

# Отчет по лабораторной работе №6

Смирнов Дмитрий Романович, НММбд-03-22

## Содержание

1	Цель работы .....	1
2	Выполнение лабораторной работы .....	1
3	Вопросы для самопроверки.....	9
4	Выводы.....	10

## 1 Цель работы

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

## 2 Выполнение лабораторной работы

Открою Midnight Commander с помощью команды mc, перейду в каталог /work/study/2022-2023/Архитектура компьютера/study\_2022-2023\_arh-rc/labs/lab06 и создам папку lab06, а в ней файл lab6.asm

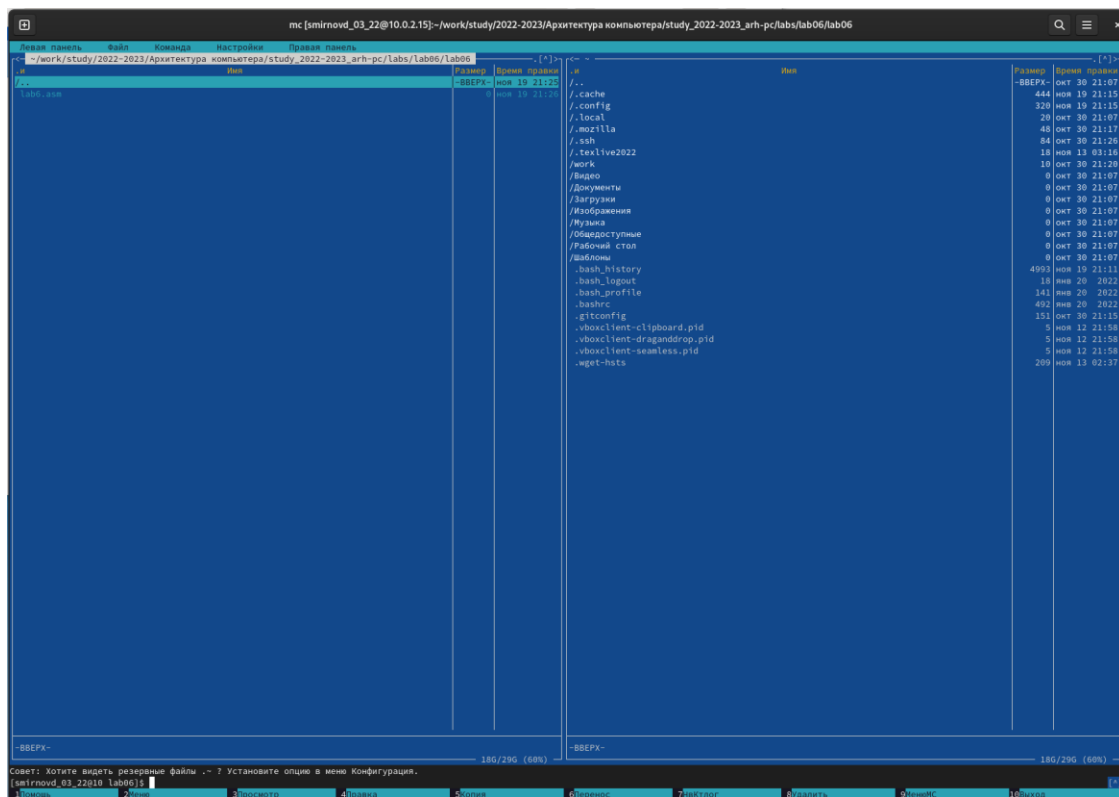


Рис. 1: Рис1

С помощью функциональной клавиши F4 открою файл lab6.asm для редактирования во встроенном редакторе.

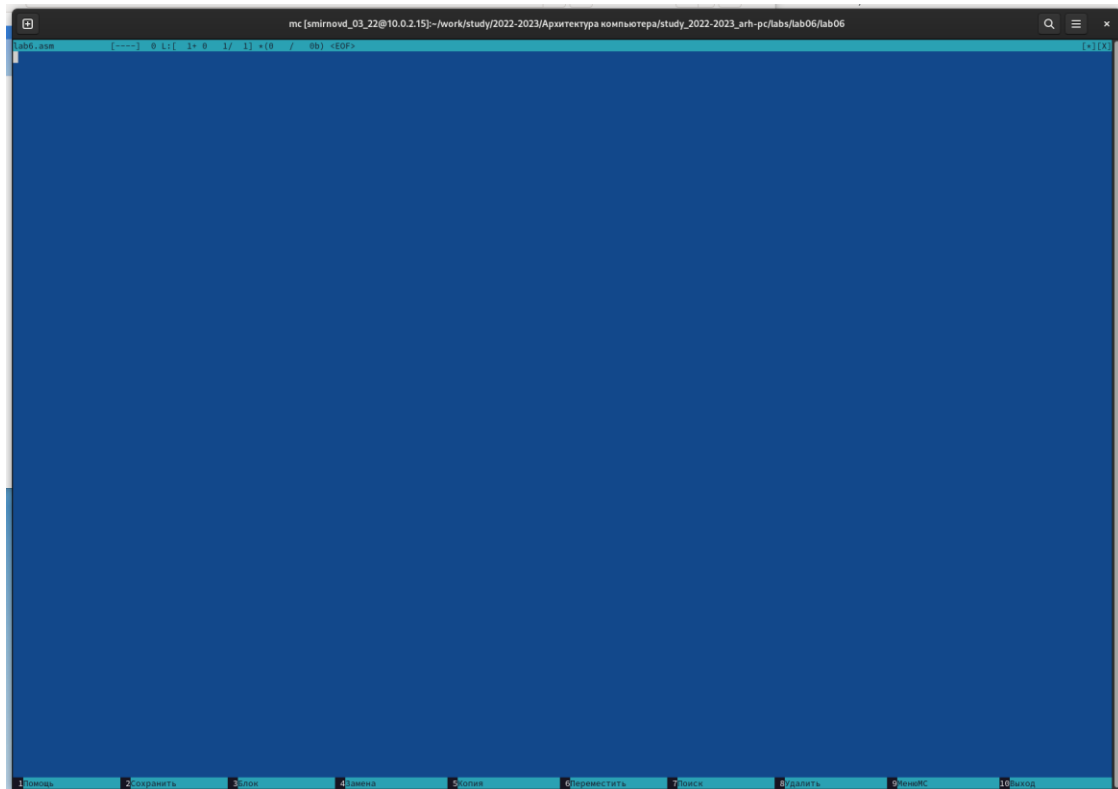
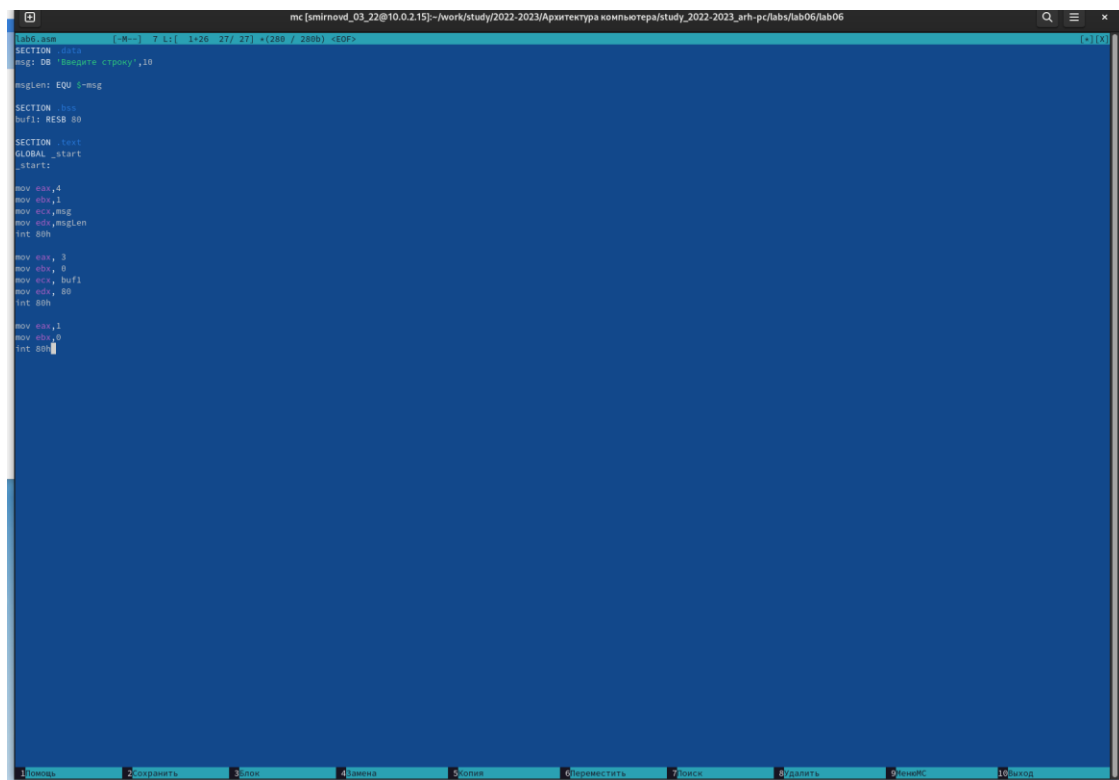


Рис. 2: Рис2

Введу текст программы из листинга 6.1, сохраню изменения и закрою файл.



```
lab6.asm [-M-] 7 1: 1+26 27/ 27 4(288 / 2888) <ESP>
SECTION .data
msg: DB "Здравствуйте",10
msgLen: EQU $-msg

SECTION .text
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

mov ebx,4
mov ebx,1
mov ecx,msg
mov edx,msgLen
int 80h

mov ebx,3
mov ebx,0
mov ecx,buf1
mov edx,80
int 80h

mov ebx,1
mov ebx,0
int 80h
```

Рис. 3: Рис3

С помощью функциональной клавиши F3 открою файл lab6.asm для просмотра

```
msc [smirnovd_03_22@10.0.2.15]:~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/labs/lab06/lab06
asm lab6.asm
nasm -f elf lab6.asm
ld -o lab6 lab6.o
./lab6
Введите строку:
```

```
SECTION .data
msg: DB "Введите строку",10
msglen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

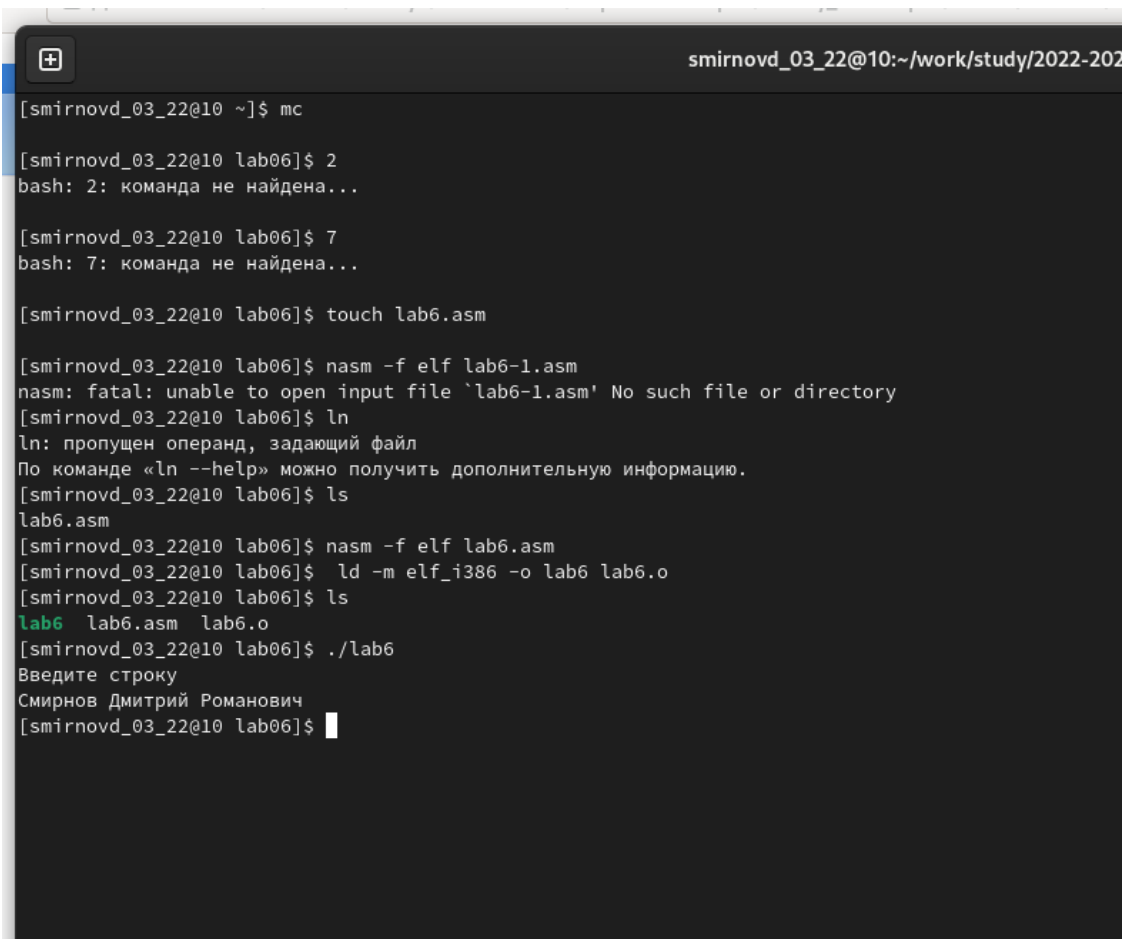
mov eax,4
mov ebx,1
mov ecx,msg
mov edx,msglen
int 80h

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

mov eax,1
mov ebx,0
int 80h
```

Рис. 4: Рис4

Оттранслирую текст программы lab6.asm в объектный файл. Выполню компоновку объектного файла и запущу получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры. На запрос введу свои ФИО.



```
smirnovd_03_22@10:~/work/study/2022-2023
[smirnovd_03_22@10 ~]$ mc
[smirnovd_03_22@10 lab06]$ 2
bash: 2: команда не найдена...
[smirnovd_03_22@10 lab06]$ 7
bash: 7: команда не найдена...
[smirnovd_03_22@10 lab06]$ touch lab6.asm
[smirnovd_03_22@10 lab06]$ nasm -f elf lab6-1.asm
nasm: fatal: unable to open input file `lab6-1.asm' No such file or directory
[smirnovd_03_22@10 lab06]$ ln
ln: пропущен операнд, задающий файл
По команде «ln --help» можно получить дополнительную информацию.
[smirnovd_03_22@10 lab06]$ ls
lab6.asm
[smirnovd_03_22@10 lab06]$ nasm -f elf lab6.asm
[smirnovd_03_22@10 lab06]$ ld -m elf_i386 -o lab6 lab6.o
[smirnovd_03_22@10 lab06]$ ls
lab6 lab6.asm lab6.o
[smirnovd_03_22@10 lab06]$ ./lab6
Введите строку
Смирнов Дмитрий Романович
[smirnovd_03_22@10 lab06]$
```

Рис. 5: Рис5

Скачаю файл in\_out.asm со страницы курса в ТУИС. И скопирую в папку /work/study/2022-2023/Архитектура компьютера/study\_2022-2023\_arch-pc/labs/lab06/lab06

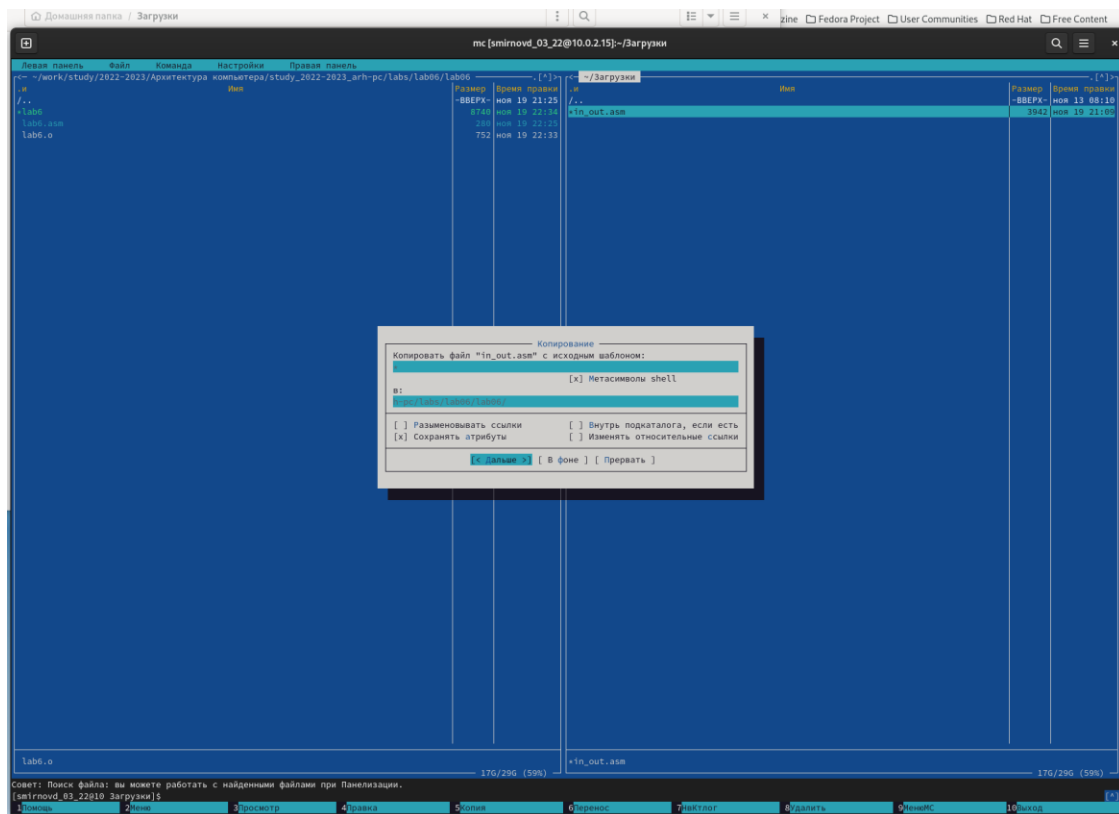


Рис. 6: Рис6

С помощью функциональной клавиши F6 создам копию файла lab6.asm с именем lab6-2.asm. Выделите файл lab6.asm, нажмите клавишу F6 , введите имя файла lab6-2.asm и нажму клавишу Enter

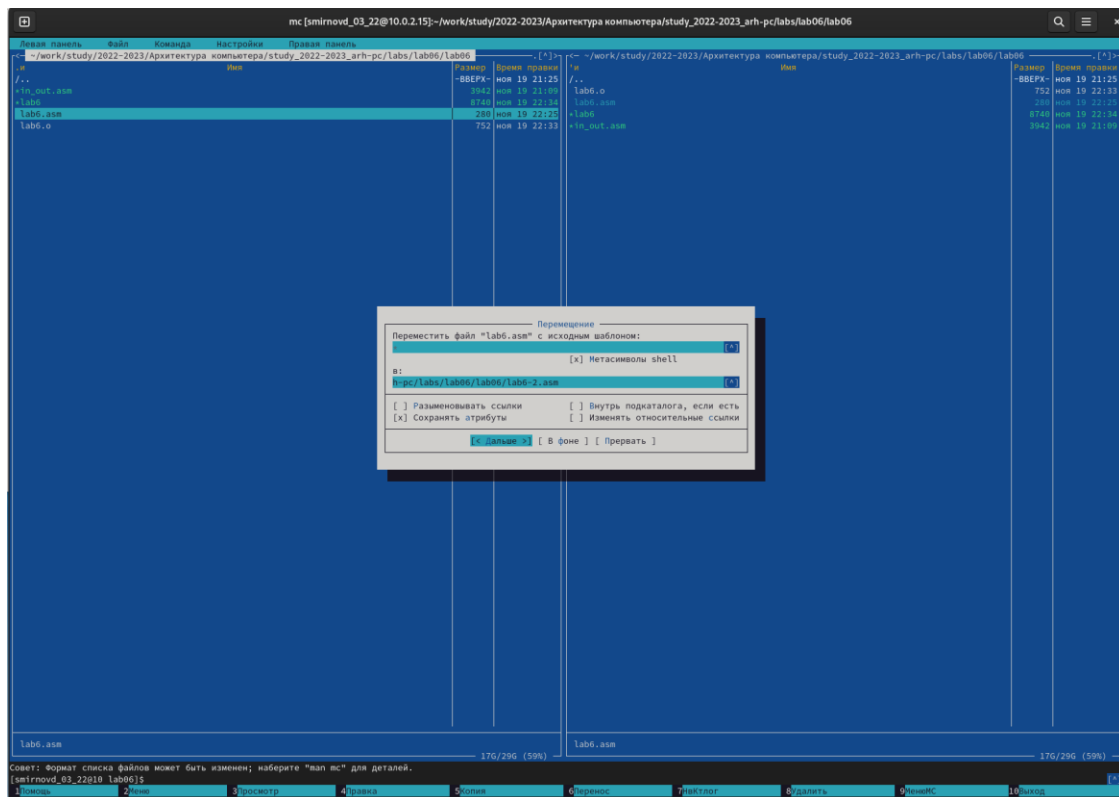


Рис. 7: Рис7

Исправлю текст программы в файле lab6-2.asm с использованием подпрограмм из внешнего файла in\_out.asm и запущу программу

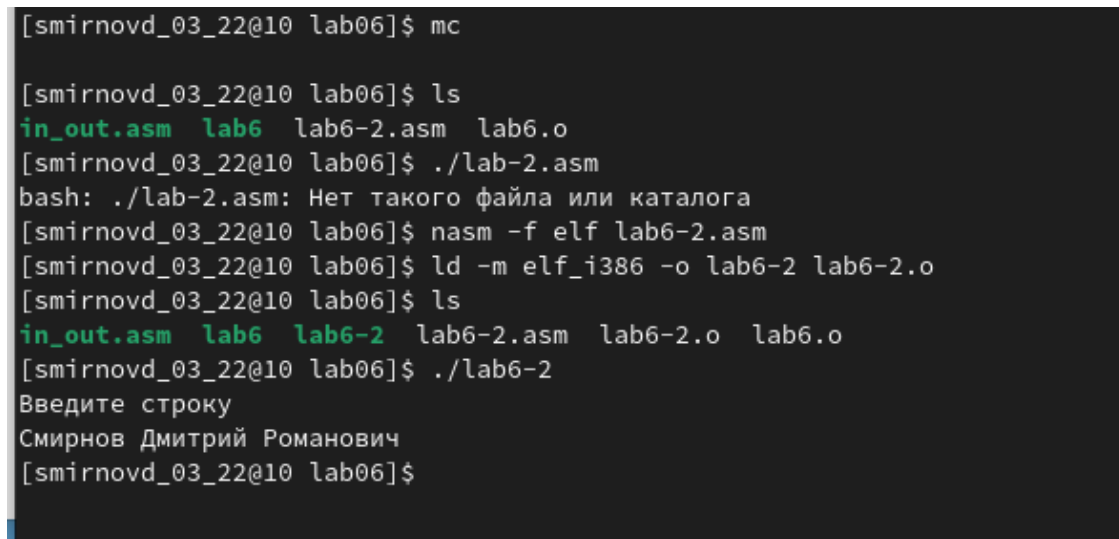


Рис. 8: Рис8



```
smirnovd_03_22@10:~/work/study/2022-2023/Архитектура компьютера/study_2
GNU nano 6.0 /home/smirnovd_03_22/work/study/2022-2023/Архитектура компьютера/study_2022-
#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 sprintLF

    mov ecx, buf1
    mov edx, 80
    call sread

    call quit
```

Рис. 9: Рис9

В файле lab6-2.asm заменю подпрограмму sprintLF на sprint. Создам исполняемый файл и проверю его работу

```
[smirnovd_03_22@10 lab06]$ ^C
[smirnovd_03_22@10 lab06]$ nasm -f elf lab6-2.asm
[smirnovd_03_22@10 lab06]$ ld -m elf_i386 -o lab6-2 lab6-21.o
ld: невозможно найти lab6-21.o: Нет такого файла или каталога
[smirnovd_03_22@10 lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[smirnovd_03_22@10 lab06]$ ls
in_out.asm lab6 lab6-2 lab6-2.asm lab6-2.o lab6.o
[smirnovd_03_22@10 lab06]$ ./lab6-2
Введите строку Смирнов Дмитрий Романович
[smirnovd_03_22@10 lab06]$
```

Рис. 10: Рис10

Разница в работе программ видна, вторая программа не переводит пользователя на новую строку.

### 3 Вопросы для самопроверки

1. Каково назначение mc? Основное назначение данной программы - упростить и сделать более наглядной работу с файлами в системе Linux.

2. Какие операции с файлами можно выполнить как с помощью команд `bash`, так и с помощью меню (комбинаций клавиш) `mc`? Приведите несколько примеров.
3. Какова структура программы на языке ассемблера `NASM`? 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`).
4. Для описания каких данных используются секции `bss` и `data` в языке ассемблера `NASM`? Эти секции используются для определения кода запуска
5. Для чего используются компоненты `db`, `dw`, `dd`, `dq` и `dt` языка ассемблера `NASM`? `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 (четверное слово); • `DT` (`define ten bytes`) — определяет переменную размером в 10 байт.
6. Какое произойдёт действие при выполнении инструкции `mov eax, esi`?
7. Для чего используется инструкция `int 80h`? Системный вызов `exit` является обязательным в конце любой программы на языке ассемблер. Для обозначения конца программы перед вызовом инструкции `int 80h` необходимо поместить в регистр `eax` значение 1, а в регистр `ebx` код завершения 0.

## 4 Выводы

Я Приобрел практические навыки работы в `Midnight Commander` и освоил инструкций языка ассемблера `mov` и `int`.