answer to labrotary work 10

Discipline: Computer Architecture

Ерфан Хосейнабади

Content

List of illustrations

List of Tables

1 Work Goal

Acquiring skills in writing programs for working with files.

2 Assignment

1. Creating files in programs.
2. Changing file permissions for different user groups.
3. Completing independent assignments based on the materials of the laboratory work.

3 Theoretical Introduction

The GNU/Linux OS is a multi-user operating system. To protect the data of one user from the actions of other users, special mechanisms for access control to files exist. Besides access restriction, this mechanism allows other users access to data for collaborative work.

4 Performing the Laboratory Work

I create a directory for the programs of laboratory work No. 10 (Fig. -fig. 4.1).

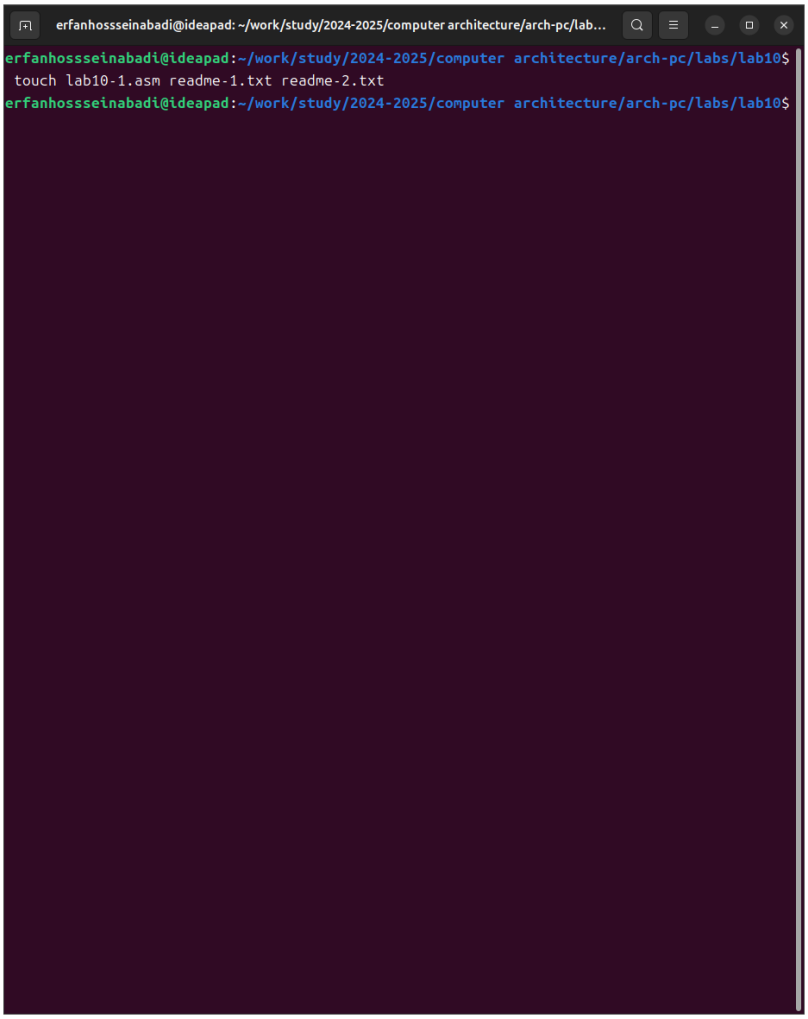

Fig. 4.1: Creating a working directory

I enter the program from the first listing into the created file (Fig. -fig. 4.2).

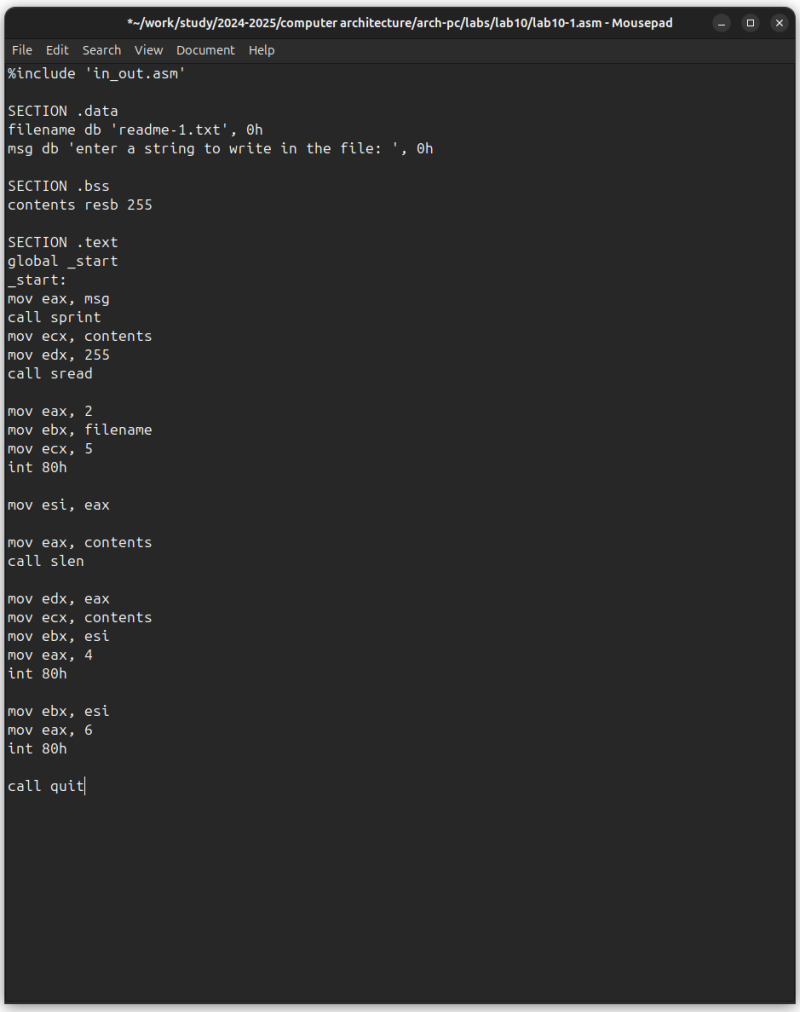

Fig. 4.2: Program of the first listing

I run the program; it prompts for a string input, after which it creates a text file with the string entered by the user (Fig. -fig. 4.3).

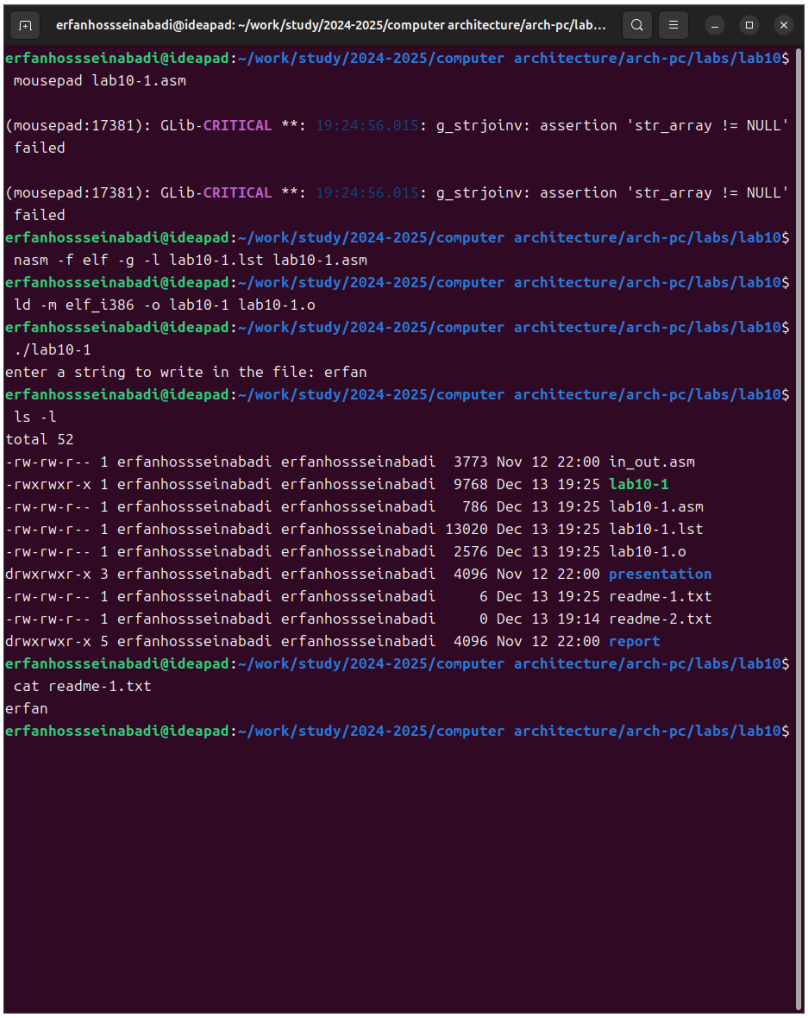

Fig. 4.3: Running the program of the first listing

I change the owner's permissions, prohibiting the execution of the file, after which the system refuses to execute the file because I, the owner, have prohibited myself from executing the program (Fig. -fig. 4.4).

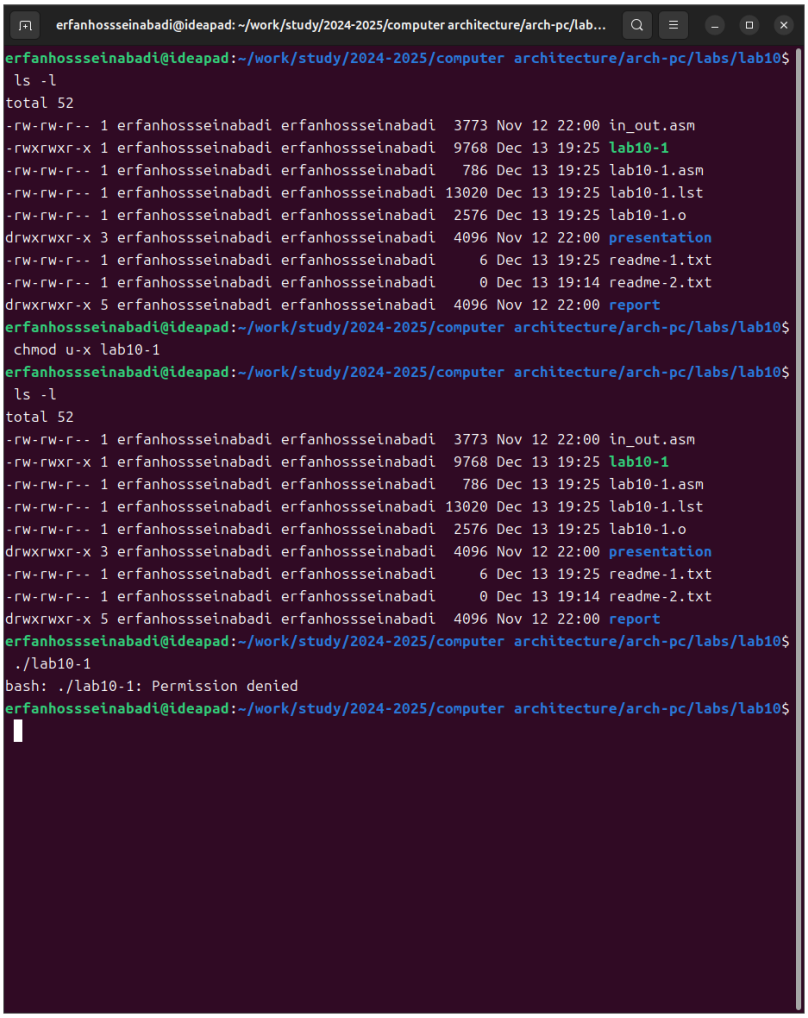

Fig. 4.4: Demonstration of the chmod command

I add the execution permission to the owner for the source program file; the executable text file interprets each line as a command. Since none of the lines are bash commands, the program does absolutely nothing (Fig. -fig. 4.5).

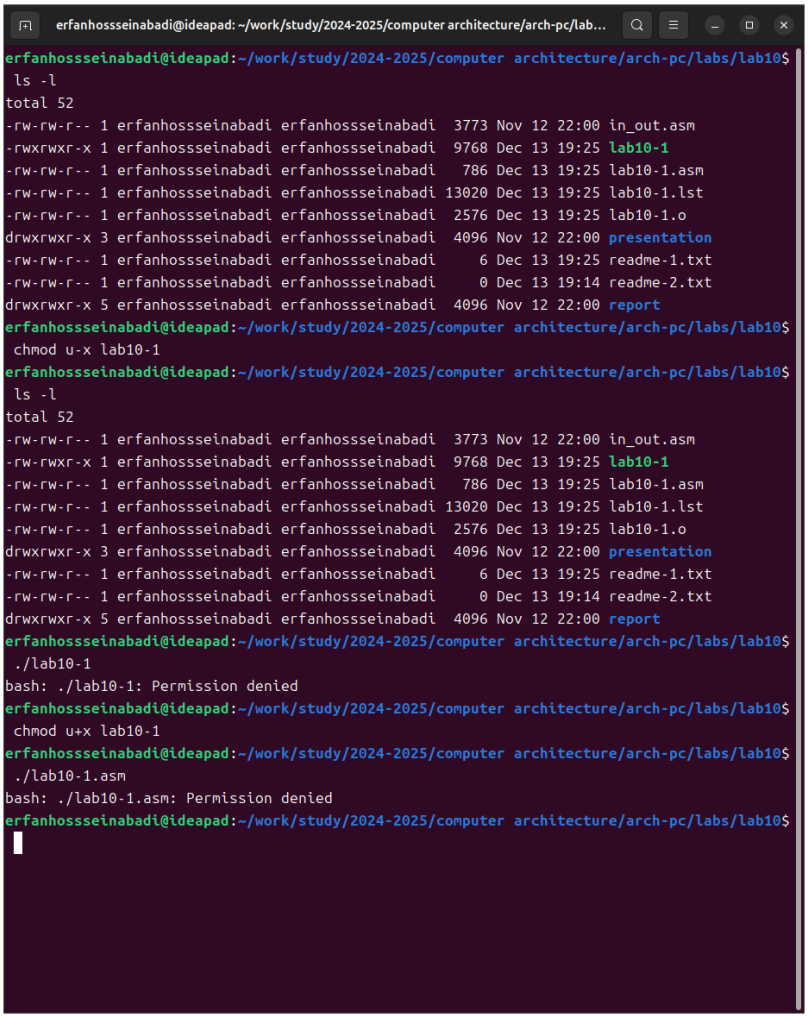

Fig. 4.5: Running the text file

According to my variant, I need to set the corresponding permissions to the text files created at the beginning of the laboratory work:

1. In symbolic form for the 1st readme file –x -w- -w-

2. In binary system for the 2nd readme file 001 011 101

I convert the group of bits to the octal system; I adjust the symbolic notation to the syntax and obtain the necessary arguments for chmod (Fig. -fig. 4.6).

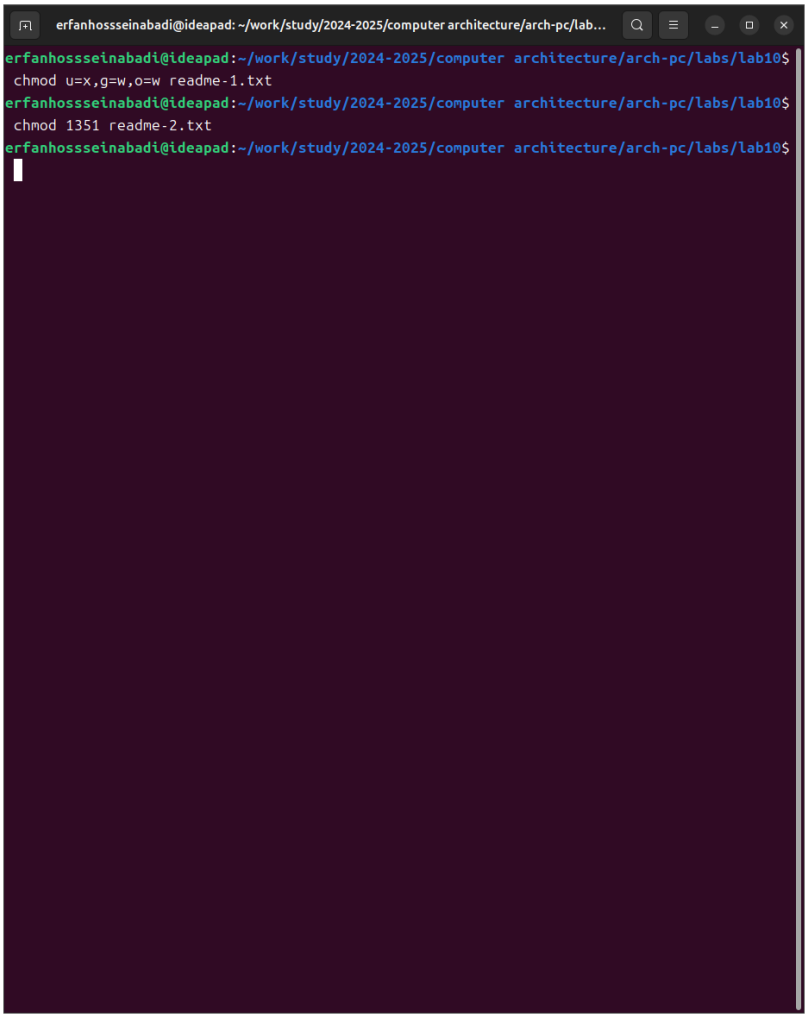

Fig. 4.6: Symbolic and numerical notations

4.1 Independent Work Assignment

I write a program, transliterate and compile it. The program should display a prompt, request input from the keyboard, and create a text file with the string specified in the program and the user's input.

I run the program, check the presence and content of the created text file; the program works correctly (Fig. -fig. 4.7).

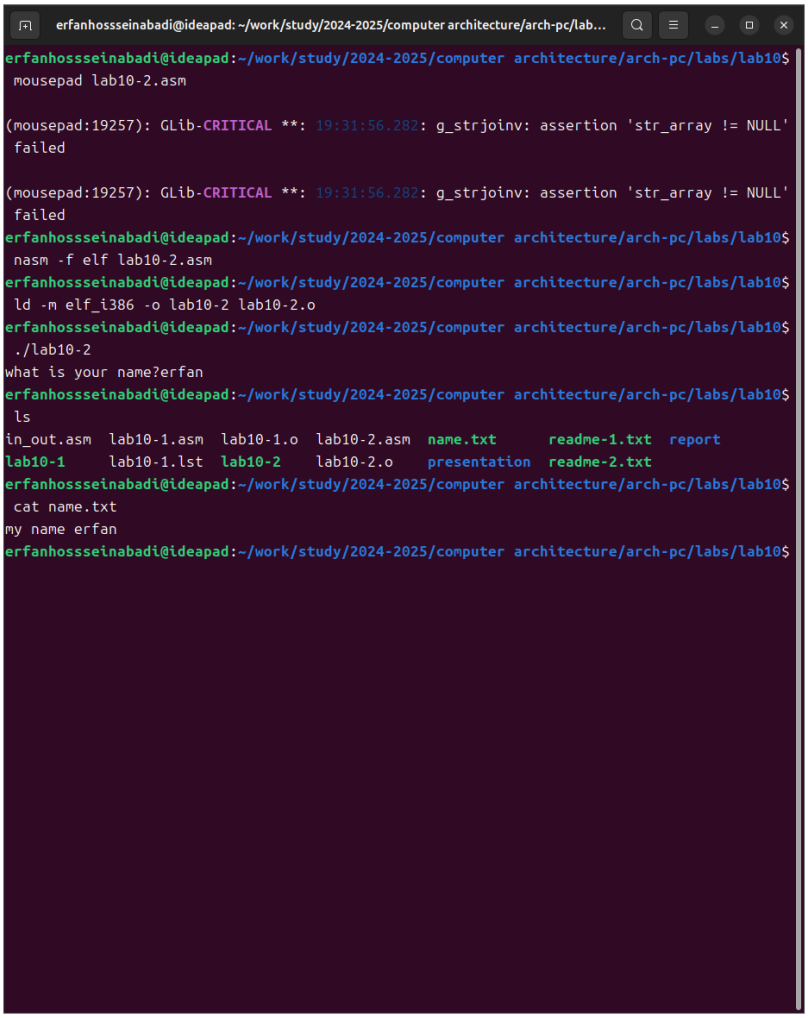

Fig. 4.7: Demonstration of the program's operation

Program code:

```
%include 'in_out.asm'

SECTION .data

filename db 'name.txt', 0

prompt db 'what is your name?', 0
```

```
intro db 'my name is ', 0

SECTION .bss

name resb 255

SECTION .text

global _start

_start:

mov eax, prompt

call sprint

mov ecx, name

mov edx, 255

call sread

mov eax, 8

mov ebx, filename

mov ecx, 0744o
```

```
int 80h

mov esi, eax

mov eax, intro

call slen

mov edx, eax

mov ecx, intro

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 Conclusions

In the process of performing the laboratory work, I acquired skills in writing programs for working with files and learned how to edit file permissions.

6 References

1. Course on TUIS

2. Programming in NASM Assembler Language Stolyarov A. V.