

# **Лабораторная работа номер 5**

**Отчёт**

Виноградова Мария Андреевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
4.1	Порядок выполнения лабораторной работы . . . . .	9
4.2	Задание для самостоятельной работы . . . . .	15
<b>5</b>	<b>Выводы</b>	<b>19</b>

# Список иллюстраций

4.1	Вводим в консоль команду ms . . . . .	9
4.2	Переходим в каталог . . . . .	10
4.3	Создаем каталог функциональной клавишей F7 . . . . .	10
4.4	Воспользуемся командой touch . . . . .	11
4.5	Открываем файл функциональной клавишей, заполняем и сохраняем	11
4.6	Открываем файл и убеждаемся, что файл содержит текст программы	12
4.7	Проверяем, как работает данная программа . . . . .	12
4.8	Скачиваем файл . . . . .	12
4.9	Копируем скаченный файл . . . . .	13
4.10	Создаем копию файла клавишей F6 . . . . .	13
4.11	Проверяем скопировался ли файл . . . . .	14
4.12	Открываем и заполняем файл . . . . .	14
4.13	Смотрим, как сработала программа . . . . .	14
4.14	Редактируем файл . . . . .	15
4.15	Смотрим, как сработала программа и сравниваем с прошлой . . . .	15
4.16	Создаем копию файла lab5-1.asm . . . . .	16
4.17	Редактируем файл . . . . .	16
4.18	Проверяем правильность написания программы . . . . .	17
4.19	Создаем копию файла lab5-2.asm . . . . .	17
4.20	Редактируем файл . . . . .	18
4.21	Проверяем правильность написания программы . . . . .	18

## **Список таблиц**

# 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 не инициализированных данных (тех, под которые во время компиляции и только отводится память, а значение присваивается входе 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 (четырёхбайтное 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, src—источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непо-

средственные 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 Порядок выполнения лабораторной работы

Открываем Midnight Commander (рис. 4.1).

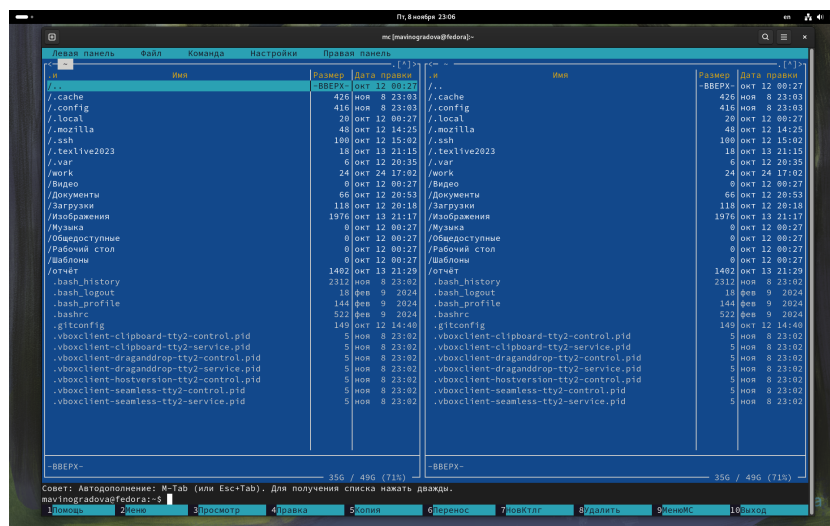


Рис. 4.1: Вводим в консоль команду mc

Переходим в каталог, созданный при выполнении 4 ЛБ (рис. 4.2).

Левая панель				Правая панель			
Файл				Команда			
Настройка				Правая панель			
-				-			
Имя	Размер	Дата правки	Имя	Размер	Дата правки	Имя	Размер
./	-BBERX-	окт 24 17:02	./	-BBERX-	окт 12 00:27	./	-BBERX-
/lab04	104	окт 24 18:51	./cache	426	ноя 8 23:03	./local	20
			./config	416	ноя 8 23:03	./mozilla	48
			./ssh	100	окт 12 15:02	./texlive2023	18
			./var	6	окт 12 20:35	./work	24
			/work	0	окт 12 00:27	/video	0
			/Документы	66	окт 12 20:53	/Загрузки	118
			/Изображения	1976	окт 12 20:18	/Рабочий стол	0
			/Музыка	0	окт 12 00:27	/Шаблоны	0
			/Общедоступные	0	окт 12 00:27	./bash_history	1402
			./Рабочий стол	0	окт 12 00:27	./bash_logout	2312
			./Шаблоны	0	окт 12 00:27	./bash_profile	16
			./bash_history	1402	окт 13 21:29	./bashrc	144
			./bash_logout	2312	ноя 8 23:02	./gitconfig	522
			./bash_profile	16	фев 9 2024	./vboxClient-clipboard-tty2-control.pid	149
			./bashrc	144	фев 9 2024	./vboxClient-clipboard-tty2-service.pid	5
			./gitconfig	149	окт 12 14:40	./vboxClient-draganddrop-tty2-control.pid	5
			./vboxClient-clipboard-tty2-control.pid	5	ноя 8 23:02	./vboxClient-draganddrop-tty2-service.pid	5
			./vboxClient-clipboard-tty2-service.pid	5	ноя 8 23:02	./vboxClient-hostversion-tty2-control.pid	5
			./vboxClient-draganddrop-tty2-control.pid	5	ноя 8 23:02	./vboxClient-hostversion-tty2-service.pid	5
			./vboxClient-draganddrop-tty2-service.pid	5	ноя 8 23:02	./vboxClient-seamless-tty2-control.pid	5
			./vboxClient-hostversion-tty2-control.pid	5	ноя 8 23:02	./vboxClient-seamless-tty2-service.pid	5
			./vboxClient-hostversion-tty2-service.pid	5	ноя 8 23:02		
			./vboxClient-seamless-tty2-control.pid	5	ноя 8 23:02		
			./vboxClient-seamless-tty2-service.pid	5	ноя 8 23:02		

Рис. 4.2: Переходим в каталог

Создаем каталог lab05 (рис. 4.3).

Левая панель				Правая панель			
Файл				Команда			
Настройка				Правая панель			
-				-			
Имя	Размер	Дата правки	Имя	Размер	Дата правки	Имя	Размер
./	-BBERX-	окт 24 17:02	./	-BBERX-	окт 12 00:27	./	-BBERX-
/lab04	104	окт 24 18:51	./cache	426	ноя 8 23:03	./local	20
			./config	416	ноя 8 23:03	./mozilla	48
			./ssh	100	окт 12 15:02	./texlive2023	18
			./var	6	окт 12 20:35	./work	24
			/work	0	окт 12 00:27	/video	0
			/Документы	66	окт 12 20:53	/Загрузки	118
			/Изображения	1976	окт 12 20:18	/Рабочий стол	0
			/Музыка	0	окт 12 00:27	/Шаблоны	0
			/Общедоступные	0	окт 12 00:27	./bash_history	1402
			./Рабочий стол	0	окт 12 00:27	./bash_logout	2312
			./Шаблоны	0	окт 12 00:27	./bash_profile	16
			./bash_history	1402	окт 13 21:29	./bashrc	144
			./bash_logout	2312	ноя 8 23:02	./gitconfig	522
			./bash_profile	16	фев 9 2024	./vboxClient-clipboard-tty2-control.pid	149
			./bashrc	144	фев 9 2024	./vboxClient-clipboard-tty2-service.pid	5
			./gitconfig	149	окт 12 14:40	./vboxClient-draganddrop-tty2-control.pid	5
			./vboxClient-clipboard-tty2-control.pid	5	ноя 8 23:02	./vboxClient-draganddrop-tty2-service.pid	5
			./vboxClient-clipboard-tty2-service.pid	5	ноя 8 23:02	./vboxClient-hostversion-tty2-control.pid	5
			./vboxClient-draganddrop-tty2-control.pid	5	ноя 8 23:02	./vboxClient-hostversion-tty2-service.pid	5
			./vboxClient-draganddrop-tty2-service.pid	5	ноя 8 23:02	./vboxClient-seamless-tty2-control.pid	5
			./vboxClient-hostversion-tty2-control.pid	5	ноя 8 23:02	./vboxClient-seamless-tty2-service.pid	5
			./vboxClient-hostversion-tty2-service.pid	5	ноя 8 23:02		
			./vboxClient-seamless-tty2-control.pid	5	ноя 8 23:02		
			./vboxClient-seamless-tty2-service.pid	5	ноя 8 23:02		

Рис. 4.3: Создаем каталог функциональной клавишей F7

Создаем файл lab5-1.asm (рис. 4.4).

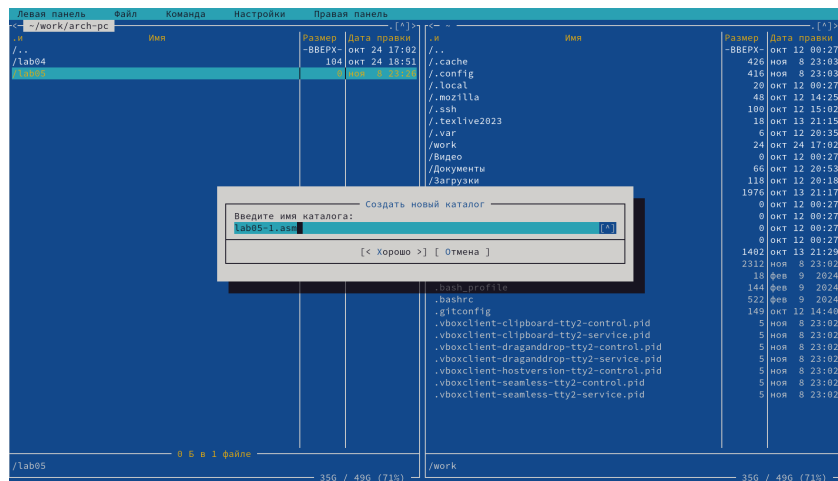


Рис. 4.4: Воспользуемся командой touch

Открываем файл для редактирования и заполняем его по листингу (рис. 4.5).

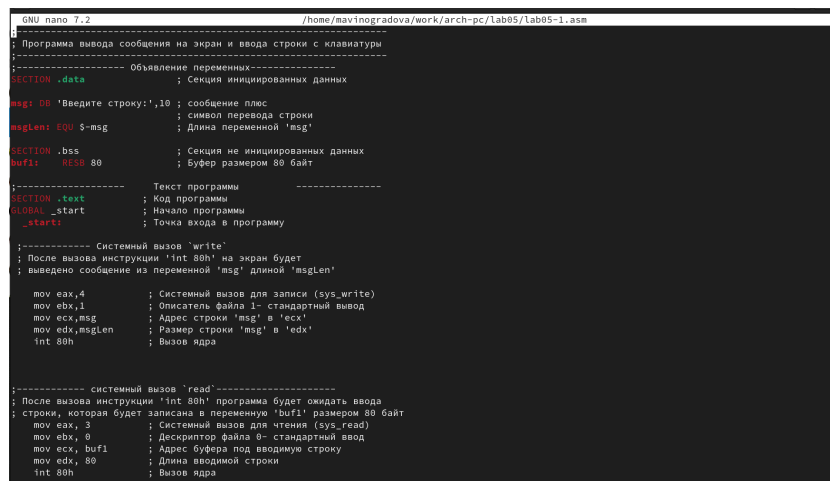


Рис. 4.5: Открываем файл функциональной клавишей, заполняем и сохраняем

Открывем файл для просмотра (рис. 4.6).

```

msglen: EQU $-msg          ; символ перевода строки
                           ; Длина переменной 'msg'
SECTION .bss               ; Секция не инициализированных 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)
mov ebx, 0                 ; Дескриптор файла 0- стандартный ввод
mov ecx, buf1              ; Адрес буфера под вводимую строку
mov edx, 80                ; Длина вводимой строки
int 80h                   ; Вызов ядра

;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу

mov eax, 1                 ; Системный вызов для выхода (sys_exit)
mov ebx, 0                 ; Выход с кодом возврата 0 (без ошибок)
int 80h                   ; Вызов ядра

```

Рис. 4.6: Открываем файл и убеждаемся, что файл содержит текст программы

Транслируем текст программы и запускаем исполняемый файл (рис. 4.7).

```

mavinogradova@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
mavinogradova@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
mavinogradova@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Виноградова Мария Андреевна
mavinogradova@fedora:~/work/arch-pc/lab05$

```

Рис. 4.7: Проверяем, как работает данная программа

Скачиваем файл со страницы курса (рис. 4.8).

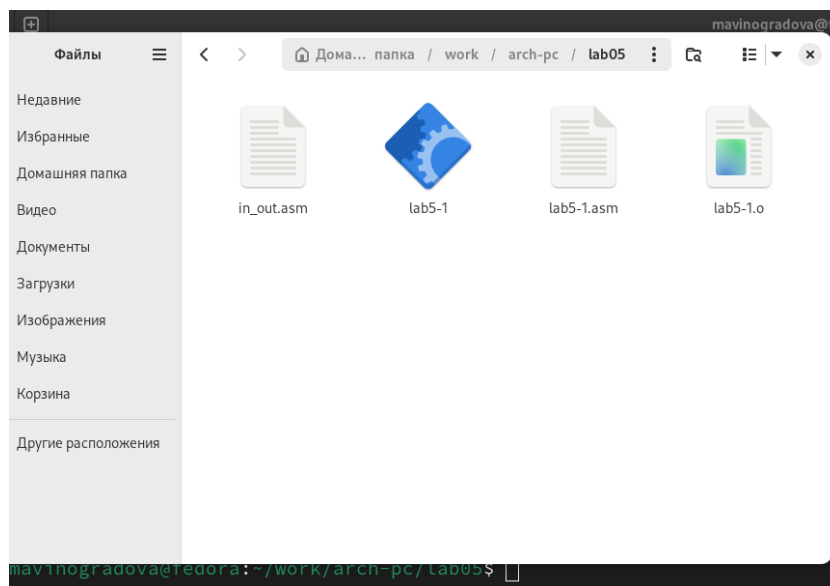


Рис. 4.8: Скачиваем файл

Копируем файл в нужную директорию (рис. 4.9).

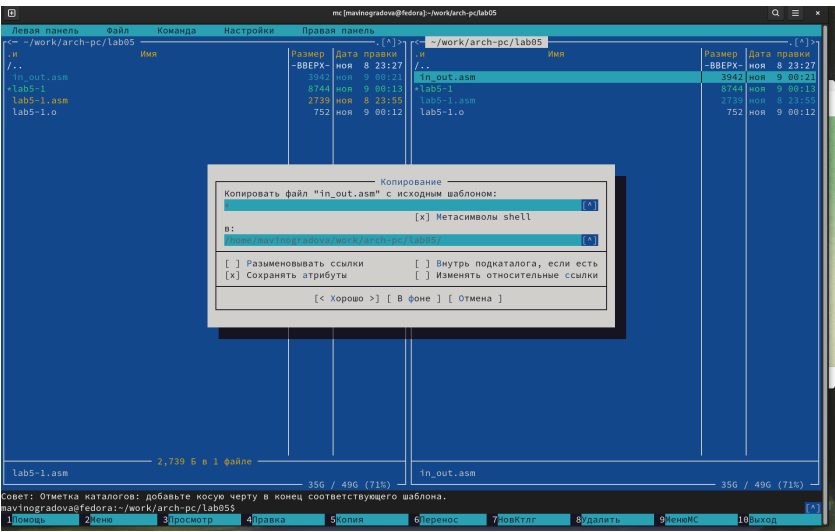


Рис. 4.9: Копируем скаченный файл

Создаем копию файла lab5-1.asm (рис. 4.10).

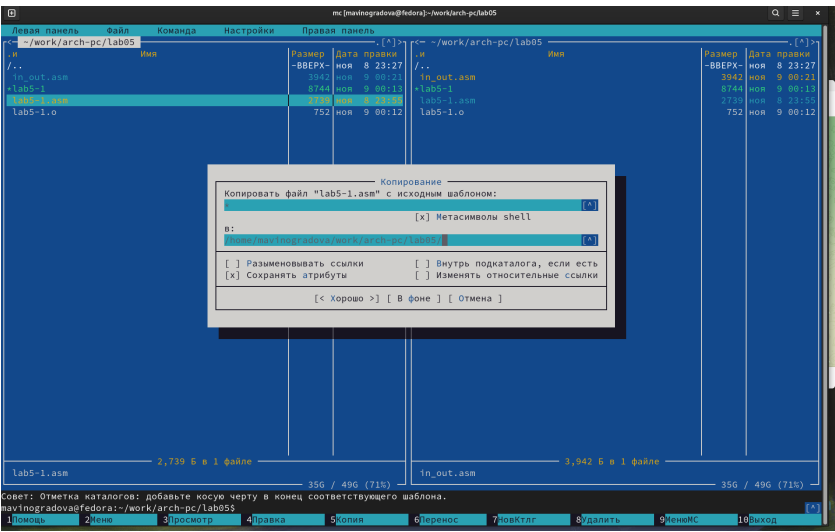


Рис. 4.10: Создаем копию файла клавишей F6

Проверяем созданный файл (рис. 4.11).

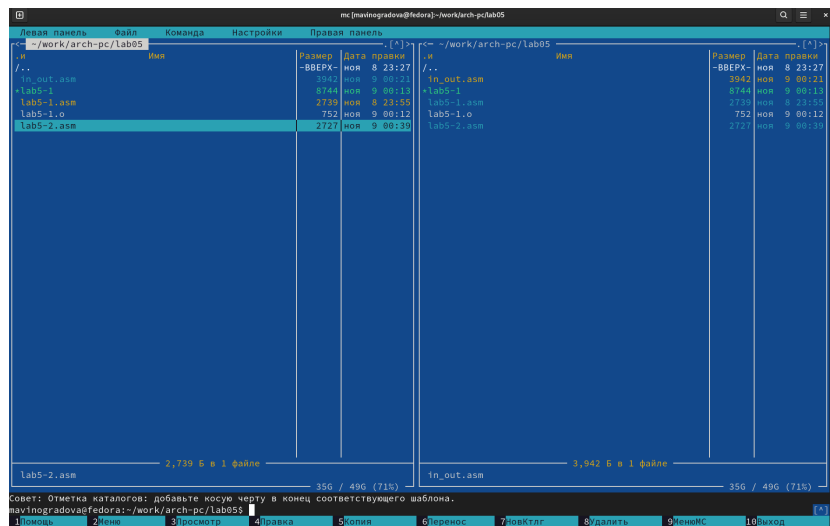


Рис. 4.11: Проверяем скопировался ли файл

Открываем новый файл и заполняем его в соответствии с листингом (рис. 4.12).

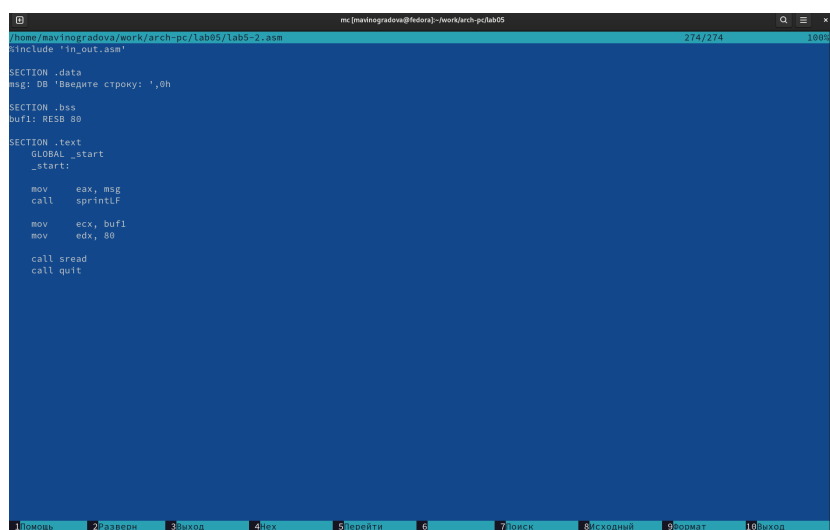


Рис. 4.12: Открываем и заполняем файл

Транслируем и запускаем новый файл (рис. 4.13).

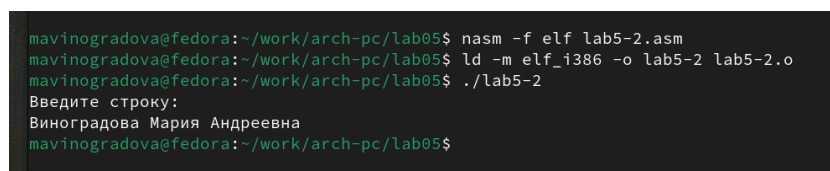
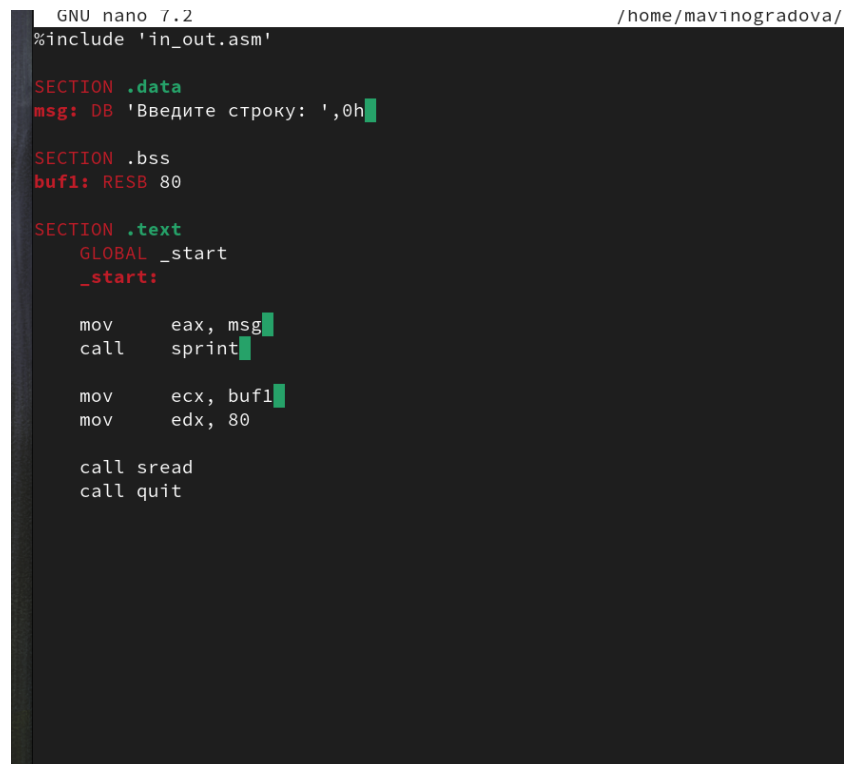


Рис. 4.13: Смотрим, как сработала программа

Снова открываем файл для редактирования и меняем `sprintLF` на `sprint` (рис. 4.14).



```
GNU nano 7.2 /home/mavinogradova/
#include 'in_out.asm'

SECTION .data
msg: DB 'Введите строку: ',0h

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

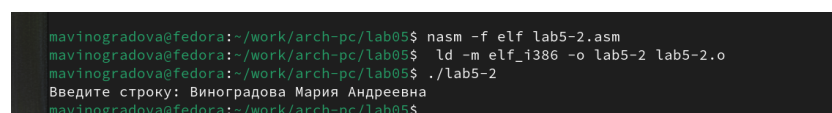
    mov     eax, msg
    call    sprint

    mov     ecx, buf1
    mov     edx, 80

    call    sread
    call    quit
```

Рис. 4.14: Редактируем файл

Транслируем и запускаем файл (рис. 4.15).



```
mavinogradova@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mavinogradova@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
mavinogradova@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Виноградова Мария Андреевна
mavinogradova@fedora:~/work/arch-pc/lab05$
```

Рис. 4.15: Смотрим, как сработала программа и сравниваем с прошлой

Таким образом можем понять, что команда `sprint` выводит текст в той же строке, а `sprintLF` переносит на новую строку.

## 4.2 Задание для самостоятельной работы

Создаем копию файла `lab5-1.asm` и называем его так же (рис. 4.16).

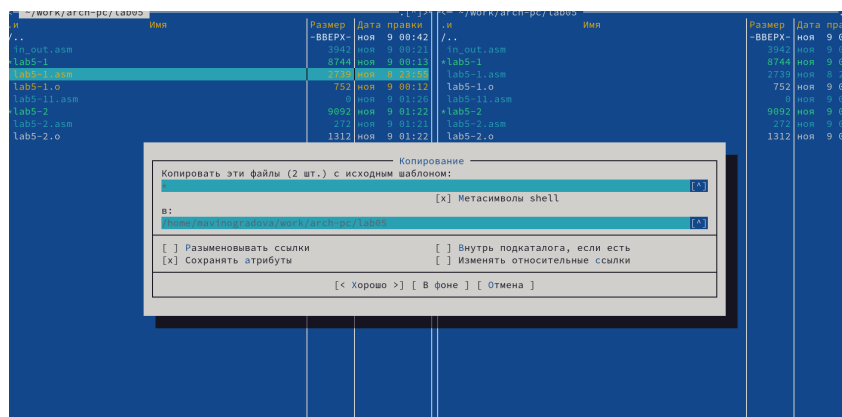


Рис. 4.16: Создаем копию файла lab5-1.asm

Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль (рис. 4.17).

```

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)
    mov ebx, 0          ; Дескриптор 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
    mov ebx, 1
    mov ecx, buf1
    mov edx, 80
    int 80h

;----- Системный вызов `exit`-----
; После вызова инструкции `int 80h` программа завершит работу

    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: Редактируем файл

Транслируем файл и запускаем программу (рис. 4.18).



```

mavinogradova@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
mavinogradova@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
mavinogradova@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Виноградова Мария Андреевна
Виноградова Мария Андреевна

```

Рис. 4.18: Проверям правильность написания программы

Создаем копию файла lab5-2.asm и называем его так же (рис. 4.19).

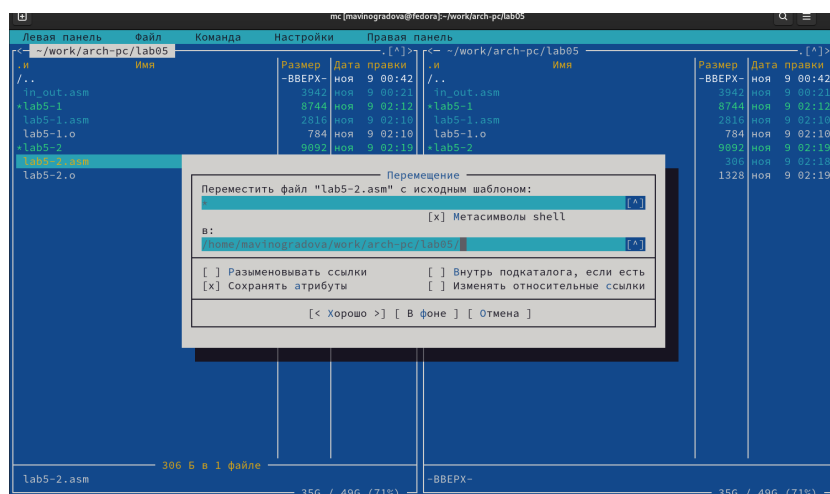


Рис. 4.19: Создаем копию файла lab5-2.asm

Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль (рис. 4.20).

```
%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    sprintf

    mov     ecx, buf1
    mov     edx, 80

    call    sread
    mov     eax, buf1
    call    sprint
    call    quit
```

Рис. 4.20: Редактируем файл

Транслируем файл и запускаем программу (рис. 4.21).

```
mavinogradova@fedora:~/work/arch-pc/lab05$ mc
mavinogradova@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mavinogradova@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
mavinogradova@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Виноградова Мария Андреевна
Виноградова Мария Андреевна
mavinogradova@fedora:~/work/arch-pc/lab05$ mc
mavinogradova@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mavinogradova@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
mavinogradova@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Виноградова Мария Андреевна
Виноградова Мария Андреевна
mavinogradova@fedora:~/work/arch-pc/lab05$
```

Рис. 4.21: Проверяем правильность написания программы

## 5 Выводы

Мы приобрели навыки работы с Midnight Commander и освоили инструкцию mov.