

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра компьютерных и информационных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

дисциплина: Архитектура компьютера

Студент: Симакова Алина

Группа: НКАбд-05-25

МОСКВА

2025 г.

Оглавление

1 Цель работы.....	3
2 Задание.....	4
3 Теоретическое введение.....	5
4 Выполнение лабораторной работы.....	7
4.1 Основы работы с Midnight Commander.....	7
4.2 Работа в NASM.....	12
4.3 Подключение внешнего файла.....	14
5 Задания для самостоятельной работы.....	16
6 Выводы.....	20
Список литературы.....	21

1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в Midnight Commander, освоение инструкций языка ассемблера `mov` и `int`.

2 Задание

На основе методических указаний освоить основы работы с тс, изучить структуру программы на языке ассемблера NASM, провести подключение внешнего файла и выполнить задания 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. Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция иницированных (известных во время компиляции) данных (SECTION .data) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (SECTION .bss). Для объявления иницированных данных в секции .data используются директивы DB, DW, DD, DQ и DT, которые резервируют память и указывают, какие значения должны храниться в этой памяти:

- DB (define byte) — определяет переменную размером в 1 байт;
- DW (define word) — определяет переменную размером в 2 байта (слово);
- DD (define double word) — определяет переменную размером в 4 байта (двойное слово);
- DQ (define quad word) — определяет переменную размером в 8 байт (учетверённое слово);
- DT (define ten bytes) — определяет переменную размером в 10 байт.

Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву DB в связи с особенностями хранения данных в оперативной памяти. Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике.

mov dst,src

Здесь операнд dst — приёмник, а src — источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непосредственные значения (const). Инструкция языка ассемблера int предназначена для вызова прерывания с указанным номером.

int n

Здесь n — номер прерывания, принадлежащий диапазону 0 – 255. При программировании в Linux с использованием вызовов ядра `sys_calls` $n=80h$ (принято задавать в шестнадцатеричной системе счисления).

4 Выполнение лабораторной работы

4.1 Основы работы с Midnight Commander

В домашней директории ввожу соответствующую команду в терминале (рис. 4.1) и открываю Midnight Commander (рис. 4.2).



Рис. 4.1: Открытие Midnight Commander

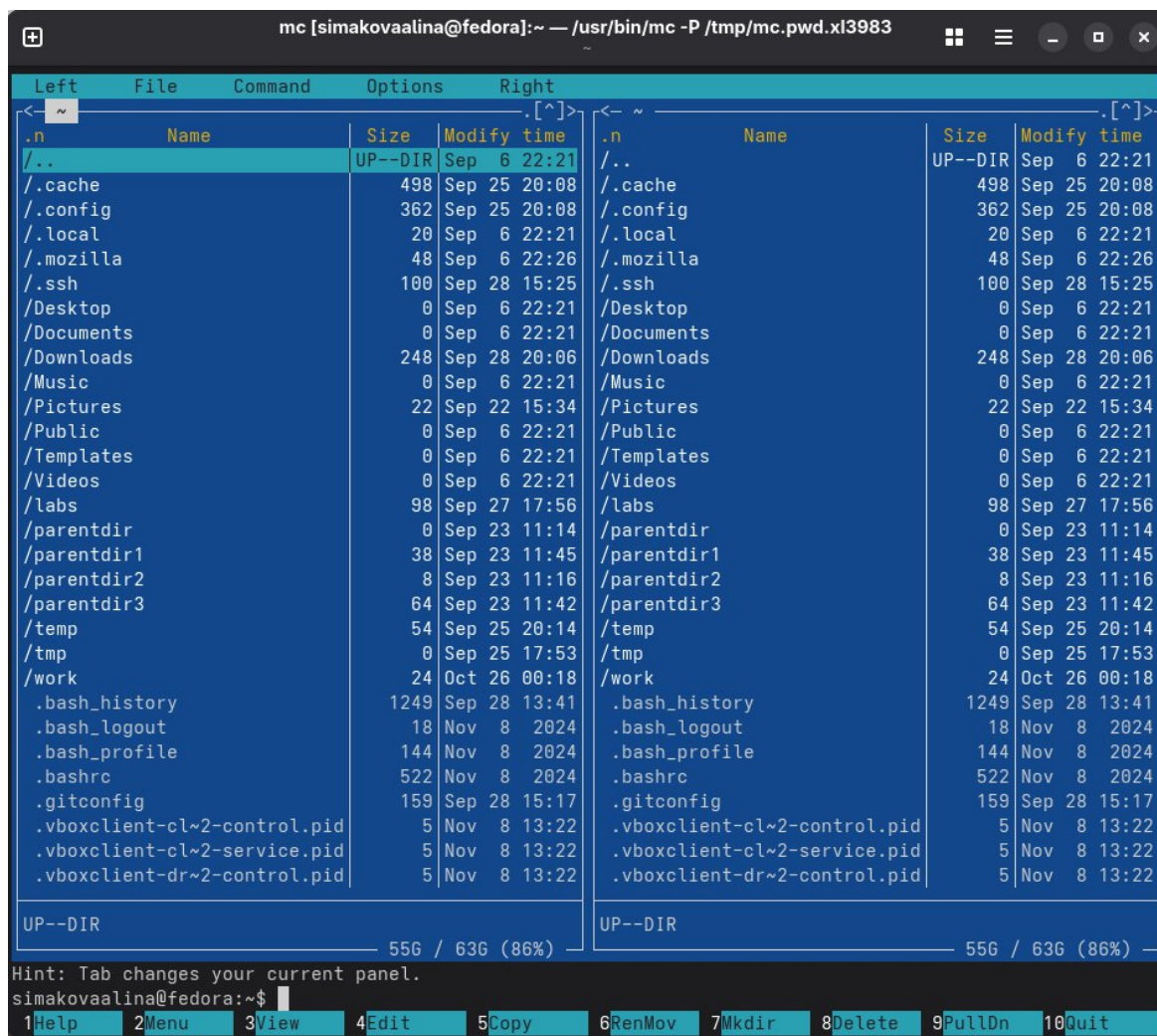


Рис. 4.2: Интерфейс Midnight Commander

Перехожу в созданный каталог в предыдущей лабораторной работе (рис. 4.3).

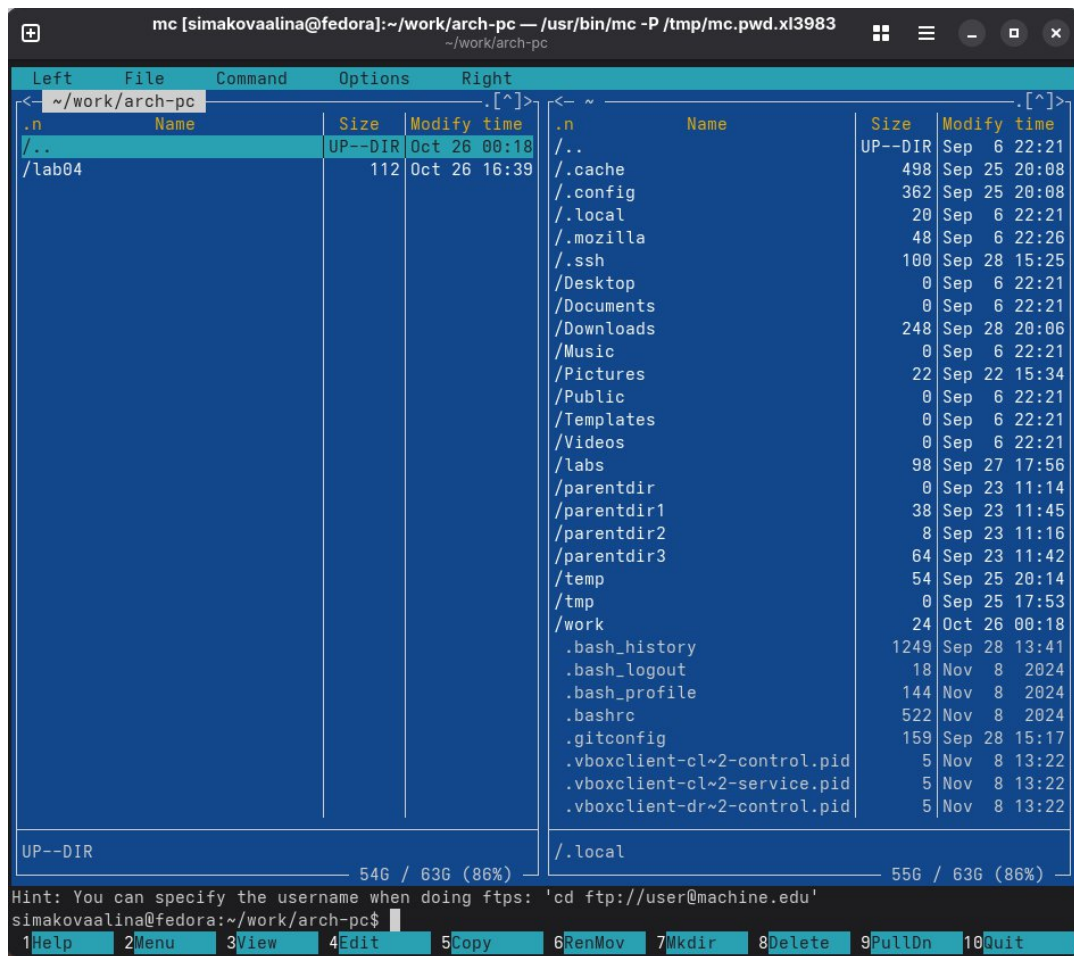


Рис. 4.3: Открытый каталог arch-pc

Далее с помощью функциональной клавиши, создаю подкаталог lab05, в котором буду работать (рис. 4.4).

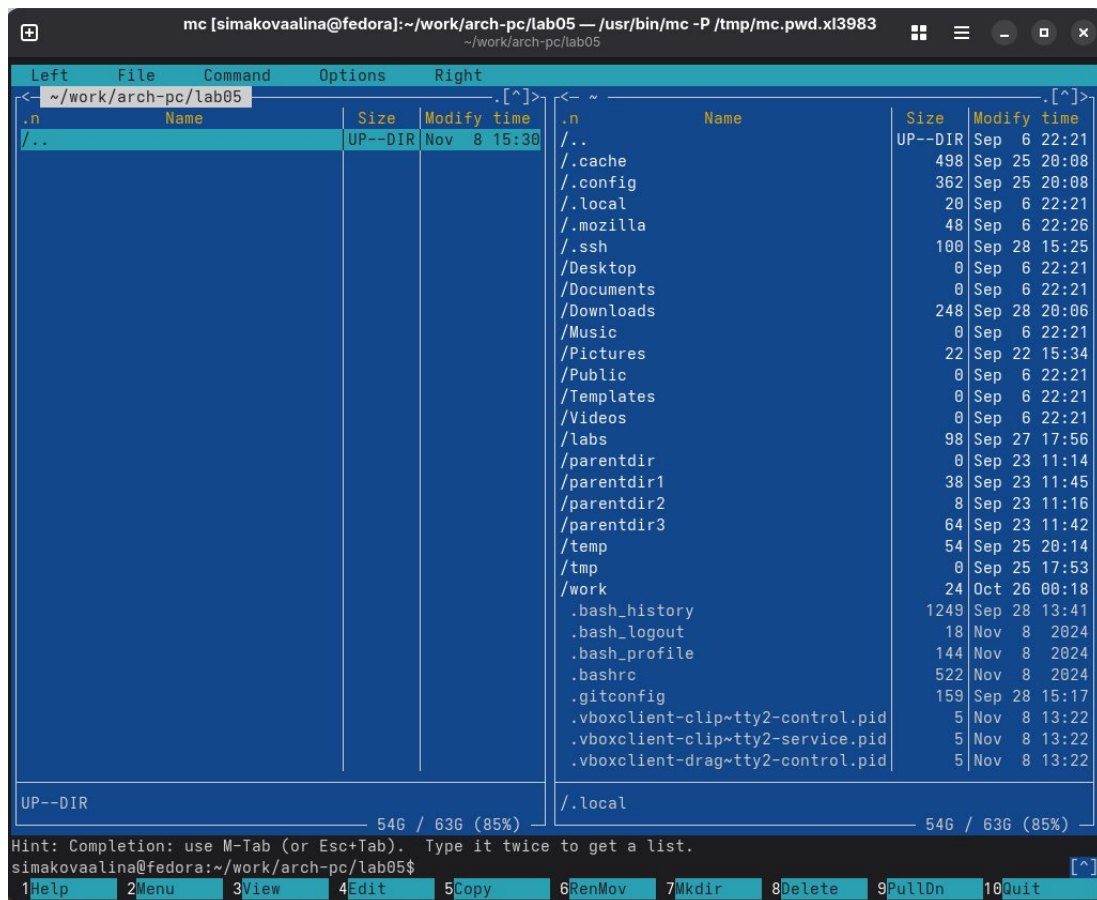


Рис. 4.4: Создание рабочего подкаталога

В строке ввода ввожу команду touch и создаю файл (рис. 4.5).

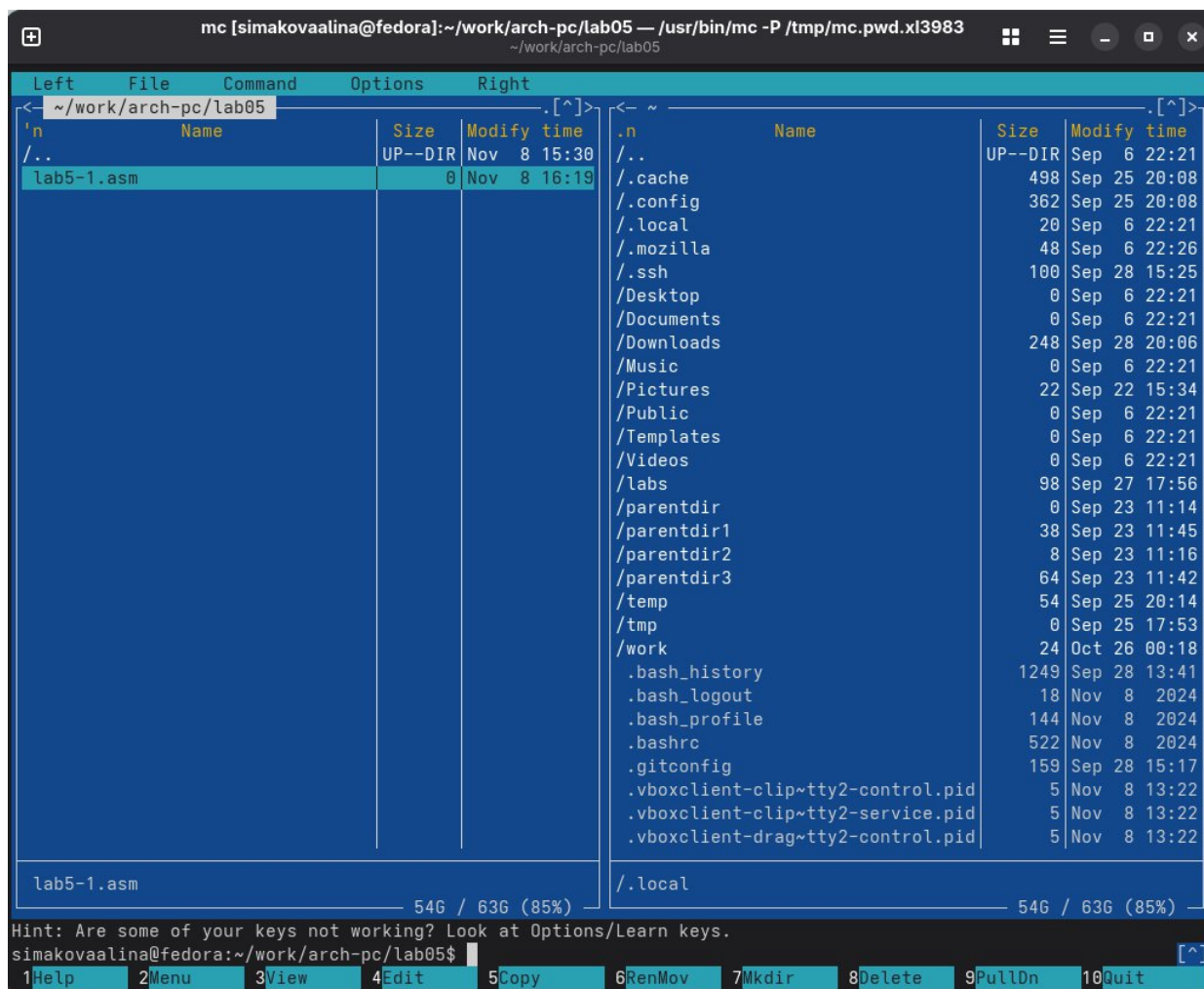


Рис. 4.5: Создание файла в Midnight Commander

4.2 Работа в NASM

С помощью F4 открываю только что созданный файл и вношу код с листинга (рис. 4.6).

```
lab5-1.asm [-M--] 15 L: [ 1+23 24/ 25] *(388 / 389b) 0010 0x00A [*][X]
SECTION .data
msg: DB 'Введите строку', 10
msgLen: EQU $-msg

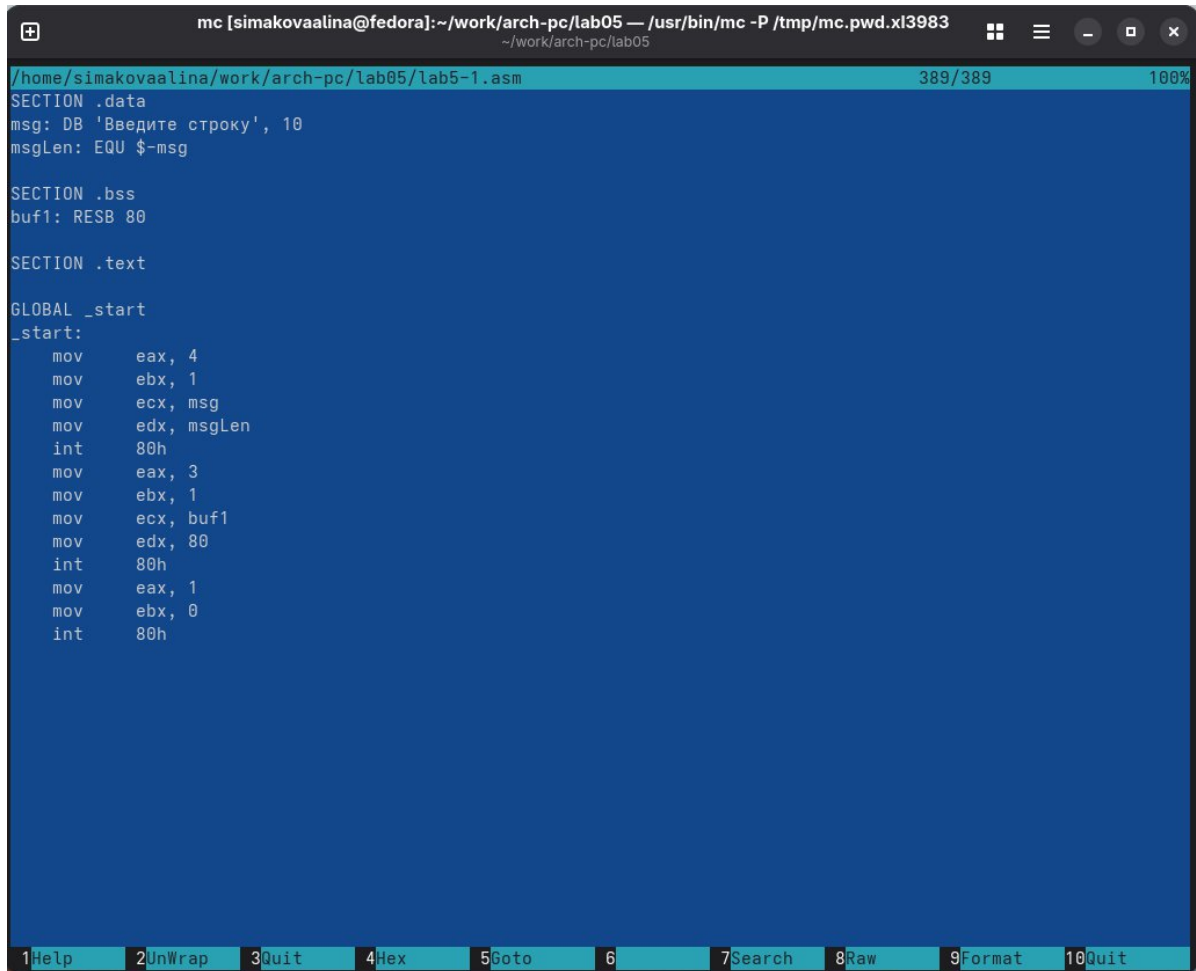
SECTION .bss
buf1: RESB 80

SECTION .text

GLOBAL _start
_start:
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, msg
    mov     edx, msgLen
    int     80h
    mov     eax, 3
    mov     ebx, 1
    mov     ecx, buf1
    mov     edx, 80
    int     80h
    mov     eax, 1
    mov     ebx, 0
    int     80h
```

Рис. 4.6: Редактирование файла в Midnight Commander

Затем проверяю сохраненные изменения с помощью клавиши F3 (рис. 4.7).



```
mc [simakovaalina@fedora]:~/work/arch-pc/lab05 — /usr/bin/mc -P /tmp/mc.pwd.xl3983
~/work/arch-pc/lab05

/home/simakovaalina/work/arch-pc/lab05/lab5-1.asm 389/389 100%
SECTION .data
msg: DB 'Введите строку', 10
msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

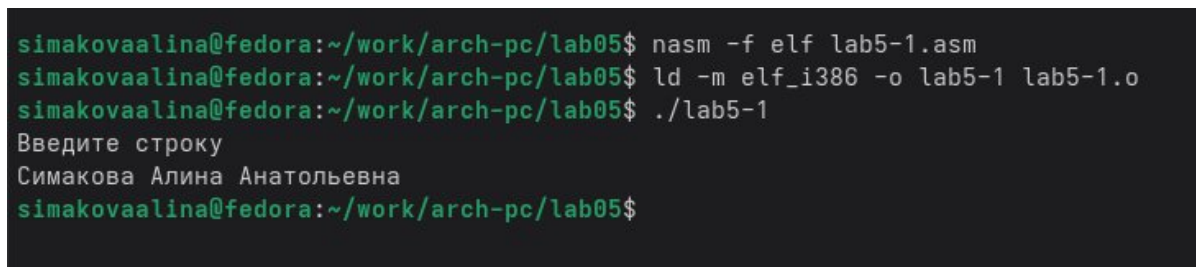
SECTION .text

GLOBAL _start
_start:
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, msg
    mov     edx, msgLen
    int     80h
    mov     eax, 3
    mov     ebx, 1
    mov     ecx, buf1
    mov     edx, 80
    int     80h
    mov     eax, 1
    mov     ebx, 0
    int     80h

1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format 10Quit
```

Рис. 4.7: Проверка сохранения сделанных изменений

Транслирую и компоную измененный файл, после запускаю (рис. 4.8).



```
simakovaalina@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
simakovaalina@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
simakovaalina@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку
Симакова Алина Анатольевна
simakovaalina@fedora:~/work/arch-pc/lab05$
```

Рис. 4.8: Трансляция, компоновка и последующий запуск программы

4.3 Подключение внешнего файла

Скачанный с ТУИС файл сохраняю в общую папку на своем компьютере, на виртуальной машине в интерфейсе Midnight Commander перехожу в директорию общей папки, копирую файл в рабочий подкаталог. (рис. 4.9).

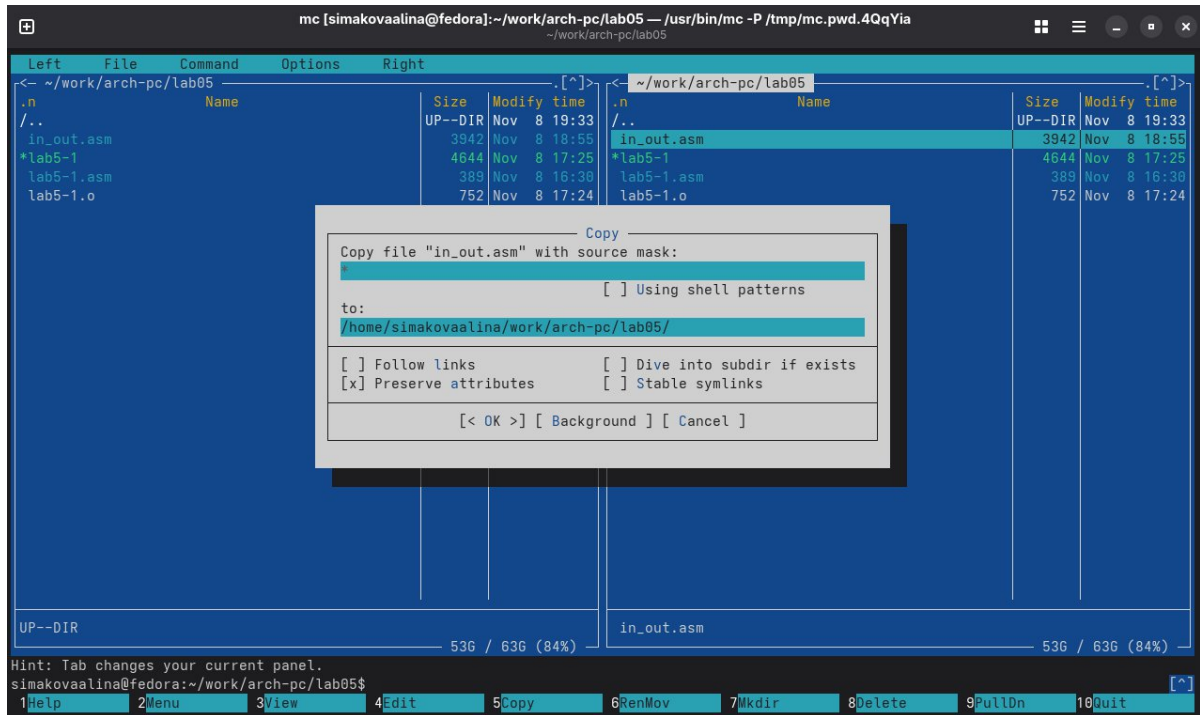


Рис. 4.9: Копирование файла в рабочий каталог

После этого создаю копию файла для последующей работы с ним (рис. 4.10)

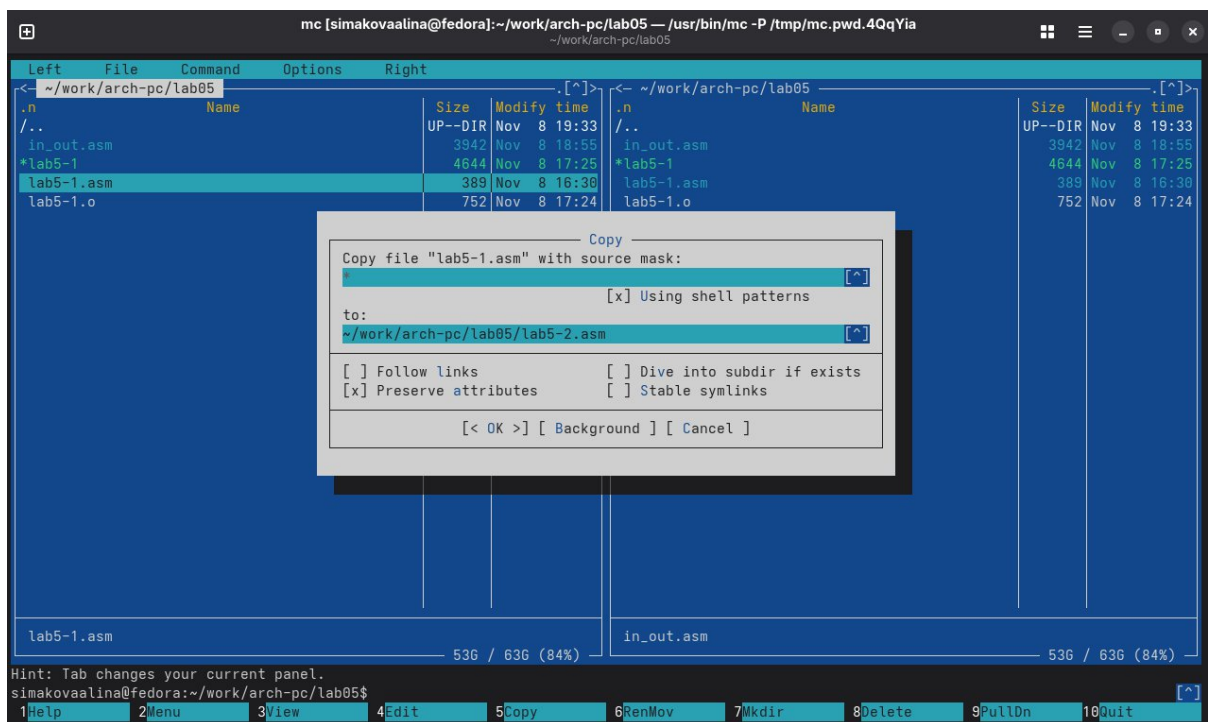


Рис 4.10 Создание копии файла в Midnight Commander

В копии файла подключаю подпрограмму из подключенного файла (рис. 4.11).

```
%include 'in_out.asm'

SECTION .data

msg: DB 'Введите строку: ', 0h

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

    mov eax, msg
    call sprintLF

    mov ecx, buf1
    mov edx, 80

    call sread
```

Рис 4.11: Изменение программы

Далее транслирую, компоную и запускаю программу с подключенным файлом (рис. 4.12).

```
simakovaalina@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Симакова Алина Анатольевна
simakovaalina@fedora:~/work/arch-pc/lab05$
```

Рис 4.12: Запуск измененной программы

Редактирую файл и заменяю в нём подпрограмму sprintLF на sprint. Разница подпрограмма в том, что вторая вызывает ввод на той же строке (рис. 4.13).

```
simakovaalina@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
simakovaalina@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
simakovaalina@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Симакова Алина Анатольевна
simakovaalina@fedora:~/work/arch-pc/lab05$
```

Рис. 4.13: Запуск измененной программы с другой подпрограммой

5 Задания для самостоятельной работы

Для начала создаю копию lab5-1.asm и редактирую так, чтобы в конце выводилась введенная мною строка с клавиатуры (рис. 5.1).

```
SECTION .data
msg: DB 'Введите строку:', 10
msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text

GLOBAL _start
_start:
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, msg
    mov     edx, msgLen
    int     80h
    mov     eax, 3
    mov     ebx, 0
    mov     ecx, buf1
    mov     edx, 80
    int     80h
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, buf1
    mov     edx, buf1
    int     80h
    mov     eax, 1
    mov     ebx, 0
    int     80h
```

Рис. 5.1: – Редактирование копии

Затем транслирую, компоную и запускаю программу (рис. 5.2).

```
simakovaalina@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1copy.asm
simakovaalina@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1copy lab5-1copy.o
simakovaalina@fedora:~/work/arch-pc/lab05$ ./lab5-1copy
Введите строку:
Симакова Алина Анатольевна
Симакова Алина Анатольевна
simakovaalina@fedora:~/work/arch-pc/lab05$
```

Рис. 5.2: – Редактирование копии

Прикладываю код:

```
SECTION .data
```

```
msg: DB 'Введите строку:', 10
```



```
msgLen: EQU $-msg
```

```
SECTION .bss
```

```
buf1: RESB 80
```

```
SECTION .text
```

```
GLOBAL _start
```

```
_start:
```

```
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, msg
    mov     edx, msgLen
    int     80h
    mov     eax, 3
    mov     ebx, 0
    mov     ecx, buf1
    mov     edx, 80
    int     80h
    mov     eax, 4
    mov     ebx, 1
    mov     ecx, buf1
    mov     edx, buf1
    int     80h
    mov     eax, 1
    mov     ebx, 0
    int     80h
```

Убедившись в корректности работы программы, создаю копию lab5-2.asm, редактирую так, чтобы в конце выводилась введенная мною строка с клавиатуры. (рис. 5.3)

```

#include 'in_out.asm'

SECTION .data
msg: DB 'Введите строку:', 0h
msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

    mov eax, msg
    call sprint

    mov ecx, buf1
    mov edx, 80

    call sread

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

    call quit

```

Рис. 5.3 – Редактирование копии

Транслирую, компоную и запускаю свою программу (рис. 5.4).

```

simakovaalina@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2copy.asm
simakovaalina@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2copy lab5-2copy.o
simakovaalina@fedora:~/work/arch-pc/lab05$ ./lab5-2copy
Введите строку: Симакова Алина Анатольевна
Симакова Алина Анатольевна
simakovaalina@fedora:~/work/arch-pc/lab05$

```

Рис. 5.4: Запуск своей программы

Прикладываю код:

```

#include 'in_out.asm'

SECTION .data
msg: DB 'Введите строку: ', 0h
msgLen: EQU $-msg

```

SECTION .bss

buf1: **RESB** 80

SECTION .text

GLOBAL _start

_start:

mov **eax**, msg

call sprint

mov **ecx**, buf1

mov **edx**, 80

call sread

mov **eax**, 4

mov **ebx**, 1

mov **ecx**, buf1

int 80h

call quit

6 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы в Midnight Commander, а также освоила инструкции языка ассемблера `mov` и `int`.

Список литературы

1. <https://esystem.rudn.ru/mod/resource/view.php?id=1030495>
2. https://esystem.rudn.ru/pluginfile.php/2089085/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%965.%20%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B%20%D1%81%20Midnight%20Commander%20%28%29.%20%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D1%8B%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM.%20%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D1%8B%D0%B5%20%D0%B2%D1%8B%D0%B7%D0%BE%D0%B2%D1%8B%20%D0%B2%20%D0%9E%D0%A1%20GNU%20Linux.pdf
3. <https://esystem.rudn.ru/mod/page/view.php?id=1030492>
4. <https://esystem.rudn.ru/mod/resource/view.php?id=1030496>