

Отчёт по лабораторной работе №5

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

Малкина Дарья Александровна

Содержание

| | | |
|----------|------------------------------------------|-----------|
| 1 | Цель работы | 5 |
| 2 | Выполнение лабораторной работы | 6 |
| 3 | Выполнение самостоятельной работы | 11 |
| 4 | Вывод | 13 |

Список иллюстраций

| | | |
|-----|----------------------------------------------------|----|
| 2.1 | Открываем Midnight Commander | 6 |
| 2.2 | Создаем каталог lab05 | 7 |
| 2.3 | Редактируем файл lab5-1.asm | 7 |
| 2.4 | Проверяем изменения | 8 |
| 2.5 | Запускаем исполняемый файл lab5-1 | 8 |
| 2.6 | Копируем файл in_out.asm в каталок lab05 | 9 |
| 2.7 | Создаем копию файла lab5-1.asm | 9 |
| 2.8 | Запускаем исполняемый файл lab5-2 | 10 |
| 3.1 | Редактируем файл lab5-3.asm | 11 |
| 3.2 | Запускаем исполняемый файл lab5-3 | 11 |
| 3.3 | Редактируем файл lab5-4.asm | 12 |
| 3.4 | Запускаем исполняемый файл lab5-4 | 12 |

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

1. Открываем Midnight Commander:

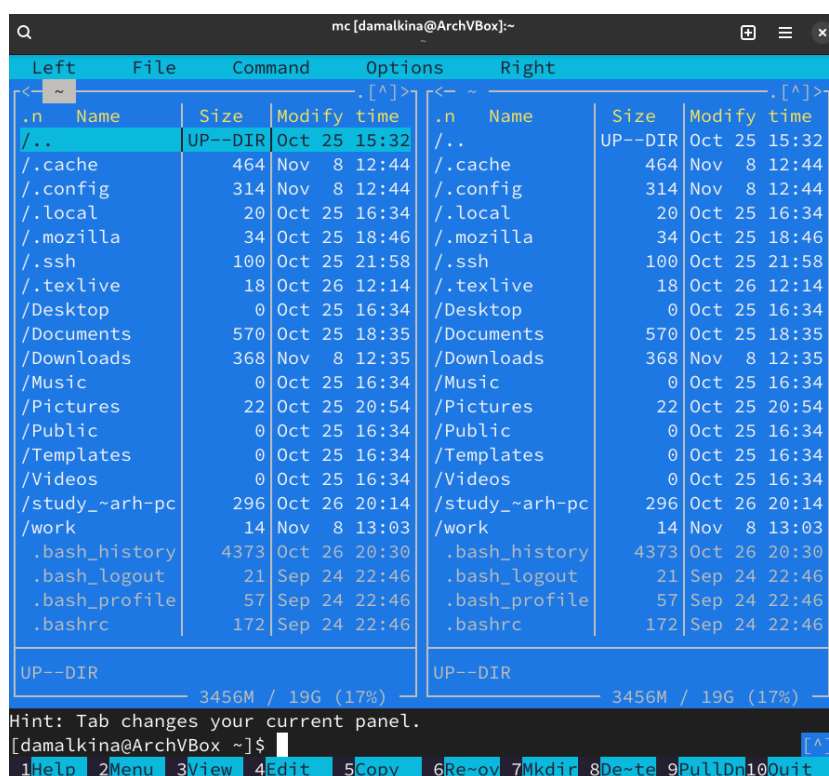


Рис. 2.1: Открываем Midnight Commander

2. Переходим в каталог ~/work/arch-rc и создаем папку lab05, переходим в созданный каталог:

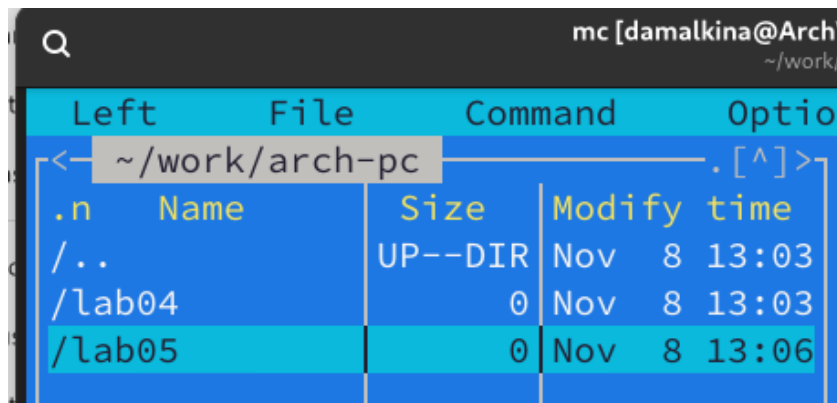


Рис. 2.2: Создаем каталог lab05

- В каталоге lab05 с помощью команды touch создаем файл lab5-1.asm, открываем его для редактирования во встроенном редакторе и вводим текст программы:

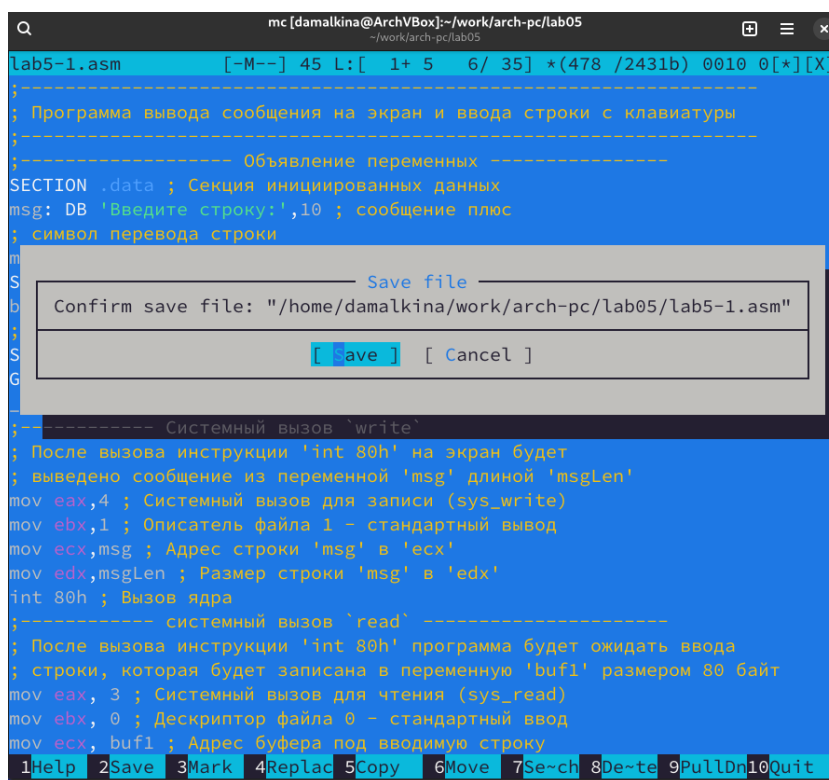
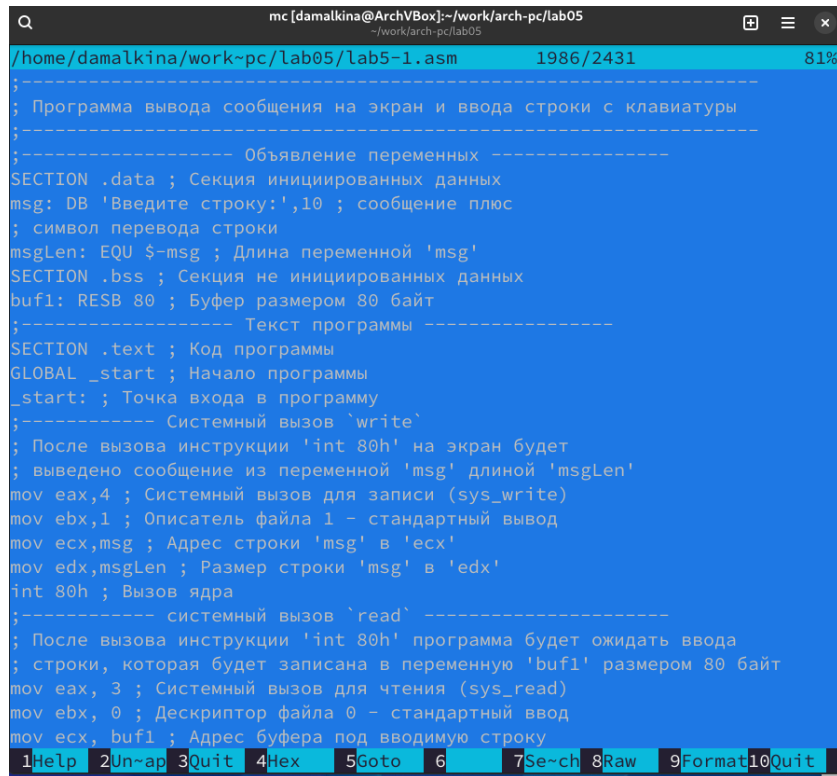


Рис. 2.3: Редактируем файл lab5-1.asm

- Сохраняем изменения и открываем файл для просмотра, чтобы убедиться,

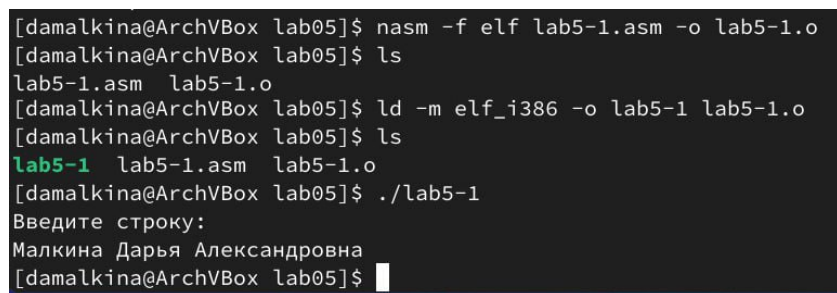
что файл содержит текст программы:



```
Q mc [damalkina@ArchVBox]:~/work/arch-pc/lab05
~/work/arch-pc/lab05
/home/damalkina/work-pc/lab05/lab5-1.asm 1986/2431 81%
;-----
; 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 ;Descriptor файла 0 - стандартный ввод
mov ecx,buf1 ; Адрес буфера под вводимую строку
1Help 2Un-ap 3Quit 4Hex 5Goto 6 7Se-ch 8Raw 9Format10Quit
```

Рис. 2.4: Проверяем изменения

5. Транслируем текст программы в объектный файл, выполняем компоновку и запускаем получившийся исполняемый файл:



```
[damalkina@ArchVBox lab05]$ nasm -f elf lab5-1.asm -o lab5-1.o
[damalkina@ArchVBox lab05]$ ls
lab5-1.asm lab5-1.o
[damalkina@ArchVBox lab05]$ ld -m elf_i386 -o lab5-1 lab5-1.o
[damalkina@ArchVBox lab05]$ ls
lab5-1 lab5-1.asm lab5-1.o
[damalkina@ArchVBox lab05]$ ./lab5-1
Введите строку:
Малкина Дарья Александровна
[damalkina@ArchVBox lab05]$
```

Рис. 2.5: Запускаем исполняемый файл lab5-1

6. Скачиваем файл in_out.asm и копируем его в каталог с файлом lab5-1.asm:

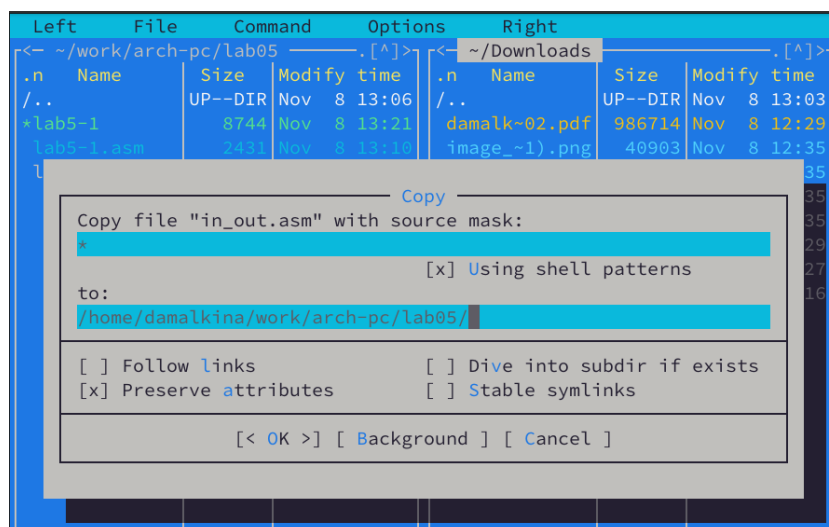


Рис. 2.6: Копируем файл in_out.asm в каталог lab05

7. Создаем копию файла lab5-1.asm с именем lab5-2.asm, исправляем текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm и заменяем подпрограмму sprintLF на sprint:

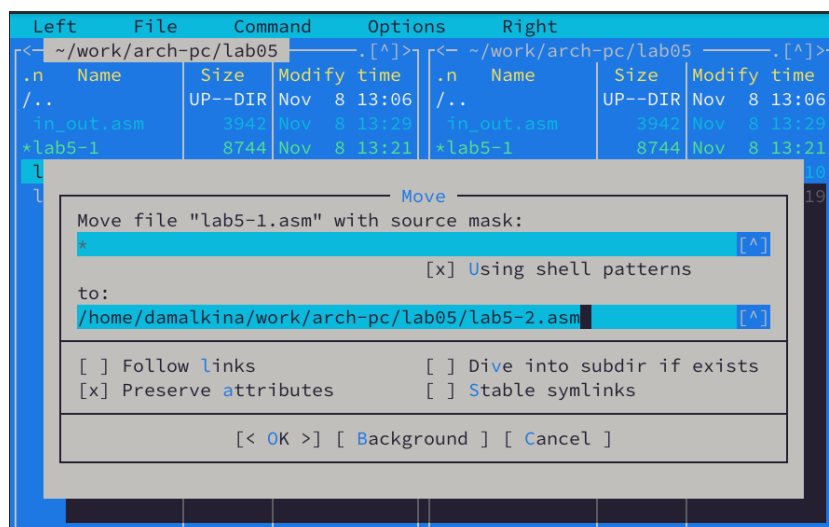


Рис. 2.7: Создаем копию файла lab5-1.asm

8. Создаем исполняемый файл и проверяем работу, заметим, что работа программы lab5-2 отличается от работы lab5-1, а именно отсутствует отступ после сообщения "Введите текст":

```
[damalkina@ArchVBox lab05]$ nasm -f elf lab5-2.asm -o lab5-2.o
[damalkina@ArchVBox lab05]$ ls
in_out.asm lab5-1 lab5-1.asm lab5-1.o lab5-2.asm lab5-2.o
[damalkina@ArchVBox lab05]$ ld -m elf_i386 -o lab5-2 lab5-2.o
[damalkina@ArchVBox lab05]$ ls
in_out.asm lab5-1 lab5-1.asm lab5-1.o lab5-2 lab5-2.asm lab5-2.o
[damalkina@ArchVBox lab05]$ ./lab5-2
Введите строку: Малкина Дарья Александровна
[damalkina@ArchVBox lab05]$
```

Рис. 2.8: Запускаем исполнительный файл lab5-2

3 Выполнение самостоятельной работы

1. Создаем копию файла lab5-1.asm с именем lab5-3.asm, после блока read добавляем в текст программы блок write, чтобы программа выводила введенную строку на экран:

```
----- Системный вызов 'write' -----  
mov eax, 4 ; Системный вызов write  
mov ebx, 1 ;Descriptor файла 1 - стандартный вывод  
mov ecx, buf1 ; Адрес выводимой строки  
mov edx, 80 ; Длина выводимой строки  
int 80h ; Вызов ядра  
----- Системный вызов 'exit' -----
```

Рис. 3.1: Редактируем файл lab5-3.asm

2. Создаем исполняемый файл и проверяем его работу:

```
[damalkina@ArchVBox lab05]$ nasm -f elf lab5-3.asm -o lab5-3.o  
[damalkina@ArchVBox lab05]$ ls  
in_out.asm lab5-1.asm lab5-2 lab5-2.o lab5-3.o  
lab5-1 lab5-1.o lab5-2.asm lab5-3.asm  
[damalkina@ArchVBox lab05]$ ld -m elf_i386 -o lab5-3 lab5-3.o  
[damalkina@ArchVBox lab05]$ ls  
in_out.asm lab5-1.asm lab5-2 lab5-2.o lab5-3.asm  
lab5-1 lab5-1.o lab5-2.asm lab5-3 lab5-3.o  
[damalkina@ArchVBox lab05]$ ./lab5-3  
Введите строку:  
Малкина Дарья Александровна  
Малкина Дарья Александровна  
[damalkina@ArchVBox lab05]$
```

Рис. 3.2: Запускаем исполняемый файл lab5-3

3. Создаем копию файла lab5-2.asm с именем lab5-4.asm, исправляем текст программы, добавляя подпрограмму `sprint` из внешнего файла `in_out.asm`, чтобы программа выводила введенную строку на экран:

```
mov edx, 80 ; запись длины вводимого сообщения в 'EDX'
call sread ; вызов подпрограммы ввода сообщения
mov eax, buf1 ; Передать адрес буфера в eax
call sprint ; вывод текста сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 3.3: Редактируем файл lab5-4.asm

4. Создаем исполняемый файл и проверяем его работу:

```
[damalkina@ArchVBox lab05]$ ls
in_out.asm lab5-1.asm lab5-2 lab5-2.o lab5-3.asm lab5-4.asm
lab5-1 lab5-1.o lab5-2.asm lab5-3 lab5-3.o
[damalkina@ArchVBox lab05]$ nasm -f elf lab5-4.asm -o lab5-4.o
[damalkina@ArchVBox lab05]$ ls
in_out.asm lab5-1.asm lab5-2 lab5-2.o lab5-3.asm lab5-4.asm
lab5-1 lab5-1.o lab5-2.asm lab5-3 lab5-3.o lab5-4.o
[damalkina@ArchVBox lab05]$ ld -m elf_i386 -o lab5-4 lab5-4.o
[damalkina@ArchVBox lab05]$ ls
in_out.asm lab5-1.o lab5-2.o lab5-3.o lab5-4.o
lab5-1 lab5-2 lab5-3 lab5-4
lab5-1.asm lab5-2.asm lab5-3.asm lab5-4.asm
[damalkina@ArchVBox lab05]$ ./lab5-4
Введите строку: Малкина Дарья Александровна
Малкина Дарья Александровна
[damalkina@ArchVBox lab05]$
```

Рис. 3.4: Запускаем исполняемый файл lab5-4

4 Вывод

В ходе выполнения лабораторной работы мы познакомились с Midnight Commander и приобрели навыки работы с ним. Освоили инструкции языка ассемблера `mov` и `int`, в процессе написания программ, а также научились работать с подпрограммами из внешнего файла.