

# Отчет по лабораторной работе №5

## Дисциплина: Архитектура компьютера

Баазова Нина Эдгаровна

### Содержание

#### 1 Цель работы

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

#### 2 Задание

1. Выполнение порядка лабораторной работы №5
2. Задание для самостоятельной работы

#### 3 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной.

#### 4 Выполнение лабораторной работы

1. Порядок выполнения лабораторной работы:

1.1 Откроем Midnight Commander с помощью команды mc. Используя клавиши ↑, ↓ и Enter, перейдём в каталог ~/work/arch-rc, созданный при выполнении лабораторной работы №4. Затем с помощью функциональной клавиши F7 создаем папку lab05 и перейдем в созданный каталог.

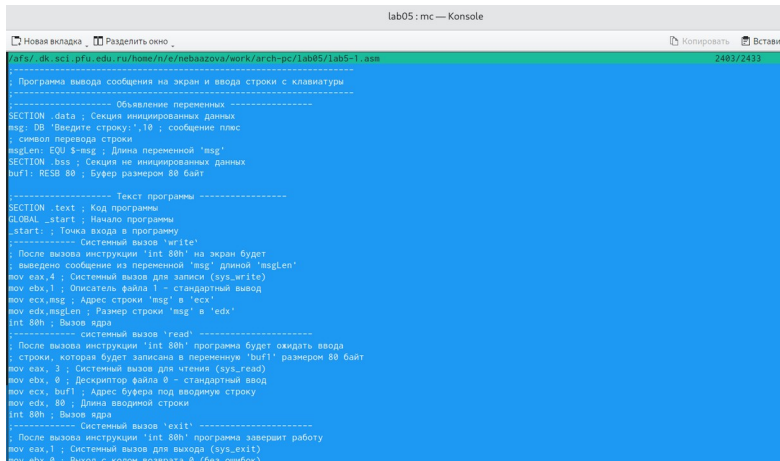


*рис 2*

[illegible]

рис 3

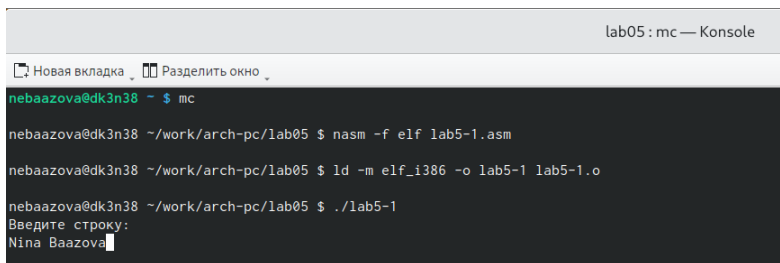
1.4 С помощью функциональной клавиши F3 откроем файл lab5-1.asm для просмотра. Убедимся, что файл содержит текст программы.



```
lab05: mc — Console
-----
; 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 переменных '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 ; Описание 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 ; Описание файла 0 - стандартный ввод
mov ecx, buf1 ; Адрес буфера под вводную строку
mov edx, 80 ; Длина вводной 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)
```

рис 4

1.5 Оттранслируем текст программы lab5-1.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры. На запрос введите "Ваши ФИО".



```
lab05: mc — Console
-----
nebaazova@dk3n38 ~ $ mc
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Nina Baazova
```

рис 5

2.1 Подключение внешнего файла in\_out.asm: Для начала скачаем файл in\_out.asm со страницы курса в ТУИС, затем подключим его. Но нужно помнить, что in\_out.asm должен лежать в том же каталоге, что и файл с программой, в которой он используется.

В одной из панелей mc откроем каталог с файлом lab5-1.asm. В другой панели каталог со скаченным файлом in\_out.asm (для перемещения между панелями используем Tab). Скопируем файл in\_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5.

С помощью функциональной клавиши F6 создадим копию файла lab5-1.asm с именем lab5-2.asm. Выделяем файл lab5-1.asm, нажимаем клавишу F6, вводим имя файла lab5-2.asm и нажимаем клавишу Enter.

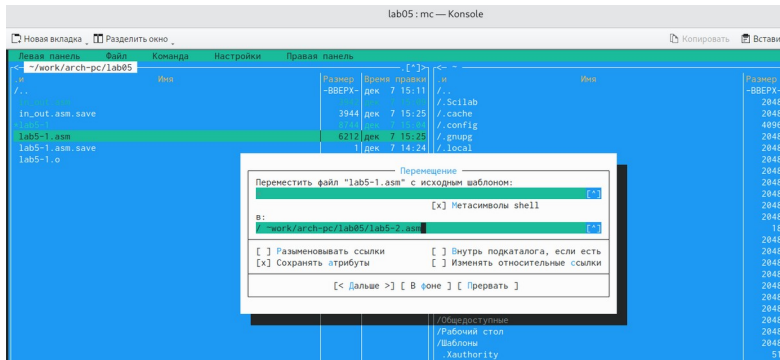


рис 6

2.2 Исправим текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in\_out.asm (используем подпрограммы sprintLF, sread и quit) в соответствии с листингом 5.2.

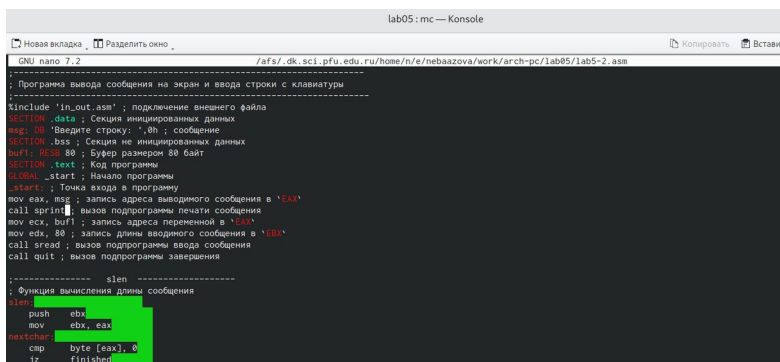


рис 7

Создаем исполняемый файл и проверьте его работу.



рис 8

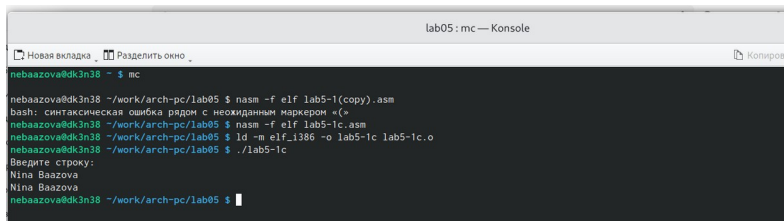
2.3 В файле lab5-2.asm заменим подпрограмму sprintLF на sprint. Создаем исполняемый файл и проверьте его работу.

В чем же разница? А разница в том, что sprintLF переводит на следующую строку, а sprint не переводит.

**ВЫВОД:** Мы ознакомились с работой в Midnight Commander, научились подключать внешний файл in\_out.asm.

## 2. Задание для самостоятельной работы:

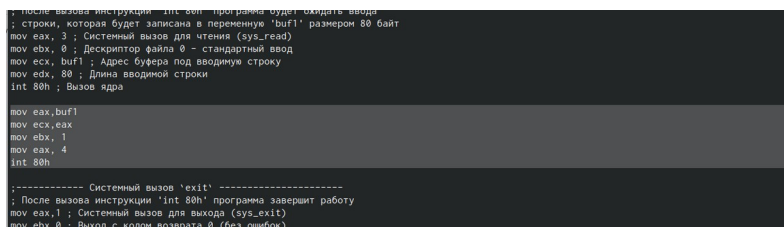
1.1 Создаем копию файла lab5-1.asm. Вносим изменения в программу (без использования внешнего файла in\_out.asm), так чтобы она работала по следующему алгоритму: • вывести приглашение типа “Введите строку:”; • ввести строку с клавиатуры; • вывести введенную строку на экран.



```
lab05:mc — Konsole
nebaazova@dk3n38 ~ $ mc
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1(copy).asm
bash: синтаксическая ошибка рядом с неожиданным маркером «(»
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1c.asm
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1c lab5-1c.o
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-1c
Введите строку:
Nina Baazova
Nina Baazova
nebaazova@dk3n38 ~/work/arch-pc/lab05 $
```

рис 9

Получим исполняемый файл и проверим его работу. На приглашение ввести строку введем свое имя и фамилию.



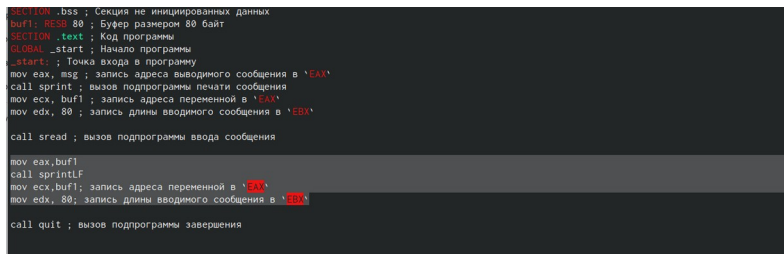
```
; после вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
mov eax, 3 ; Системный вызов для чтения (sys_read)
mov ebx, 0 ; Регистр файла 0 - стандартный ввод
mov ecx, buf1 ; Адрес буфера под вводимую строку
mov edx, 80 ; Длина вводимой строки
int 80h ; Вызов ядра

mov eax, buf1
mov ecx, eax
mov ebx, 1
mov eax, 4
int 80h

;----- 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)
```

рис 10

1.2 Создаем копию файла lab5-2.asm. Исправим текст программы с использованием под-программ из внешнего файла in\_out.asm, так чтобы она работала по следующему алгоритму: • вывести приглашение типа “Введите строку:”; • ввести строку с клавиатуры; • вывести введенную строку на экран.



```
section .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
section .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
mov eax, msg ; запись адреса выводимого сообщения в 'EAX'
call sprint ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'ECX'
mov edx, 80 ; запись длины выводимого сообщения в 'EDX'

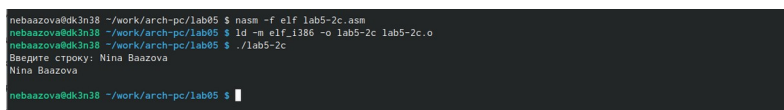
call read ; вызов подпрограммы ввода сообщения

mov eax, buf1
call sprintf
mov ecx, buf1; запись адреса переменной в 'ECX'
mov edx, 80; запись длины выводимого сообщения в 'EDX'

call quit ; вызов подпрограммы завершения
```

рис 11

Создаем исполняемый файл и проверяем его работу.



```
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2c.asm
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2c lab5-2c.o
nebaazova@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-2c
Введите строку: Nina Baazova
Nina Baazova
nebaazova@dk3n38 ~/work/arch-pc/lab05 $
```

рис 12

ВЫВОД: Мы выполнили задание для самостоятельной работы.

## 5 Вывод лабораторной работы

Мы приобрели практические навыки работы в Midnight Commander и освоили инструкции языка ассемблера `mov` и `int`.