

# **Отчёт по лабораторной работе №5**

**Архитектура компьютеров и операционные системы.**

Брыляков Никита Евгеньевич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
4.1	Начало работы . . . . .	7
4.2	Подключение внешнего файла in_out.asm . . . . .	11
4.3	Выполнение заданий для самостоятельной работы . . . . .	14
<b>5</b>	<b>Вывод</b>	<b>17</b>
<b>6</b>	<b>Список литературы</b>	<b>18</b>

## Список иллюстраций

4.1	Открытие mc . . . . .	7
4.2	Переход в каталог . . . . .	8
4.3	Переход в созданный каталог . . . . .	8
4.4	Создание файла . . . . .	9
4.5	Открытие файла . . . . .	9
4.6	Ввод текста программы . . . . .	10
4.7	Просмотр файла . . . . .	10
4.8	Компиляция и исполнение файла . . . . .	11
4.9	Скачивание файла . . . . .	11
4.10	Копирование файла . . . . .	12
4.11	Создание копии файла . . . . .	12
4.12	Изменение программы . . . . .	13
4.13	Создание и проверка . . . . .	13
4.14	Изменение подпрограммы . . . . .	13
4.15	Создание и проверка . . . . .	14
4.16	Копирование файла . . . . .	14
4.17	Редактирование . . . . .	15
4.18	Проверка . . . . .	15
4.19	Копирование файла . . . . .	16
4.20	Редактирование . . . . .	16
4.21	Проверка . . . . .	16

# 1 Цель работы

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

## 2 Задание

1. Начало работы
2. Подключение внешнего файла in\_out.asm
3. 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. Инструкция языка ассемблера `int $n` предназначена для вызова прерывания с указанным номером. В общем виде она записывается в виде `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 Начало работы

Открываю Midnight Commander (рис. [4.1]).

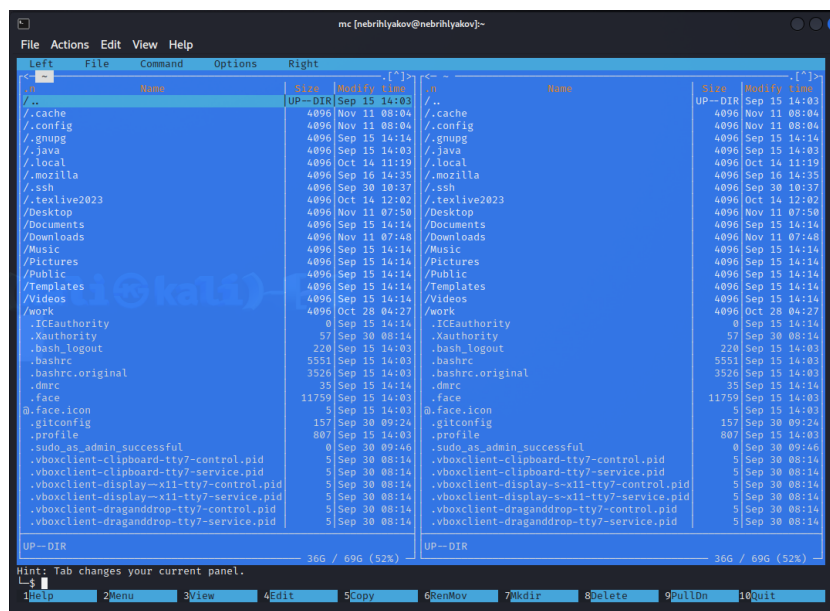


Рис. 4.1: Открытие mc

Перехожу в каталог ~/work/arch-рс созданный при выполнении лабораторной работы №4 (рис. [4.2]).

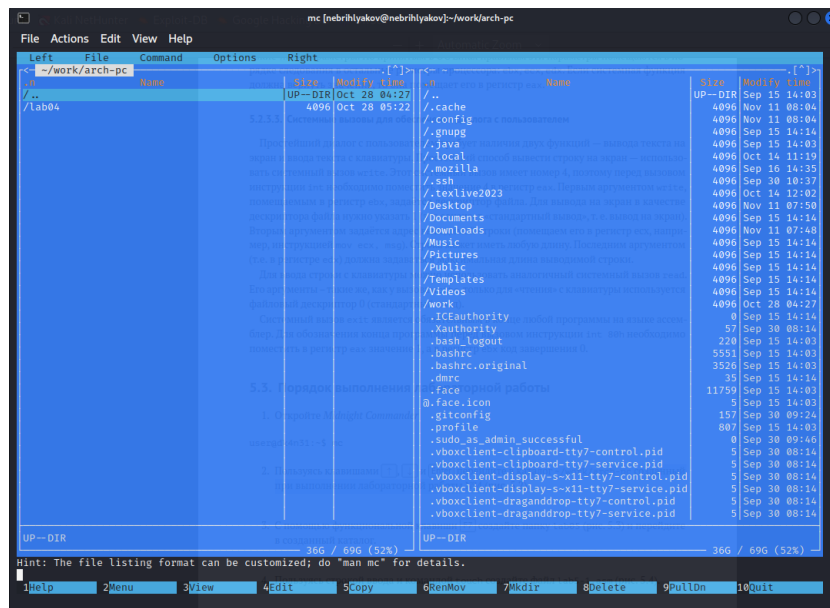


Рис. 4.2: Переход в каталог

Создаю папку lab05 и перехожу в созданный каталог. (рис. [4.3]).

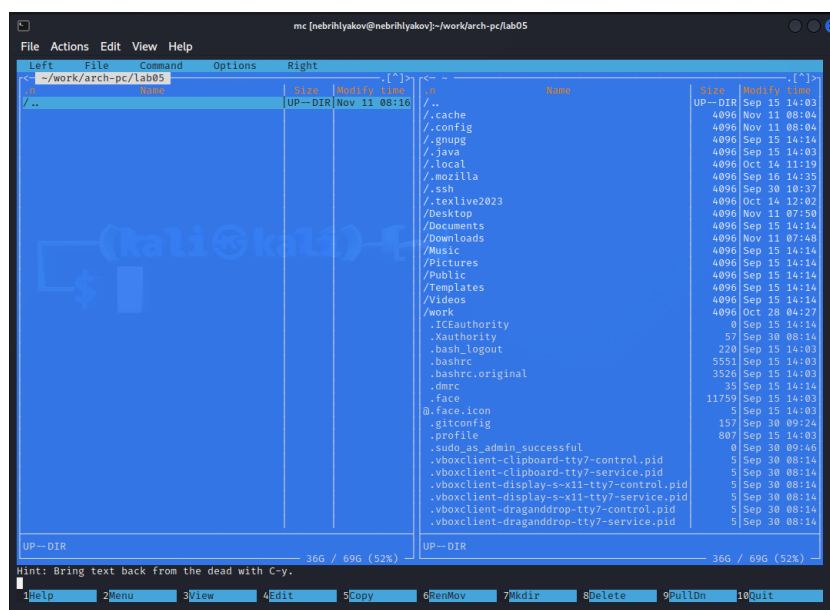


Рис. 4.3: Переход в созданный каталог

Создаю файл lab5-1.asm(рис. [4.4]).



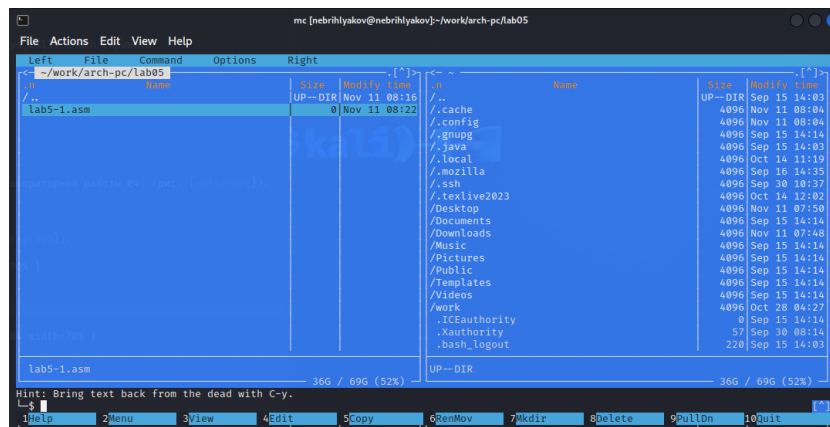


Рис. 4.4: Создание файла

Открываю файл lab5-1.asm для редактирования во встроенном редакторе (рис. [4.5]).

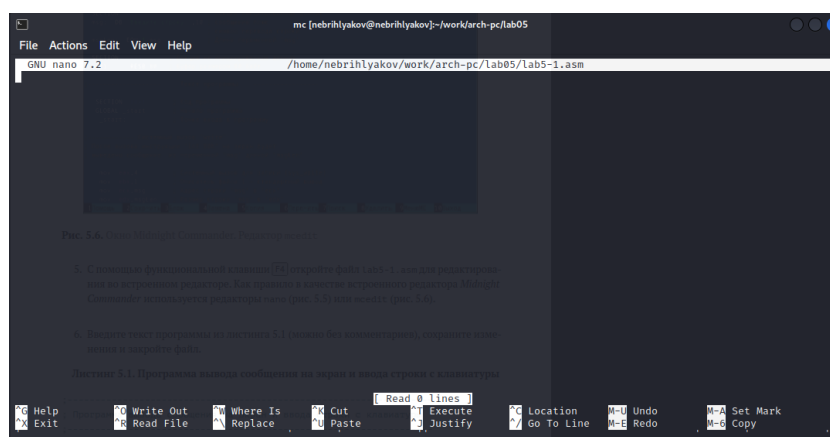


Рис. 4.5: Открытие файла

Ввожу текст программы и сохраняю (рис. [4.6]).



Рис. 4.6: Ввод текста программы

Открываю файл lab5-1.asm для просмотра и убеждаюсь в наличии текста.(рис. [4.7]).

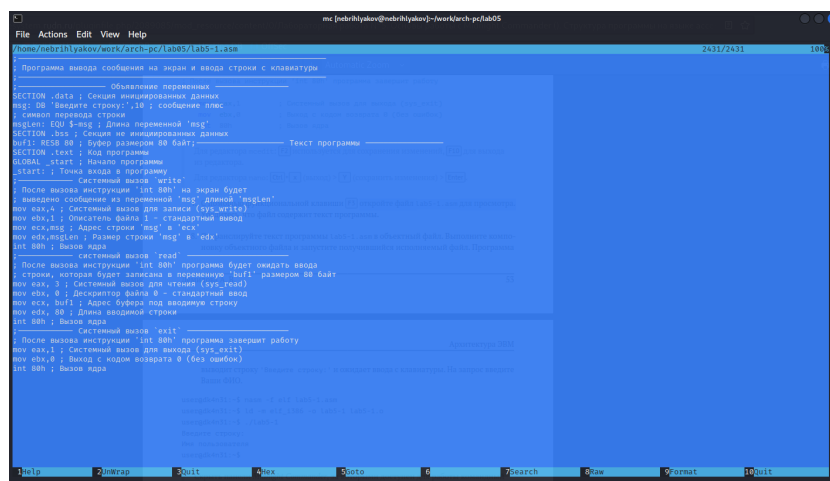


Рис. 4.7: Просмотр файла

Оттранслирую текст программы lab5-1.asm в объектный файл. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры. На запрос ввожу своё ФИО (рис. [4.8]).

```
└─$ nasm -f elf lab5-1.asm

ld -m elf_i386 -o lab5-1 lab5-1.o
./lab5-1
Введите строку:
Брыляков Никита Евгеньевич
```

Рис. 4.8: Компиляция и исполнение файла

## 4.2 Подключение внешнего файла in\_out.asm

Скачиваю файл in\_out.asm со страницы курса в ТУИС (рис. [4.9]).

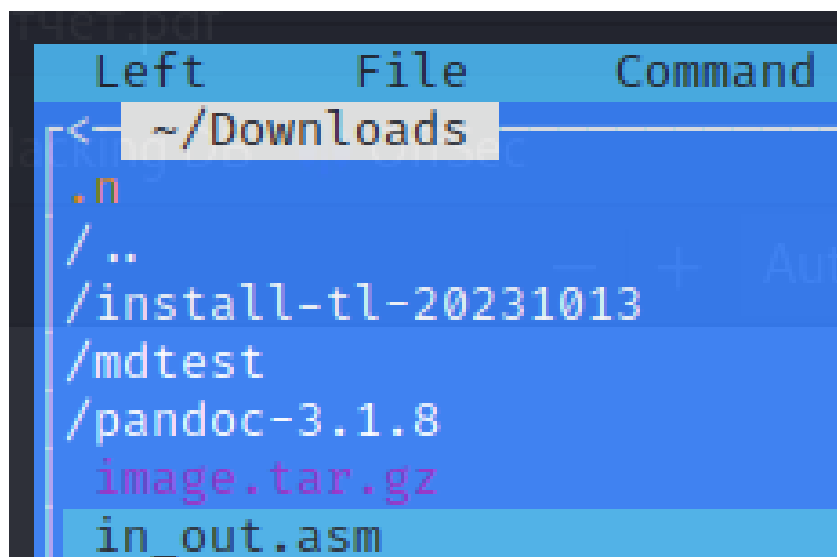


Рис. 4.9: Скачивание файла

Копирую файл in\_out.asm из каталога Downloads в каталог lab05 (рис. [4.10]).

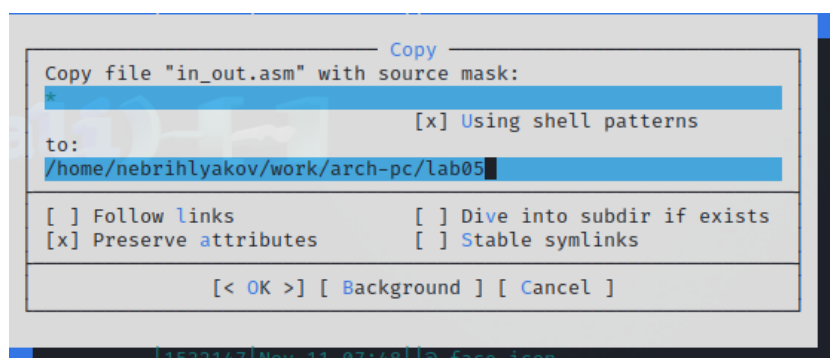


Рис. 4.10: Копирование файла

Создаю копию файла lab5-1.asm с именем lab5-2.asm (рис. [4.11]).

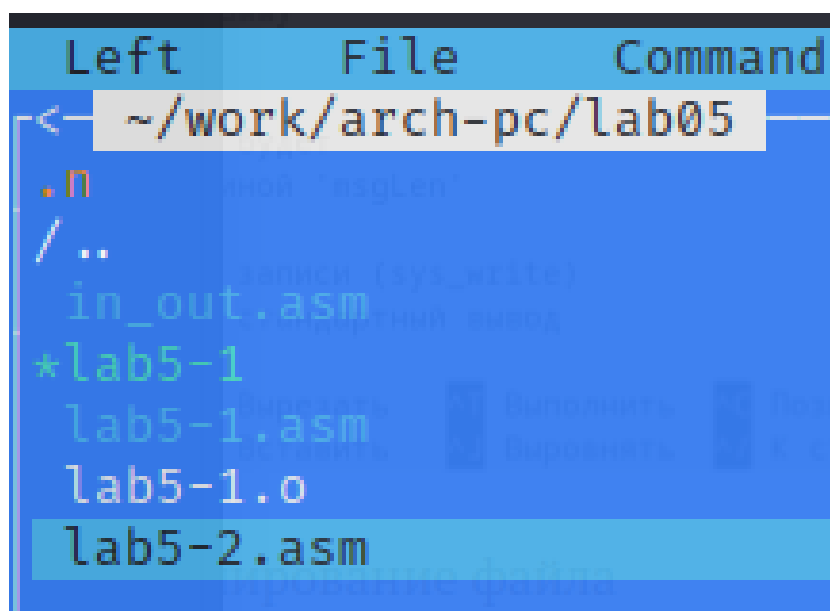


Рис. 4.11: Создание копии файла

Изменяю содержимое файла lab-2.asm (рис. [4.12]).

```

GNU nano 7.2 /home/nebrihlyakov/work/arch-pc/lab05/lab5-2.asm
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;
#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 'ECX'
mov edx, 80 ; запись длины вводимого сообщения в 'EDX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

```

Рис. 4.12: Изменение программы

Создаю исполняемый файл и проверяю его работу. (рис. [4.13]).

```

nasm -f elf lab5-2.asm

└─$ ld -m elf_i386 -o lab5-2 lab5-2.o

└─$ ./lab5-2
Введите строку:
Брыляков Никита Евгеньевич

```

Рис. 4.13: Создание и проверка

Открываю файл lab5-2.asm. Изменяю в нем подпрограмму sprintf на sprint. (рис. [4.14]).

```

GNU nano 7.2 /home/nebrihlyakov/work/arch-pc/lab05/lab5-2.asm
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;
#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 sprint ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'ECX'
mov edx, 80 ; запись длины вводимого сообщения в 'EDX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

```

Рис. 4.14: Изменение подпрограммы

Создаю исполняемый файл и проверяю его работу. (рис. [4.15]).

```
└─$ nasm -f elf lab5-2.asm

└─$ ld -m elf_i386 -o lab5-2 lab5-2.o

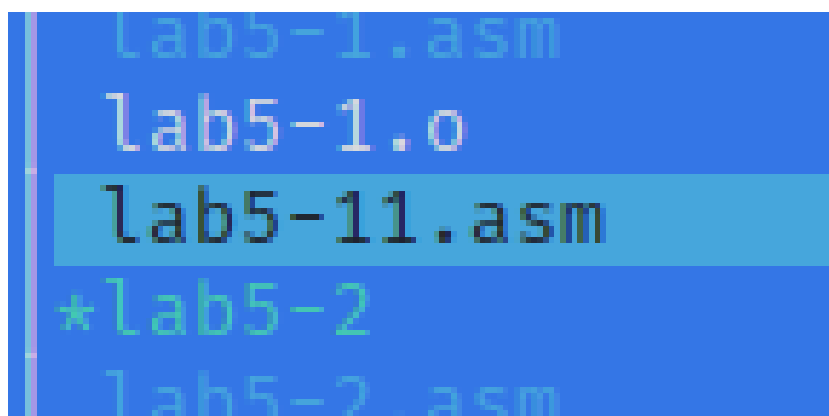
└─$ ./lab5-2
Введите строку: Брыляков Никита Евгеньевич
```

Рис. 4.15: Создание и проверка

Разница в том, что в случае с подпрограммой `sprintLF` исполняемый файл запрашивает ввод с новой строки, а с подпрограммой `sprint` запрашивает ввод без переноса на новую строку

### 4.3 Выполнение заданий для самостоятельной работы

Создаю копию файла `lab5-1.asm`, называю её `lab5-11.asm`. (рис. [4.16]).



```
lab5-1.asm
lab5-1.o
lab5-11.asm
*lab5-2
lab5-2.asm
```

Рис. 4.16: Копирование файла

Открываю программу и редактирую по заданию. (рис. [4.17]).

```

GNU nano 7.2
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10
msgLen: EQU $-msg ; Длина переменной 'msg'
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,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
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
mov edx,80 ; Длина вводимой строки
int 80h ; Вызов ядра
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,buf1 ; Адрес строки buf1 в ecx
mov edx,buf1 ; Размер строки buf1
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

```

Рис. 4.17: Редактирование

Получаю исполняемый файл и проверяю его работу. (рис. [4.18]).

```

└─$ nasm -f elf lab5-11.asm
└─$ ld -m elf_i386 -o lab5-11 lab5-11.o
./lab5-11
Введите строку:
Брыляков Никита Евгеньевич
Брыляков Никита Евгеньевич

```

Рис. 4.18: Проверка

Создаю копию файла lab5-2.asm, называю её lab5-22.asm. (рис. [4.19]).

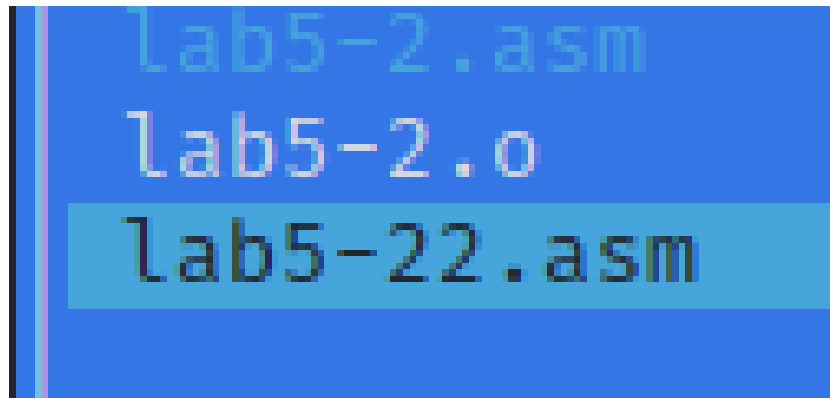


Рис. 4.19: Копирование файла

Открываю программу и редактирую по заданию. (рис. [??]).

```
GNU nano 7.2 /home/nebrihlyakov/work/arch-pc/lab05/lab5-22.asm
#include "in_out.asm"
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 sprint ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в "ECX"
mov edx, 80 ; запись длины вводимого сообщения в "EDX"
call sread ; вызов подпрограммы ввода сообщения
mov eax, 4 ; Системный вызов для записи (sys_write)
mov ebx, 1 ; Описатель файла '1' - стандартный вывод
mov ecx, buf1 ; Адрес строки buf1 в ecx
int 80h ; Вызов ядра
call quit ; вызов подпрограммы завершения
```

Рис. 4.20: Редактирование

Получаю исполняемый файл и проверяю его работу. (рис. [4.21]).

```
nasm -f elf lab5-22.asm

└─$ ld -m elf_i386 -o lab5-22 lab5-22.o

└─$ ./lab5-22
Введите строку: Брыляков Никита Евгеньевич
Брыляков Никита Евгеньевич
```

Рис. 4.21: Проверка



## 5 Вывод

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

## 6 Список литературы

- [illegible]