**Отчет по лабораторной работе №5**

#### Дисциплина: архитектура компьютера

**Группа: НКАбд-04-23**

**Зоригоо Номун**

**Содержание**

1. [Цель работы](#_bookmark0) 4
2. [Задание](#_bookmark1) 5
3. [Теоретическое введение](#_bookmark2) 6
4. [Выполнение лабораторной работы](#_bookmark3) 8

[4.1 Основы работы с mc](#_bookmark4) . . . . . . . . . . . . . . . . . . . . . . . . . 8

* 1. [Структура программы на языке ассемблера NASM](#_bookmark10) 10
  2. [Подключение внешнего файла](#_bookmark16) 11
  3. [Выполнение заданий для самостоятельной работы](#_bookmark24) 14

1. [Выводы](#_bookmark31) 19
2. [Список литературы](#_bookmark32) 20

**Список иллюстраций**

[4.1 Открытый mc](#_bookmark5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

[4.2 Перемещение между директориями](#_bookmark6) . . . . . . . . . . . . . . . . 9 [4.3 Создание каталога](#_bookmark7) . . . . . . . . . . . . . . . . . . . . . . . . . . 9

[4.4 Перемещение между директориями](#_bookmark8) . . . . . . . . . . . . . . . . 9 [4.5 Создание файла](#_bookmark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

* 1. [Открытие файла для редактирования](#_bookmark11) 10
  2. [Редактирование файла](#_bookmark12) 10
  3. [Открытие файла для просмотра](#_bookmark13) 11
  4. [Компиляция файла и передача на обработку компоновщику](#_bookmark14) 11
  5. [Исполнение файла](#_bookmark15) 11
  6. [Скачанный файл](#_bookmark17) 12
  7. [Копирование файла](#_bookmark18) 12
  8. [Копирование файла](#_bookmark19) 13
  9. [Редактирование файла](#_bookmark20) 13
  10. [Исполнение файла](#_bookmark21) 13
  11. [Отредактированный файл](#_bookmark22) 14
  12. [Исполнение файла](#_bookmark23) 14
  13. [Копирование файла](#_bookmark25) 15
  14. [Редактирование файла](#_bookmark26) 15
  15. [Исполнение файла](#_bookmark27) 16
  16. [Копирование файла](#_bookmark28) 17
  17. [Редактирование файла](#_bookmark29) 17
  18. [Исполнение файла](#_bookmark30) 18

# Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в Midnight Commander, освоение инструкций языка ассемблера mov и int.

# Задание

* 1. Основы работы с mc
  2. Структура программы на языке ассемблера NASM
  3. Подключение внешнего файла
  4. Выполнение заданий для самостоятельной работы

# Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управ- лению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Про- грамма 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 разме- ром 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 перемен- ных 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 програм- мировании в Linux с использованием вызовов ядра sys\_calls n=80h (принято задавать в шестнадцатеричной системе счисления).

# Выполнение лабораторной работы

## Основы работы с mc

Открываю Midnight Commander, введя в терминал mc (рис. [4.1).](#_bookmark5)

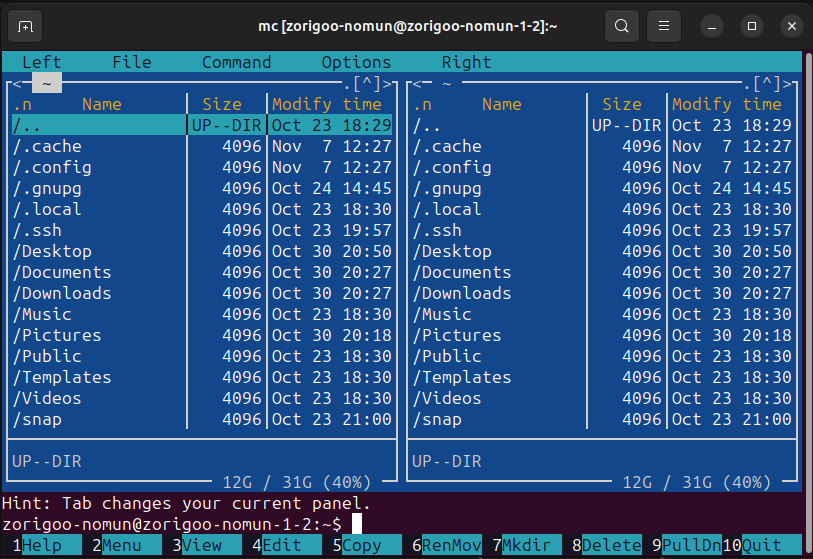


Рис. 4.1: Открытый mc

Перехожу в каталог ~/work/study/2023-2024/Архитектура Компьютера/arch-pc, используя файловый менеджер mc (рис. [4.2)](#_bookmark6)

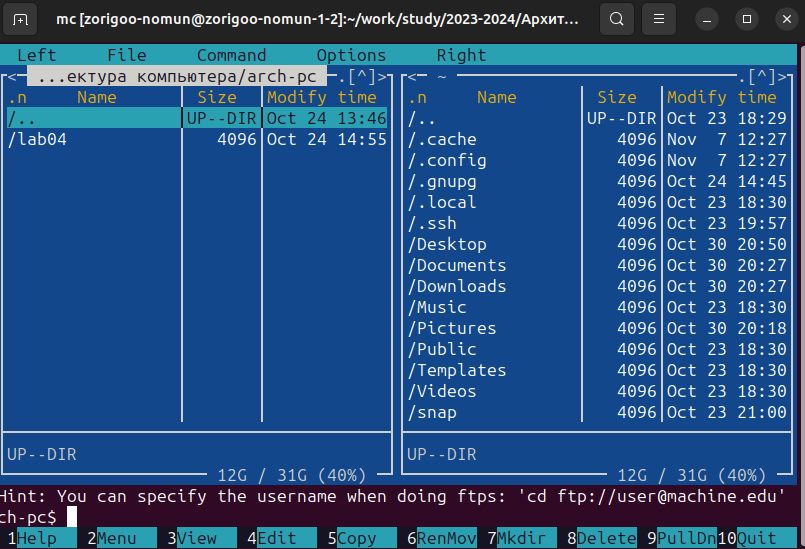


Рис. 4.2: Перемещение между директориями

С помощью функциональной клавиши F7 создаю каталог lab05 (рис. [4.3).](#_bookmark7)

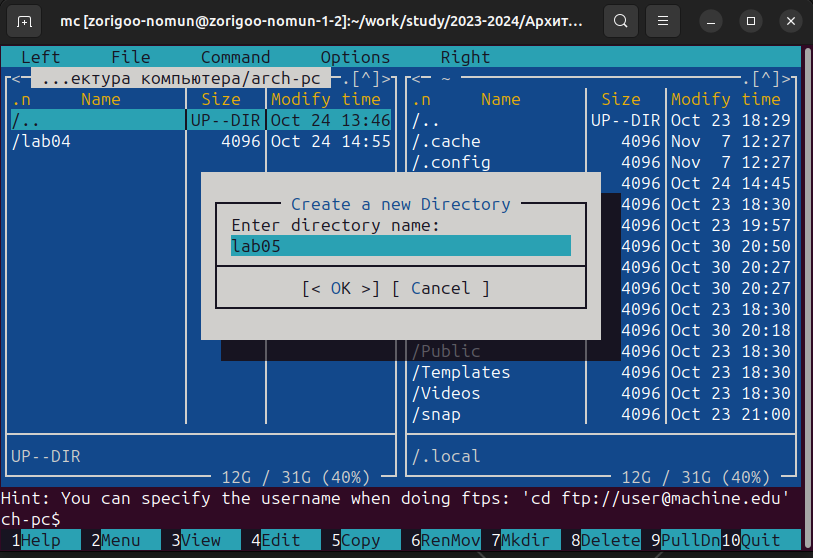


Рис. 4.3: Создание каталога

Переходу в созданный каталог (рис. [4.4).](#_bookmark8)

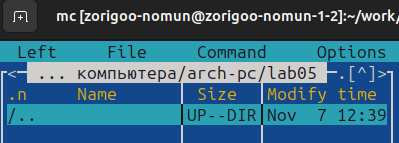


Рис. 4.4: Перемещение между директориями

В строке ввода прописываю команду touch lab5-1.asm, чтобы создать файл, в котором буду работать (рис. [4.5).](#_bookmark9)



Рис. 4.5: Создание файла

## Структура программы на языке ассемблера NASM

С помощью функциональной клавиши F4 открываю созданный файл для ре- дактирования в редакторе nano (рис. [4.6).](#_bookmark11)

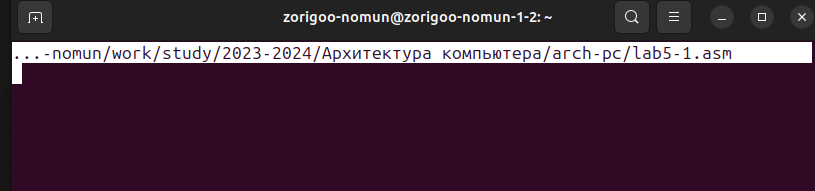


Рис. 4.6: Открытие файла для редактирования

Ввожу в файл код программы для запроса строки у пользователя (рис. [4.7).](#_bookmark12)

Далее выхожу из файла (Ctrl+X), сохраняя изменения (Y, Enter).

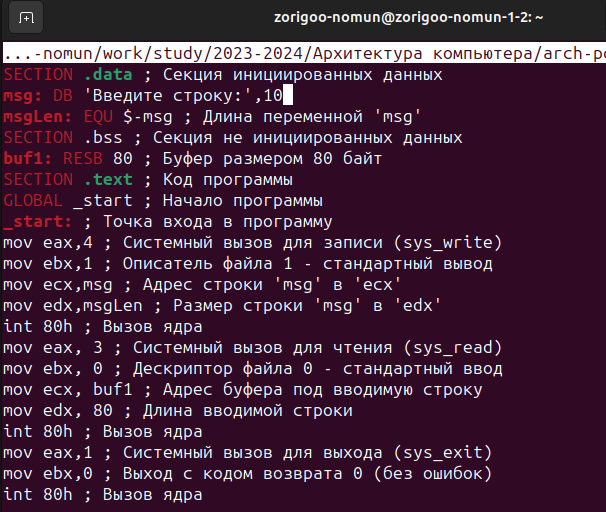


Рис. 4.7: Редактирование файла

С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы (рис. [4.8).](#_bookmark13)

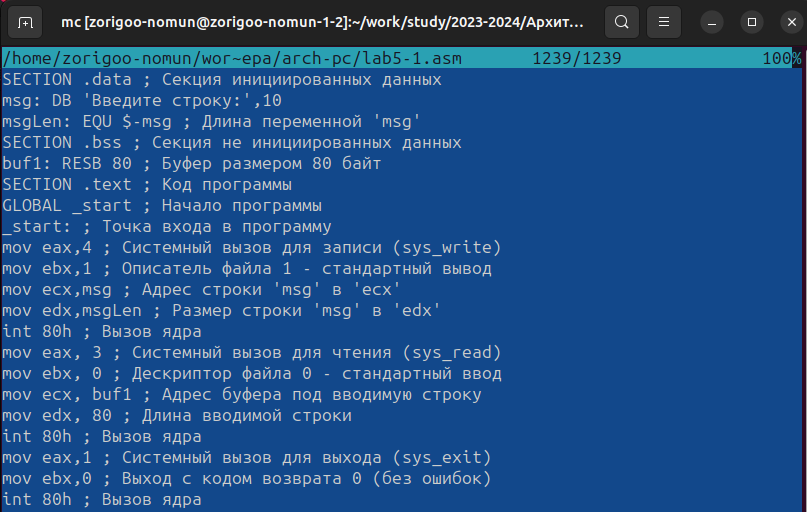
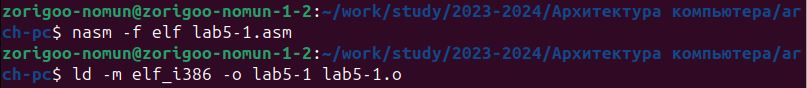


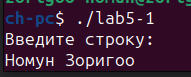
Рис. 4.8: Открытие файла для просмотра

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 (рис. [4.9).](#_bookmark14) Создался исполняемый файл lab5-1.

Рис. 4.9: Компиляция файла и передача на обработку компоновщику

Запускаю исполняемый файл. Программа выводит строку “Введите строку:”

и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. [4.10).](#_bookmark15)

 Рис. 4.10: Исполнение файла

## Подключение внешнего файла

Скачиваю файл in\_out.asm со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. [4.11).](#_bookmark17)

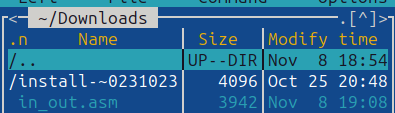


Рис. 4.11: Скачанный файл

С помощью функциональной клавиши F5 копирую файл in\_out.asm из каталога Загрузки в созданный каталог lab05 (рис. [4.12).](#_bookmark18)

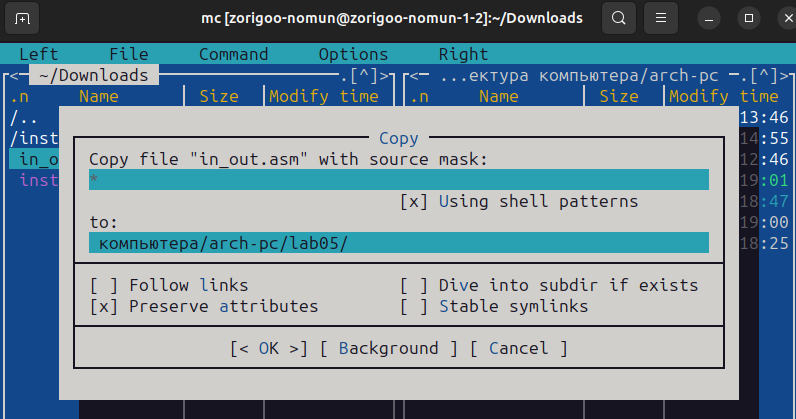


Рис. 4.12: Копирование файла

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. [4.13).](#_bookmark19)

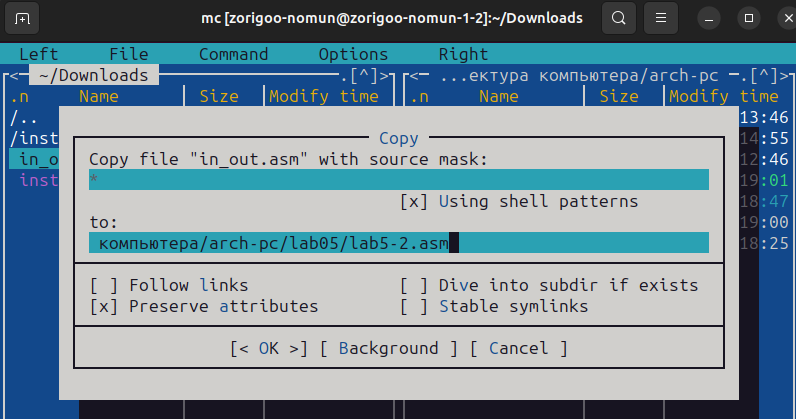


Рис. 4.13: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano (рис. [4.14),](#_bookmark20) чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.

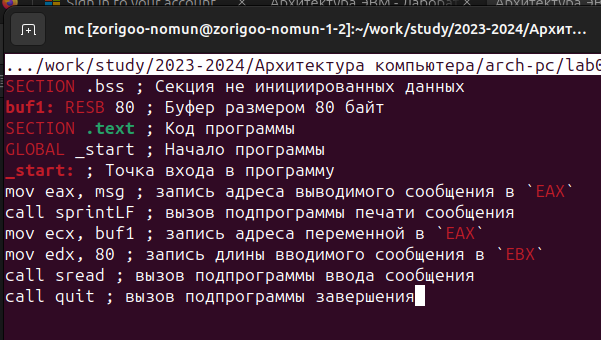


Рис. 4.14: Редактирование файла

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. [4.15).](#_bookmark21)

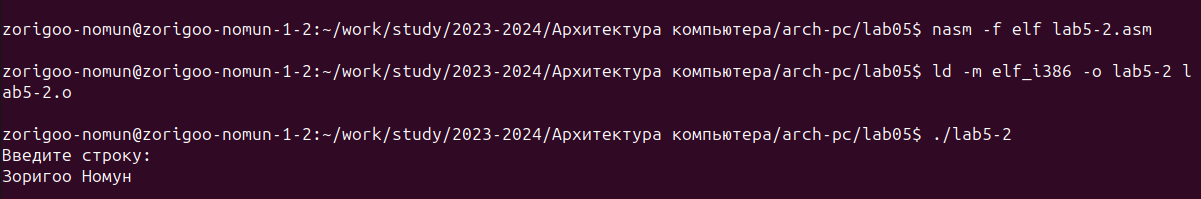


Рис. 4.15: Исполнение файла

Открываю файл lab5-2.asm для редактирования в nano функциональной кла- вишей F4. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий (рис. [4.16).](#_bookmark22)

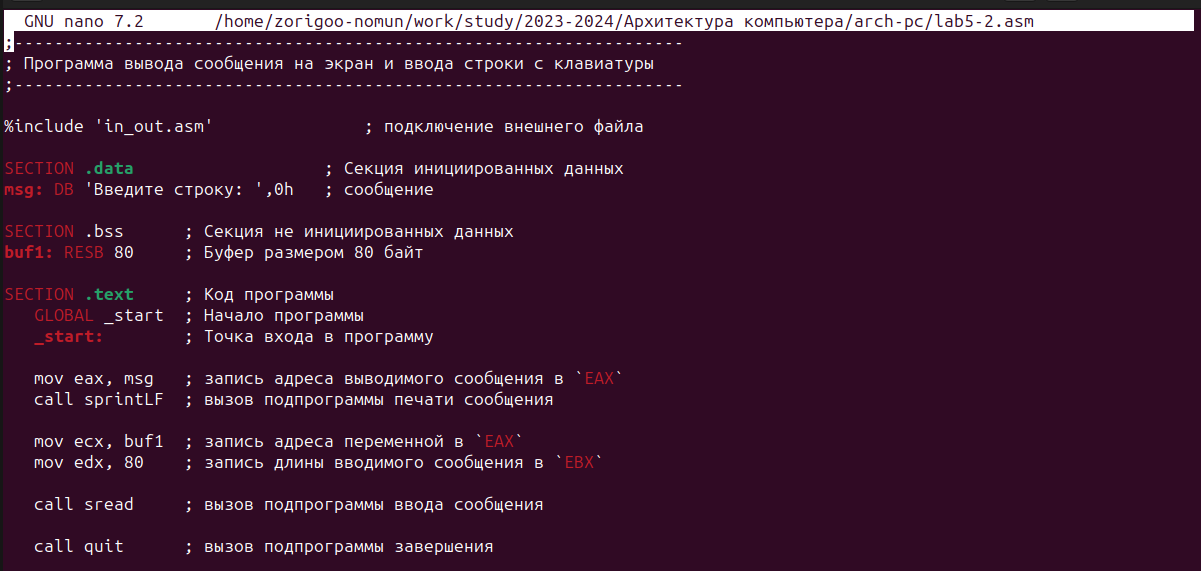
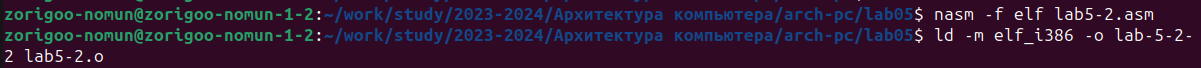


Рис. 4.16: Отредактированный файл

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. [4.17).](#_bookmark23)



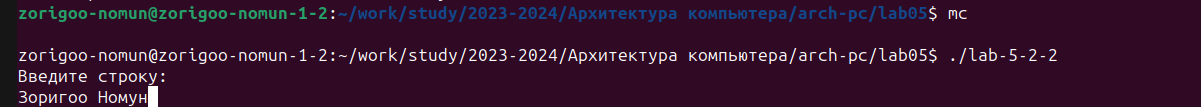


Рис. 4.17: Исполнение файла

Разница между первым исполняемым файлом lab6-2 и вторым lab6-2-2 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.

## Выполнение заданий для самостоятельной работы

* + 1. Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функци- ональной клавиши F5 (рис. [4.18).](#_bookmark25)

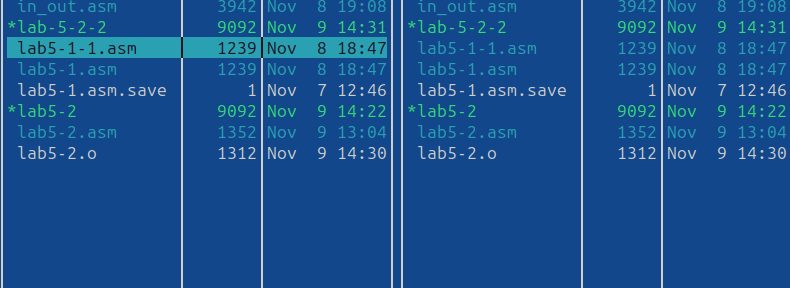


Рис. 4.18: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для ре- дактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [4.19).](#_bookmark26)

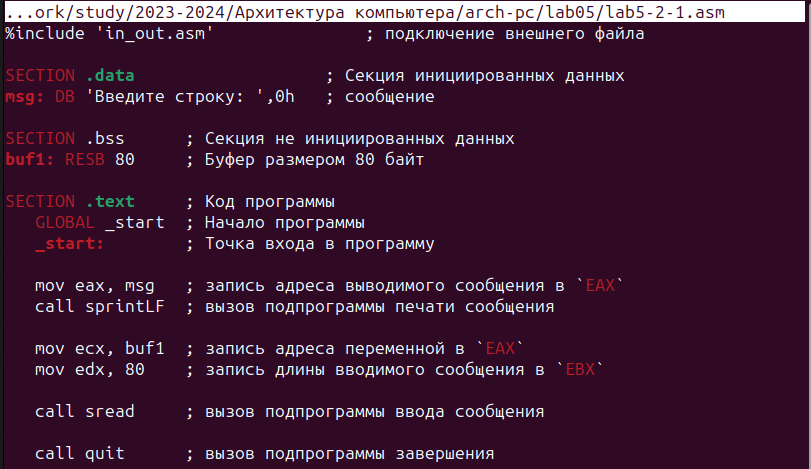


Рис. 4.19: Редактирование файла

* + 1. 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. [4.20).](#_bookmark27)

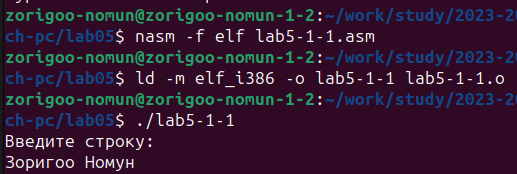


Рис. 4.20: Исполнение файла

Код программы из пункта 1:

**SECTION** .data ; Секция инициированных данных msg: DB 'Введите строку:',10

msgLen: EQU $-msg ; Длина переменной 'msg' **SECTION** .bss ; Секция не инициированных данных buf1: RESB 80 ; Буфер размером 80 байт **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 ; 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 ; 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

**int** 80h ; Вызов ядра

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

* + 1. Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функци- ональной клавиши F5 (рис. [4.21).](#_bookmark28)

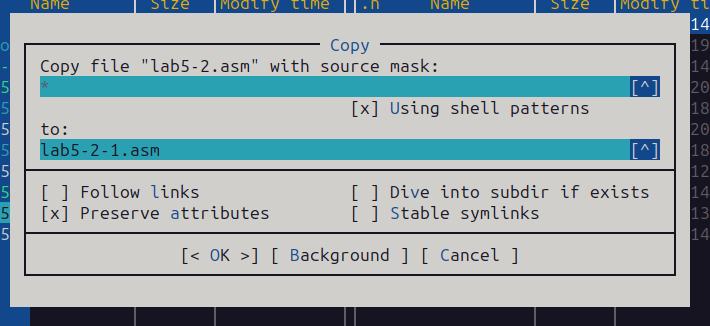


Рис. 4.21: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для ре- дактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [4.22).](#_bookmark29)

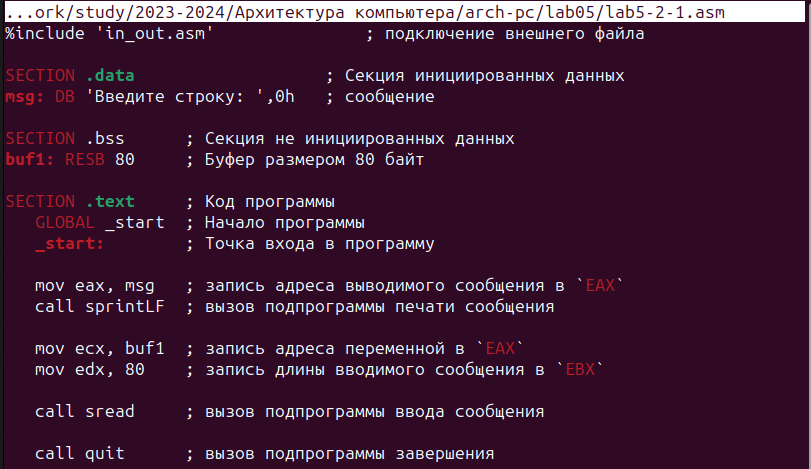


Рис. 4.22: Редактирование файла

* + 1. 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. [4.23).](#_bookmark30)

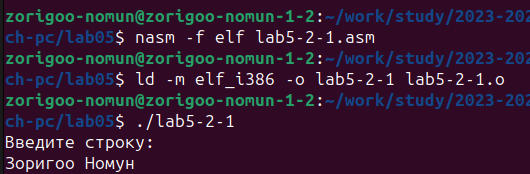


Рис. 4.23: Исполнение файла

Код программы из пункта 3:

%include 'in\_out.asm'

**SECTION** .data ; Секция инициированных данных msg: DB 'Введите строку: ',0h ; сообщение **SECTION** .bss ; Секция не инициированных 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

# Выводы

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

# Список литературы

* 1. [Лабораторная работа №5](https://esystem.rudn.ru/pluginfile.php/1584633/mod_resource/content/1/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%966.pdf)