

Отчёт по лабораторной работе №10

Дисциплина: архитектура компьютеров и операционные системы

Сахно Никита НКАбд-05-23

Содержание

1	Цель работы	1
2	Задание	1
3	Теоретическое введение.....	1
4	Выполнение лабораторной работы.....	2
4.1	Написание программ для работы с файлами.....	2
4.2	Задание для самостоятельной работы.....	4
5	Выводы	7
6	Список литературы	7

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 `EDX`, 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 `EDX`, 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

Создаю каталог для программ лабораторной работы № 10, перехожу в него и создаю файлы `lab10-1.asm`, `readme-1.txt` и `readme-2.txt`. (рис. 8)

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

Figure 1: Создание файлов для лабораторной работы

Ввожу в файл lab10-1.asm текст программы, записывающей в файл сообщения, из листинга 10.1. (рис. 8)

```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab10/lab10-1.asm Изм
; --- Печать сообщения `msg`
mov eax,msg
call sprint
; ---- Запись введенной с клавиатуры строки в `contents`
mov ecx, contents
mov edx, 255
call sread
; --- Открытие существующего файла (`sys_open`)
mov ecx, 2 ; открываем для записи (2)
mov ebx, filename
mov eax, 5
int 80h
; --- Запись дескриптора файла в `esi`
mov esi, eax
; --- Расчет длины введенной строки
mov eax, contents ; в `eax` запишется количество
call slen ; введенных байтов
; --- Записываем в файл `contents` (`sys_write`)
mov edx, eax
mov ecx, contents
mov ebx, esi
mov eax, 4
int 80h
; --- Закрываем файл (`sys_close`)
mov ebx, esi
mov eax, 6
int 80h
call quit
```

Figure 2: Ввод текста программы из листинга 10.1

Создаю исполняемый файл и проверяю его работу. (рис. 8)

```
[nvsakhno@fedora lab10]$ nasm -f elf lab10-1.asm
[nvsakhno@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[nvsakhno@fedora lab10]$ ./lab10-1
Введите строку для записи в файл: Hello world!
[nvsakhno@fedora lab10]$ cat readme-1.txt
Hello world!
```

Figure 3: Запуск исполняемого файла

Далее с помощью команды `chmod` `u-x` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение и пытаюсь выполнить файл. (рис. 8)

```
[nvsakhno@fedora lab10]$ chmod u-x lab10-1
[nvsakhno@fedora lab10]$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
[nvsakhno@fedora lab10]$
```

Figure 4: Запрет на выполнение файла

Файл не выполняется, т.к в команде я указала “`u`” - владелец (себя), “`-`” - отменить набор прав, “`x`” - право на исполнение.

С помощью команды `chmod u+x` изменяю права доступа к файлу `lab10-1.asm` с исходным текстом программы, добавив права на исполнение, и пытаюсь выполнить его. (рис. 8)

```
[nvsakhno@fedora lab10]$ chmod u+x lab10-1.asm
[nvsakhno@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 «;»
```

Figure 5: Добавление прав на исполнение

Текстовый файл начинает исполнение, но не исполняется, т.к не содержит в себе команд для терминала.

В соответствии со своим вариантом (10) в таблице 10.4 предоставляю права доступа к файлу `readme1.txt` представленные в символьном виде, а для файла `readme-2.txt` – в двоичном виде:

`r- r- rwx, 001 100 010`

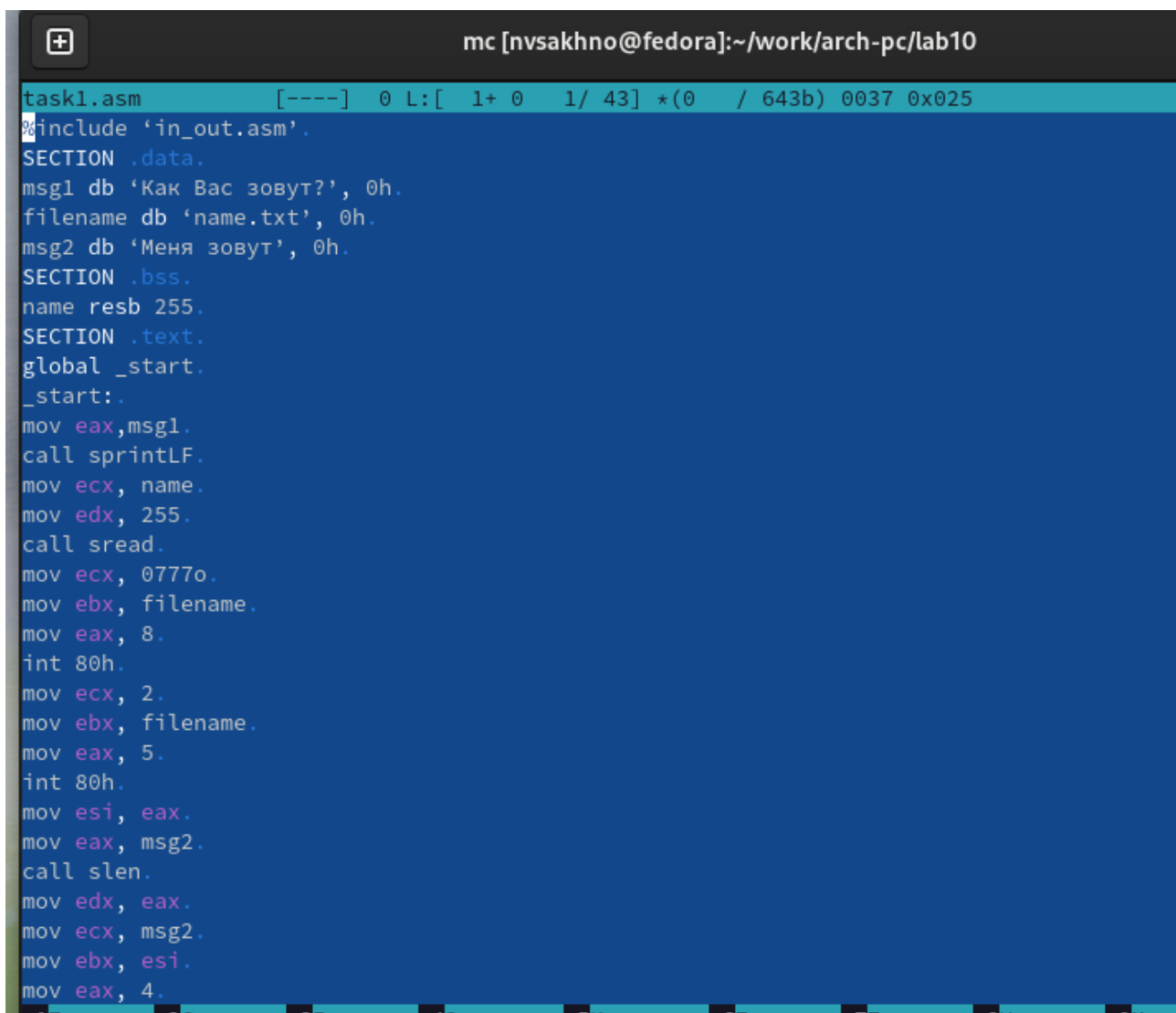
И проверяю правильность выполнения с помощью команды `ls -l`. (рис. 8)

```
[1]+ Остановлен ./lab10-1
[nvsakhno@fedora lab10]$ chmod 640 readme-1.txt # r-- r-- rwx
[nvsakhno@fedora lab10]$ chmod 640 readme-2.txt # 001 100 010
[nvsakhno@fedora lab10]$ ls -l
итого 20
-rw-r--r--. 1 nvsakhno nvsakhno 3942 ноя  8 14:29 in_out.asm
-rwxr-xr-x. 1 nvsakhno nvsakhno 1524 дек 15 22:15 lab10-1
-rwxr--r--. 1 nvsakhno nvsakhno 1142 дек 15 22:15 lab10-1.asm
-rw-r--r--. 1 nvsakhno nvsakhno 1472 дек 15 22:15 lab10-1.o
-rw-r-----. 1 nvsakhno nvsakhno   13 дек 15 22:15 readme-1.txt
-rw-r-----. 1 nvsakhno nvsakhno    0 дек 15 22:05 readme-2.txt
```

Figure 6: Предоставление прав доступа в символьном и двоичном виде

4.2 Задание для самостоятельной работы

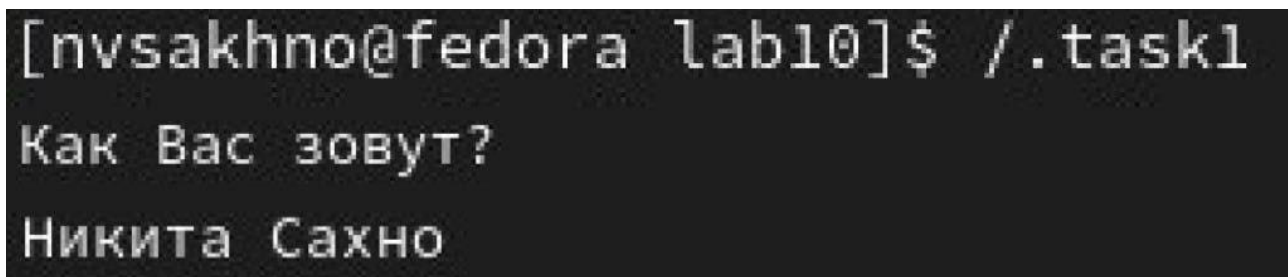
Пишу код программы, выводящей приглашения “Как Вас зовут?”, считывающей с клавиатуры фамилию и имя и создающую файл, в который записывается сообщение “Меня зовут”ФИ””. (рис. 8)



```
mc [nvsakhno@fedora]:~/work/arch-pc/lab10
task1.asm [----] 0 L:[ 1+ 0 1/ 43] *(0 / 643b) 0037 0x025
%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.
```

Figure 7: Написание текста программы

Создаю исполняемый файл и проверяю его работу. Проверяю наличие файла и его содержимое с помощью команд ls и cat. (рис. 8)



```
[nvsakhno@fedora lab10]$ ./task1
Как Вас зовут?
Никита Сахно
```

Figure 8: Запуск исполняемого файла и проверка его работы

Программа работает корректно.

Код программы:

```
%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

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

6 Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005 — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.

13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).