

# **Лабораторная работа №10**

**Дисциплина: Архитектура компьютера**

Малюга Валерия Васильевна

# 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>
4.1	Написание программ для работы с файлами . . . . .	8
4.2	Задание для самостоятельной работы . . . . .	11
<b>5</b>	<b>Выводы</b>	<b>15</b>

## Список иллюстраций

4.1	Запуск программы и проверка его работы . . . . .	9
4.2	Запрет на выполнение файла . . . . .	10
4.3	Добавление прав на исполнение . . . . .	10
4.4	Предоставление прав доступа в символьном и двоичном виде . .	10
4.5	Запуск исполняемого файла и проверка его работы . . . . .	12

# **1 Цель работы**

Приобретение навыков написания программ для работы с файлами.

## 2 Задание

1. Написание программ для работы с файлами.
2. Задание для самостоятельной работы.

### 3 Теоретическое введение

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.

Для изменения прав доступа служит команда `chmod`, которая понимает как символьное, так и числовое указание прав.

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.

Для чтения данных из файла служит системный вызов `sys_read`, который использует следующие аргументы: количество байтов для чтения в регистре EDI, адрес в памяти для записи прочитанных данных в ECX, файловый дескриптор в EBX и номер системного вызова `sys-read` (3) в EAX. Как и для записи, прежде чем читать из файла, его необходимо открыть, что позволит получить дескриптор файла.

Для правильного закрытия файла служит системный вызов `sys_close`, который использует один аргумент – дескриптор файла в регистре EBX. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

Для изменения содержимого файла служит системный вызов `sys_lseek`, который использует следующие аргументы: исходная позиция для смещения EDI, значение смещения в байтах в ECX, файловый дескриптор в EBX и номер системного вызова `sys_lseek` (19) в EAX. Значение смещения можно задавать в байтах.

Удаление файла осуществляется системным вызовом `sys_unlink`, который использует один аргумент – имя файла в регистре EBX.

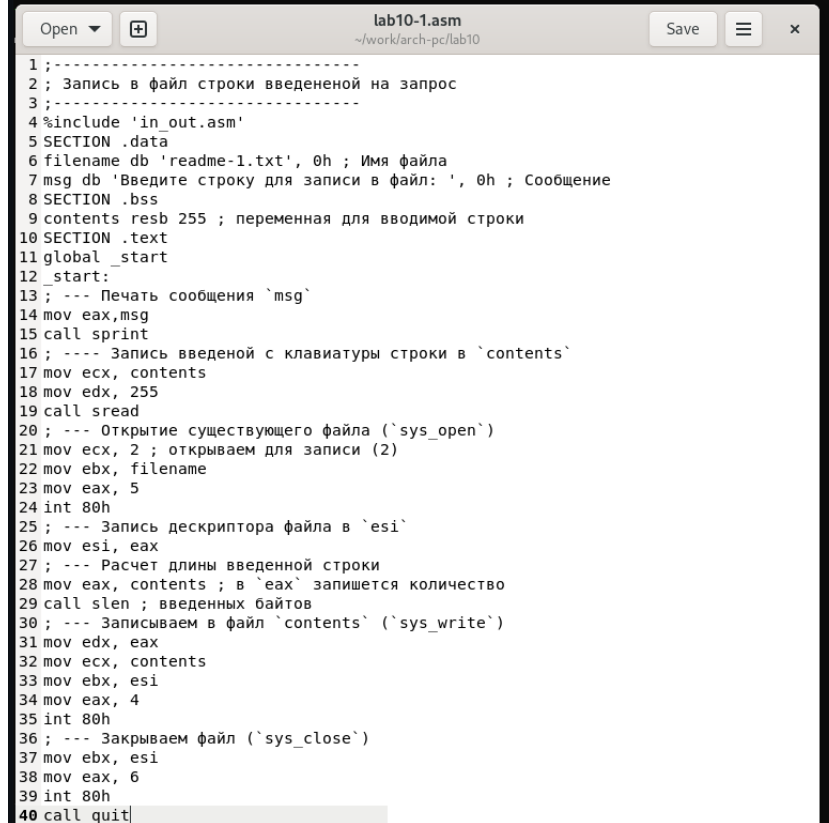
## **4 Выполнение лабораторной работы**

### **4.1 Написание программ для работы с файлами**

Создала каталог для программ лабораторной работы № 10, перешла в него и создала файлы lab10-1.asm, readme-1.txt и readme-2.txt. Ввела в файл lab10-1.asm текст программы из листинга 10.1. Создала исполняемый файл и проверила его работу (рис. 4.1).



```
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ touch lab10-1.asm readme-1.txt readme-2.txt
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ gedit lab10-1.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ nasm -f elf lab10-1.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: лабораторная №10
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ cat readme-1.txt
лабораторная №10
```



```
1 ;-----
2 ; Запись в файл строки введенной на запрос
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 filename db 'readme-1.txt', 0h ; Имя файла
7 msg db 'Введите строку для записи в файл: ', 0h ; Сообщение
8 SECTION .bss
9 contents resb 255 ; переменная для вводимой строки
10 SECTION .text
11 global _start
12 _start:
13 ; --- Печать сообщения `msg`
14 mov eax,msg
15 call sprint
16 ; ---- Запись введенной с клавиатуры строки в `contents`
17 mov ecx, contents
18 mov edx, 255
19 call sread
20 ; --- Открытие существующего файла (`sys_open`)
21 mov ecx, 2 ; открываем для записи (2)
22 mov ebx, filename
23 mov eax, 5
24 int 80h
25 ; --- Запись дескриптора файла в `esi`
26 mov esi, eax
27 ; --- Расчет длины введенной строки
28 mov eax, contents ; в `eax` запишется количество
29 call slen ; введенных байтов
30 ; --- Записываем в файл `contents` (`sys_write`)
31 mov edx, eax
32 mov ecx, contents
33 mov ebx, esi
34 mov eax, 4
35 int 80h
36 ; --- Закрываем файл (`sys_close`)
37 mov ebx, esi
38 mov eax, 6
39 int 80h
40 call quit
```

Рис. 4.1: Запуск программы и проверка его работы

С помощью команды `chmod u-x` изменила права доступа к исполняемому файлу `lab10-1`, запретив его выполнение. Попыталась выполнить файл. Объяснение результата: файл не выполняется, т.к в команде я указала “u” - владелец (себя), “-” - отменить набор прав, “x” - право на исполнение (рис. 4.2).

```

vvmalyuga@Malyuga:~/work/arch-pc/lab10$ cat readme-1.txt
лабораторная №10
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ chmod u-x lab10-1
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ./lab10-1
-bash: ./lab10-1: Permission denied
vvmalyuga@Malyuga:~/work/arch-pc/lab10$

```

Рис. 4.2: Запрет на выполнение файла

С помощью команды `chmod u+x` изменила права доступа к файлу `lab10-1.asm` с исходным текстом программы, добавив права на исполнение. Попыталась выполнить его. Объяснение результата: текстовый файл начинает исполнение, но не исполняется, т.к не содержит в себе команд для терминала (рис. 4.3).

```

vvmalyuga@Malyuga:~/work/arch-pc/lab10$ chmod u+x lab10-1.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ./lab10-1.asm
./lab10-1.asm: line 1: syntax error near unexpected token `;'
./lab10-1.asm: line 1: `;-----'
vvmalyuga@Malyuga:~/work/arch-pc/lab10$

```

Рис. 4.3: Добавление прав на исполнение

В соответствии со своим вариантом (11) предоставила права доступа к файлу `readme1.txt` представленные в символьном виде, а для файла `readme-2.txt` – в двоичном виде: `-x r- -w- 000 100 111`. Проверила правильность выполнения с помощью команды `ls -l` (рис. 4.4).

```

vvmalyuga@Malyuga:~/work/arch-pc/lab10$ chmod 640 readme-1.txt # --x r-- -w-
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ chmod 640 readme-2.txt # 000 100 111
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ls -l
total 28
-rw-r--r-- 1 vvmalyuga vvmalyuga 3942 Nov  8 15:39 in_out.asm
-rw-r-xr-x 1 vvmalyuga vvmalyuga 9164 Dec  9 23:13 lab10-1
-rwxr--r-- 1 vvmalyuga vvmalyuga 1289 Dec  9 23:12 lab10-1.asm
-rw-r--r-- 1 vvmalyuga vvmalyuga 1472 Dec  9 23:13 lab10-1.o
-rw-r----- 1 vvmalyuga vvmalyuga  31 Dec  9 23:14 readme-1.txt
-rw-r----- 1 vvmalyuga vvmalyuga   0 Dec  9 22:22 readme-2.txt
vvmalyuga@Malyuga:~/work/arch-pc/lab10$

```

Рис. 4.4: Предоставление прав доступа в символьном и двоичном виде

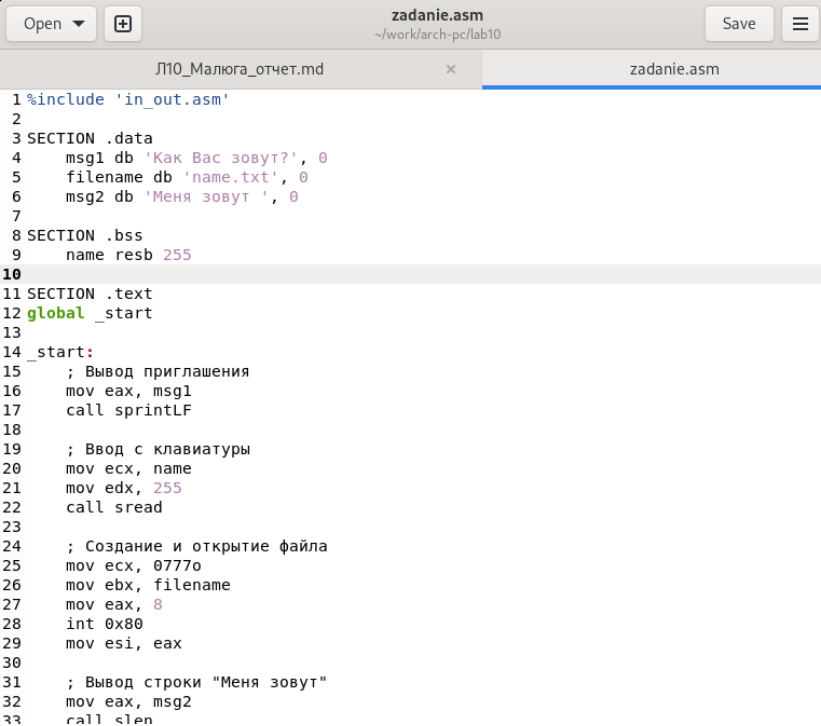
## 4.2 Задание для самостоятельной работы

Написала 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

Создала исполняемый файл и проверила его работу. Проверила наличие файла и его содержимое с помощью команд `ls` и `cat` (рис. 4.5).

```
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ touch zadanie.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ gedit zadanie.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ nasm -f elf zadanie.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ld -m elf_i386 -o zadanie zadanie.o
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ls
in_out.asm  lab10-1.asm  readme-1.txt  zadanie      zadanie.o
lab10-1     lab10-1.o    readme-2.txt  zadanie.asm
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ./zadanie
Как Вас зовут?
Малюга Валерия
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ ls
in_out.asm  lab10-1.asm  name.txt      readme-2.txt  zadanie.asm
lab10-1     lab10-1.o    readme-1.txt  zadanie      zadanie.o
vvmalyuga@Malyuga:~/work/arch-pc/lab10$ cat name.txt
Меня зовут Малюга Валерия
```



```
1 %include 'in_out.asm'
2
3 SECTION .data
4     msg1 db 'Как Вас зовут?', 0
5     filename db 'name.txt', 0
6     msg2 db 'Меня зовут ', 0
7
8 SECTION .bss
9     name resb 255
10
11 SECTION .text
12 global _start
13
14 _start:
15     ; Вывод приглашения
16     mov eax, msg1
17     call sprintf
18
19     ; Ввод с клавиатуры
20     mov ecx, name
21     mov edx, 255
22     call sread
23
24     ; Создание и открытие файла
25     mov ecx, 0777o
26     mov ebx, filename
27     mov eax, 8
28     int 0x80
29     mov esi, eax
30
31     ; Вывод строки "Меня зовут"
32     mov eax, msg2
33     call strlen
```

Рис. 4.5: Запуск исполняемого файла и проверка его работы

Прилагаю код:

```
%include 'in_out.asm'

SECTION .data

    msg1 db 'Как Вас зовут?', 0

    filename db 'name.txt', 0

    msg2 db 'Меня зовут ', 0

SECTION .bss

    name resb 255
```

```

SECTION .text
global _start
_start:

    ; вывод приглашения
    mov eax, msg1
    call sprintLF

    ; ввод с клавиатуры
    mov ecx, name
    mov edx, 255
    call sread

    ; создание и открытие файла
    mov ecx, 0777o
    mov ebx, filename
    mov eax, 8
    int 0x80
    mov esi, eax

    ; вывод строки "Меня зовут"
    mov eax, msg2
    call slen
    mov edx, eax
    mov ecx, msg2
    mov ebx, esi
    mov eax, 4
    int 0x80

    ; вывод введенного имени
    mov eax, name
    call slen
    mov edx, eax
    mov ecx, name

```

```
mov ebx, esi
mov eax, 4
int 0x80

; закрытие файла

mov ebx, esi
mov eax, 6
int 0x80
call quit
```

## **5 Выводы**

Благодаря данной лабораторной работе я приобрела навыки написания программ для работы с файлами.