

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Козин Иван Евгеньевич

Содержание

1 Цель работы

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

2 Задание

1. Основы работы с `mc`
2. Структура программы на языке ассемблера NASM
3. Подключение внешнего файла
4. Выполнение заданий 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 инициированных (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 инициированных 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

Введя соответствующую команду в терминале (рис. 1), я открываю Midnight Commander (рис. 2).

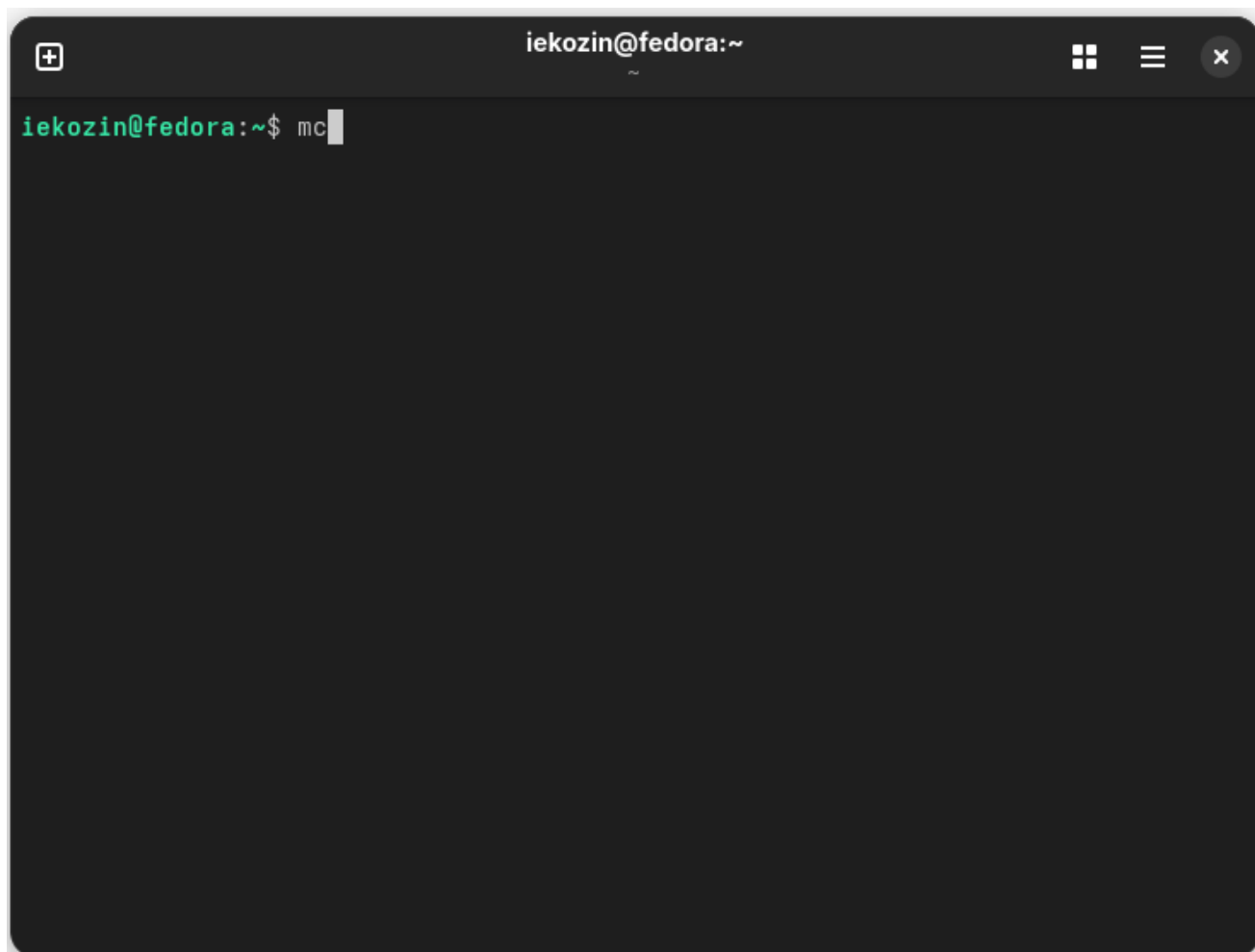


Рис. 1: Открытие Midnight Commander



Рис. 2: Интерфейс Midnight Commander

Перехожу в созданный каталог в предыдущей лабораторной работе (рис. 3).

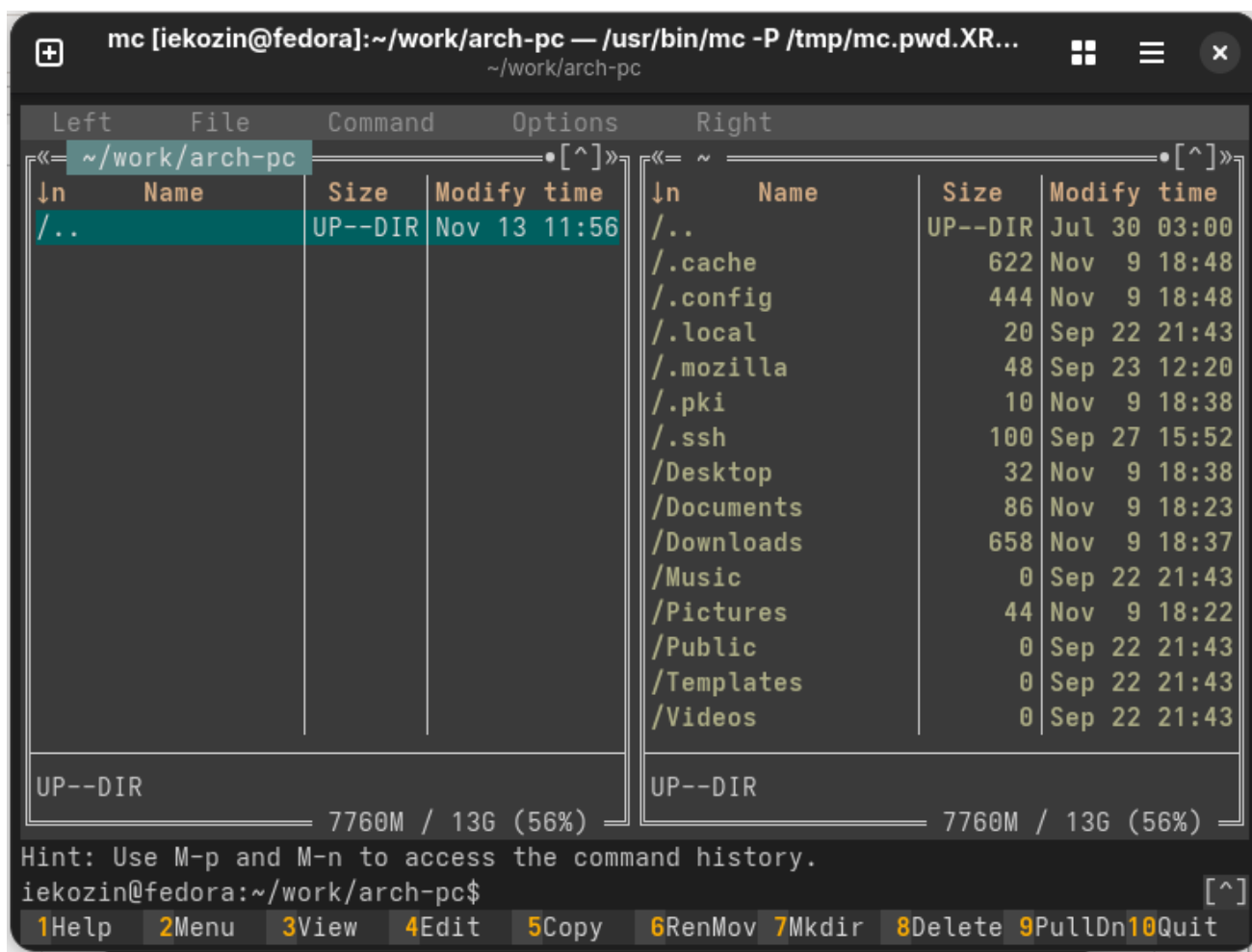


Рис. 3: Открытый каталог arch-pc

С помощью функциональной клавиши, я создаю подкаталог lab05, в котором буду работать (рис. 4). это делается при помощи F7

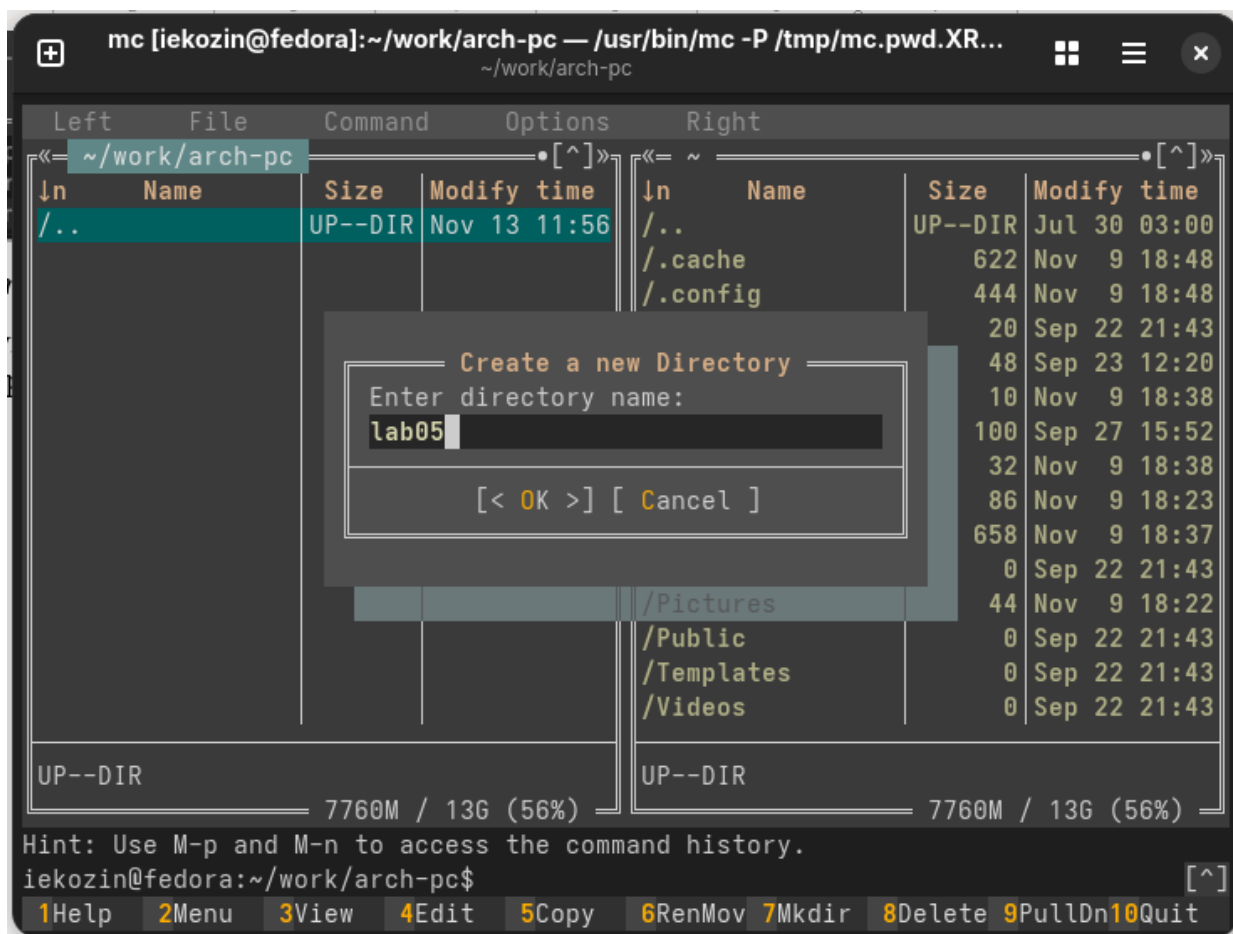


Рис. 4: Создание рабочего подкаталога

В строке ввода вводжу команду touch и создаю файл (рис. 5).

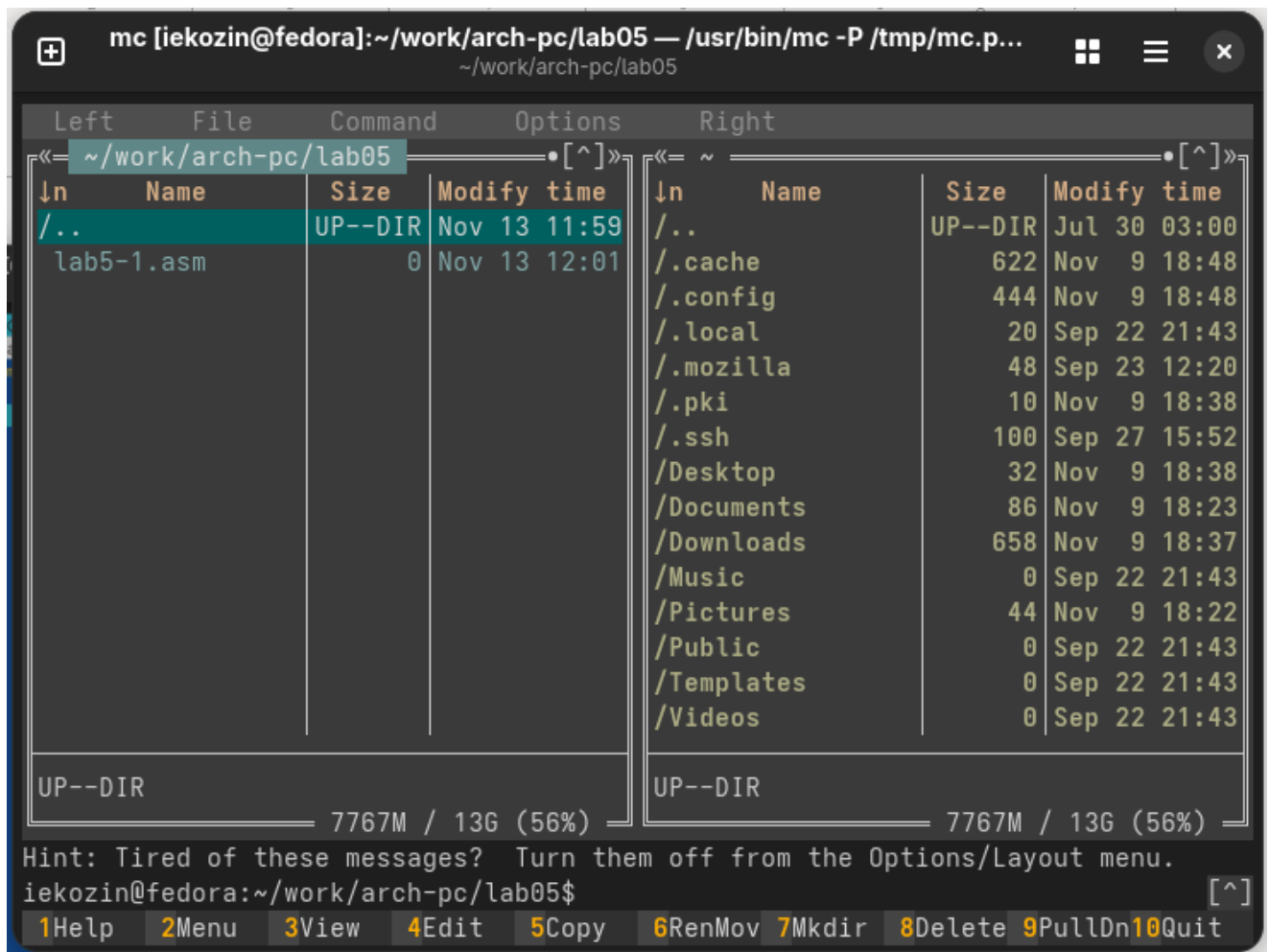


Рис. 5: Создание файла в Midnight Commander

4.2 Работа в NASM

С помощью F4 открываю только что созданный файл и вношу код с листинга (рис. 6).

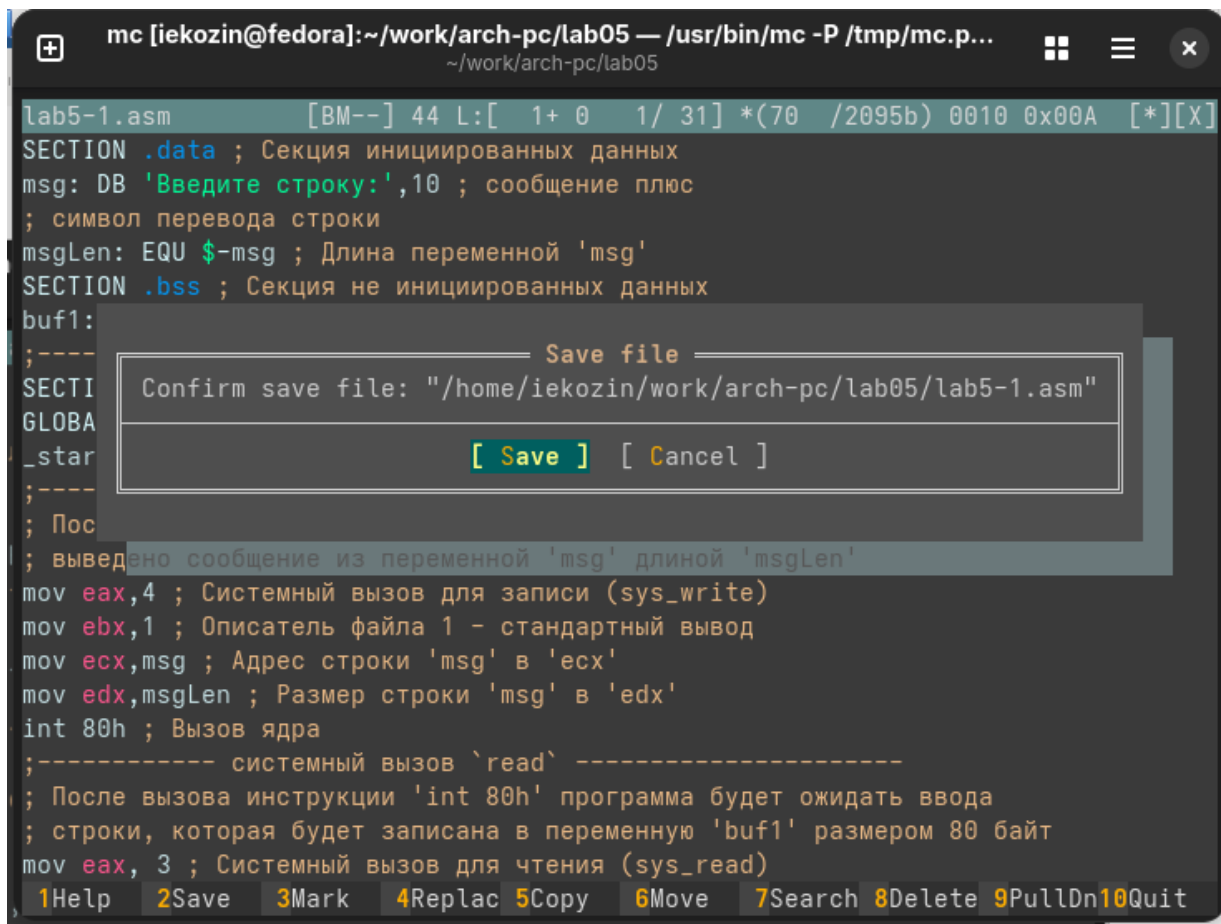
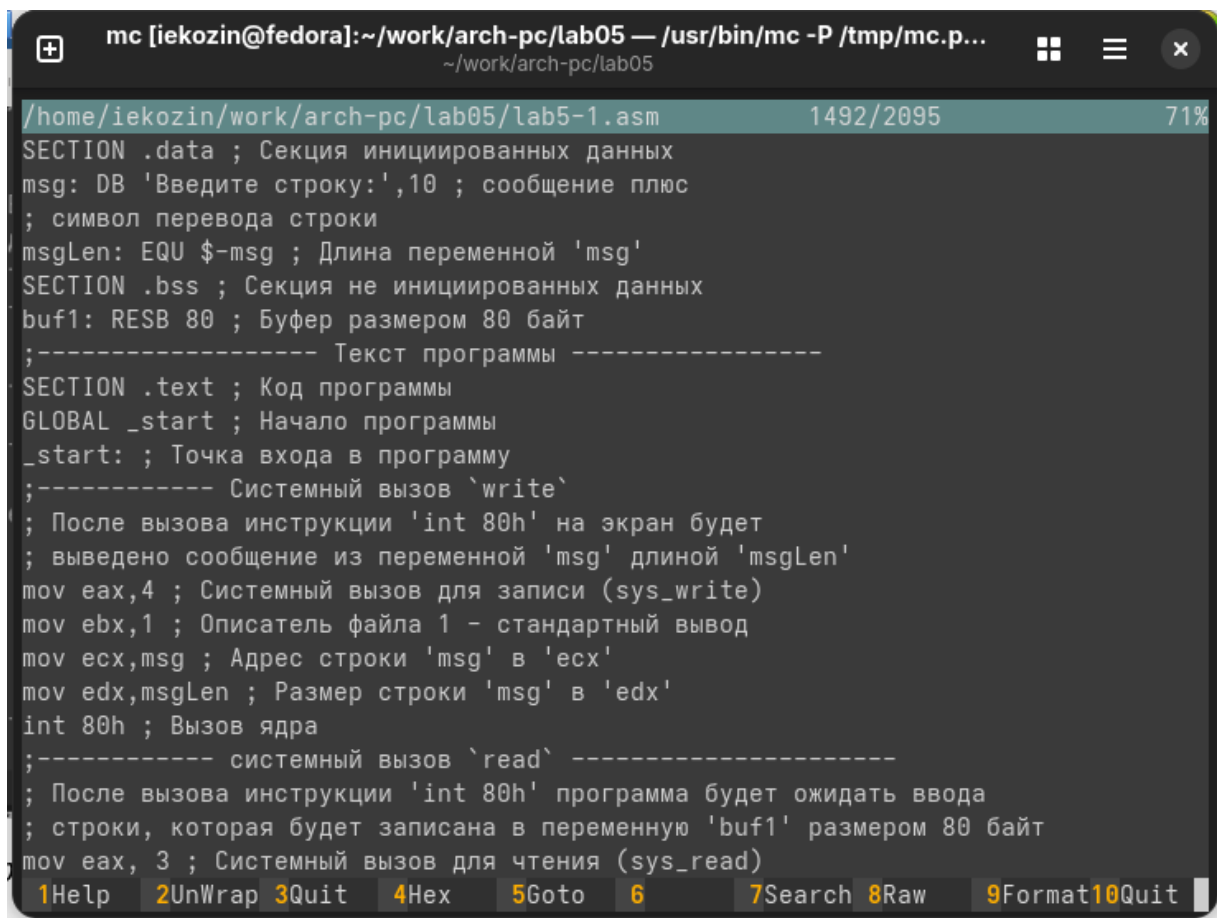


Рис. 6: Редактирование файла в Midnight Commander

Проверяю сохраненные изменения с помощью клавиши F3 (рис. 7).

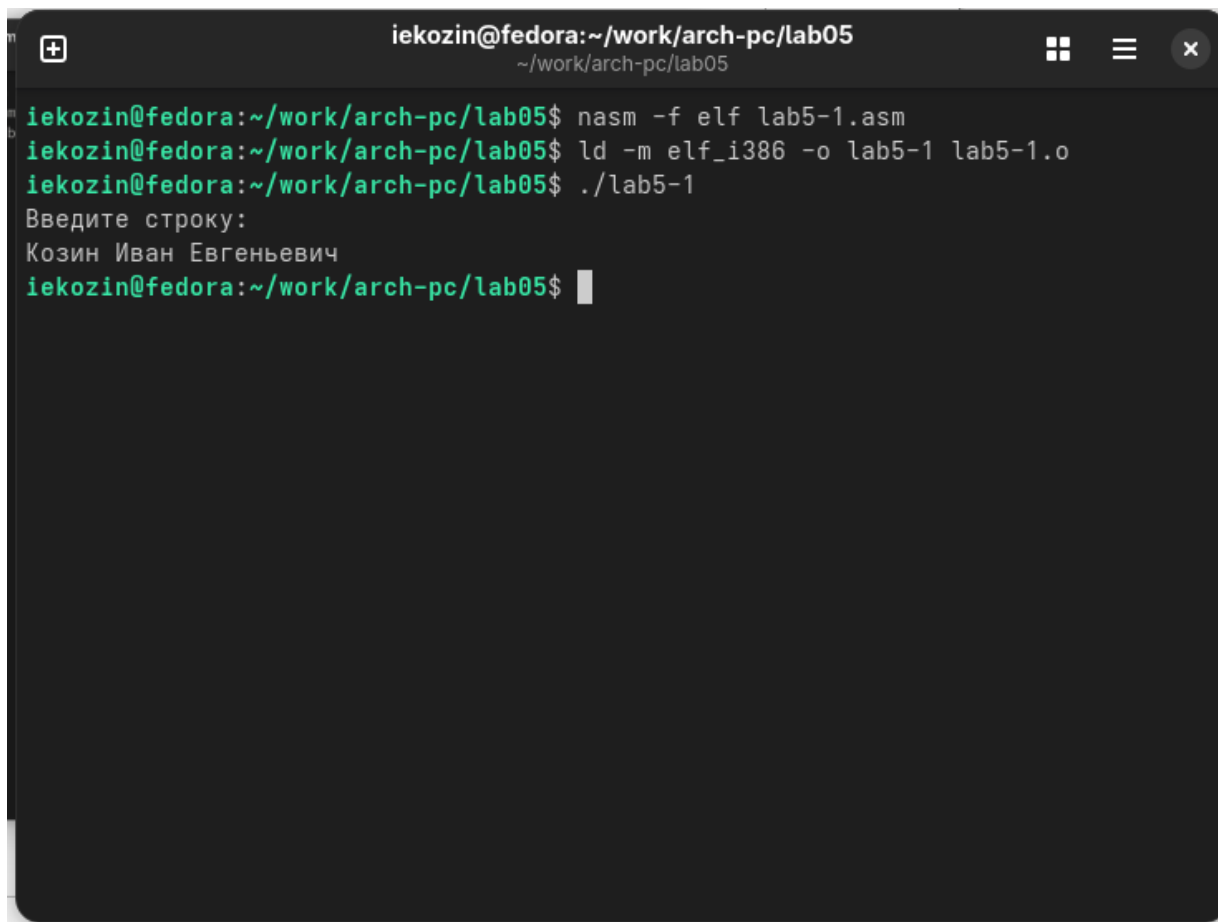


The screenshot shows the MC editor window with the title bar "mc [iekozin@fedora]:~/work/arch-pc/lab05 — /usr/bin/mc -P /tmp/mc.p...". The file being edited is "/home/iekozin/work/arch-pc/lab05/lab5-1.asm" with a progress indicator "1492/2095" and a zoom level of "71%". The code is written in assembly with Russian comments. It defines a data section for a message, a bss section for a buffer, and a text section for the program code. The code includes system calls for writing to the screen and reading from the user. At the bottom, there is a menu bar with options: 1Help, 2UnWrap, 3Quit, 4Hex, 5Goto, 6, 7Search, 8Raw, 9Format, 10Quit.

```
/home/iekozin/work/arch-pc/lab05/lab5-1.asm 1492/2095 71%
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
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 байт
mov eax, 3 ; Системный вызов для чтения (sys_read)
1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format10Quit
```

Рис. 7: Проверка сохранения сделанных изменений

Транслирую и компоную измененный файл, запускаю (рис. 8).

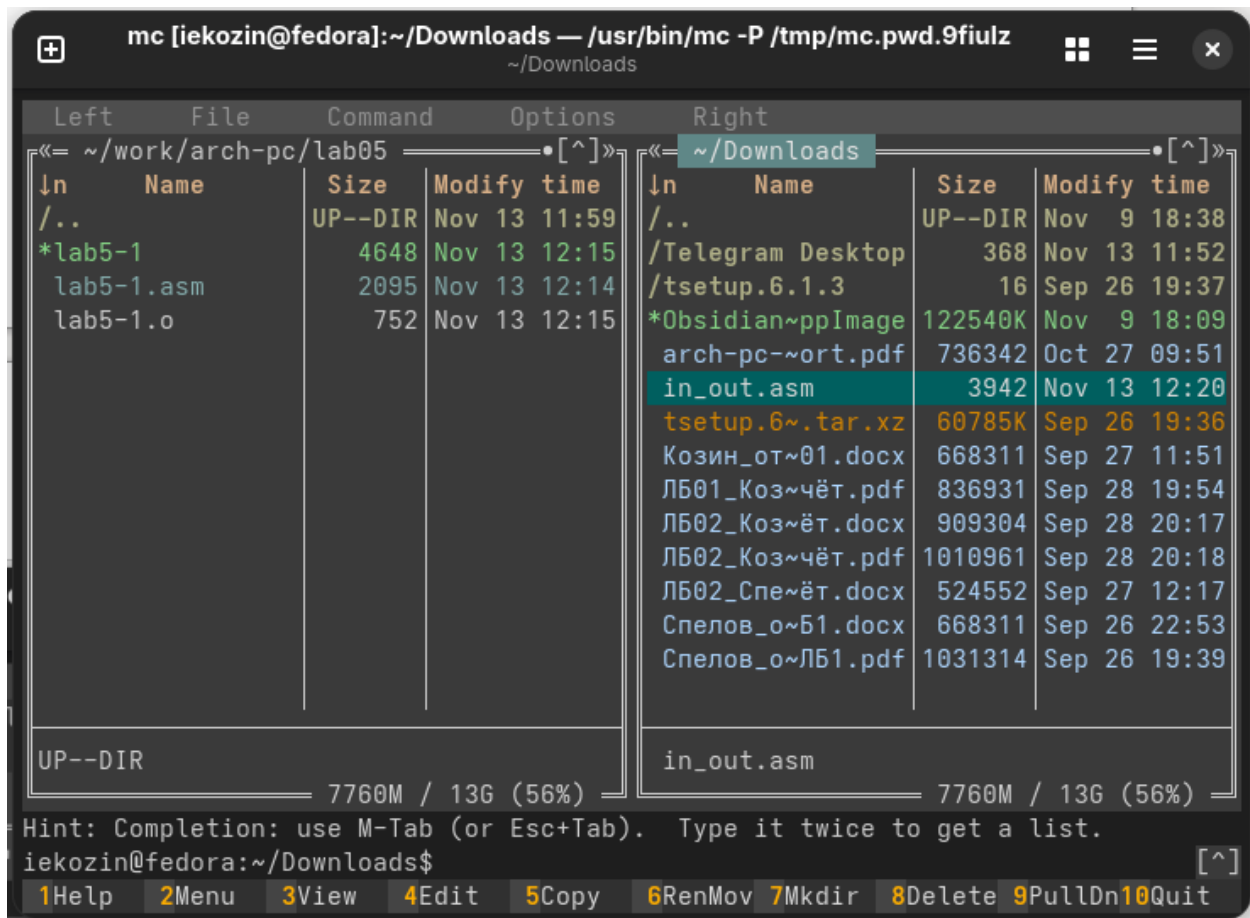
A terminal window with a dark background. The title bar shows the user 'iekozin@fedora' and the current directory '~/work/arch-pc/lab05'. The terminal contains the following text:

```
iekozin@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
iekozin@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
iekozin@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Козин Иван Евгеньевич
iekozin@fedora:~/work/arch-pc/lab05$
```

Рис. 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. 9).



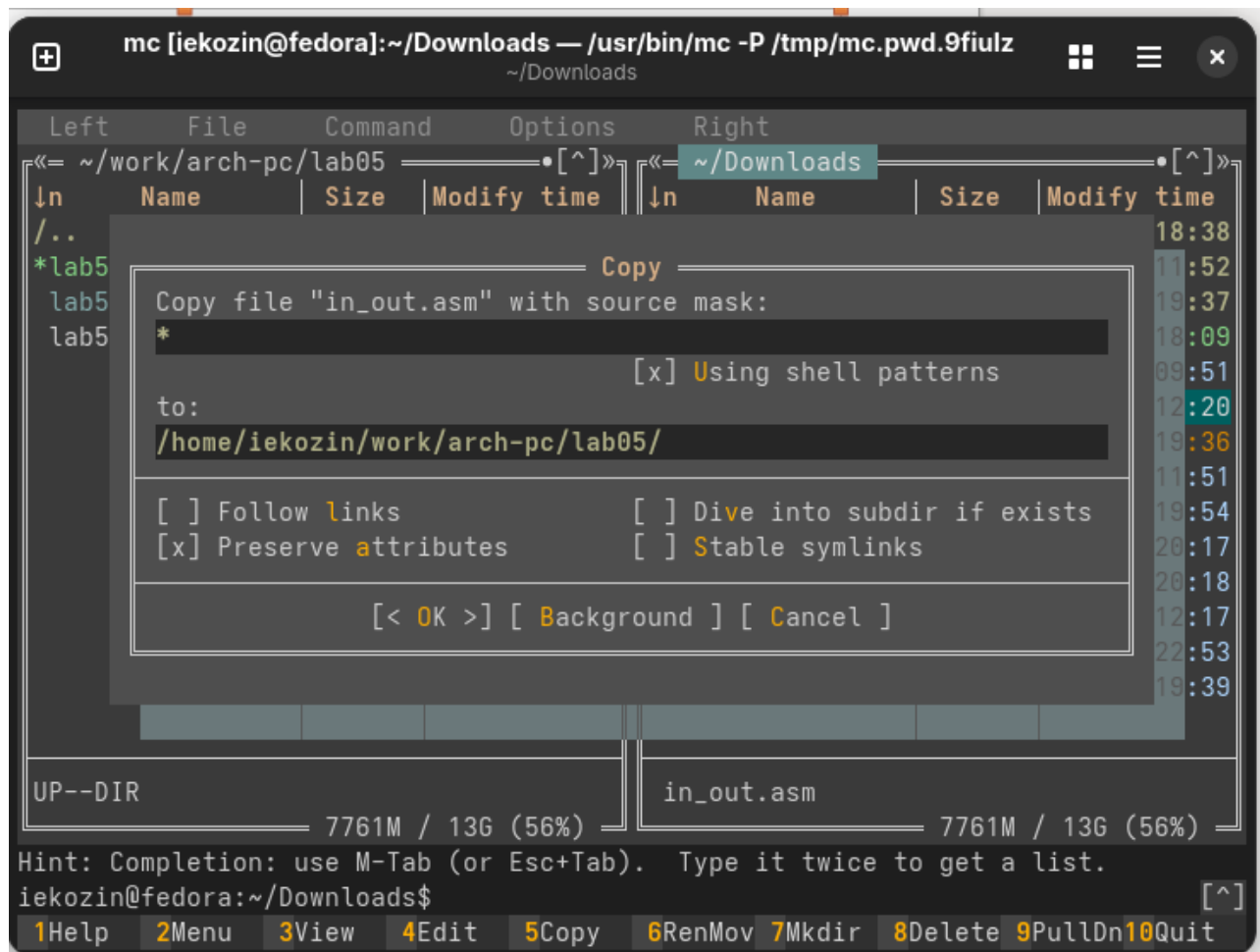


Рис. 9: Копирование файла в рабочий каталог

Создаю копию файла для последующей работы с ним (рис. 10).

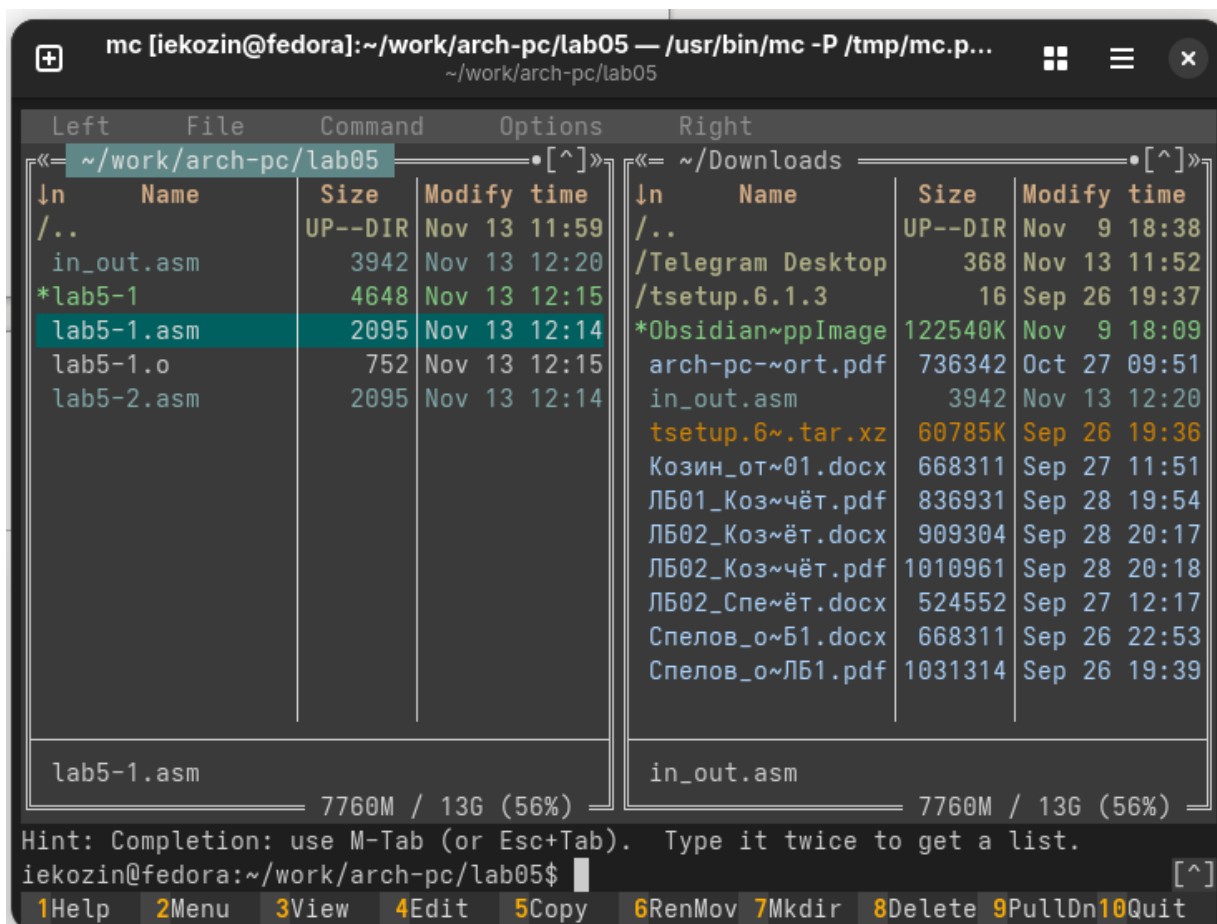
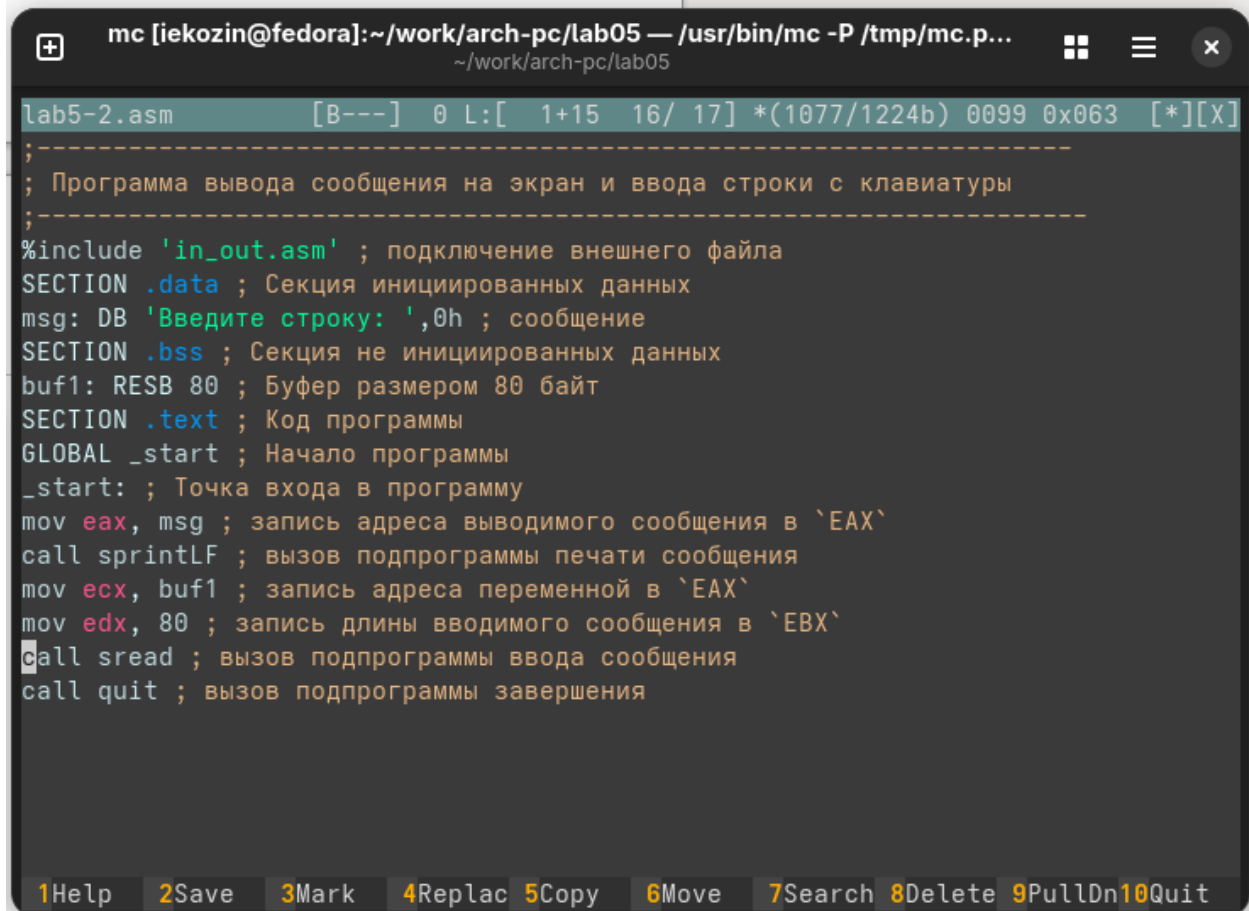


Рис. 10: Создание копии файла в Midnight Commander

В копии файла подключаю подпрограмм из подключенного файла (рис. 11).



```
mc [iekozin@fedora]:~/work/arch-pc/lab05 — /usr/bin/mc -P /tmp/mc.p...
~/work/arch-pc/lab05
lab5-2.asm [B---] 0 L:[ 1+15 16/ 17] *(1077/1224b) 0099 0x063 [*][X]
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не инициированных 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 sprintLF ; 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

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Рис. 11: Изменение программы

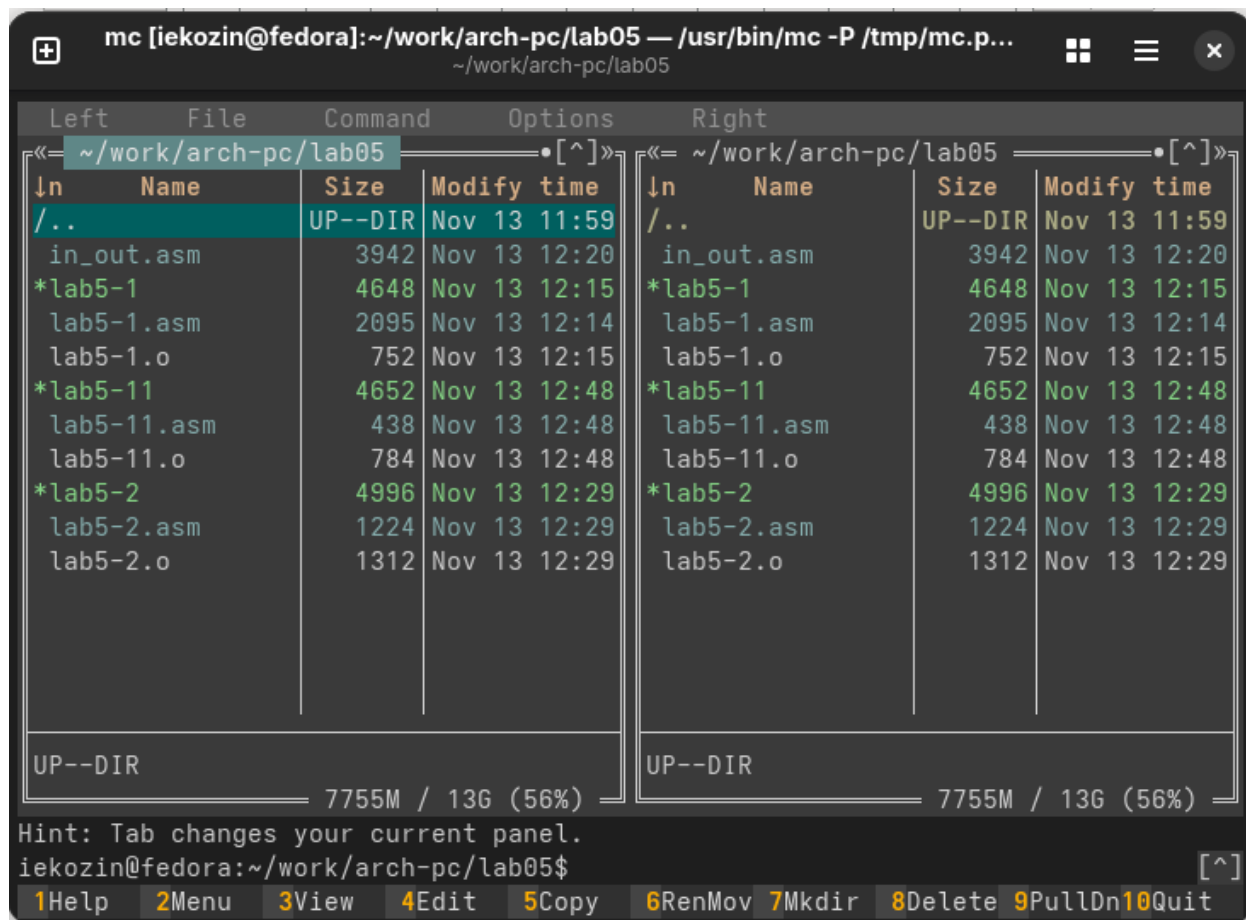
```
iekozin@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
iekozin@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
iekozin@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Козин Иван Евгеньевич
iekozin@fedora:~/work/arch-pc/lab05$
```

Транслирую, компоную и запускаю программу с подключенным файлом (рис. 12).

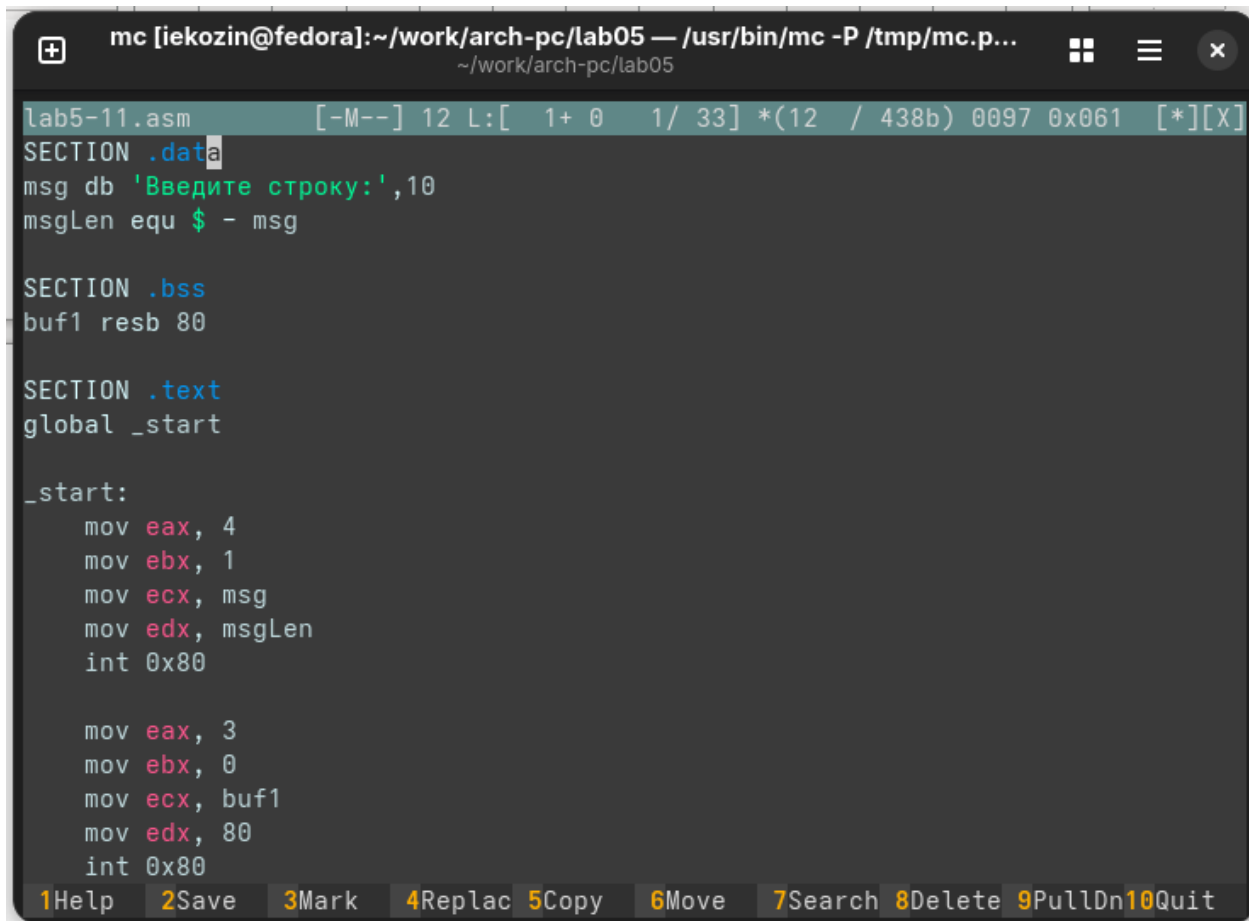
Рис. 12: Запуск измененной программы

4.4 Задание для самостоятельной работы

Создаю копию lab5-1.asm, редактирую так, чтобы в конце выводилась введенная мною строка с клавиатуры (рис. 14).



PS: я немного забыл сделать скриншот



```
lab5-11.asm [-M--] 12 L:[ 1+ 0 1/ 33] *(12 / 438b) 0097 0x061 [*][X]
SECTION .data
msg db 'Введите строку:',10
msgLen equ $ - msg

SECTION .bss
buf1 resb 80

SECTION .text
global _start

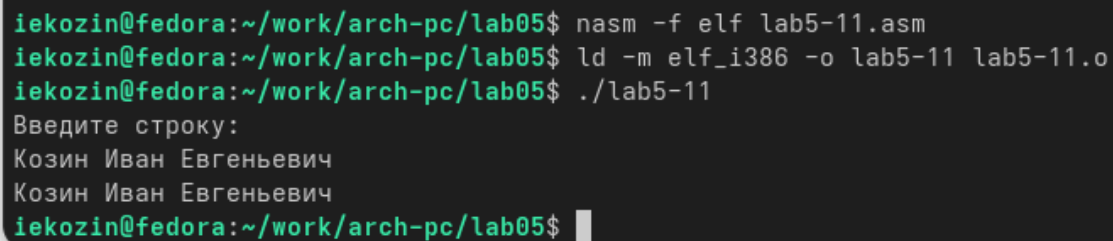
_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, msg
    mov edx, msgLen
    int 0x80

    mov eax, 3
    mov ebx, 0
    mov ecx, buf1
    mov edx, 80
    int 0x80

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn10Quit
```

Рис. 14: Редактирование копии

Транслирую, компоную и запускаю свою программу (рис. 15).



```
iekozin@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-11.asm
iekozin@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-11 lab5-11.o
iekozin@fedora:~/work/arch-pc/lab05$ ./lab5-11
Введите строку:
Козин Иван Евгеньевич
Козин Иван Евгеньевич
iekozin@fedora:~/work/arch-pc/lab05$
```

Рис. 15: Запуск своей программы

Код прикладываю

```
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 0x80
```

```
    mov eax, 3
```

```
    mov ebx, 0
```

```
    mov ecx, buf1
```

```
    mov edx, 80
```

```
    int 0x80
```

```
    mov esi, eax
```

```
    mov eax, 4
```

```
    mov ebx, 1
```

```
    mov ecx, buf1
```

```
    mov edx, esi
```

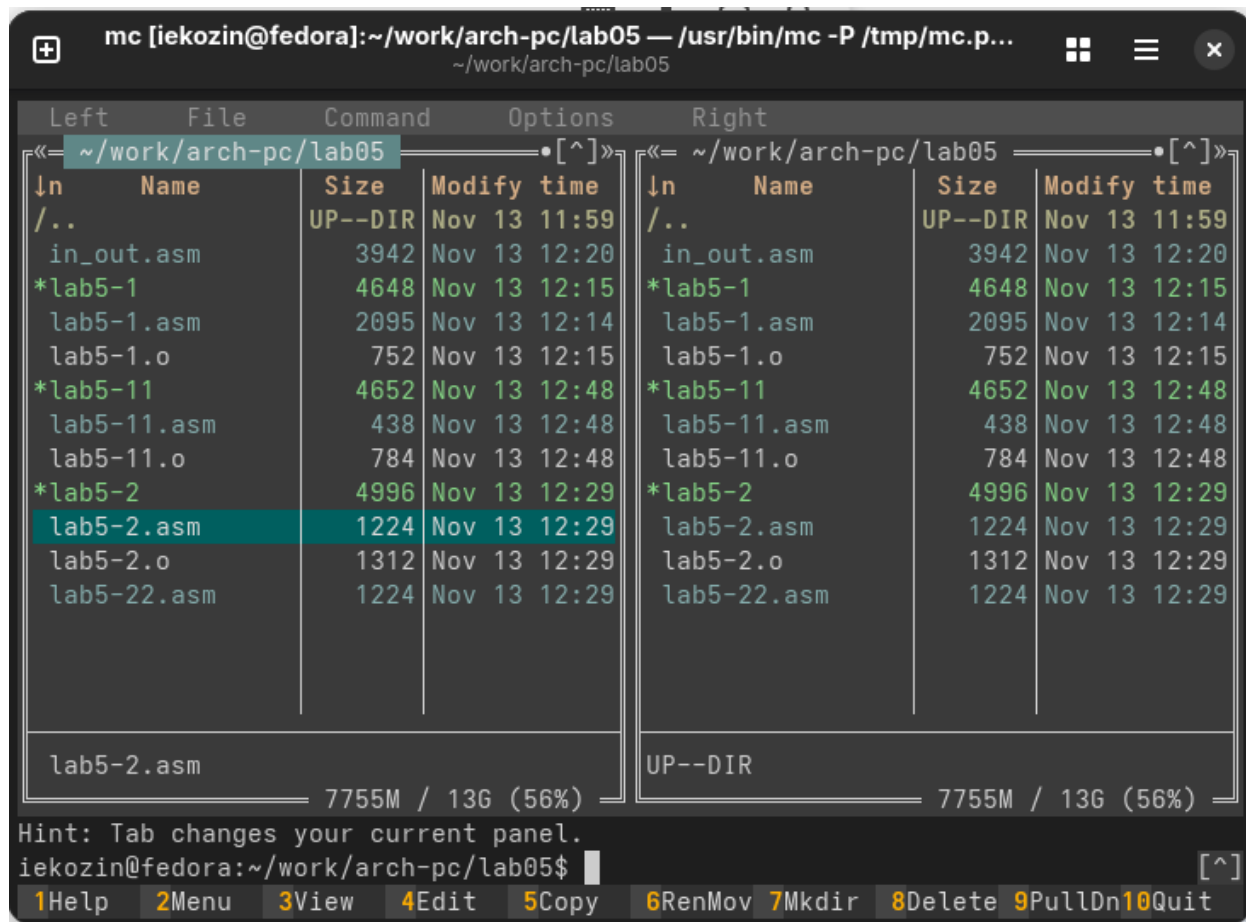
```
    int 0x80
```

```
    mov eax, 1
```

```
    xor ebx, ebx
```

```
int 0x80
```

Создаю копию lab5-2.asm, редактирую так, чтобы в конце выводилась введенная мною строка с клавиатуры (рис. 16).



```
mc [iekozin@fedora]:~/work/arch-pc/lab05 — /usr/bin/mc -P /tmp/mc.p...  
~ /work/arch-pc/lab05  
lab5-22.asm [-M--] 0 L: [ 1+ 1 2/ 28] *(22 / 333b) 0010 0x00A [*][X]  
%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  
  
1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Рис. 16: Редактирование копии

Транслирую, компоную и запускаю свою программу (рис. 17).

```
iekozin@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-22.asm  
iekozin@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-22 lab5-22.o  
iekozin@fedora:~/work/arch-pc/lab05$ ./lab5-22  
Введите строку: Козин Иван Евгеньевич  
Козин Иван Евгеньевич  
iekozin@fedora:~/work/arch-pc/lab05$
```

Рис. 17: Запуск своей программы

Код прикладываю:

```
%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
```

5 Выводы

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

Список литературы

1. Пример выполнения лабораторной работы
2. Курс на ТУИС
3. Лабораторная работа №5
4. Программирование на языке ассемблера NASM Столяров А. В.