

Отчёт по лабораторной работе №10

Дисциплина: Архитектура Компьютера

Егор Витальевич Кузьмин

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	8
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Работа с директориями и создание, редактирование файла	8
4.2	Создание исполняемого файла и его работа	9
4.3	Запрет на исполнение файла	9
4.4	Добавление прав на исполнение	10
4.5	Предоставление прав доступа в символьном виде	10
4.6	Предоставление прав доступа в числовом виде	11
4.7	Создание и редактирование файла	11
4.8	Редактирование файла	12

1 Цель работы

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

2 Задание

- 0. Общее ознакомление с программами для работы с файлами.
- 1. Работа с файлами средствами NASM.
- 2. 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.

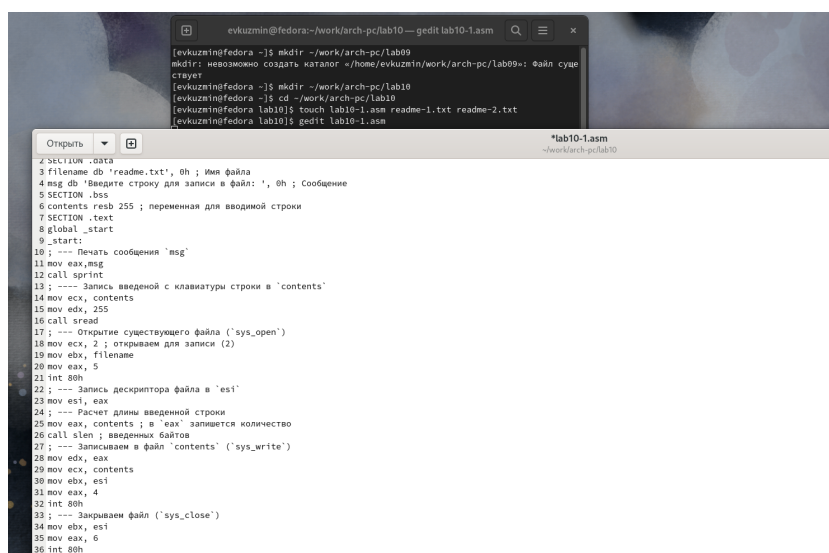
Для изменения прав доступа служит команда `chmod`, которая понимает как символьное, так и числовое указание прав. Для того чтобы назначить файлу `/home/debugger/README` права `rw-r`, то есть разрешить владельцу чтение и запись, группе только чтение, остальным пользователям — ничего. В символьном представлении 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 `lab8-1.asm`, вставляю в него следующую программу: (рис. 4.1).



The image shows a terminal window and a text editor. The terminal window displays the following commands and output:

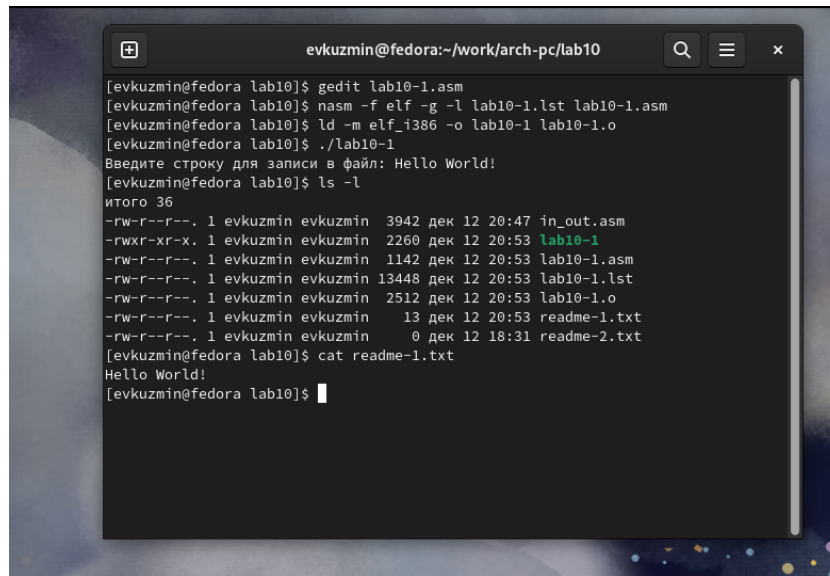
```
evkuzmin@fedora:~/work/arch-pc/lab10$ mkdir -p /work/arch-pc/lab10
mkdir: невозможно создать каталог «/home/evkuzmin/work/arch-pc/lab10»: файл уже
существует
evkuzmin@fedora:~/work/arch-pc/lab10$ cd /work/arch-pc/lab10
evkuzmin@fedora:~/work/arch-pc/lab10$ touch lab10-1.asm
evkuzmin@fedora:~/work/arch-pc/lab10$ gedit lab10-1.asm
```

The text editor shows the following assembly code:

```
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 вводной строки
7 SECTION .text
8 global _start
9 _start:
10 ; --- Печать сообщения 'msg'
11 mov eax, msg
12 call sprintf
13 ; ---- Запись введенной с клавиатуры строки в 'contents'
14 mov ecx, contents
15 mov edx, 255
16 call fread
17 ; --- Открытие существующего файла ('sys_open')
18 mov ecx, 2 ; открываем для записи (2)
19 mov ebx, filename
20 mov eax, 5
21 int 80h
22 ; --- Запись дескриптора файла в 'esi'
23 mov esi, eax
24 ; --- Расчет длины введенной строки
25 mov ecx, contents ; в 'eax' запишется количество
26 call strlen ; введенных байтов
27 ; --- Записываем в файл 'contents' ('sys_write')
28 mov edx, eax
29 mov ecx, contents
30 mov ebx, esi
31 mov eax, 4
32 int 80h
33 ; --- Закрываем файл ('sys_close')
34 mov ebx, esi
35 mov eax, 6
36 int 80h
37 ; ---
```

Рис. 4.1: Работа с директориями и создание, редактирование файла

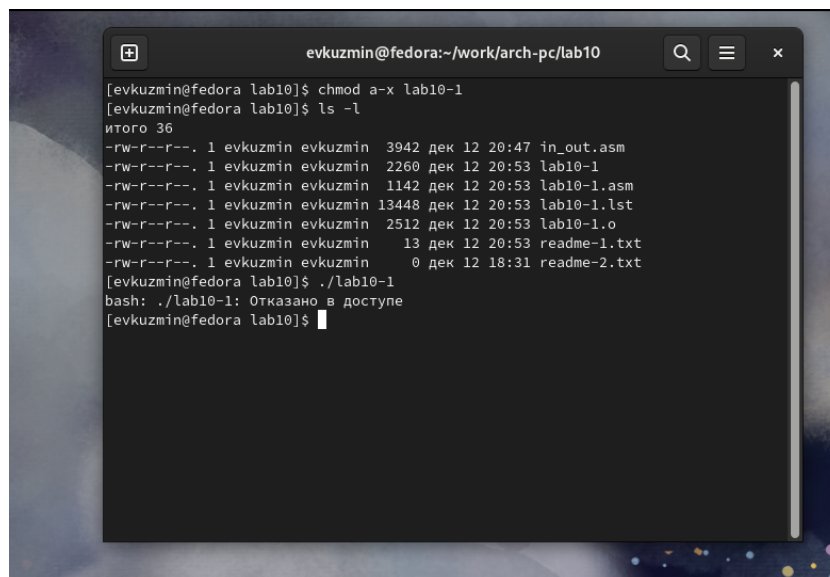
Создаю исполняемый файл и проверяю его работу. (рис. 4.2).



```
evkuzmin@fedora:~/work/arch-pc/lab10
[evkuzmin@fedora lab10]$ gedit lab10-1.asm
[evkuzmin@fedora lab10]$ nasm -f elf -g -l lab10-1.lst lab10-1.asm
[evkuzmin@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[evkuzmin@fedora lab10]$ ./lab10-1
Введите строку для записи в файл: Hello World!
[evkuzmin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 evkuzmin evkuzmin 3942 дек 12 20:47 in_out.asm
-rwxr-xr-x. 1 evkuzmin evkuzmin 2260 дек 12 20:53 lab10-1
-rw-r--r--. 1 evkuzmin evkuzmin 1142 дек 12 20:53 lab10-1.asm
-rw-r--r--. 1 evkuzmin evkuzmin 13448 дек 12 20:53 lab10-1.lst
-rw-r--r--. 1 evkuzmin evkuzmin 2512 дек 12 20:53 lab10-1.o
-rw-r--r--. 1 evkuzmin evkuzmin 13 дек 12 20:53 readme-1.txt
-rw-r--r--. 1 evkuzmin evkuzmin 0 дек 12 18:31 readme-2.txt
[evkuzmin@fedora lab10]$ cat readme-1.txt
Hello World!
[evkuzmin@fedora lab10]$
```

Рис. 4.2: 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. 4.3).

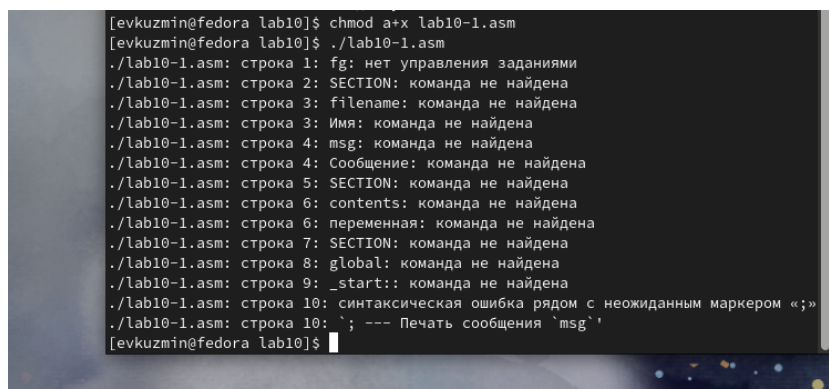


```
evkuzmin@fedora:~/work/arch-pc/lab10
[evkuzmin@fedora lab10]$ chmod a-x lab10-1
[evkuzmin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 evkuzmin evkuzmin 3942 дек 12 20:47 in_out.asm
-rw-r--r--. 1 evkuzmin evkuzmin 2260 дек 12 20:53 lab10-1
-rw-r--r--. 1 evkuzmin evkuzmin 1142 дек 12 20:53 lab10-1.asm
-rw-r--r--. 1 evkuzmin evkuzmin 13448 дек 12 20:53 lab10-1.lst
-rw-r--r--. 1 evkuzmin evkuzmin 2512 дек 12 20:53 lab10-1.o
-rw-r--r--. 1 evkuzmin evkuzmin 13 дек 12 20:53 readme-1.txt
-rw-r--r--. 1 evkuzmin evkuzmin 0 дек 12 18:31 readme-2.txt
[evkuzmin@fedora lab10]$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
[evkuzmin@fedora lab10]$
```

Рис. 4.3: 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

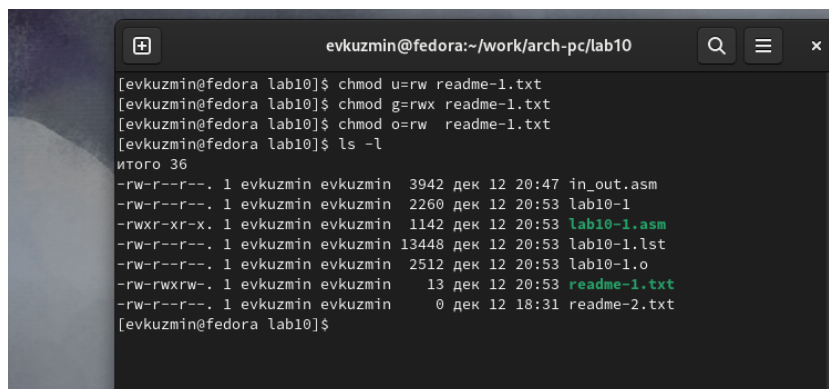
его. (рис. 4.4).



```
[evkuzmin@fedora lab10]$ chmod a+x lab10-1.asm
[evkuzmin@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`'
[evkuzmin@fedora lab10]$
```

Рис. 4.4: 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 7-м вариантом изменяю права доступа к файлу `readme-1.txt` согласно таблице. Потом проверяю правильность выполнения команды с помощью `ls -l`. (рис. 4.5).



```
evkuzmin@fedora:~/work/arch-pc/lab10
[evkuzmin@fedora lab10]$ chmod u=rw readme-1.txt
[evkuzmin@fedora lab10]$ chmod g=rwx readme-1.txt
[evkuzmin@fedora lab10]$ chmod o=rw readme-1.txt
[evkuzmin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 evkuzmin evkuzmin 3942 дек 12 20:47 in_out.asm
-rw-r--r--. 1 evkuzmin evkuzmin 2260 дек 12 20:53 lab10-1
-rwxr-xr-x. 1 evkuzmin evkuzmin 1142 дек 12 20:53 lab10-1.asm
-rw-r--r--. 1 evkuzmin evkuzmin 13448 дек 12 20:53 lab10-1.lst
-rw-r--r--. 1 evkuzmin evkuzmin 2512 дек 12 20:53 lab10-1.o
-rw-rwxrwx. 1 evkuzmin evkuzmin 13 дек 12 20:53 readme-1.txt
-rw-r--r--. 1 evkuzmin evkuzmin 0 дек 12 18:31 readme-2.txt
[evkuzmin@fedora lab10]$
```

Рис. 4.5: Предоставление прав доступа в символьном виде

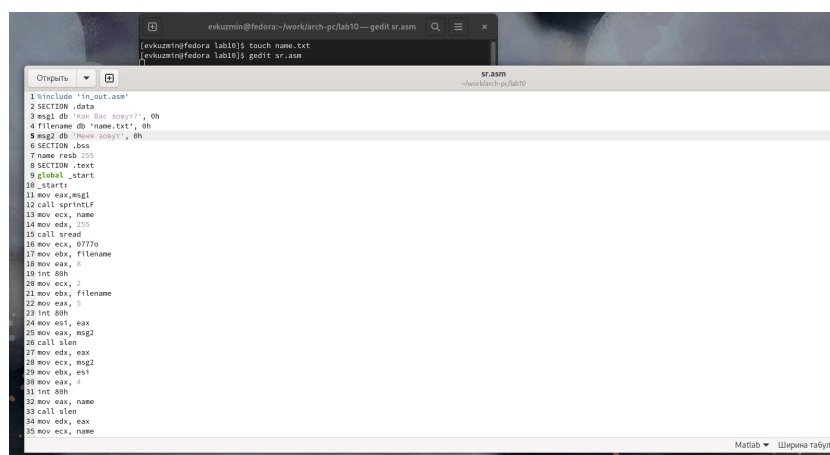
Аналогично поступаю с файлом `readme-2.txt`, только уже в числовом виде. (рис. 4.6).

```
[evkuzmin@fedora lab10]$ chmod 577 readme-2.txt # 101 111 111 = 577
[evkuzmin@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 evkuzmin evkuzmin 3942 дек 12 20:47 in_out.asm
-rw-r--r--. 1 evkuzmin evkuzmin 2260 дек 12 20:53 lab10-1
-rwxr-xr-x. 1 evkuzmin evkuzmin 1142 дек 12 20:53 lab10-1.asm
-rw-r--r--. 1 evkuzmin evkuzmin 13448 дек 12 20:53 lab10-1.lst
-rw-r--r--. 1 evkuzmin evkuzmin 2512 дек 12 20:53 lab10-1.o
-rw-rwxrwx. 1 evkuzmin evkuzmin 13 дек 12 20:53 readme-1.txt
-r-xrwxrwx. 1 evkuzmin evkuzmin 0 дек 12 18:31 readme-2.txt
[evkuzmin@fedora lab10]$
```

Рис. 4.6: Предоставление прав доступа в числовом виде

4.2) Выполнение заданий для самостоятельной работы.

Создаю файлы `sr.asm` и `name.txt` для корректной работы программы записи моего имени, введенного с клавиатуры, в соответствующий файл. Далее пишу непосредственно сам код: (рис. 4.7).



```
1 include "in_out.asm"
2 SECTION .data
3 msg1 db "Ваше имя: ", 0h
4 filename db "name.txt", 0h
5 msg2 db "Ваше имя: ", 0h
6 SECTION .bss
7 name resb 255
8 SECTION .text
9 global _start
10 _start:
11 mov eax, msg1
12 call printf
13 mov ecx, name
14 mov edi, 255
15 call read
16 mov ecx, 0770
17 mov ebx, filename
18 mov eax, 0
19 int 80h
20 mov ecx, 2
21 mov ebx, filename
22 mov eax, 0
23 int 80h
24 mov esi, eax
25 mov eax, msg2
26 call len
27 mov edi, eax
28 mov ecx, msg2
29 mov ebx, esi
30 mov eax, 0
31 int 80h
32 mov ebx, name
33 call len
34 mov edi, eax
35 mov ecx, name
```

Рис. 4.7: Создание и редактирование файла

Создаю исполняемый файл, проверяю его работу, далее ввожу необходимые команды, и убеждаюсь в правильности работы программы. (рис. 4.8).

```
[evkuzmin@fedora lab10]$ nasm -f elf -g -l sr.lst sr.asm
[evkuzmin@fedora lab10]$ ld -m elf_i386 -o sr sr.o
[evkuzmin@fedora lab10]$ ./sr
Как Вас зовут?
Кузьмин Егор
[evkuzmin@fedora lab10]$ ls
in_out.asm  lab10-1.asm  lab10-1.o  readme-1.txt  sr      sr.lst
lab10-1     lab10-1.lst  name.txt   readme-2.txt  sr.asm  sr.o
[evkuzmin@fedora lab10]$ cat name.txt
Меня зовут: Кузьмин Егор
[evkuzmin@fedora lab10]$
```

Рис. 4.8: Редактирование файла

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

```
““%include 'in_out.asm'
SECTION .data
msg1 db 'Как Вас зовут?', 0h
filename db 'name.txt', 0h
msg2 db 'Меня зовут', 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