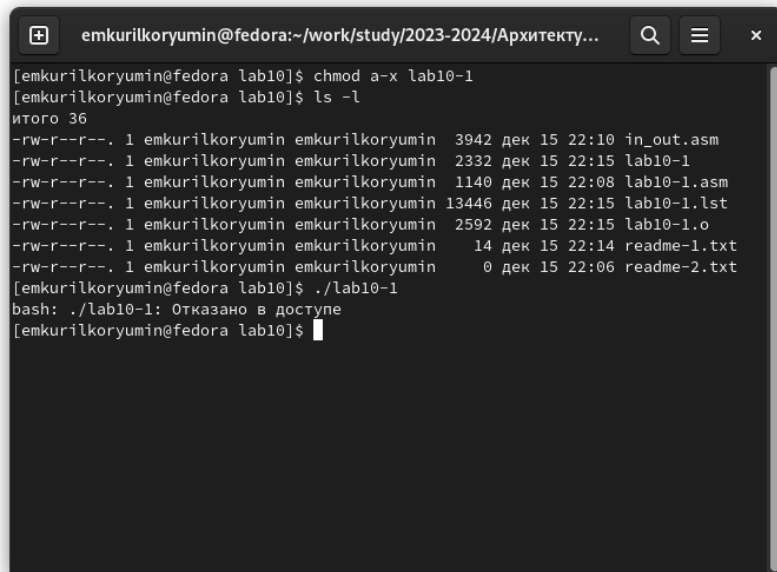
Создаю исполняемый файл и проверяю его работу. (рис.2).



```
[emkurilkoryumin@fedora lab10]$ nasm -f elf -g -l lab10-1.lst lab10-1.asm
[emkurilkoryumin@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[emkurilkoryumin@fedora lab10]$ ./lab10-1
Введите строку для записи в файл: Hello World!
[emkurilkoryumin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 3942 дек 15 22:10 in_out.asm
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 2332 дек 15 22:15 lab10-1
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 1140 дек 15 22:08 lab10-1.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 13446 дек 15 22:15 lab10-1.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2592 дек 15 22:15 lab10-1.o
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 14 дек 15 22:14 readme-1.txt
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 0 дек 15 22:06 readme-2.txt
[emkurilkoryumin@fedora lab10]$ cat readme-1.txt
Hello World!
[emkurilkoryumin@fedora lab10]$
```

Рис. 4.1: Создание исполняемого файла и его работа

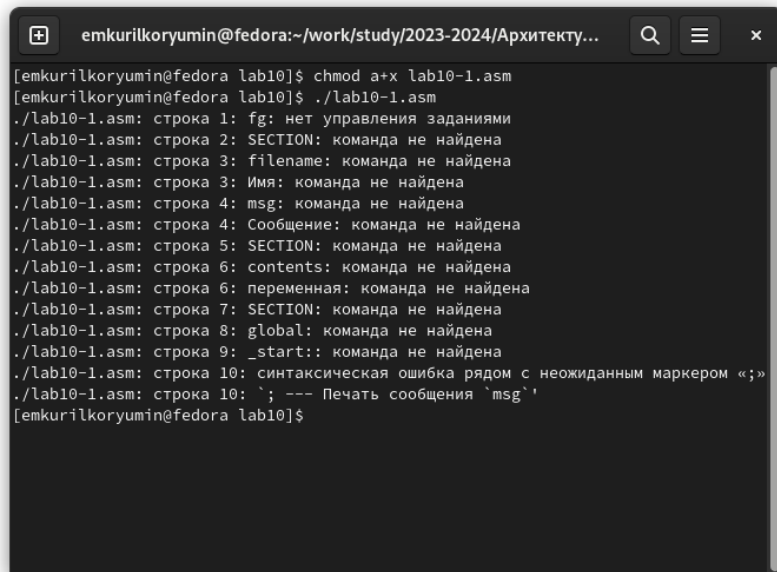
Далее, с помощью команды `chmod a-x` изменяю права доступа к исполняемому файлу, запретив его выполнение, что и проверяю соответственно. (рис.3).

A terminal window titled 'emkurilkoryumin@fedora:~/work/study/2023-2024/Архитекту...' with search and menu icons. The terminal shows the following commands and output:

```
[emkurilkoryumin@fedora lab10]$ chmod a-x lab10-1
[emkurilkoryumin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 3942 дек 15 22:10 in_out.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2332 дек 15 22:15 lab10-1
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 1140 дек 15 22:08 lab10-1.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 13446 дек 15 22:15 lab10-1.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2592 дек 15 22:15 lab10-1.o
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 14 дек 15 22:14 readme-1.txt
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 0 дек 15 22:06 readme-2.txt
[emkurilkoryumin@fedora lab10]$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
[emkurilkoryumin@fedora lab10]$
```

Рис. 4.2: Запрет на исполнение файла

Файл не исполняется, все верно. Следом я с помощью команды `chmod a+x` добавляю права на исполнение самого файла `lab10-1.asm` и пытаюсь выполнить его. (рис.4).



```
emkurilkoryumin@fedora:~/work/study/2023-2024/Архитекту...
[emkurilkoryumin@fedora lab10]$ chmod a+x lab10-1.asm
[emkurilkoryumin@fedora lab10]$ ./lab10-1.asm
./lab10-1.asm: строка 1: fg: нет управления заданиями
./lab10-1.asm: строка 2: SECTION: команда не найдена
./lab10-1.asm: строка 3: filename: команда не найдена
./lab10-1.asm: строка 3: Имя: команда не найдена
./lab10-1.asm: строка 4: msg: команда не найдена
./lab10-1.asm: строка 4: Сообщение: команда не найдена
./lab10-1.asm: строка 5: SECTION: команда не найдена
./lab10-1.asm: строка 6: contents: команда не найдена
./lab10-1.asm: строка 6: переменная: команда не найдена
./lab10-1.asm: строка 7: SECTION: команда не найдена
./lab10-1.asm: строка 8: global: команда не найдена
./lab10-1.asm: строка 9: _start:: команда не найдена
./lab10-1.asm: строка 10: синтаксическая ошибка рядом с неожиданным маркером «;»
./lab10-1.asm: строка 10: `;` --- Печать сообщения `msg`
[emkurilkoryumin@fedora lab10]$
```

Рис. 4.3: Добавление прав на исполнение

Текстовый файл начинает работу, но не исполняется, т.к. не содержит в себе команд для терминала. Затем в соответствии со своим 4-м вариантом изменяю права доступа к файлу readme-1.txt согласно таблице. Потом проверяю правильность выполнение команды с помощью `ls -l`. (рис.5).

```
emkurilkoryumin@fedora:~/work/study/2023-2024/Архитекту...
[emkurilkoryumin@fedora lab10]$ cd
[emkurilkoryumin@fedora ~]$ cd ~/work/study/2023-2024/Архитектура\ компьютера/ar
ch-pc/lab10
[emkurilkoryumin@fedora lab10]$ chmod u=w readme-1.txt
[emkurilkoryumin@fedora lab10]$ chmod g= readme-1.txt
[emkurilkoryumin@fedora lab10]$ chmod o=w readme-1.txt
[emkurilkoryumin@fedora lab10]$ ls -l
итого 72
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 3942 дек 15 22:10 in_out.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2332 дек 15 22:15 lab10-1
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 1140 дек 15 22:08 lab10-1.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 13446 дек 15 22:15 lab10-1.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2592 дек 15 22:15 lab10-1.o
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 2364 дек 15 22:29 lab10-2
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 611 дек 15 22:28 lab10-2.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 14537 дек 15 22:28 lab10-2.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2656 дек 15 22:28 lab10-2.o
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 54 дек 15 22:29 name.txt
--w-----w. 1 emkurilkoryumin emkurilkoryumin 14 дек 15 22:14 readme-1.txt
-r-xr-xr-x. 1 emkurilkoryumin emkurilkoryumin 0 дек 15 22:06 readme-2.txt
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 42 дек 15 22:26 text.txt
[emkurilkoryumin@fedora lab10]$
```

Рис. 4.4: Предоставление прав доступа в символьном виде

Аналогично поступаю с файлом readme-2.txt, только уже в числовом виде.  
(рис.6).

```
[emkurilkoryumin@fedora lab10]$ chmod 110 readme-2.txt #001 011 110
[emkurilkoryumin@fedora lab10]$ ls -l
итого 72
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 3942 дек 15 22:10 in_out.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2332 дек 15 22:15 lab10-1
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 1140 дек 15 22:08 lab10-1.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 13446 дек 15 22:15 lab10-1.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2592 дек 15 22:15 lab10-1.o
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 2364 дек 15 22:29 lab10-2
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 611 дек 15 22:28 lab10-2.asm
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 14537 дек 15 22:28 lab10-2.lst
-rw-r--r--. 1 emkurilkoryumin emkurilkoryumin 2656 дек 15 22:28 lab10-2.o
-rwxr-xr-x. 1 emkurilkoryumin emkurilkoryumin 54 дек 15 22:29 name.txt
--w-----w. 1 emkurilkoryumin emkurilkoryumin 14 дек 15 22:14 readme-1.txt
---x---x---. 1 emkurilkoryumin emkurilkoryumin 0 дек 15 22:06 readme-2.txt
```

Рис. 4.5: Предоставление прав доступа в числовом виде

#### 4.2) Выполнение заданий для самостоятельной работы.

Создаю файлы lab10-2.asm и name.txt для корректной работы программы записи моего имени, введенного с клавиатуры, в соответствующий файл. Далее пишу непосредственно сам код: (рис.7).

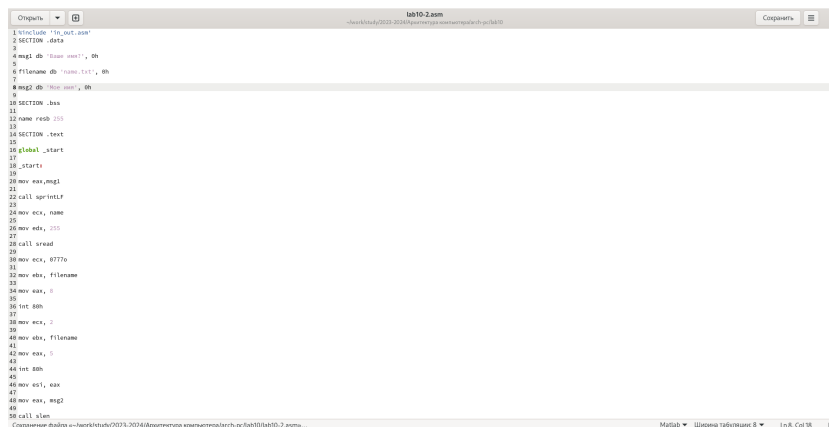


Рис. 4.6: Создание и редактирование файла

Создаю исполняемый файл, проверяю его работу, далее ввожу необходимые команды, и убеждаюсь в правильности работы программы. (рис.8).

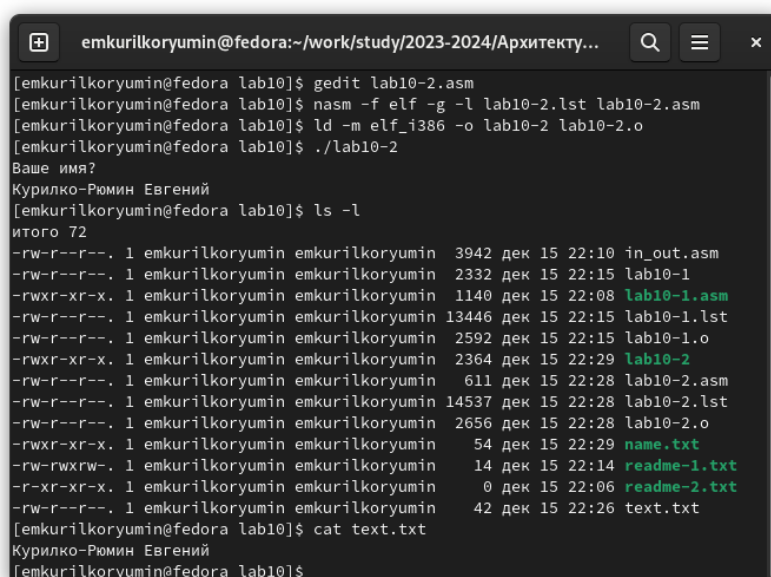


Рис. 4.7: Редактирование файла

Листинг 4.1 - Программа, приглашающая написать имя, и записывающая его в файл.

““%include ‘in\_out.asm’ SECTION .data

```

msg1 db 'Ваше имя?', 0h
filename db 'name.txt', 0h
msg2 db 'Мое имя', 0h
SECTION .bss
name resb 255
SECTION .text
global _start
_start:
mov eax,msg1
call sprintLF
mov ecx, name
mov edx, 255
call sread
mov ecx, 0777o
mov ebx, filename
mov eax, 8
int 80h
mov ecx, 2
mov ebx, filename
mov eax, 5
int 80h
mov esi, eax
mov eax, msg2
call slen
mov edx, eax
mov ecx, msg2
mov ebx, esi
mov eax, 4
int 80h

```

```
mov eax, name  
call slen  
mov edx, eax  
mov ecx, name  
mov ebx, esi  
mov eax, 4  
int 80h  
mov ebx, esi  
mov eax, 6  
int 80h  
call quit ““
```

## 5 Выводы

При выполнении лабораторной работы я приобрел практический опыт в написании программ с использованием циклов и обработкой аргументов командной строки.



# Список литературы

Архитектура компьютера и ЭВМ