**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**по лабораторной работе № 5**

*дисциплина: Архитектура компьютера*

Студент: Юсупова Алина Руслановна

Группа: НКАбд-06-25

**МОСКВА**

2025 г.

**Содержание**

1. Цель работы………………………………………………………………….……3
2. Задания………………………………………………………………….…………4
3. Теоретическое введение……………………………………..……..…………….5
4. Выполнение лабораторной работы……………………………………………...6

4.1 Основы работы с Midnight Commander……………………………………..…..7

4.2 Работа в NASM………………………………………………………………..….9

4.3 Подключение внешнего файла in\_out.asm……………………………………..11

4.4 Задания для самостоятельной работы……………………………………….…13

5. Выводы…………………………………………………………………………..19

Список литературы…………………………………………………………………….20

**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 размеров 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 (учетве- рённое 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 intпредназначена 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 c Midnight Commander**

Ввожу команду ‘mc’ в терминале (рис. 1), cследовательно открывается Midnight Commander (рис. 2).

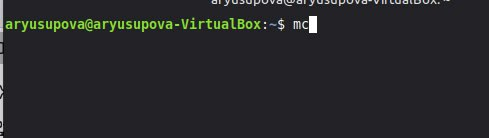


Рис. 1. Отркрытие Midnight Commander.

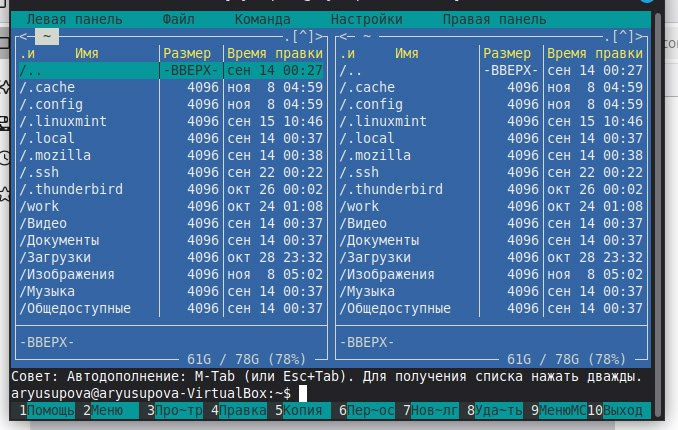


Рис. 2. Интерфейс Midnight Commander.

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

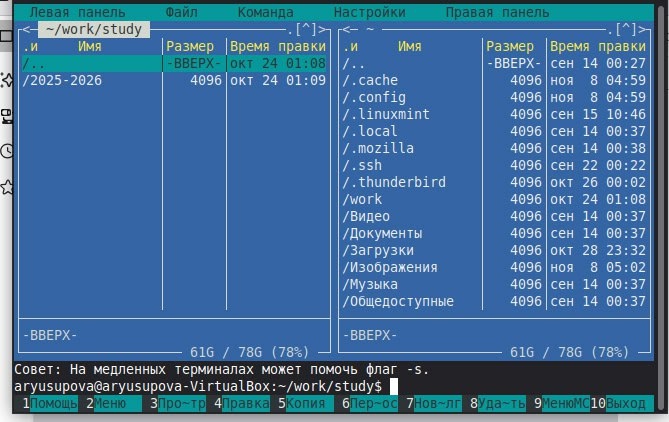


Рис. 3. Открытый каталог arch-pc.

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

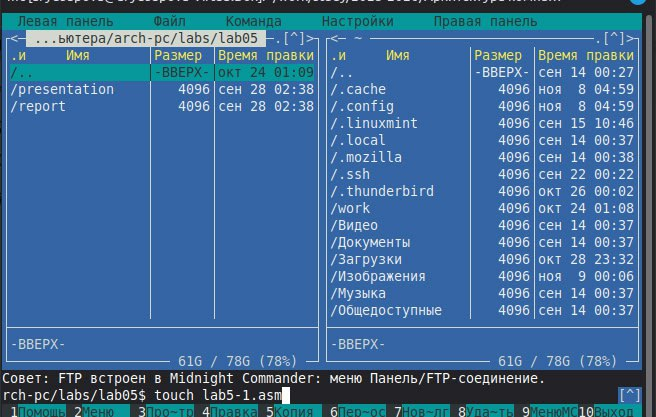


Рис. 4.Создание рабочего подкаталога

В строке ввода вводжу команду “touch” и создаю новый файл lab5-1.asm (рис. 5-6).

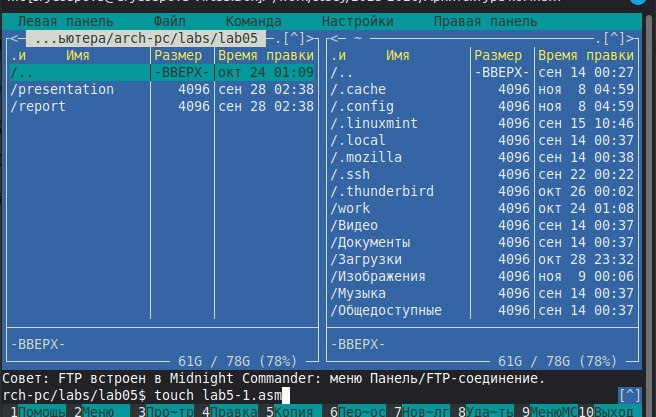


Рис. 5. Создание файла в Midnight Commander

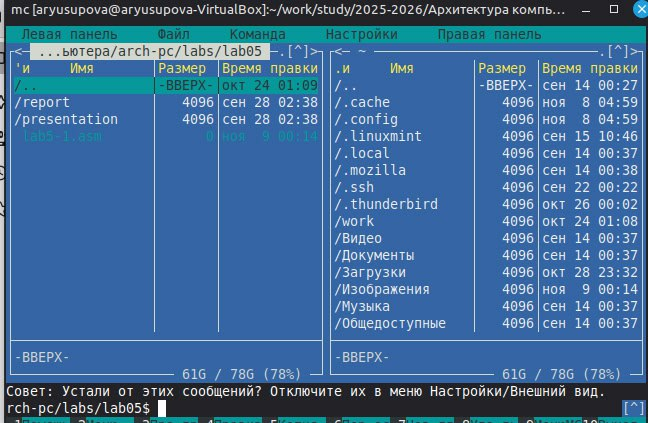


Рис.6. Создание файла в Midnight Commander.

**4.2 Работа в NASM.**

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

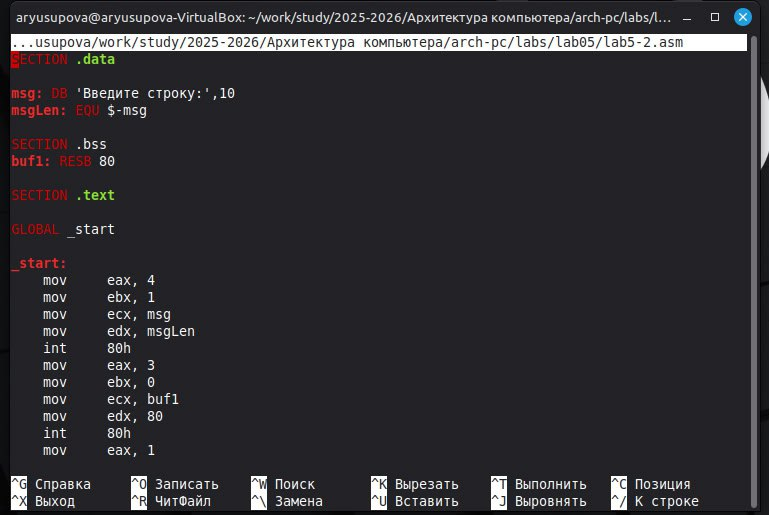


Рис. 6: Редактирование файла в Midnight Commander

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

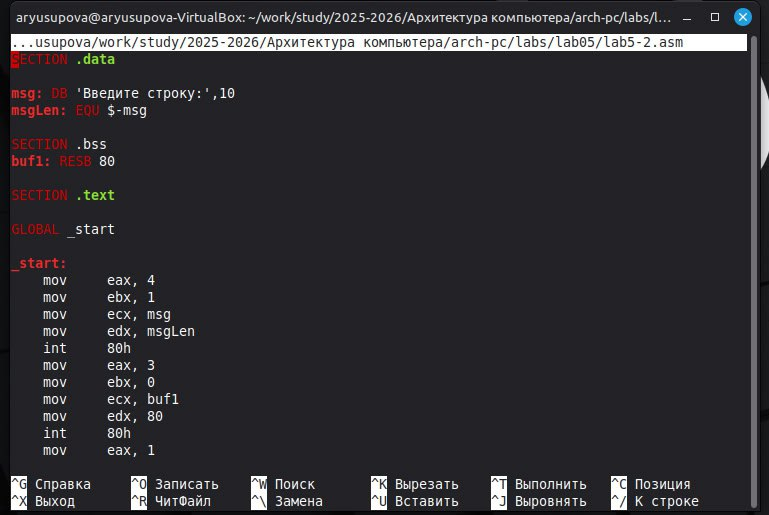


Рис. 7: Проверка сохранения сделанных изменений

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

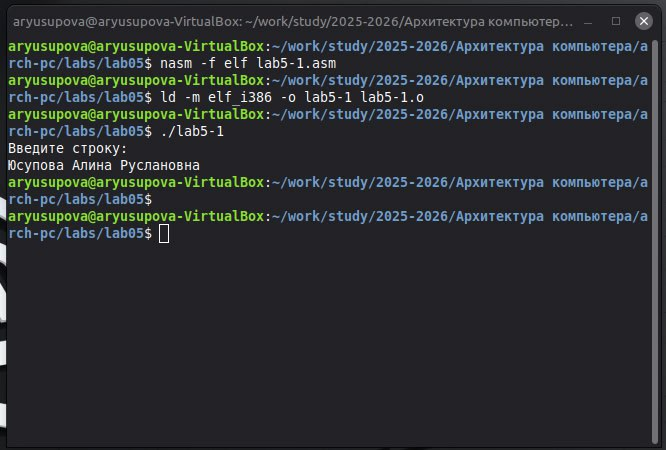


Рис. 8: Трансляция, компоновка и последующий запуск программы.

**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).

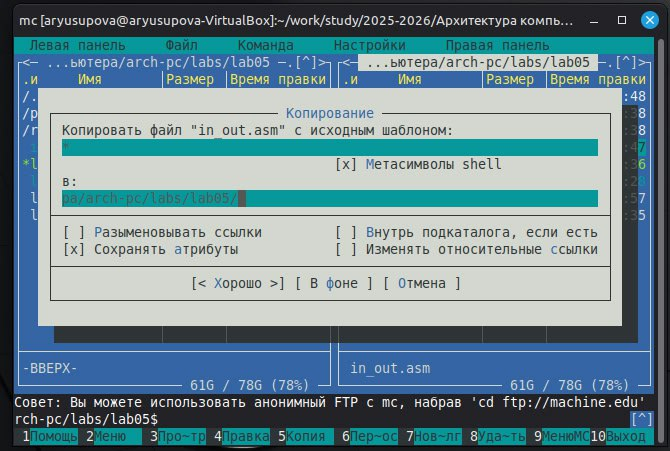


Рис. 9: Копирование файла в рабочий каталог

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

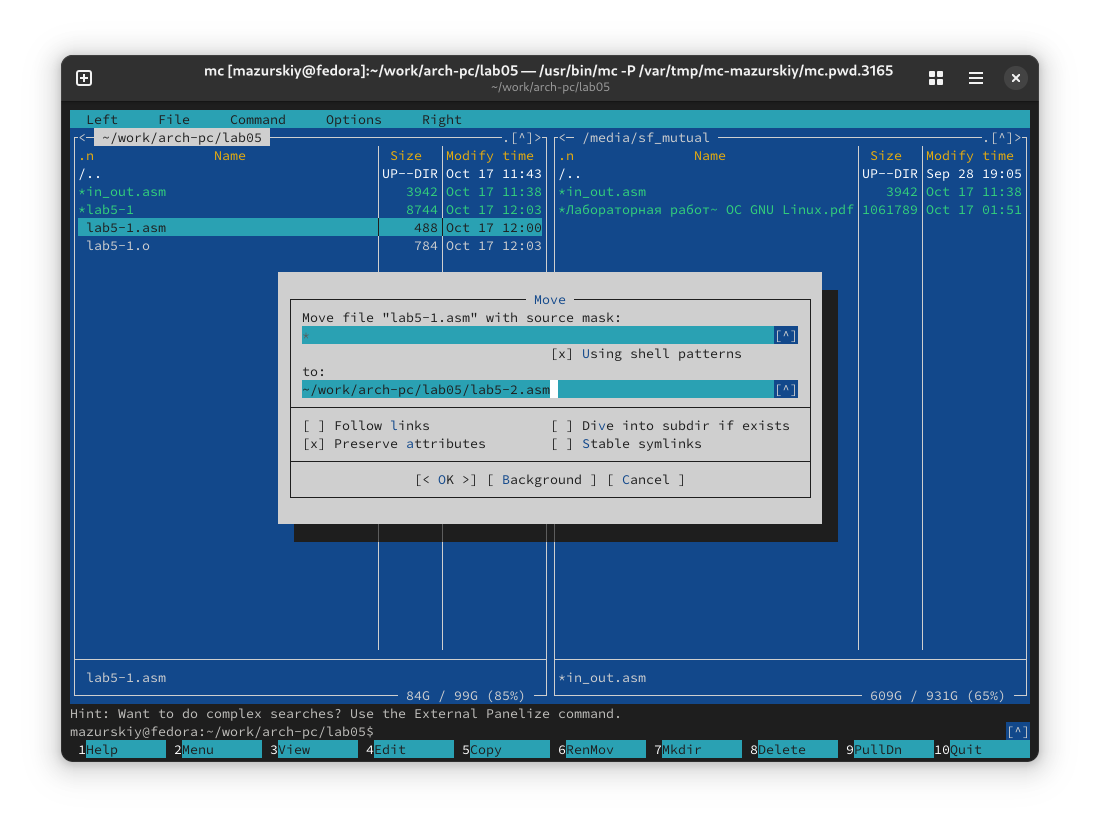


Рис. 10: Создание копии файла в Midnight Commander

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

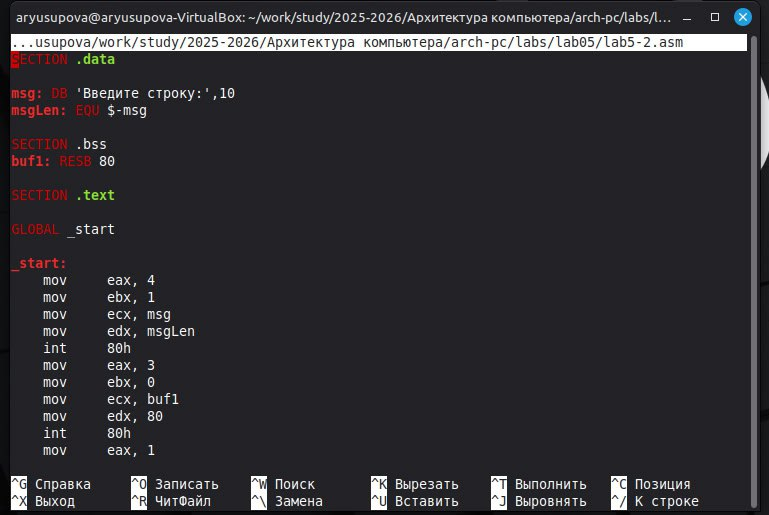


Рис. 11: Изменение программы

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

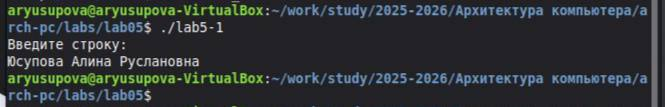


Рис. 12: Запуск измененной программы

Редактирую файл lab5-2.asm и заменяю в нем подпрограмму sprintLF на sprint. Разница подпрограмм в том, что вторая вызывает ввод на той же строке (рис. 13).

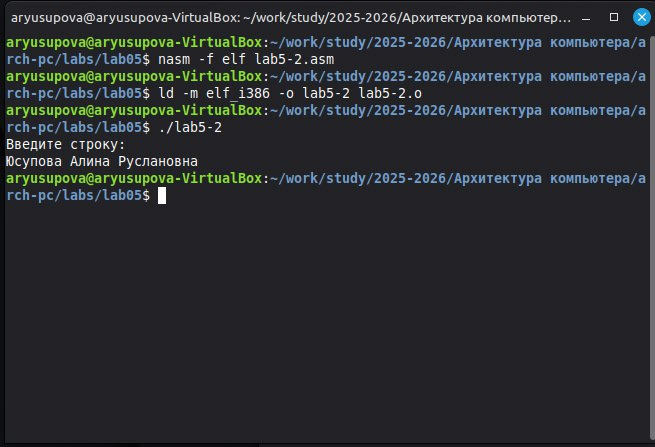


Рис. 13. Редактирование файла lab5-2.asm.

**4.4 Задание для самостоятельной работы.**

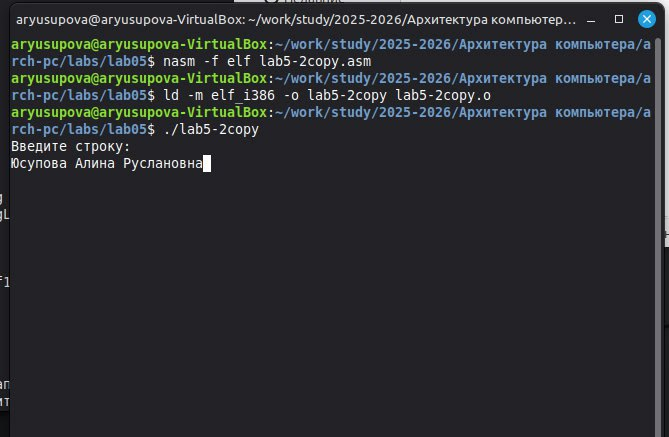


Рис. 13: Запуск изменной программы с другой подпрограммой

Создаю копию lab5-1.asm, именую её как lab5-1copy.asm, редактирую так, чтобы в конце выводилась введеная мною строка с клавиатуры(фамилия) (рис. 14).

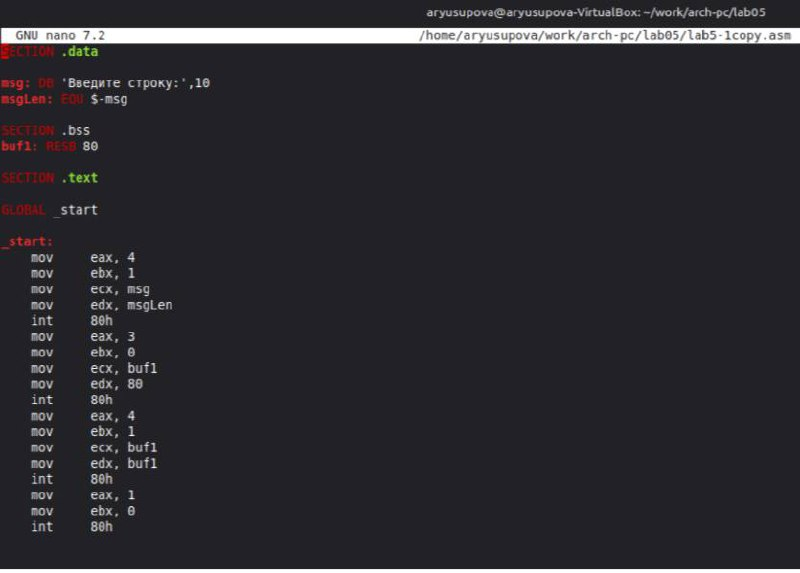


Рис. 14: Редактирование копии

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

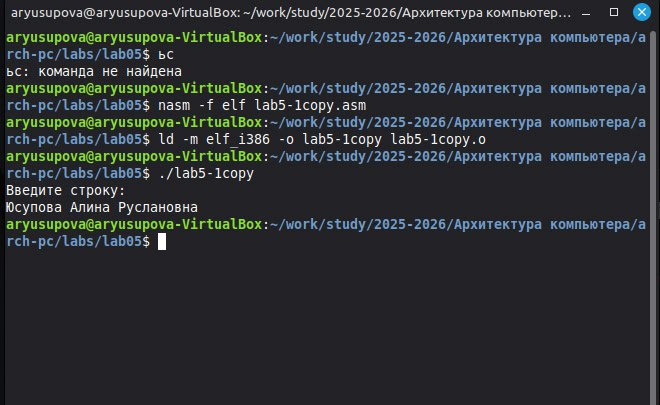


Рис. 15: Запуск своей программы

Код программы

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, именую lab5-2copy.asm(рис.16), редактирую так, чтобы в конце выводилась введеная мною строка с клавиатуры (рис. 17).

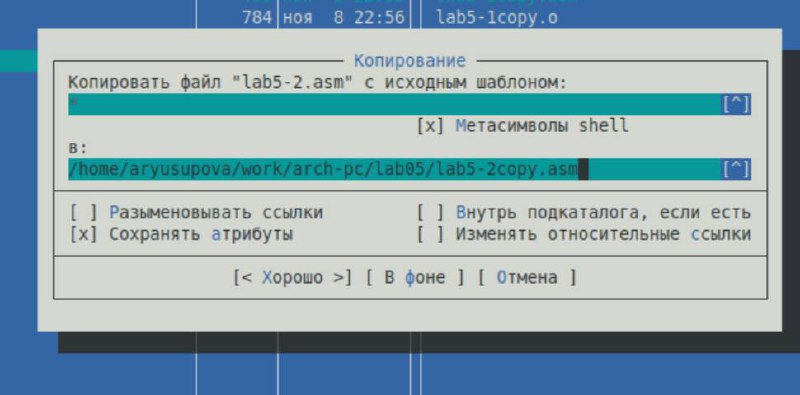


Рис. 16. Создание копии файла lab5-2.asm.



Рис.17. Редактирование копии lab5-2copy.asm.

Транслирую, компоную и запускаю свою программу lab5-2copy.asm (рис. 18).

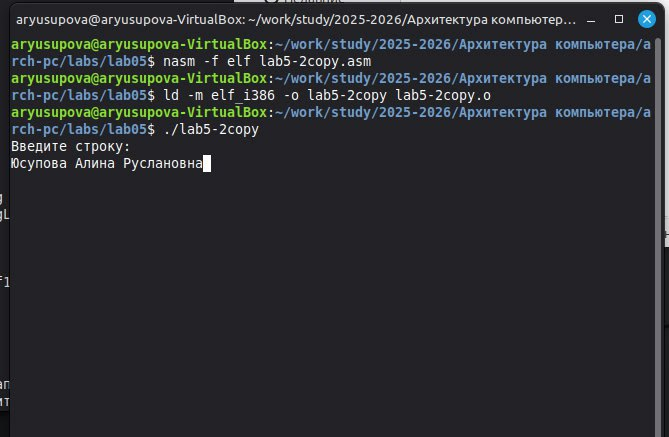


Рис. 18. Редактирование, компоновка и запуск своей программы lab5-2copy.asm.

Код программы

%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**

В ходе данной лабораторной работы были отработаны практические навыки работы с Midnight Commander и изучены ассемблерные инструкции mov и int.

**Список литературы**

1. <https://esystem.rudn.ru/mod/resource/view.php?id=1030553>

2. Расширенный ассемблер: NASM. — 2021. — URL:

<https://www.opennet.ru/docs/RUS/nasm/>.

3. GDB: The GNU Project Debugger. — URL:

<https://www.gnu.org/software/gdb/>.

4. <https://github.com/alyusupova/study_2025-2026_arh-pc.git>