

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Стещенко Артём Срегеевич

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).

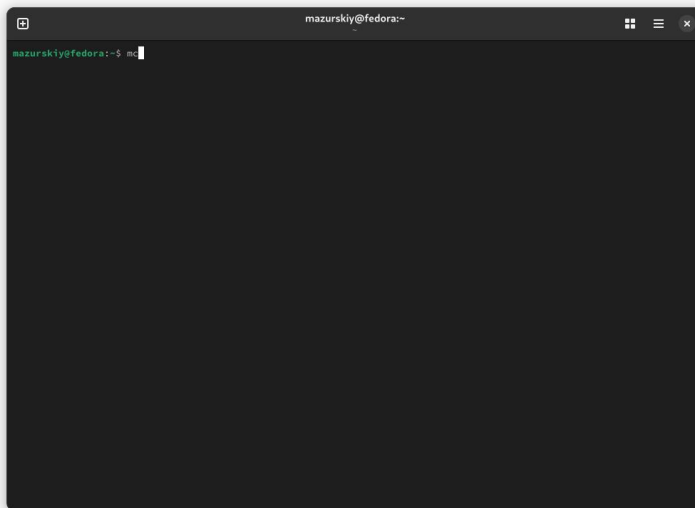


Рис. 1: Открытие Midnight Commander

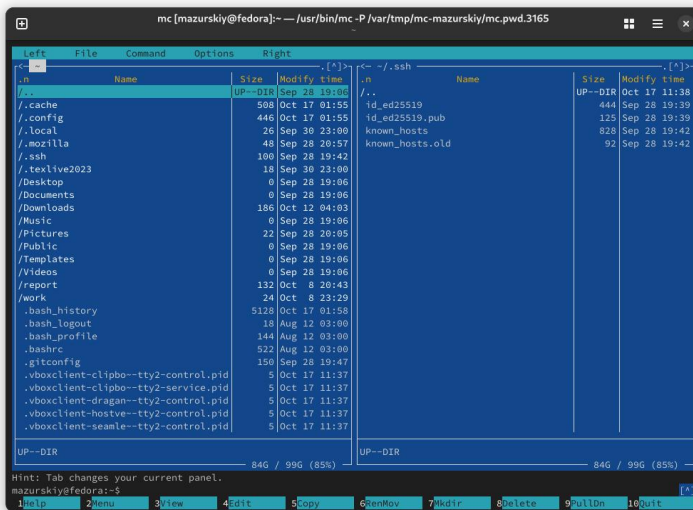


Рис. 2: Интерфейс Midnight Commander

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

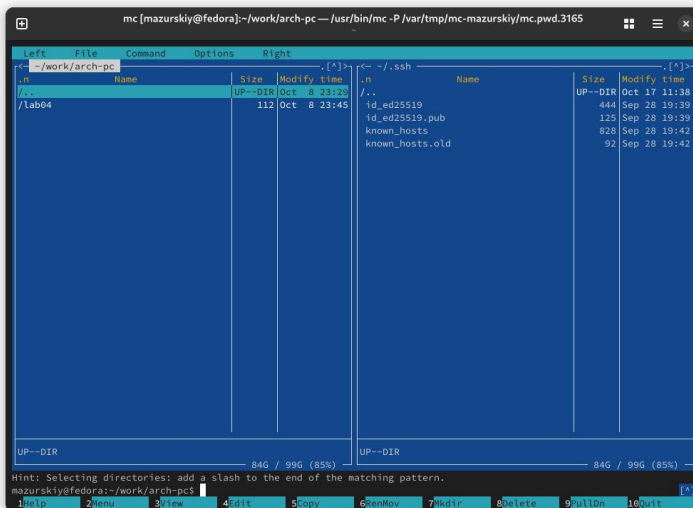


Рис. 3: Открытый каталог arch-pc

С помощью функциональной клавиши, я создаю подкаталог lab05, в котором буду работать (рис. 4).

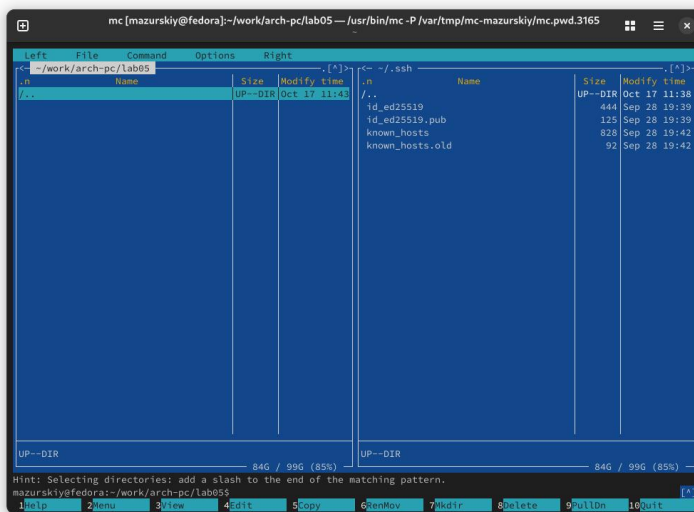


Рис. 4: Создание рабочего подкаталога

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

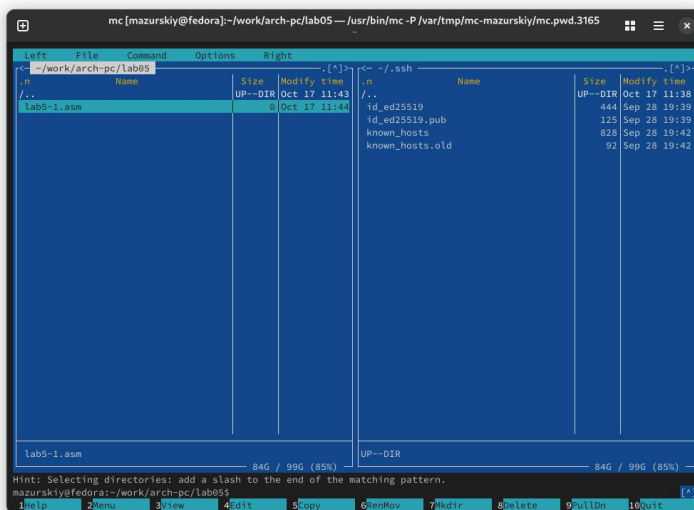
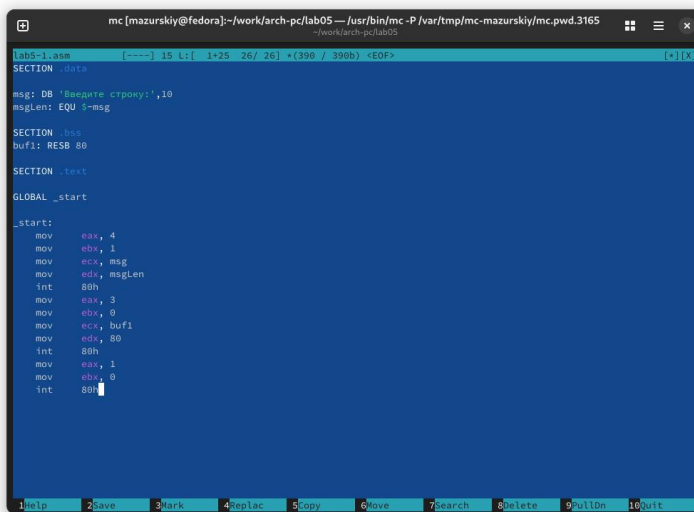


Рис. 5: Создание файла в Midnight Commander

4.2 Работа в NASM

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

A screenshot of the Midnight Commander file manager in a terminal window. The window title is 'mc [mazurskiy@fedora]~/work/arch-pc/lab05 — /usr/bin/mc -P /var/tmp/mc-mazurskiy/mc.pwd.3165'. The status bar at the top shows 'lab5-1.asm' and file statistics. The main pane displays the assembly code of 'lab5-1.asm'. The code includes sections for data, bss, and text, with various instructions like 'mov', 'int', and 'push' being edited. The bottom status bar shows function keys: 1help, 2save, 3mark, 4replac, 5copy, 6move, 7search, 8delete, 9pulldn, 10quit.

```
mc [mazurskiy@fedora]~/work/arch-pc/lab05 — /usr/bin/mc -P /var/tmp/mc-mazurskiy/mc.pwd.3165
lab5-1.asm  [----] 15 L: 1+25 26/ 26 *(390 / 390b) <EOF>

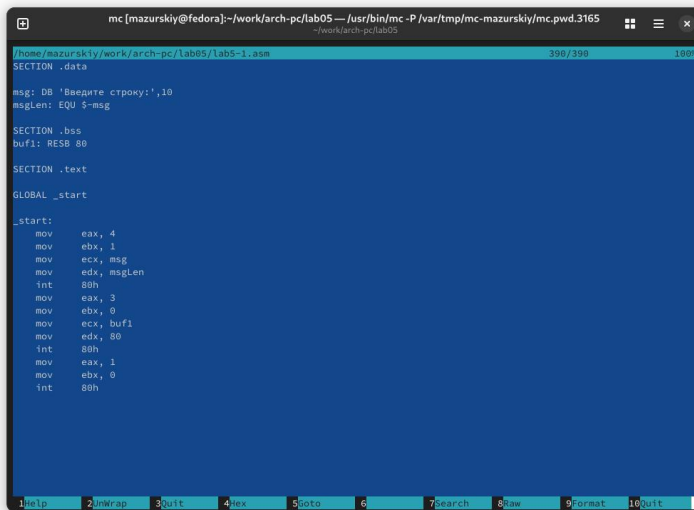
SECTION .data
msg: DB "Введите строку:",10
msglen: EQU $-msg

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. 6: Редактирование файла в Midnight Commander

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

A screenshot of the Midnight Commander file manager, similar to the previous one, but with the status bar at the top showing file statistics: '390/390' and '100%'. The assembly code in the main pane is identical to the one in Figure 6. The bottom status bar shows function keys: 1help, 2wrap, 3quit, 4lex, 5goto, 6, 7search, 8aw, 9format, 10quit.

```
mc [mazurskiy@fedora]~/work/arch-pc/lab05 — /usr/bin/mc -P /var/tmp/mc-mazurskiy/mc.pwd.3165
/home/mazurskiy/work/arch-pc/lab05/lab5-1.asm  390/390  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, 0
mov     ecx, buf1
mov     edx, 80
int     80h
mov     eax, 1
mov     ebx, 0
int     80h
```

Рис. 7: Проверка сохранения сделанных изменений

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

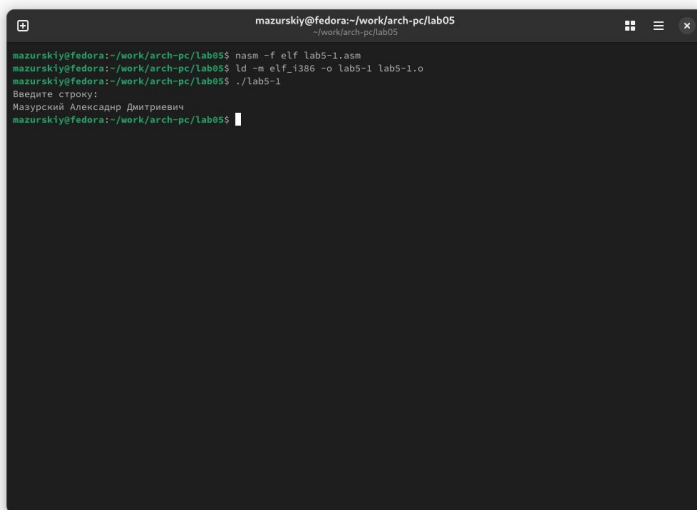


Рис. 8: Трансляция, компоновка и последующий запуск программы

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).

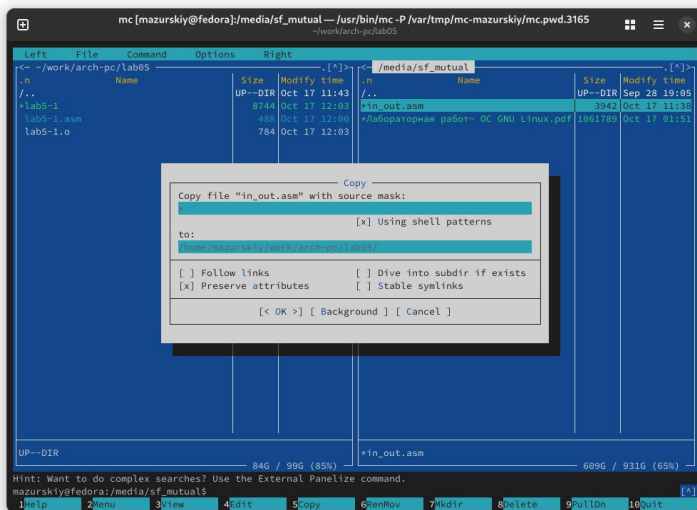


Рис. 9: Копирование файла в рабочий каталог

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

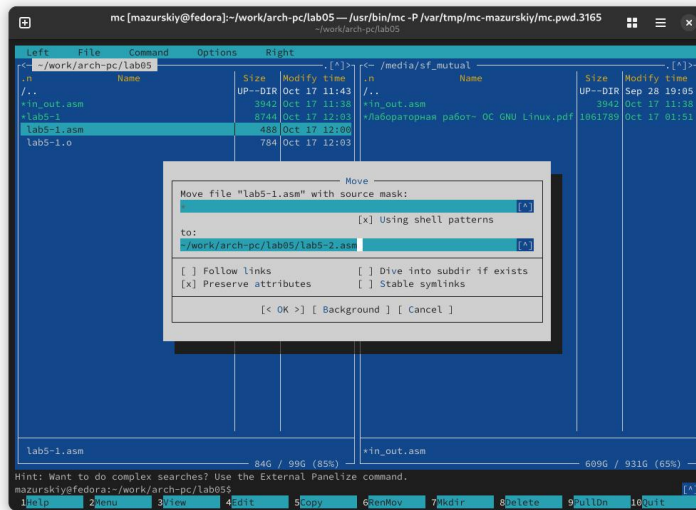


Рис. 10: Создание копии файла в Midnight Commander

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

```
%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    sprintf

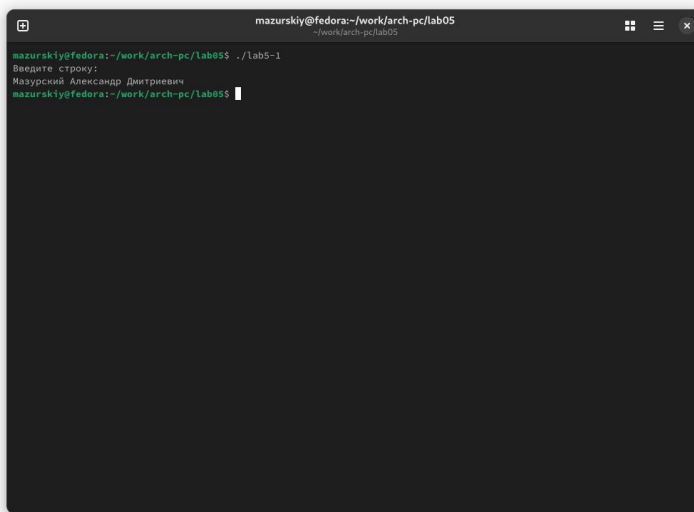
    mov     ecx, buf1
    mov     edx, 80

    call    sread

    call    quit
```

Рис. 11: Изменение программы

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

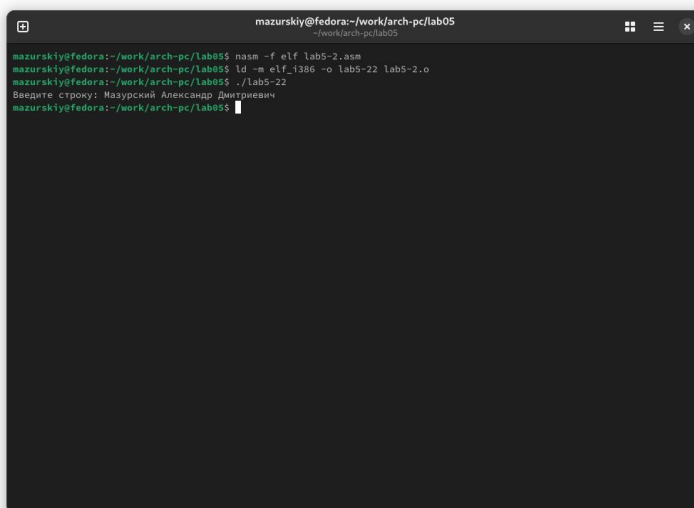


```
mazurskiy@fedora:~/work/arch-pc/lab05
mazurskiy@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Мазурский Александр Дмитриевич
mazurskiy@fedora:~/work/arch-pc/lab05$
```

Рис. 12: Запуск измененной программы

Редактирую файл и заменяю в нем подпрограмму `sprintLF` на `sprint`. Разница подпрограмм в том, что вторая вызывает ввод на той же строке (рис. 13).

4.4 Задание для самостоятельной работы



```
mazurskiy@fedora:~/work/arch-pc/lab05
mazurskiy@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mazurskiy@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-22 lab5-2.o
mazurskiy@fedora:~/work/arch-pc/lab05$ ./lab5-22
Введите строку: Мазурский Александр Дмитриевич
mazurskiy@fedora:~/work/arch-pc/lab05$
```

Рис. 13: Запуск измененной программы с другой подпрограммой

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

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

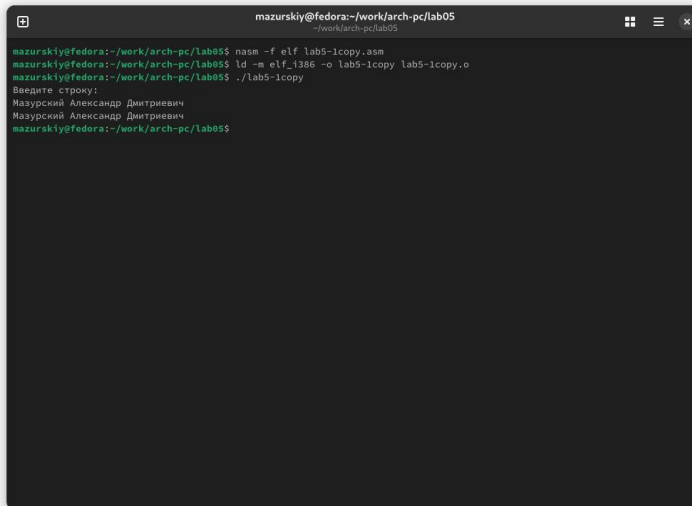
SECTION .bss
buf1: RESB 80

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. 14: Редактирование копии

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



```
mazurskiy@fedora:~/work/arch-pc/lab05
mazurskiy@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1copy.asm
mazurskiy@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1copy lab5-1copy.o
mazurskiy@fedora:~/work/arch-pc/lab05$ ./lab5-1copy
Введите строку:
Мазурский Александр Дмитриевич
mazurskiy@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     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
```

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

```
%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
```

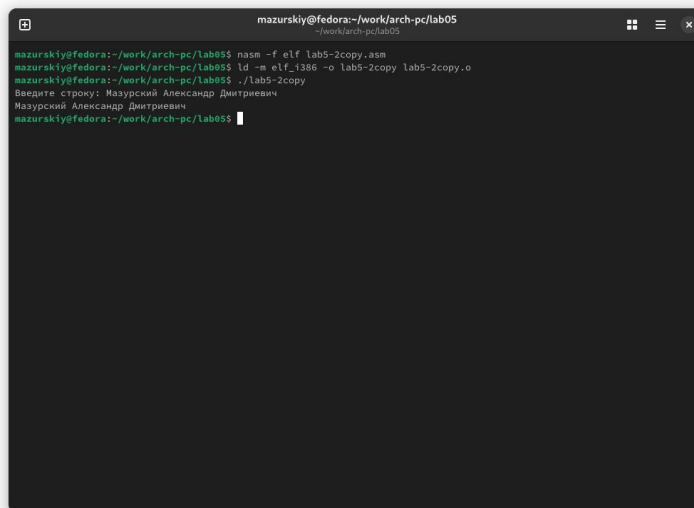
```
call sread
```

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

```
call quit
```

Рис. 16: Редактирование копии

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



```
mazurskiy@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2copy.asm  
mazurskiy@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2copy lab5-2copy.o  
mazurskiy@fedora:~/work/arch-pc/lab05$ ./lab5-2copy  
Введите строку: Мазурский Александр Дмитриевич  
Мазурский Александр Дмитриевич  
mazurskiy@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.