

Работа с файлами средствами Nasm

Лабораторная работа №10.

Дагделен Зейнап Реджеповна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
3.1	Права доступа к файлам	7
3.2	Работа с файлами средствами Nasm	8
3.2.1	Открытие и создание файла	8
3.2.2	Запись в файл	8
3.2.3	Чтение файла	9
3.2.4	Закрытие файла	9
3.2.5	Удаление файла	9
4	Выполнение лабораторной работы	10
4.1	Написание программ для работы с файлами.	10
4.2	Задание для самостоятельной работы	12
5	Выводы	15
6	Список литературы	16

Список иллюстраций

4.1	Создание каталога и файлов в нем	10
4.2	Текст программы в файле	11
4.3	Создание исполняемого файла и его запуск	11
4.4	Запрет на выполнение файла и попытка выполнения файла . . .	11
4.5	Изменение прав доступа к файлу	12
4.6	Изменение прав доступа к файлу и проверка	12
4.7	Создание файла, код программы, создание исполняемого файла, его запуск и проверка	13

Список таблиц

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

3.1 Права доступа к файлам

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. Для предоставления прав доступа другому пользователю или другой группе командой 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.

Свойства (атрибуты) файлов и каталогов можно вывести на терминал с помощью команды `ls` с ключом `-l`.

3.2 Работа с файлами средствами Nasm

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.

Общий алгоритм работы с системными вызовами в Nasm можно представить в следующем виде: 1. Поместить номер системного вызова в регистр EAX; 2. Поместить аргументы системного вызова в регистрах EBX, ECX и EDX; 3. Вызов прерывания (int 80h); 4. Результат обычно возвращается в регистр EAX.

3.2.1 Открытие и создание 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.

3.2.2 Запись в файл

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.

3.2.3 Чтение 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.

3.2.4 Заккрытие 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.

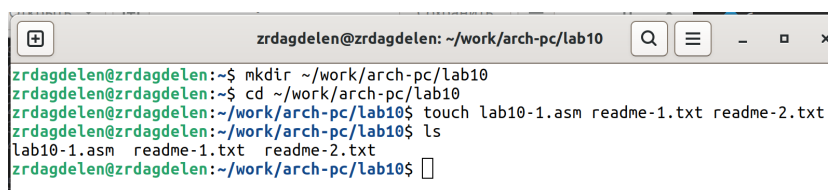
3.2.5 Удаление файла

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.

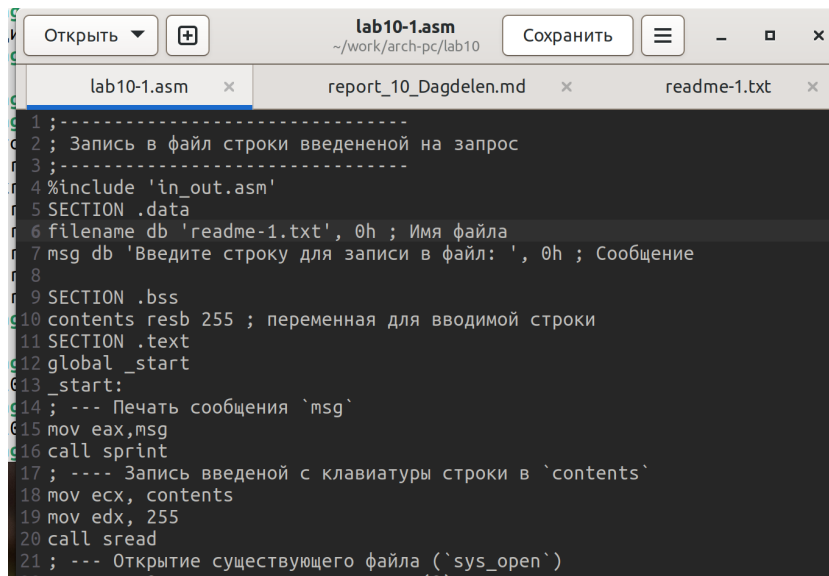
1. Создаю каталог для программ лабораторной работы № 10 благодаря команде `mkdir`, перехожу в него (с помощью `cd`) и создаю файлы `lab10-1.asm`, `readme-1.txt` и `readme-2.txt` (утилита `touch`) (рис. [-fig:001]).



```
zrdagdelen@zrdagdelen: ~/work/arch-pc/lab10
zrdagdelen@zrdagdelen:~$ mkdir ~/work/arch-pc/lab10
zrdagdelen@zrdagdelen:~$ cd ~/work/arch-pc/lab10
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ touch lab10-1.asm readme-1.txt readme-2.txt
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ls
lab10-1.asm  readme-1.txt  readme-2.txt
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

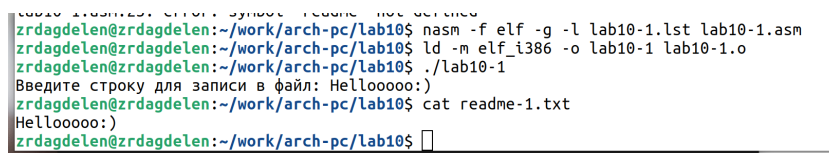
Рис. 4.1: Создание каталога и файлов в нем

2. Ввожу в файл `lab10-1.asm` текст программы из листинга 10.1 (Программа записи в файл сообщения) (рис. [-fig:002]). Создаю исполняемый файл и проверяю его работу (рис. [-fig:003]).



```
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
9 SECTION .bss
10 contents resb 255 ; переменная для вводимой строки
11 SECTION .text
12 global _start
13 _start:
14 ; --- Печать сообщения `msg`
15 mov eax,msg
16 call sprint
17 ; ---- Запись введенной с клавиатуры строки в `contents`
18 mov ecx, contents
19 mov edx, 255
20 call sread
21 ; --- Открытие существующего файла (`sys_open`)
```

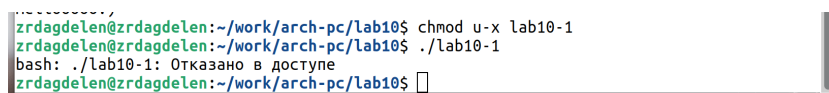
Рис. 4.2: Текст программы в файле



```
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ nasm -f elf -g -l lab10-1.lst lab10-1.asm
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: Hellooooo:)
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ cat readme-1.txt
Hellooooo:)
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

Рис. 4.3: Создание исполняемого файла и его запуск

3. С помощью команды `chmod` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение. Попытаюсь выполнить файл (рис. [4.4]). Объяснение результата: Файл не выполняется, т.к в команде указано “u” - владелец (то есть я), “-” - отменить набор прав, “x” - право на исполнение.



```
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ chmod u-x lab10-1
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

Рис. 4.4: Запрет на выполнение файла и попытка выполнения файла

4. С помощью команды `chmod` изменяю права доступа к файлу `lab10-1.asm` с исходным текстом программы, добавив права на исполнение. Попытаюсь выполнить его (рис. [4.5]).

```
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ chmod u+x lab10-1.asm
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ./lab10-1.asm
./lab10-1.asm: строка 1: синтаксическая ошибка рядом с неожиданным маркером «;»
./lab10-1.asm: строка 1: `;------'
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

Рис. 4.5: Изменение прав доступа к файлу

Объяснение результата: текстовый файл начинает исполнение, но не исполняется, т.к не содержит в себе команд для терминала. 5. В соответствии с вариантом в таблице 10.4 (у меня 13 вариант) предоставляю права доступа к файлу readme-1.txt представленные в символьном виде, а для файла readme-2.txt – в двоичном виде.

1) -w- -x — : в двоичном это 010001000, в восьмеричном это 210

2) 110 011 001 : в восьмеричном это 631

Проверяю правильность выполнения с помощью команды ls -l (рис. [4.6]).

```
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ chmod 210 readme-1.txt # -w- --x ---
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ls -l
итого 40
-rw-rw-r-- 1 zrdagdelen zrdagdelen 3942 ноя 28 19:58 in_out.asm
-rwxrwxr-x 1 zrdagdelen zrdagdelen 9700 дек 15 17:13 lab10-1
-rwxrwxr-- 1 zrdagdelen zrdagdelen 1290 дек 15 17:13 lab10-1.asm
-rw-rw-r-- 1 zrdagdelen zrdagdelen 13756 дек 15 17:13 lab10-1.lst
-rw-rw-r-- 1 zrdagdelen zrdagdelen 2512 дек 15 17:13 lab10-1.o
--w---x--- 1 zrdagdelen zrdagdelen 0 дек 15 17:34 readme-1.txt
-rw-rw-r-- 1 zrdagdelen zrdagdelen 0 дек 15 17:34 readme-2.txt
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ chmod 631 readme-2.txt # 110 011 001
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ls -l
итого 40
-rw-rw-r-- 1 zrdagdelen zrdagdelen 3942 ноя 28 19:58 in_out.asm
-rwxrwxr-x 1 zrdagdelen zrdagdelen 9700 дек 15 17:13 lab10-1
-rwxrwxr-- 1 zrdagdelen zrdagdelen 1290 дек 15 17:13 lab10-1.asm
-rw-rw-r-- 1 zrdagdelen zrdagdelen 13756 дек 15 17:13 lab10-1.lst
-rw-rw-r-- 1 zrdagdelen zrdagdelen 2512 дек 15 17:13 lab10-1.o
--w---x--- 1 zrdagdelen zrdagdelen 0 дек 15 17:34 readme-1.txt
-rw--wx-x 1 zrdagdelen zrdagdelen 0 дек 15 17:34 readme-2.txt
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

Рис. 4.6: Изменение прав доступа к файлу и проверка

4.2 Задание для самостоятельной работы

1. Напишу программу работающую по следующему алгоритму:

- Вывод приглашения “Как Вас зовут?”
- ввести с клавиатуры мои фамилию и имя
- создать файл с именем name.txt

- записать в файл сообщение “Меня зовут”
- дописать в файл строку введенную с клавиатуры
- закрыть файл

Создаю файл для программы, пишу саму программу в соответствии с требованиями, создаю исполняемый файл и проверяю его работу. Проверяю наличие файла и его содержимое с помощью команд `ls` и `cat` (рис. [4.7]).

```
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ touch zadan.asm
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ nasm -f elf -g -l zadan.lst zadan.asm
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ld -m elf_i386 -o zadan zadan.o
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ./zadan
Как Вас зовут?
Дагделен Зейнап
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ cat name.txt
Меня зовут Дагделен Зейнап
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$ ls
in_out.asm  lab10-1.asm  lab10-1.o  readme-1.txt  zadan      zadan.lst
lab10-1     lab10-1.lst  name.txt   readme-2.txt  zadan.asm  zadan.o
zrdagdelen@zrdagdelen:~/work/arch-pc/lab10$
```

Рис. 4.7: Создание файла, код программы, создание исполняемого файла, его запуск и проверка

```
%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

Я приобрела навыки написания программ для работы с файлами.

6 Список литературы

Архитектура ЭВМ