

Лабораторная работа №5

Дисциплина: Архитектура компьютера

Серебрякова Дарья Ильинична

Содержание

1	Цель работы	5
2	Задания	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Основы работы с mc	9
4.2	Подключение внешнего файла in_out.asm	13
5	Выполнение заданий для самостоятельной работы	17
6	Выводы	21
	Список литературы	22

Список иллюстраций

4.1	Открыт Midnight Commander	9
4.2	Перемещение по каталогам	10
4.3	Создание папки	10
4.4	Создание файла	10
4.5	Открытие файла для редактирования в mcedit	11
4.6	Изменение текста файла	11
4.7	Проверка сохраненных изменений	12
4.8	Трансляция текста объектный файл	12
4.9	Компоновка объектного файла	13
4.10	Запуск исполняемого файла	13
4.11	История команд	13
4.12	Скачивание файла in_out.asm	14
4.13	Создание копии файла с помощью функциональных клавиш	14
4.14	Создание копии файла	15
4.15	Использование подпрограмм из внешнего файла	15
4.16	Запуск исполняемого файла	16
4.17	Смена подпрограмм	16
5.1	Создание копии файла функциональной клавишей	17
5.2	Редактирование программы	18
5.3	Запуск исполняемого файла, проверка правильности выполнения задания	18
5.4	Создание копии файла	19
5.5	Редактирование текста программы	19
5.6	Запуск исполняемого файла	20

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 Задания

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 mc

Открываю Midnight Commander командой mc (рис. 4.1).

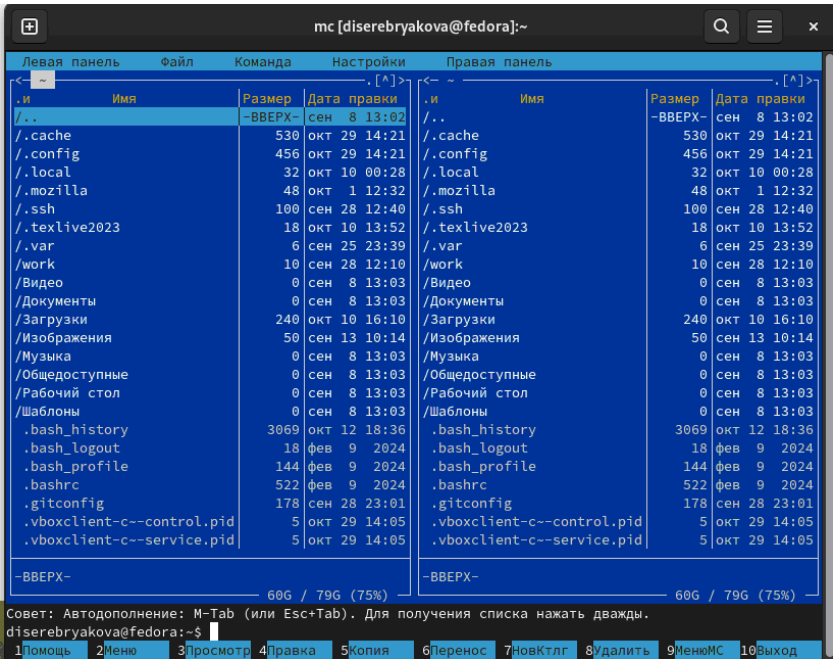


Рис. 4.1: Открыт Midnight Commander

Перехожу в каталог ~/work/study/study_2024-2025_arch-рс, используя файловый менеджер mc и клавиши (рис. 4.2).

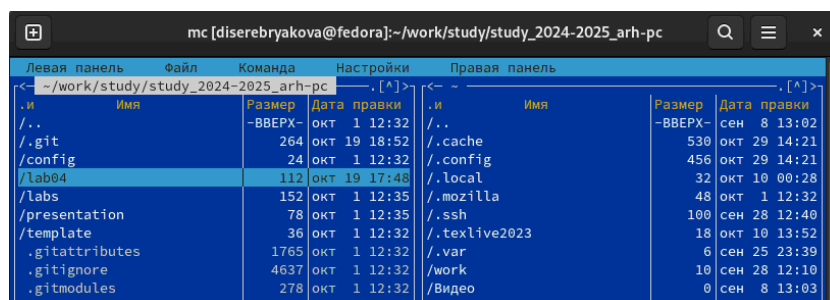


Рис. 4.2: Перемещение по каталогам

С помощью функциональной клавиши F7 создаю папку lab05 и перехожу в созданный каталог (рис. 4.3).

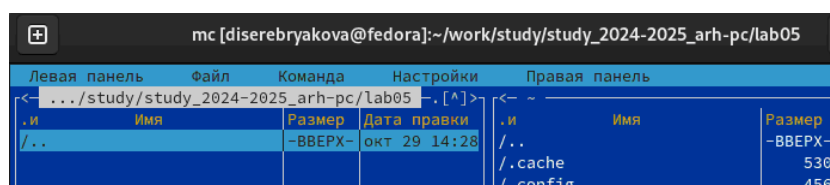


Рис. 4.3: Создание папки

Пользуясь строкой ввода и командой touch создаю файл lab5-1.asm (рис. 4.4).

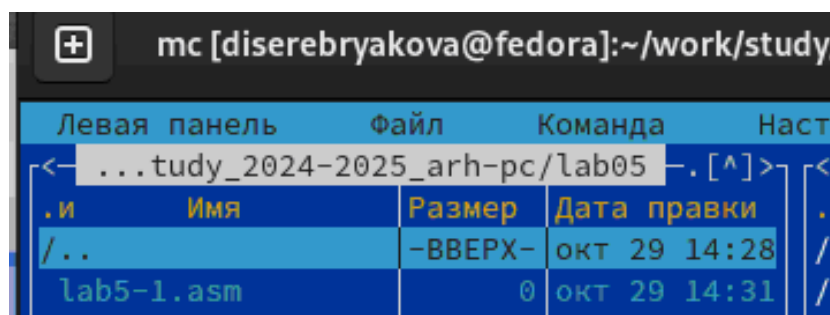


Рис. 4.4: Создание файла

С помощью функциональной клавиши F4 открываю файл lab5-1.asm для редактирования во встроенном редакторе mcedit (рис. 4.5).

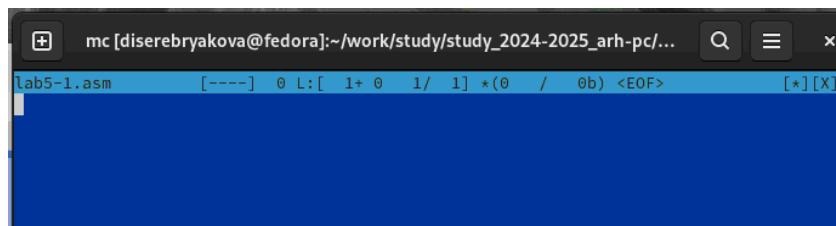


Рис. 4.5: Открытие файла для редактирования в mscedit

Ввожу текст программы из предложенного листинга 5.1, сохраняю изменения и закрываю файл (рис. 4.6).

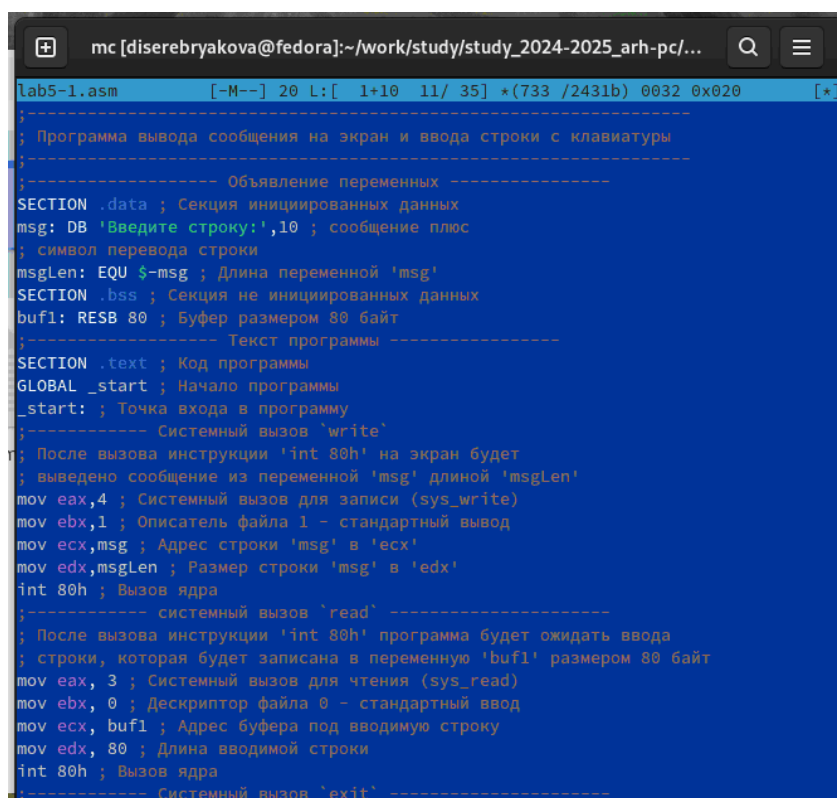


Рис. 4.6: Изменение текста файла

С помощью функциональной клавиши F3 открываю файл lab5-1.asm для просмотра. Убеждаюсь, что файл содержит текст программы (рис. 4.7).

```

mc [diserebryakova@fedora]:~/work/... x diserebryakova@fedora:~ x
/home/diserebryakova/work-5_arh-pc/lab05/lab5-1.asm 2145/2431 88
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; 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)

```

Рис. 4.7: Проверка сохраненных изменений

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-1.asm`. Создался объектный файл `lab5-1.o`. (рис. 4.8).

Левая панель	Файл	Команда	На
<--	...tudy_2024-2025_arh-pc/lab05	-. [^]>	
.и	Имя	Размер	Дата правки
/..		-ВВЕРХ-	окт 29 14:28
	lab5-1.asm	2431	окт 29 14:40
	lab5-1.o	752	окт 29 14:56

Рис. 4.8: Трансляция текста объектный файл

Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-1 lab5-1.o`. Создался исполняемый файл `lab5-1`. (рис. 4.9).

Левая панель	Файл	Команда	На
<	...	study_2024-2025_arh-pc/lab05	-. [^]>
.и	Имя	Размер	Дата правки
/..		-ВВЕРХ-	окт 29 14:28
*lab5-1		8744	окт 29 15:05
lab5-1.asm		2431	окт 29 14:40
lab5-1.o		752	окт 29 14:56

Рис. 4.9: Компоновка объектного файла

Запускаю исполняемый файл командой `./lab5-1`. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. 4.10).

```
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ./lab5-1
Введите строку:
```

Рис. 4.10: Запуск исполняемого файла

Введенные выше команды (рис. 4.11).

```
История
touch lab5-1.asm
nasm -f elf lab5-1.asm
ld -m elf_i386 -o lab5-1 lab5-1.o
./lab5-1
```

Рис. 4.11: История команд

4.2 Подключение внешнего файла `in_out.asm`

Скачиваю файл `in_out.asm` со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. 4.12).

Левая панель		Файл	Команда	Нас
<- ~/Загрузки				. [^]>
.и	Имя	Размер	Дата правки	
/..		-ВВЕРХ-	окт 29 14:05	
/Telegram Desktop		1166	окт 19 18:45	
/tsetup.5.6 (2).1		16	окт 10 13:36	
/tsetup.5.6.1		16	окт 10 13:35	
in_out.asm		3942	окт 30 15:39	
pandoc-3.~4.tar.gz		31656K	окт 10 13:44	
pandoc-cr~x.tar.xz		10727K	окт 10 13:44	
tsetup.5.~1.tar.xz		56337K	окт 10 13:35	

Рис. 4.12: Скачивание файла in_out.asm

В одной из панелей mc открываю каталог с файлом lab5-1.asm. В другой панели каталог со скаченным файлом in_out.asm. Копирую файл in_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5 (рис. 4.13).

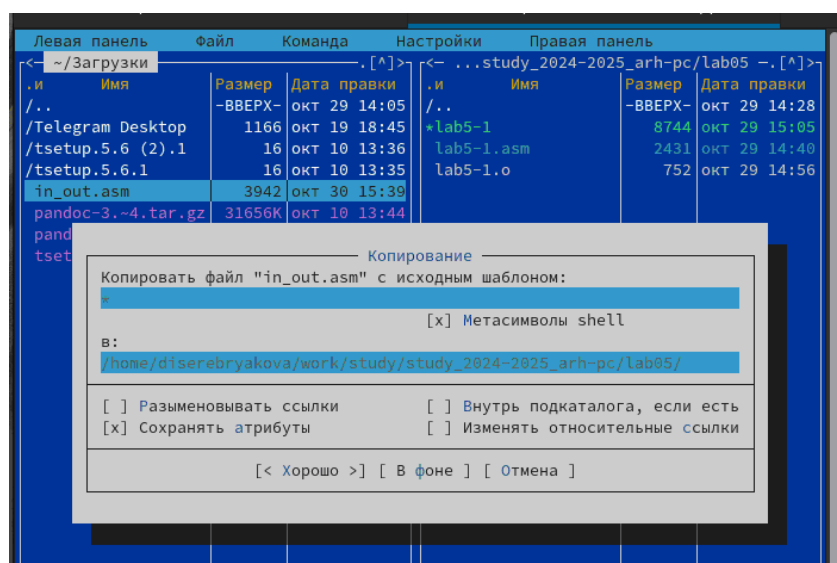


Рис. 4.13: Создание копии файла с помощью функциональных клавиш

С помощью функциональной клавиши F5 создаю копию файла lab5-1.asm с именем lab5-2.asm (рис. 4.14).

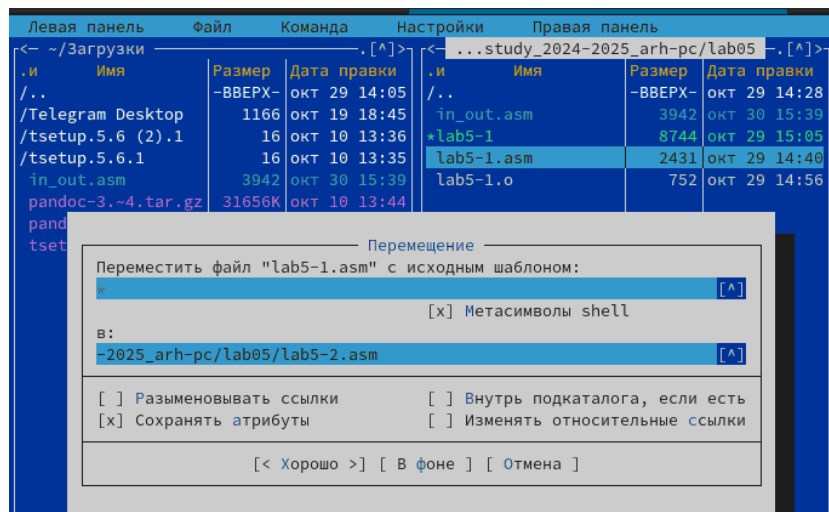


Рис. 4.14: Создание копии файла

Исправляю текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm (использую подпрограммы sprintf, sread и quit) (рис. 4.15).

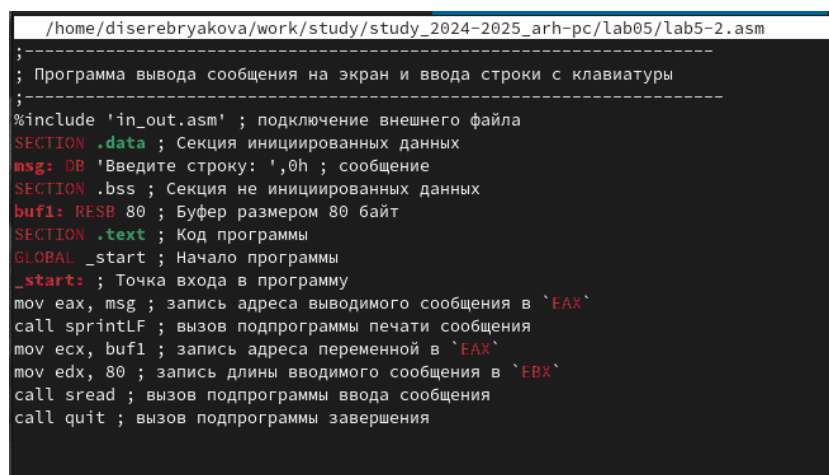


Рис. 4.15: Использование подпрограмм из внешнего файла

Создаю объектный файл lab5-2.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свои ФИО, далее программа выводит введенные мною данные (image/рис. 4.16).

```

diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ nasm -f elf lab5-2.asm
lab5-2.asm:47: error: parser: instruction expected
nasm -f elf lab5-2.asm
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ./lab5-2
Введите строку:
Серебрякова Дарья Ильинична
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$

```

Рис. 4.16: Запуск исполняемого файла

В файле lab5-2.asm заменила подпрограмму sprintLF на sprint. Создала исполняемый файл и запустила его (рис. 4.17).

```

diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ nasm -f elf lab5-2.asm
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ./lab5-2
Введите строку: Серебрякова Дарья Ильинична
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$

```

Рис. 4.17: Смена подпрограмм

Разница между исполняемым файлом lab5-2 и только что измененным этим файлом в том, что запуск первого запрашивает ввод с новой строки, а программа, которая выполняется при запуске второго, запрашивает ввод без переноса на новую строку, потому что в этом заключается различие между подпрограммами sprintLF и sprint

5 Выполнение заданий для самостоятельной работы

Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5 (рис. 5.1).

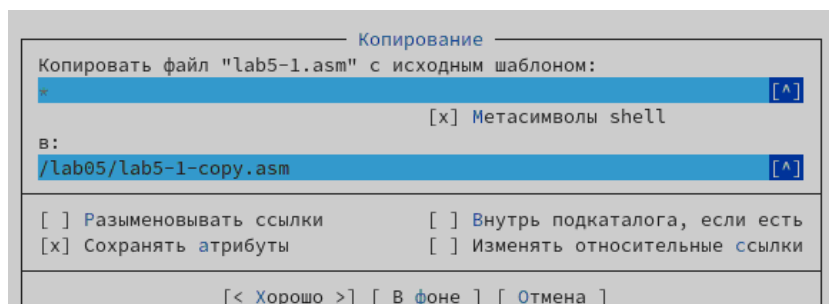


Рис. 5.1: Создание копии файла функциональной клавишей

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 5.2).

```

/home/diserebryakova/work/study/study_2024-2025_arh-pc/lab05/lab5-1-copy.asm
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; 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
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

```

Рис. 5.2: Редактирование программы

Теперь проверю правильность работы отредактированного файла. Создаю объектный файл lab5-1-сору.о, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-1-сору, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свою фамилию, далее программа выводит введенные мною данные (рис. 5.3).

```

diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ nasm -f elf lab5-1-copy.asm
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ld -m elf_i386 -o lab5-1-copy lab5-1-copy.o
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ./lab5-1-copy
Введите строку:
Серебрякова
Серебрякова
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$

```

Рис. 5.3: Запуск исполняемого файла, проверка правильности выполнения задания

Создаю копию файла lab5-2.asm с именем lab5-2-сору.asm с помощью функциональной клавиши F5 (рис. 5.4).

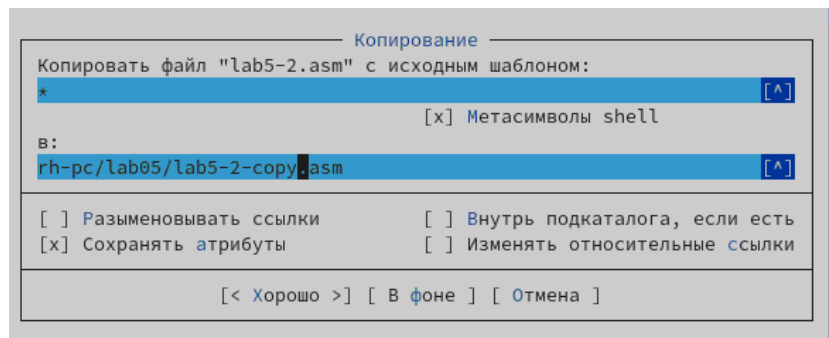


Рис. 5.4: Создание копии файла

Исправляю текст программы с использованием подпрограмм из внешнего файла `in_out.asm`, так чтобы она работала по следующему алгоритму: • вывести приглашение типа “Введите строку:” • ввести строку с клавиатуры • вывести введенную строку на экран (рис. 5.5).

```

;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не иницированных 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 `EAX`
mov edx, 80 ; запись длины вводимого сообщения в `EBX`
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. 5.5: Редактирование текста программы

Создаю объектный файл `lab5-2-сору.о`, отдаю его на обработку компоновщику, получаю исполняемый файл `lab5-2-сору`, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свою Фамилию, далее программа выводит введенные мною данные (рис. 5.6).

```
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ nasm -f elf lab5-2-copy.asm
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ld -m elf_i386 -o lab5-2-copy lab5-
2-copy.o
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$ ./lab5-2-copy
Введите строку: Serebryakova
Serebryakova
diserebryakova@fedora:~/work/study/study_2024-2025_arh-pc/lab05$
```

Рис. 5.6: Запуск исполняемого файла

6 Выводы

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

Список литературы

Лабораторная работа №5