

Лабораторная работа №5

Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM. Системные вызовы в ОС GNU Linux

Козлова Нонна Юрьевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Используем команду <code>mc</code>	8
4.2	Используем клавишу F7	9
4.3	Используем клавишу F4	10
4.4	Используем клавишу F4	11
4.5	Используем клавишу F5	11
4.6	С использованием файла <code>in_out.asm</code>	12
4.7	Программа работает верно	13

Список таблиц

1 Цель работы

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

2 Задание

Написать программу на языке ассемблера NASM.

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. Открываем Midnight Commander и переходим в каталог ~/work/arch- pc. (рис. 4.1)

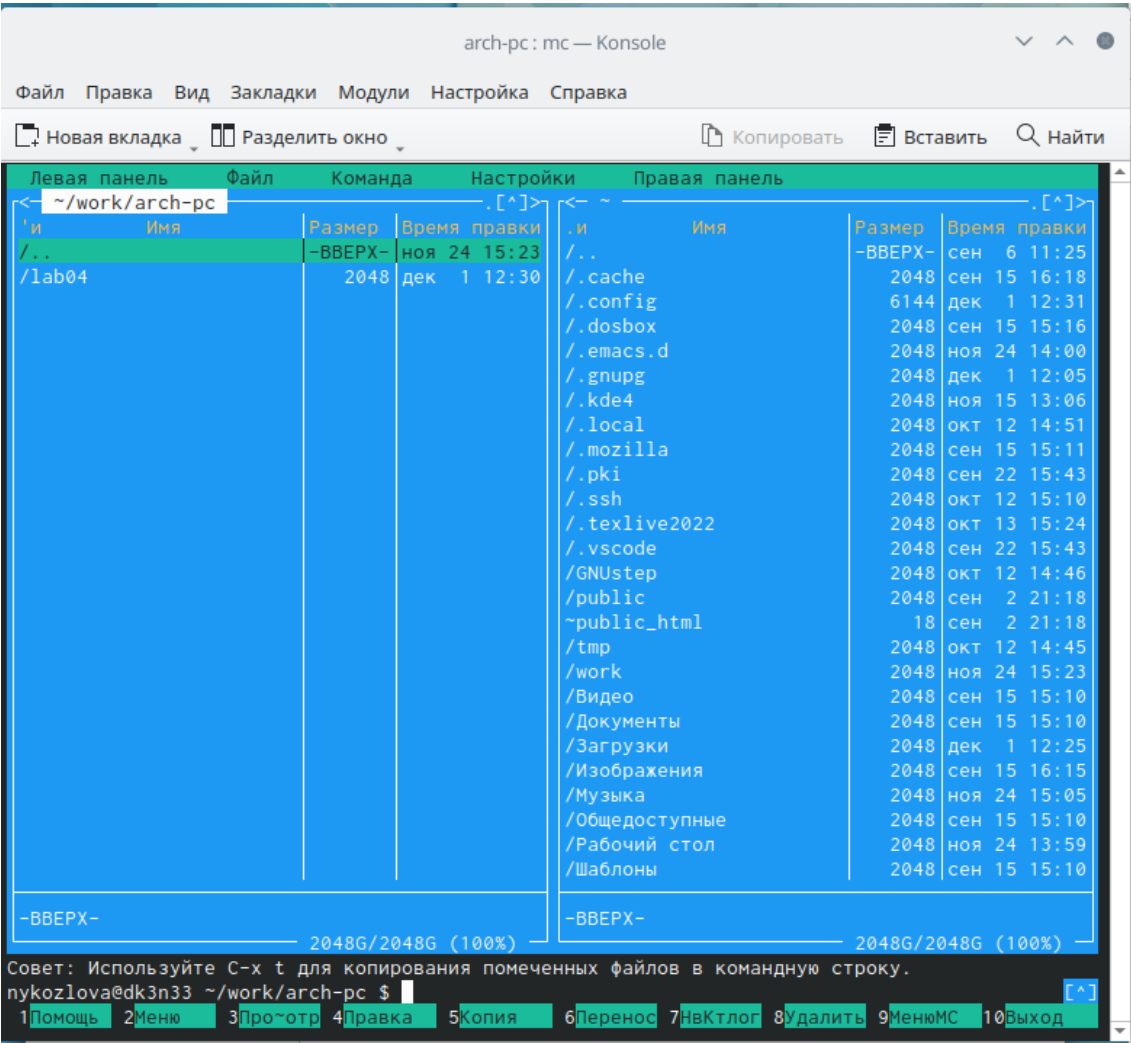


Рис. 4.1: Используем команду mc

2. Создаем папку lab05, далее в ней создаем файл lab5-1.asm. (рис. 4.2)

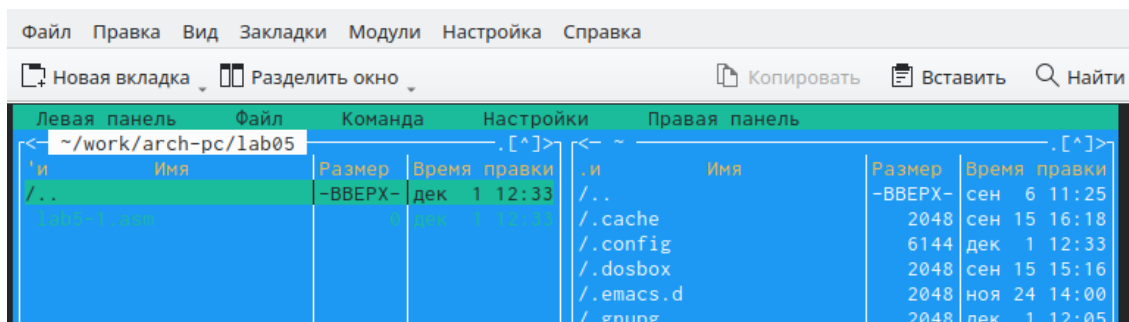


Рис. 4.2: Используем клавишу F7

3. Открываем файл lab5-1.asm для редактирования и вводим текст программы. (рис. 4.3)

```
lab5-1.asm [-M--] 64 L:[ 1+20 21/ 29] *(1424/1880b) 0010 0x00A [*][X]
;----- Объявление переменных -----
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10 ; собо
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициированных 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
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
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h
```

Рис. 4.3: Используем клавишу F4

4. Оттранслируем текст программы lab5-1.asm в объектный файл, выполним компоновку объектного файла и запустим его. (рис. 4.4)

```

nykozlova@dk3n33 ~ $ mc
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ touch lab5-1.asm
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Козлова Нонна
nykozlova@dk3n33 ~/work/arch-pc/lab05 $

```

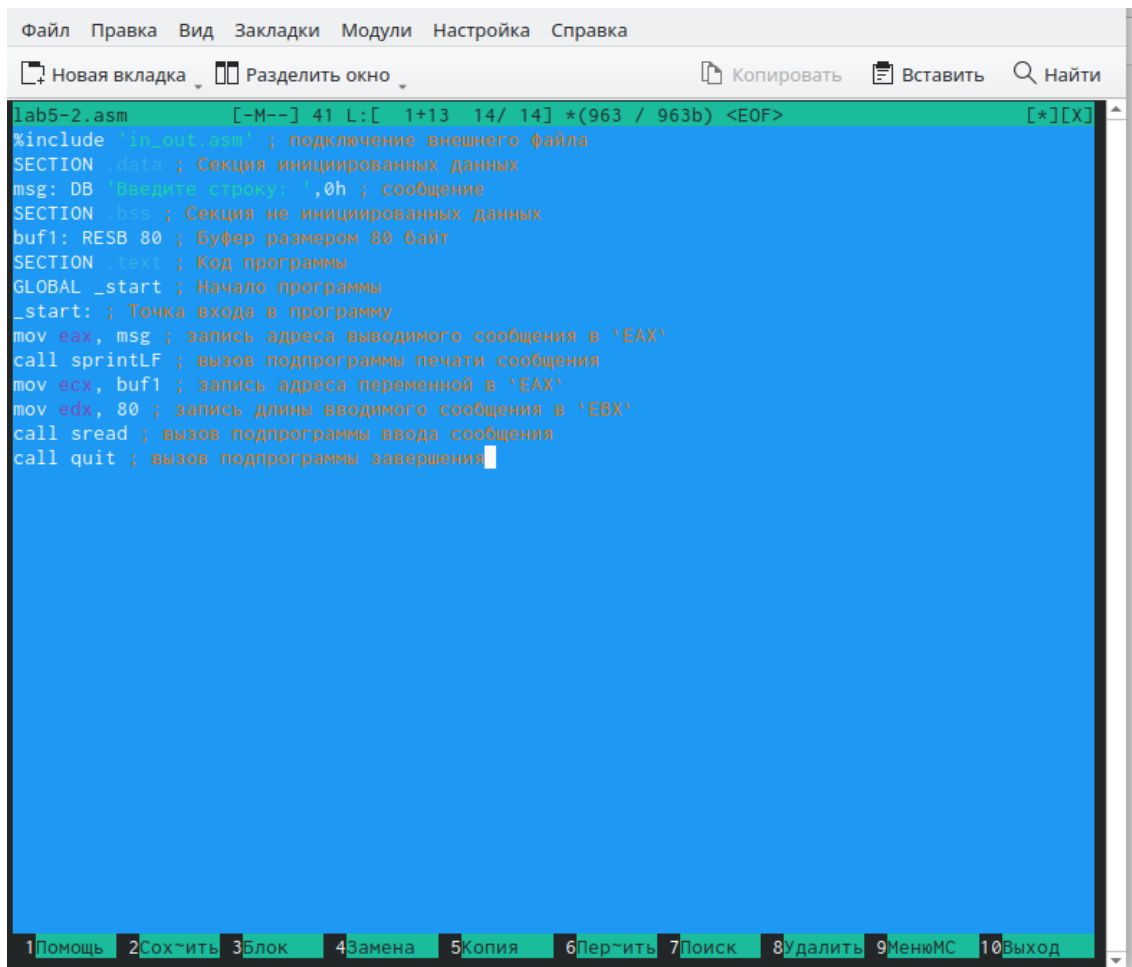
Рис. 4.4: Используем клавишу F4

5. Скачиваем файл in_out.asm со страницы курса в ТУИС.
6. Скопируем файл in_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5.
7. Создаем копию файла lab5-1.asm с именем lab5-2.asm. (рис. 4.5)

Левая панель				Правая панель			
Файл		Команда		Файл		Команда	
Имя	Размер	Время	правки	Имя	Размер	Время	правки
./	-ВВЕРХ-	дек	1 12:41	./	-ВВЕРХ-	дек	1 12:41
in_out.asm	3942	дек	1 12:40	in_out.asm	3942	дек	1 12:40
lab5-1	8744	дек	1 12:38	lab5-1	8744	дек	1 12:38
lab5-1.asm	1880	дек	1 12:36	lab5-1.asm	1880	дек	1 12:36
lab5-1.o	752	дек	1 12:38	lab5-1.o	752	дек	1 12:38
lab5-2.asm	1880	дек	1 12:36	lab5-2.asm	1880	дек	1 12:36

Рис. 4.5: Используем клавишу F5

8. Вводим текст программы вывода сообщения на экран и ввода строки с клавиатуры. (рис. 4.6)

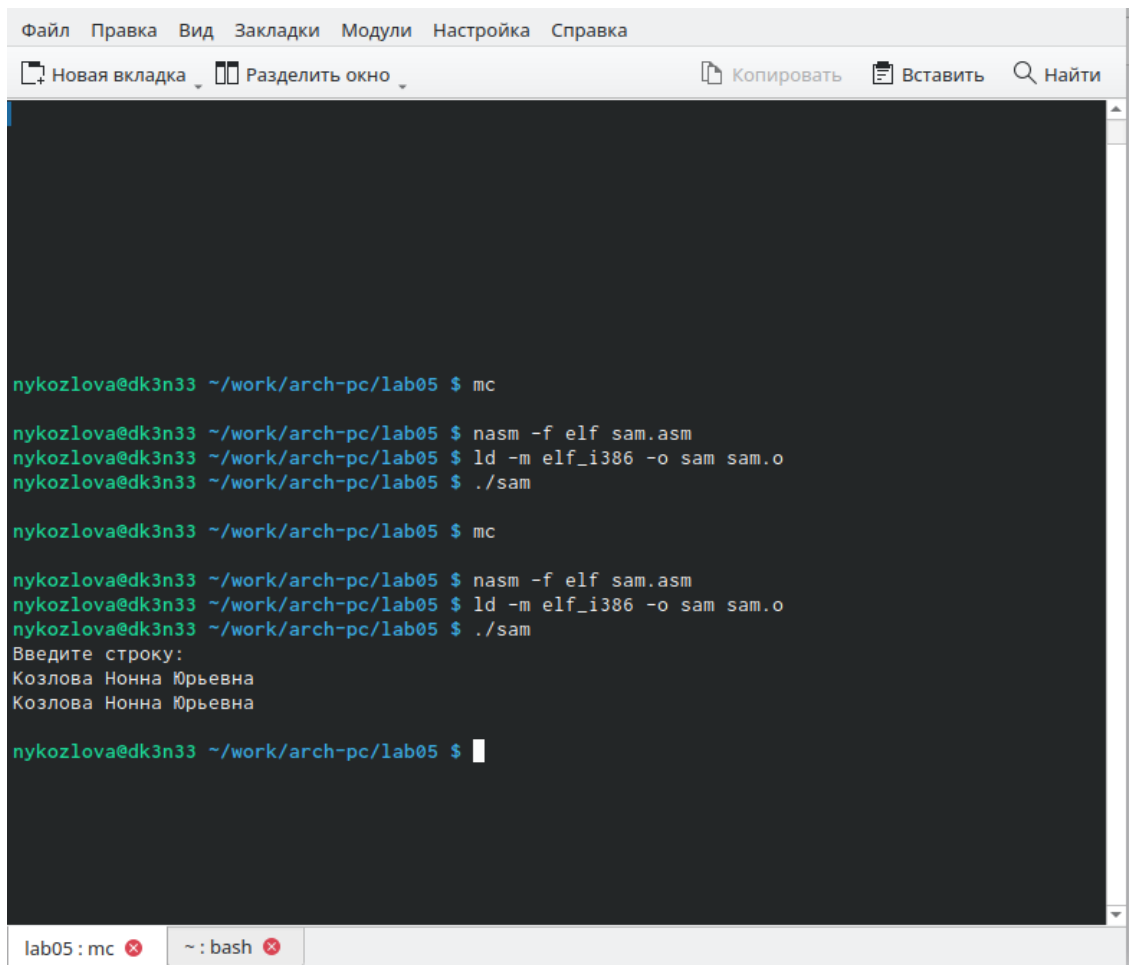


```
lab5-2.asm      [-M--] 41 L:[ 1+13 14/ 14] *(963 / 963b) <EOF>      [*][X]
#include "in_out.asm" ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB "Введите строку: ",0h ; сообщение
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 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 sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
call sread ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перенести 7Поиск 8Удалить 9МенюМС 10Выход

Рис. 4.6: С использованием файла in_out.asm

9. Создаем исполняемый файл и проверяем его работу. (рис. 4.7)



```
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
Новая вкладка  Разделить окно  Копировать  Вставить  Найти

nykozlova@dk3n33 ~/work/arch-pc/lab05 $ mc

nykozlova@dk3n33 ~/work/arch-pc/lab05 $ nasm -f elf sam.asm
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o sam sam.o
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ./sam

nykozlova@dk3n33 ~/work/arch-pc/lab05 $ mc

nykozlova@dk3n33 ~/work/arch-pc/lab05 $ nasm -f elf sam.asm
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o sam sam.o
nykozlova@dk3n33 ~/work/arch-pc/lab05 $ ./sam
Введите строку:
Козлова Нонна Юрьевна
Козлова Нонна Юрьевна

nykozlova@dk3n33 ~/work/arch-pc/lab05 $
```

lab05 : mc ~ : bash

Рис. 4.7: Программа работает верно

5 Выводы

В ходе лабораторной работы я поняла логику программ на языке ассемблера NASM. Приобрела практические навыки работы в Midnight Commander. Освоила инструкции языка ассемблера `mov` и `int`.

Список литературы