

Отчёт по лабораторной работе №10

Дисциплина: Архитектура Компьютера

Дарина Андреевна Куокконен

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 файла	8
4.2	Редактирование файла	8
4.3	Создание исполняемого файла и его работа	9
4.4	Запрет на исполнение файла	9
4.5	Добавление прав на исполнение	10
4.6	Предоставление прав доступа в символьном виде	10
4.7	Предоставление прав доступа в числовом виде	11
4.8	Редактирование файла	11
4.9	Редактирование файла	12

1 Цель работы

Цель данной работы - это приобретение практического опыта в написании программ для работы с файлами.

2 Задание

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 *lab10-1.asm*.(рис. 4.1).



```
dakuokkonen@fedora:~/work/arch-pc/lab10 — gedit lab10-1.asm
[dakuokkonen@fedora ~]$ mkdir ~/work/arch-pc/lab10
[dakuokkonen@fedora ~]$ cd ~/work/arch-pc/lab10
[dakuokkonen@fedora lab10]$ touch lab10-1.asm readme-1.txt readme-2.txt
[dakuokkonen@fedora lab10]$ gedit lab10-1.asm
```

Рис. 4.1: Работа с директориями и создание файла

Вставляю в него следующую программу: (рис. 4.2).

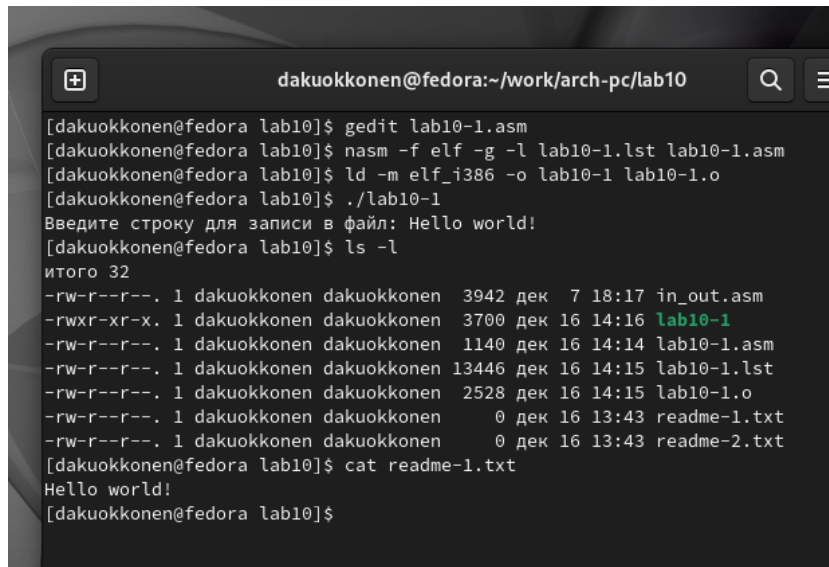


```
lab10-1.asm
~/work/arch-pc/lab10
Сохранить

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 вводной строки
7 SECTION .text
8 global _start
9 _start:
10 ; --- Печать сообщения 'msg'
11 mov eax, msg
12 call sprint
13 ; --- Запись введенной с клавиатуры строки в 'contents'
14 mov ecx, contents
15 mov ebx, 255
16 call read
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 eax, contents ; в 'eax' запишется количество
26 call strlen ; введенных байтов
27 ; --- Записываем в файл 'contents' ('sys_write')
28 mov ebx, esi
29 mov ecx, contents
30 mov ebx, esi
31 mov eax, 4
32 int 80h
33 ; --- Закрываем файл ('sys_close')
```

Рис. 4.2: Редактирование файла

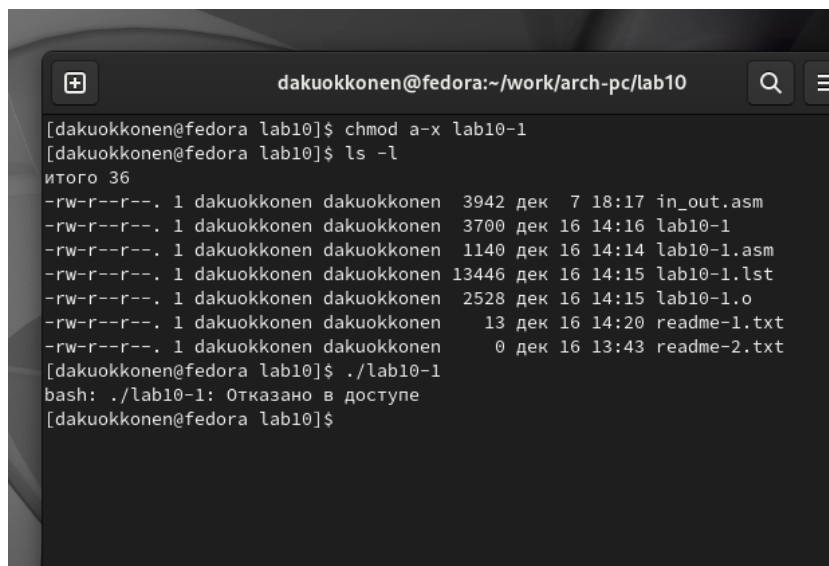
Создаю исполняемый файл и проверяю его работу. (рис. 4.3).



```
dakuokkonen@fedora:~/work/arch-pc/lab10
[dakuokkonen@fedora lab10]$ gedit lab10-1.asm
[dakuokkonen@fedora lab10]$ nasm -f elf -g -l lab10-1.lst lab10-1.asm
[dakuokkonen@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[dakuokkonen@fedora lab10]$ ./lab10-1
Введите строку для записи в файл: Hello world!
[dakuokkonen@fedora lab10]$ ls -l
итого 32
-rw-r--r--. 1 dakuokkonen dakuokkonen 3942 дек 7 18:17 in_out.asm
-rwxr-xr-x. 1 dakuokkonen dakuokkonen 3700 дек 16 14:16 lab10-1
-rw-r--r--. 1 dakuokkonen dakuokkonen 1140 дек 16 14:14 lab10-1.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 13446 дек 16 14:15 lab10-1.lst
-rw-r--r--. 1 dakuokkonen dakuokkonen 2528 дек 16 14:15 lab10-1.o
-rw-r--r--. 1 dakuokkonen dakuokkonen 0 дек 16 13:43 readme-1.txt
-rw-r--r--. 1 dakuokkonen dakuokkonen 0 дек 16 13:43 readme-2.txt
[dakuokkonen@fedora lab10]$ cat readme-1.txt
Hello world!
[dakuokkonen@fedora lab10]$
```

Рис. 4.3: Создание исполняемого файла и его работа

Далее, с помощью команды *chmod a-x* изменяю права доступа к исполняемому файлу, запретив его выполнение, далее проверяю его работу. (рис. 4.4).

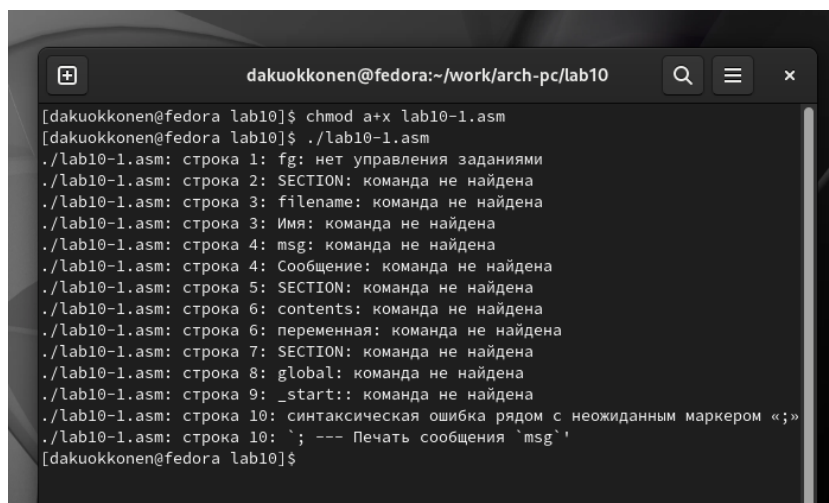


```
dakuokkonen@fedora:~/work/arch-pc/lab10
[dakuokkonen@fedora lab10]$ chmod a-x lab10-1
[dakuokkonen@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 dakuokkonen dakuokkonen 3942 дек 7 18:17 in_out.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 3700 дек 16 14:16 lab10-1
-rw-r--r--. 1 dakuokkonen dakuokkonen 1140 дек 16 14:14 lab10-1.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 13446 дек 16 14:15 lab10-1.lst
-rw-r--r--. 1 dakuokkonen dakuokkonen 2528 дек 16 14:15 lab10-1.o
-rw-r--r--. 1 dakuokkonen dakuokkonen 13 дек 16 14:20 readme-1.txt
-rw-r--r--. 1 dakuokkonen dakuokkonen 0 дек 16 13:43 readme-2.txt
[dakuokkonen@fedora lab10]$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
[dakuokkonen@fedora lab10]$
```

Рис. 4.4: Запрет на исполнение файла

Файл не исполняется, все верно. Далее я с помощью команды *chmod a+x* добав-

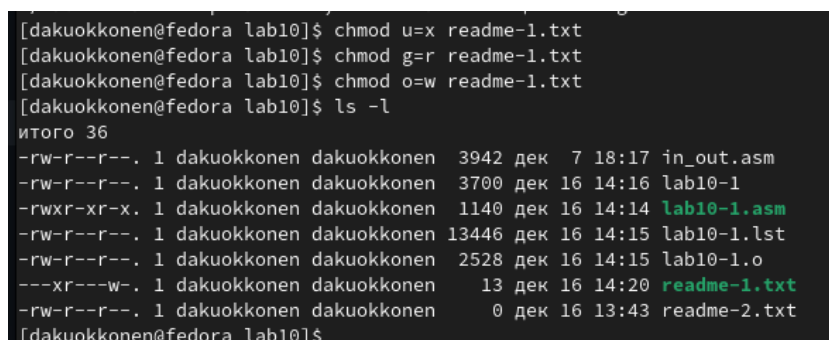
ляю права на исполнение самого файла lab10-1.asm и пытаюсь выполнить его. (рис. 4.5).



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

Рис. 4.5: Добавление прав на исполнение

Текстовый файл начинает работу, но не исполняется, так как не содержит в себе команд для терминала. Затем в соответствии со своим 11-м вариантом, я изменяю права доступа к файлу readme-1.txt согласно таблице. И проверяю правильность выполнение команды с помощью *ls -l*. (рис. 4.6).



```
[dakuokkonen@fedora lab10]$ chmod u=x readme-1.txt
[dakuokkonen@fedora lab10]$ chmod g=r readme-1.txt
[dakuokkonen@fedora lab10]$ chmod o=w readme-1.txt
[dakuokkonen@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 dakuokkonen dakuokkonen 3942 дек 7 18:17 in_out.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 3700 дек 16 14:16 lab10-1
-rwxr-xr-x. 1 dakuokkonen dakuokkonen 1140 дек 16 14:14 lab10-1.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 13446 дек 16 14:15 lab10-1.lst
-rw-r--r--. 1 dakuokkonen dakuokkonen 2528 дек 16 14:15 lab10-1.o
---xr---w-. 1 dakuokkonen dakuokkonen 13 дек 16 14:20 readme-1.txt
-rw-r--r--. 1 dakuokkonen dakuokkonen 0 дек 16 13:43 readme-2.txt
[dakuokkonen@fedora lab10]$
```

Рис. 4.6: Предоставление прав доступа в символьном виде

Аналогично поступаю с файлом readme-2.txt, только уже в числовом виде. (рис. 4.7).

```
[dakuokkonen@fedora lab10]$ chmod 577 readme-2.txt # 000 100 111 = 577
[dakuokkonen@fedora lab10]$ ls -l
итого 36
-rw-r--r--. 1 dakuokkonen dakuokkonen 3942 дек 7 18:17 in_out.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 3700 дек 16 14:16 lab10-1
-rwxr-xr-x. 1 dakuokkonen dakuokkonen 1140 дек 16 14:14 lab10-1.asm
-rw-r--r--. 1 dakuokkonen dakuokkonen 13446 дек 16 14:15 lab10-1.lst
-rw-r--r--. 1 dakuokkonen dakuokkonen 2528 дек 16 14:15 lab10-1.o
---xr---w-. 1 dakuokkonen dakuokkonen 13 дек 16 14:20 readme-1.txt
-r-xrwxrwx. 1 dakuokkonen dakuokkonen 0 дек 16 13:43 readme-2.txt
[dakuokkonen@fedora lab10]$
```

Рис. 4.7: Предоставление прав доступа в числовом виде

4.2) Выполнение заданий для самостоятельной работы.

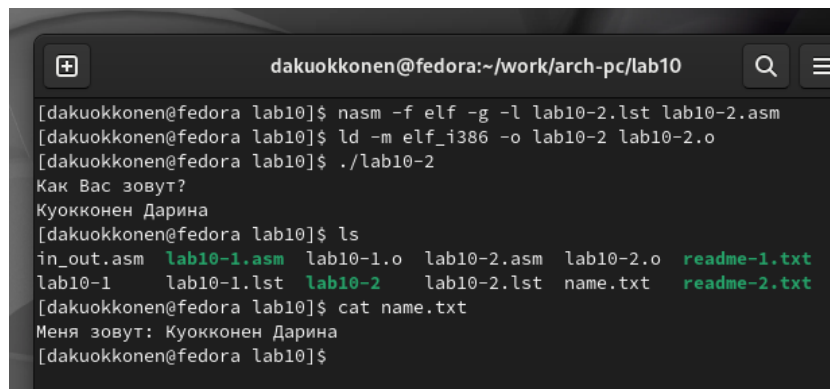
Создаю файлы lab10-2.asm и name.txt для корректной работы программы записи моего имени, введенного с клавиатуры, в соответствующий файл. Затем, я пишу сам код: (рис. 4.7).



```
1 %include "in_out.asm"
2 SECTION .data
3 msg1 db "Как вас зовут?",0h
4 filename db "name.txt",0h
5 msg2 db "Name: ",0h
6 SECTION .bss
7 name resb 255
8 SECTION .text
9 global _start
10 _start:
11 mov eax,msg1
12 call sprintf
13 mov ecx, name
14 mov edx, 255
15 call read
16 mov ecx, 0777
17 mov ebx, filename
18 mov eax, 8
19 int 80h
20 mov ecx, 2
21 mov ebx, filename
22 mov eax, 5
23 int 80h
24 mov esi, eax
25 mov eax, msg2
26 call slen
27 mov edx, eax
28 mov ecx, msg2
29 mov ebx, esi
30 mov eax, 4
31 int 80h
32 mov eax, name
33 call slen
```

Рис. 4.8: Редактирование файла

Создаю исполняемый файл, проверяю его работу, далее ввожу необходимые команды, и убеждаюсь в правильности работы программы. (рис. 4.9).



```
dakuokkonen@fedora:~/work/arch-pc/lab10
[dakuokkonen@fedora lab10]$ nasm -f elf -g -l lab10-2.lst lab10-2.asm
[dakuokkonen@fedora lab10]$ ld -m elf_i386 -o lab10-2 lab10-2.o
[dakuokkonen@fedora lab10]$ ./lab10-2
Как Вас зовут?
Куокконен Дарина
[dakuokkonen@fedora lab10]$ ls
in_out.asm  lab10-1.asm  lab10-1.o  lab10-2.asm  lab10-2.o  readme-1.txt
lab10-1     lab10-1.lst  lab10-2   lab10-2.lst  name.txt   readme-2.txt
[dakuokkonen@fedora lab10]$ cat name.txt
Меня зовут: Куокконен Дарина
[dakuokkonen@fedora lab10]$
```

Рис. 4.9: 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 225

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 данной лабораторной работы, я приобрела практический опыт в написании программ с использованием циклов и обработкой аргументов командной строки.

Список литературы

Архитектура компьютера и ЭВМ