

Отчёт по лабораторной работе №5

Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM.

Дорохов Данила Антонович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	12

Список иллюстраций

4.1	Рис 2	8
4.2	Рис 3	8
4.3	Рис 4	9
4.4	Рис 5	9
4.5	Рис 6	9
4.6	Рис 7	9
4.7	Рис 8	10
4.8	Рис 9	10
4.9	Рис 10	11
4.10	Рис 11	11
4.11	Рис 12	11

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . .	7
-----	---	---

1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера `mov` и `int`.

2 Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. [3.1] приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно об Unix см. в [1–6].

4 Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. [??])

1. Открываем Midnight Commander. (Рис. 1-2)

dadorokhov@dk3n38 ~ \$

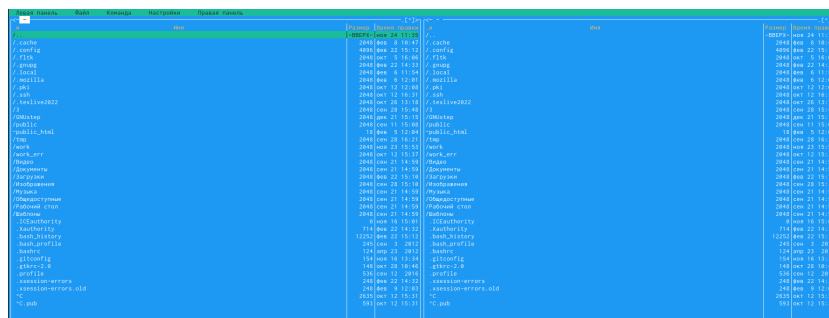


Рис. 4.1: Рис 2

2. Переходим в каталог ~/work/archpc. (Рис. 3)

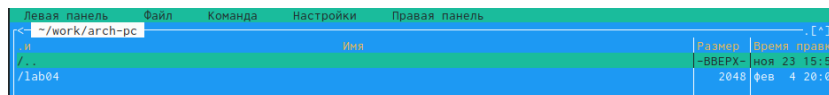


Рис. 4.2: Рис 3

3. Создаем папку lab05. (Рис. 4)

[Создать новый каталог](#)

Введите имя каталога:
lab05

[\[< Дальше >\]](#) [\[Прервать \]](#)

Рис. 4.3: Рис 4

```
dadorokhov@dk3n38 ~/work/arch-pc/lab05 $ touch lab5-1.asm
```

Рис. 4.4: Рис 5

4. Создаем файл lab5-1.asm. (Рис. 6)

```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/d/a/dadorokhov/work/arch-pc/lab05/lab5-1.asm
```

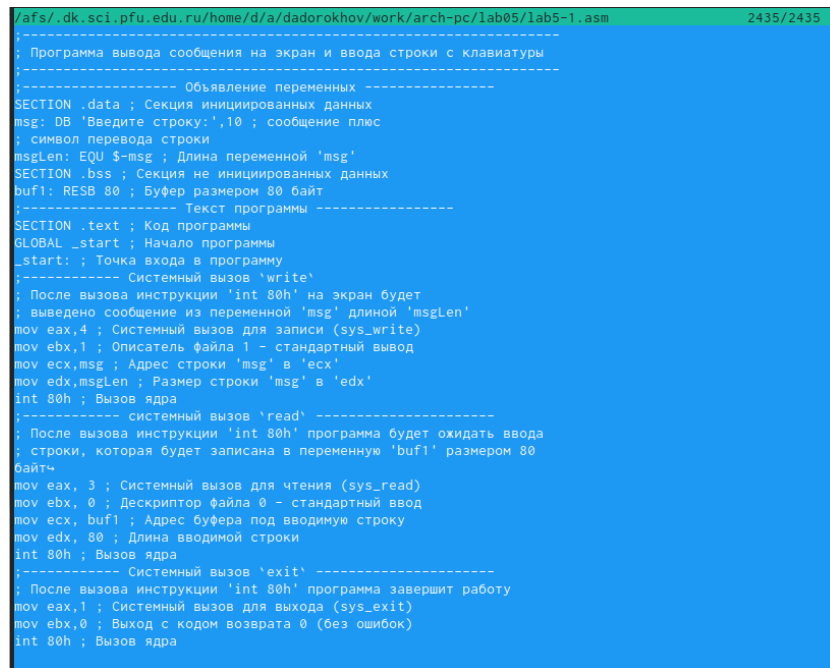
Рис. 4.5: Рис 6

5. Открываем файл lab6-1.asm для редактирования. (Рис. 7)

```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/home/d/a/dadorokhov/work/arch-pc/lab05/lab5-1.asm
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write' -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов 'read' -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80
байт
mov eax,3 ; Системный вызов для чтения (sys_read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov ecx,buf1 ; Адрес буфера под вводимую строку
mov edx,80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рис. 4.6: Рис 7

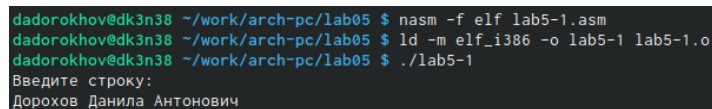
6. Вводим текст программы, сохраняем изменения и закрываем файл. (Рис. 8)



```
/afs/.dk.sci.pfu.edu.ru/home/d/a/dadorokhov/work/arch-pc/lab05/lab5-1.asm 2435/2435
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write' -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов 'read' -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80
байт
mov eax,3 ; Системный вызов для чтения (sys_read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov ecx, buf1 ; Адрес буфера под вводимую строку
mov edx, 80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рис. 4.7: Рис 8

7. Открываем файл lab5-1.asm для просмотра. Убеждаемся, что файл содержит текст программы. (Рис. 9)



```
dadorokhov@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
dadorokhov@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
dadorokhov@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Дорохов Данила Антонович
```

Рис. 4.8: Рис 9

8. Оттранслируем текст программы lab5-1.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл. (Рис. 10)

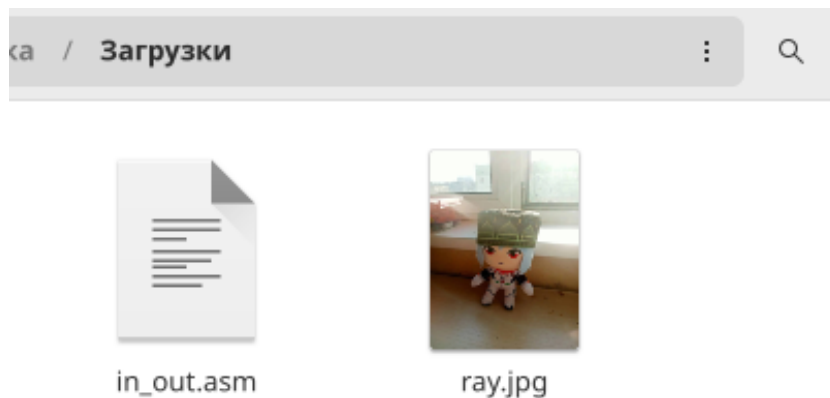


Рис. 4.9: Рис 10

9. Скачаем файл `in_out.asm` со страницы курса в ТУИС. (Рис. 11)

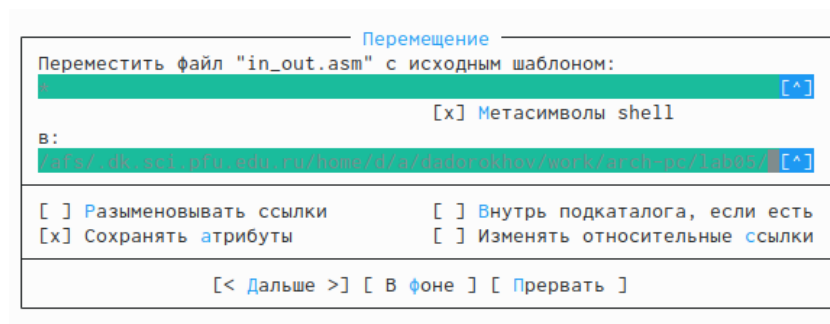


Рис. 4.10: Рис 11

10. Скопируем файл `in_out.asm` в каталог с файлом `lab5-1.asm`. (Рис. 12)

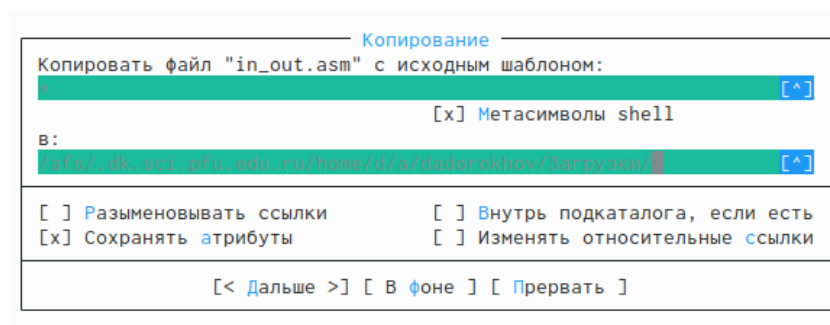


Рис. 4.11: Рис 12

5 Выводы

В ходе выполнения данной лабораторной работы я приобрел практические навыки работы в Midnight Commander и освоил инструкции языка ассемблера mov и int.

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.