

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Абронина Алиса Кирилловна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
	Список литературы	19

Список иллюстраций

4.1	Открытый ms	9
4.2	Перемещение между директориями и создание каталога	10
4.3	Перемещение между директориями и создание файла	10
4.4	Исполнение файла	11
4.5	Скачанный файл	12
4.6	Копирование файла	12
4.7	Копирование файла	13
4.8	Редактирование файла	13
4.9	Исполнение файла	14
4.10	Отредактированный файл	14
4.11	Исполнение файла	14
4.12	Копирование файла	15
4.13	Редактирование файла	16
4.14	Исполнение файла	16
4.15	Копирование файла	17
4.16	Редактирование файла	17

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. Выполнение лабораторной работы
2. 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 (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, 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

Открываю Midnight Commander, введя в терминал mc (рис. 4.1).

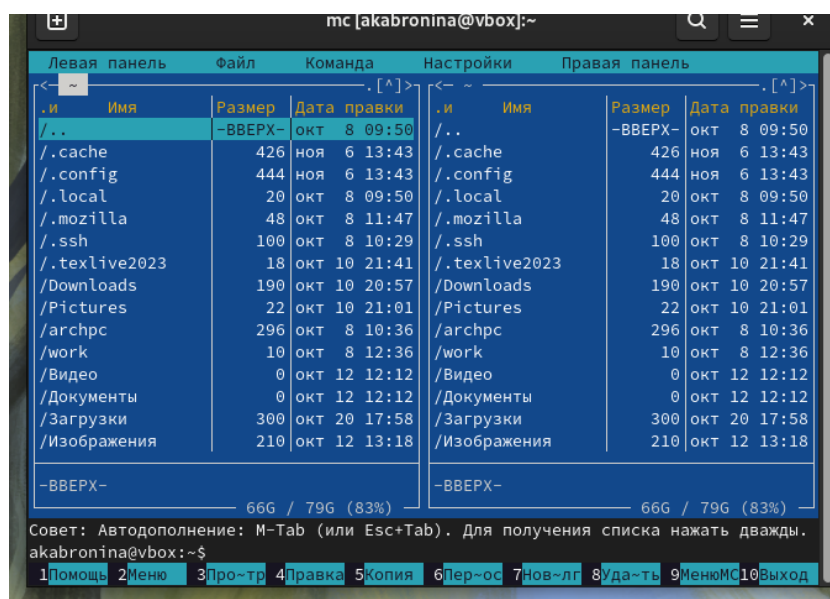


Рис. 4.1: Открытый mc

Перехожу в каталог ~/work/study/2024-2025/Архитектура Компьютера/arch-
ps, используя файловый менеджер mc (рис. 4.2) С помощью функциональной
клавиши F7 создаю каталог lab05

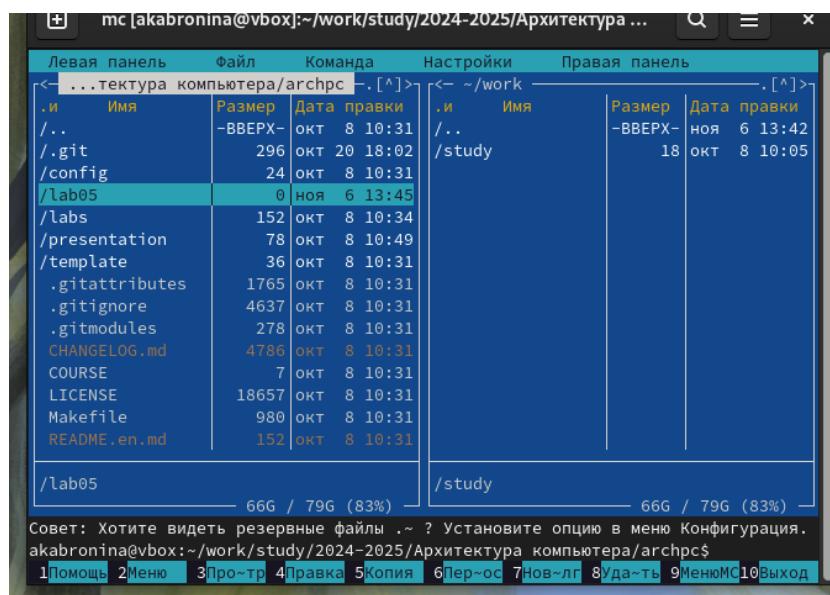


Рис. 4.2: Перемещение между директориями и создание каталога

Переходу в созданный каталог. В строке ввода прописываю команду `touch lab5-1.asm`, чтобы создать файл, в котором буду работать (рис. 4.3).

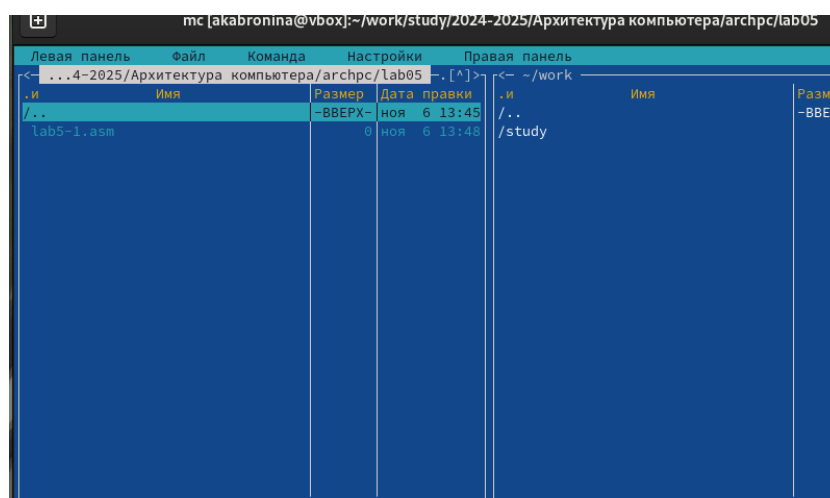
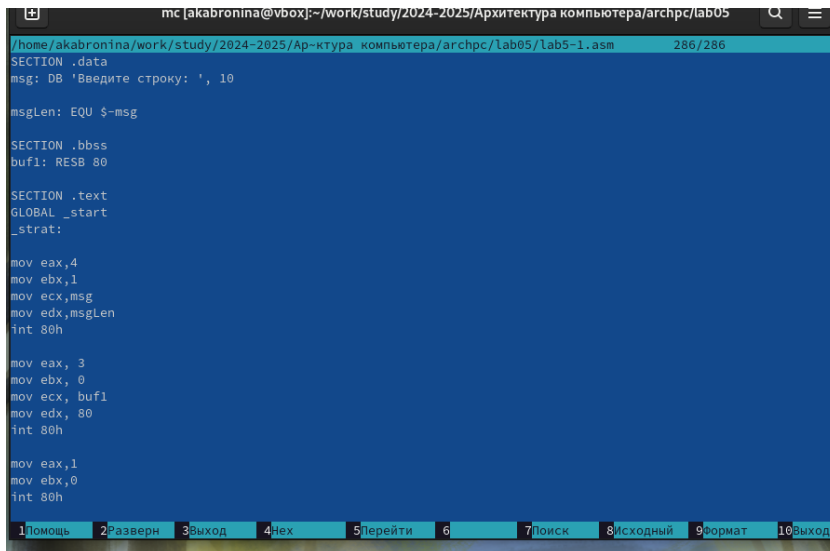


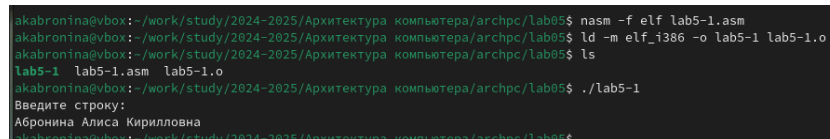
Рис. 4.3: Перемещение между директориями и создание файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Ввожу в файл код программы для запроса строки у пользователя.



```
mc [akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05]
/home/akabronina/work/study/2024-2025/Архитектура компьютера/archpc/lab05/lab5-1.asm 286/286
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
```

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-1.asm`. Создался объектный файл `lab5-1.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-1 lab5-1.o`. Создался исполняемый файл `lab5-1`. Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. 4.4).



```
akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ nasm -f elf lab5-1.asm
akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ls
lab5-1  lab5-1.asm  lab5-1.o
akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ./lab5-1
Введите строку:
Абронина Алиса Кирилловна
akabronina@vbox:~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$
```

Рис. 4.4: Исполнение файла

Скачиваю файл `in_out.asm` со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. 4.5).

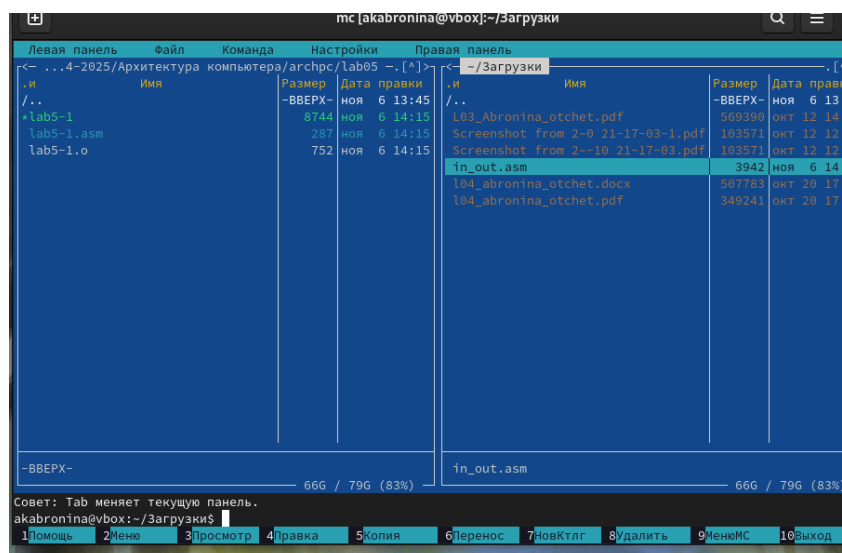


Рис. 4.5: Скачанный файл

С помощью функциональной клавиши F5 копирую файл in_out.asm из каталога Загрузки в созданный каталог lab05 (рис. 4.6).

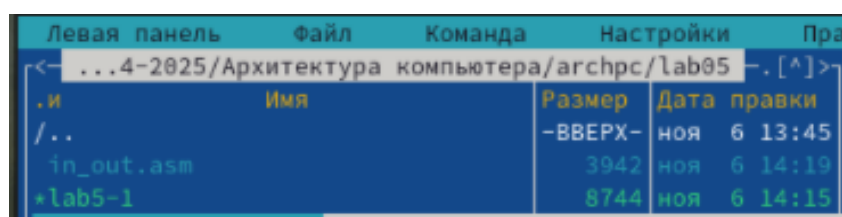


Рис. 4.6: Копирование файла

С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя для копии файла (рис. 4.7).

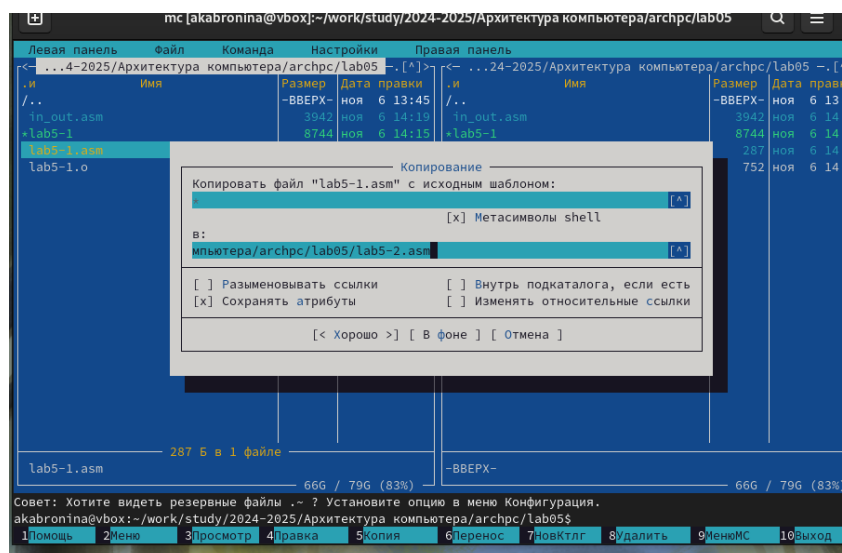


Рис. 4.7: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano (рис. 4.8), чтобы в программе использовались подпрограммы из внешнего файла in_out.asm.

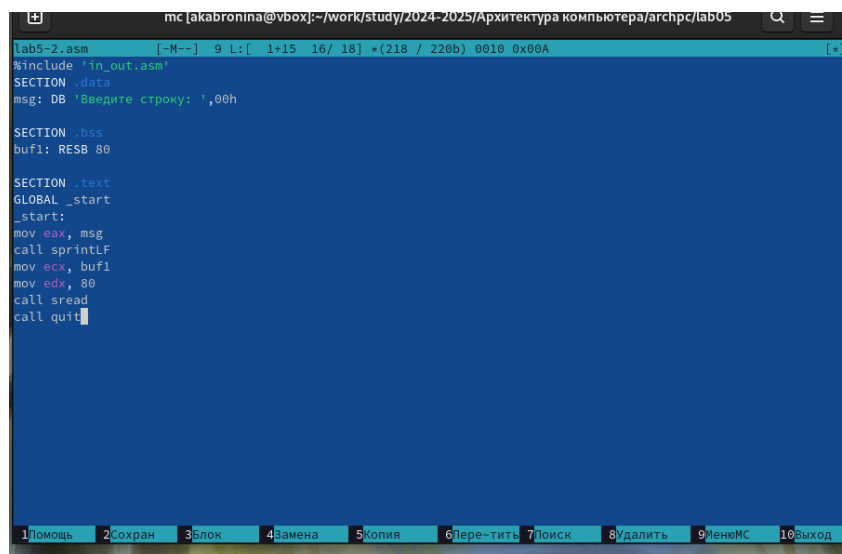


Рис. 4.8: Редактирование файла

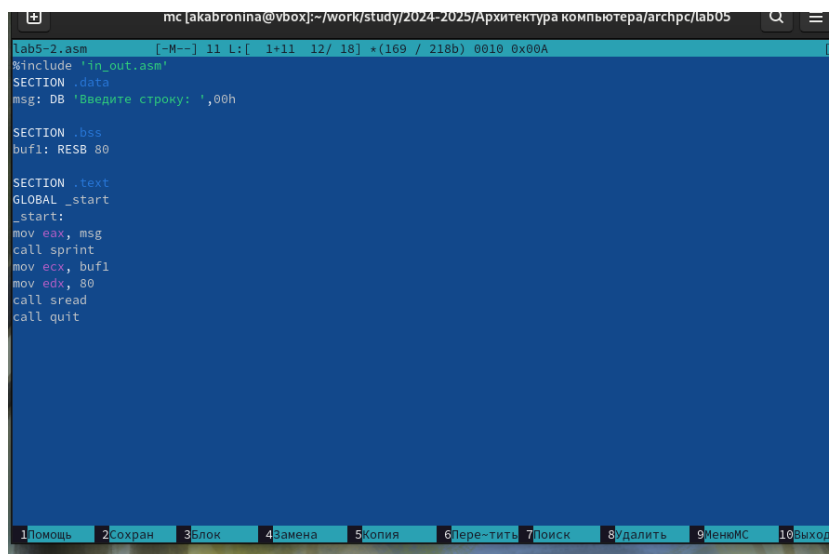
Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-2.asm`. Создался объектный файл lab5-2.o. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-2 lab5-2.o` Создался исполняемый

файл lab5-2. Запускаю исполняемый файл (рис. 4.9).

```
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ nasm -f elf lab5-2.asm
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ./lab5-2
Введите строку:
Абронина Алиса Кирилловна
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ mc
```

Рис. 4.9: Исполнение файла

Открываю файл lab5-2.asm для редактирования в nano функциональной клавишей F4. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения (рис. 4.10).



```
lab5-2.asm [-M--] 11 L: [ 1+11 12/ 18] *(169 / 218b) 0010 0x00A [*]
#include "in_out.asm"
SECTION .data
msg: DB "Введите строку: ",00h

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. 4.10: Отредактированный файл

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. 4.11).

```
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ nasm -f elf lab5-2.asm
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ./lab5-2
Введите строку: Абронина Алиса Кирилловна
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$
```

Рис. 4.11: Исполнение файла

Разница между первым исполняемым файлом lab5-2 и вторым lab5-2 в том, что запуск первого запрашивает ввод с новой строки, а программа, которая исполняется при запуске второго, запрашивает ввод без переноса на новую строку,

потому что в этом заключается различие между подпрограммами `sprintLF` и `sprint`.

Создаю копию файла `lab5-1.asm` с именем `lab5-1-1.asm` с помощью функциональной клавиши `F5` (рис. 4.12).

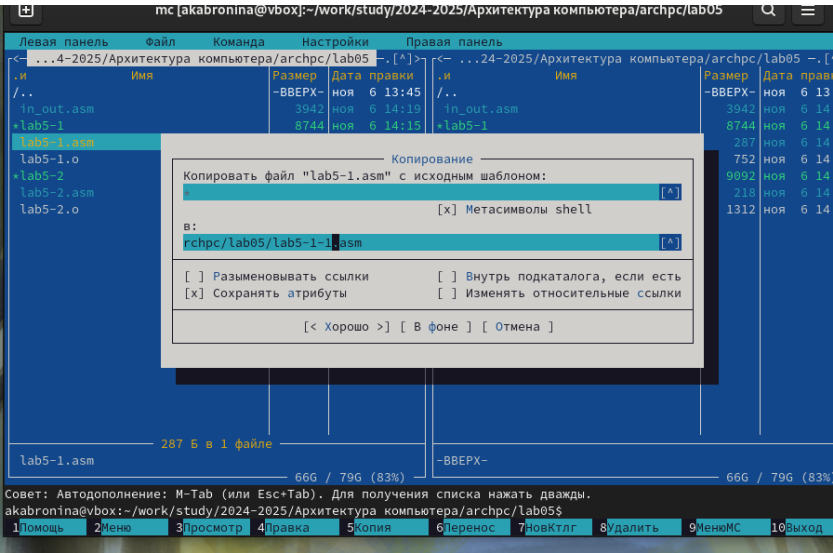


Рис. 4.12: Копирование файла

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. 4.13).

```
/home/akabronina/work/study/2024-2025/Архитектура компьютера/archpc/lab05/lab5-1-1.asm
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
```

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^D Позиция ^M-U Отмена
^X Выход ^R ЧитФайл ^N Замена ^U Вставить ^J Выровнять ^/ К строке ^M-E Повтор

Рис. 4.13: Редактирование файла

Создаю объектный файл lab5-1-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-1-1, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. 4.14).

```
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ nasm -f elf lab5-1-1.asm
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$ ./lab5-1-1
Введите строку:
Абронина Алиса Кирилловна
Абронина Алиса Кирилловна
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab05$
```

Рис. 4.14: Исполнение файла

Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5 (рис. 4.15).

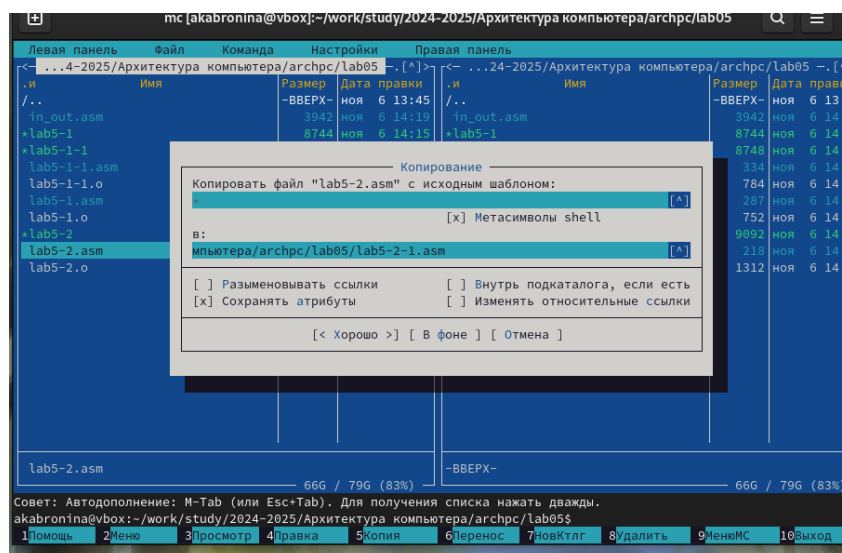


Рис. 4.15: Копирование файла

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. 4.16).

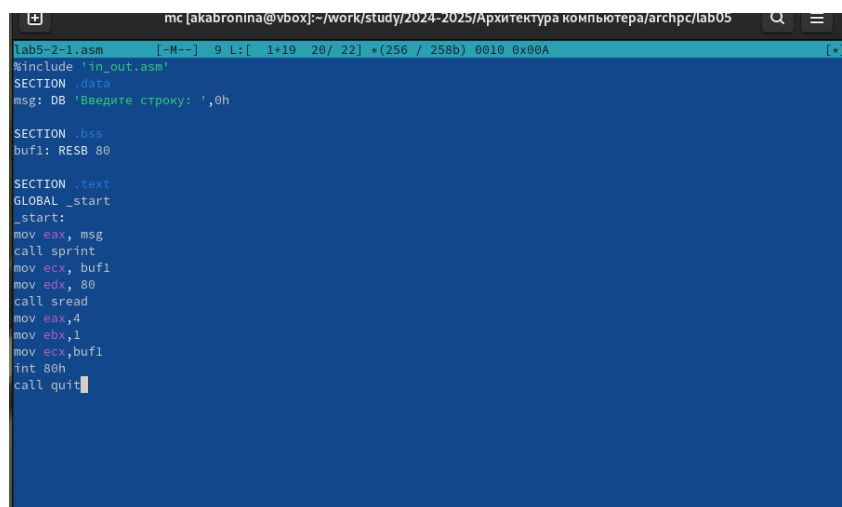


Рис. 4.16: Редактирование файла

Создаю объектный файл lab5-2-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2-1, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. ??).

```
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab0$ nasm -f elf lab5-2-1.asm
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab0$ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab0$ ./lab5-2-1
Введите строку: Абронина Алиса Кирилловна
Абронина Алиса Кирилловна
akabronina@vbox: ~/work/study/2024-2025/Архитектура компьютера/archpc/lab0$
```

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