Отчёт по лабораторной работе №5

дисциплина: Архитектура компьютеров

Терещенкова Маргарита Владимировна

Содержание

# 1 Цель работы

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 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 с mc

Открываю Midnight Commander, введя в терминал **mc**.

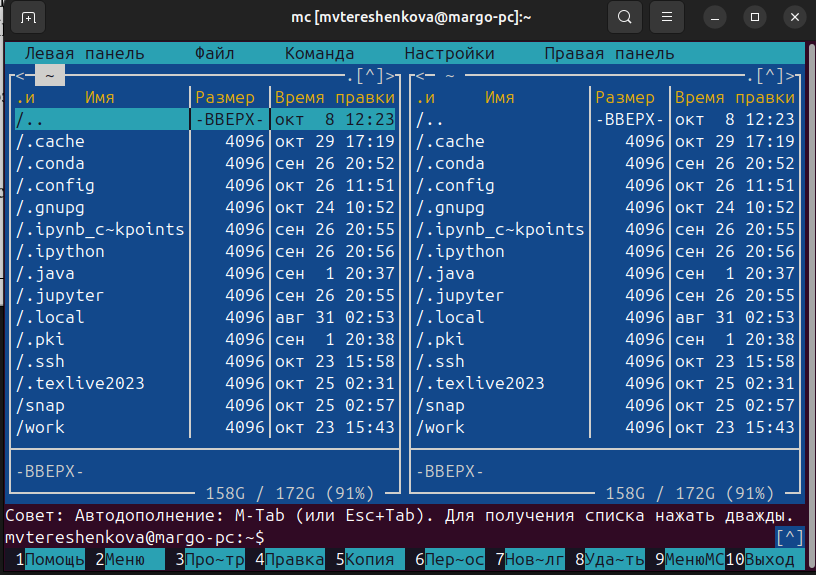


Рис. 1: *Открытый mc*

Перехожу в каталог ~/work/study/2024-2025/Архитектура Компьютера/arch-pc, используя файловый менеджер mc.

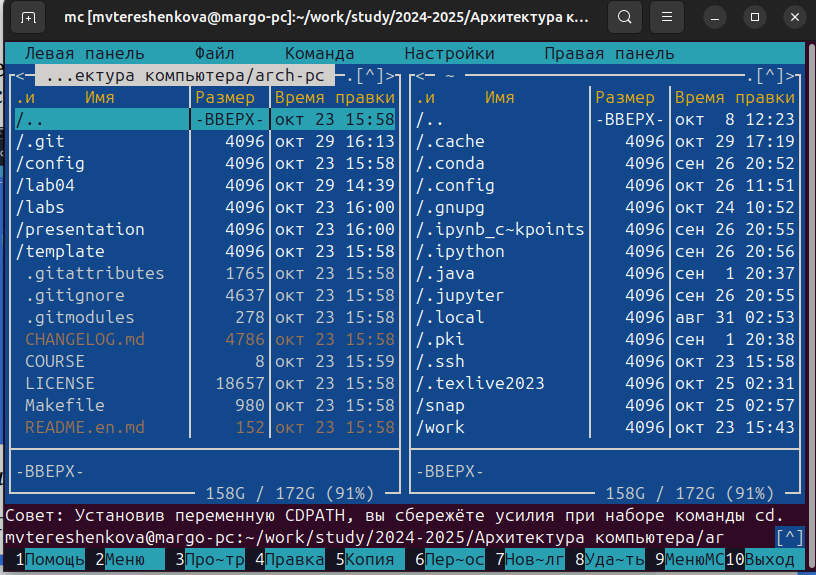


Рис. 2: *Перемещение между директориями*

С помощью функциональной клавиши **F7** создаю каталог lab05.

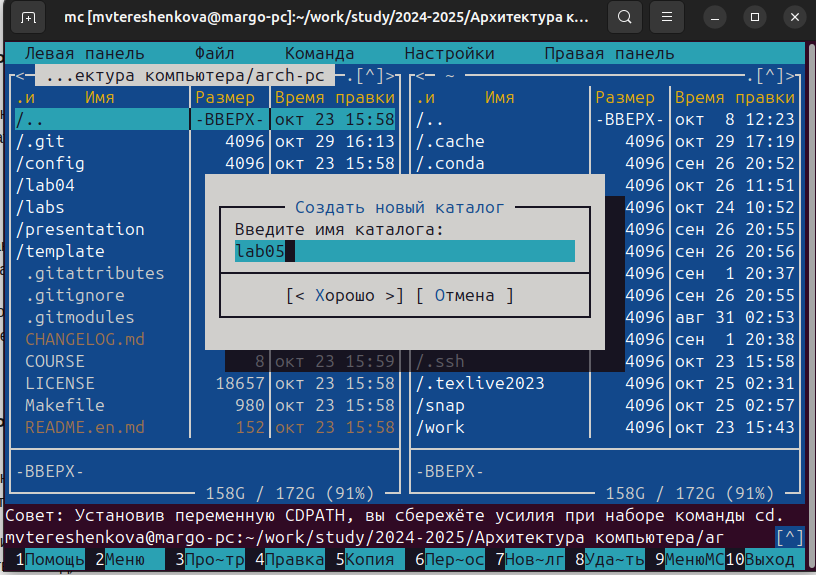


Рис. 3: *Создание каталога*

Перехожу в созданный каталог.

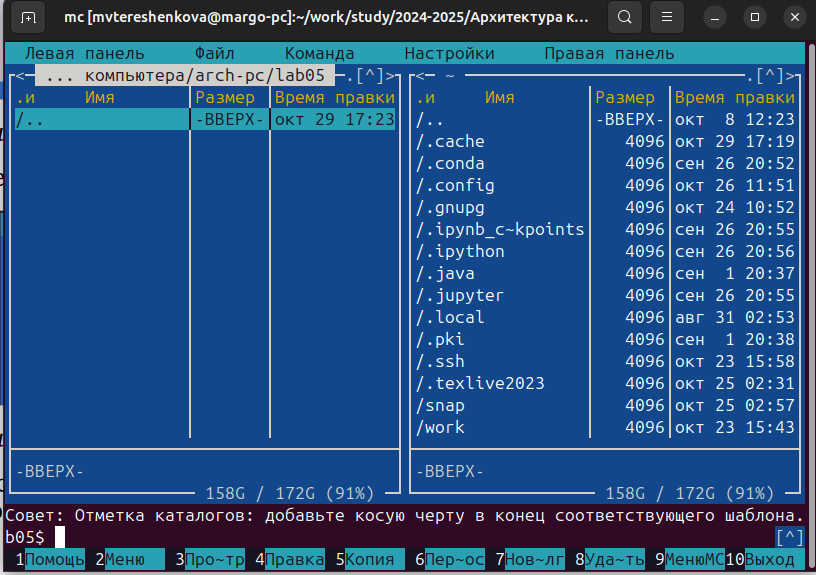


Рис. 4: *Перемещение между директориями*

В строке ввода прописываю команду **touch lab5-1.asm**, чтобы создать файл, в котором буду работать.

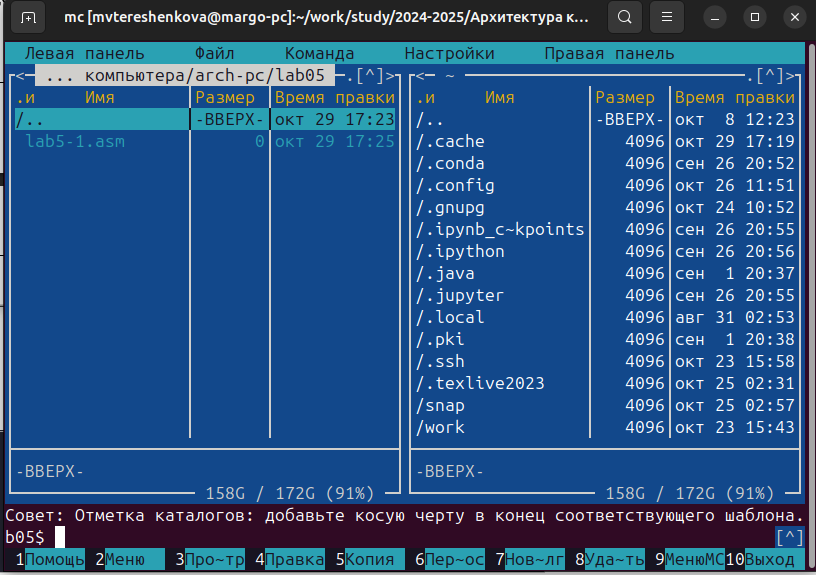


Рис. 5: *Созданный файл*

## 4.2 Структура программы на языке ассемблера NASM

С помощью функциональной клавиши **F4** открываю созданный файл для редактирования в редакторе nano.

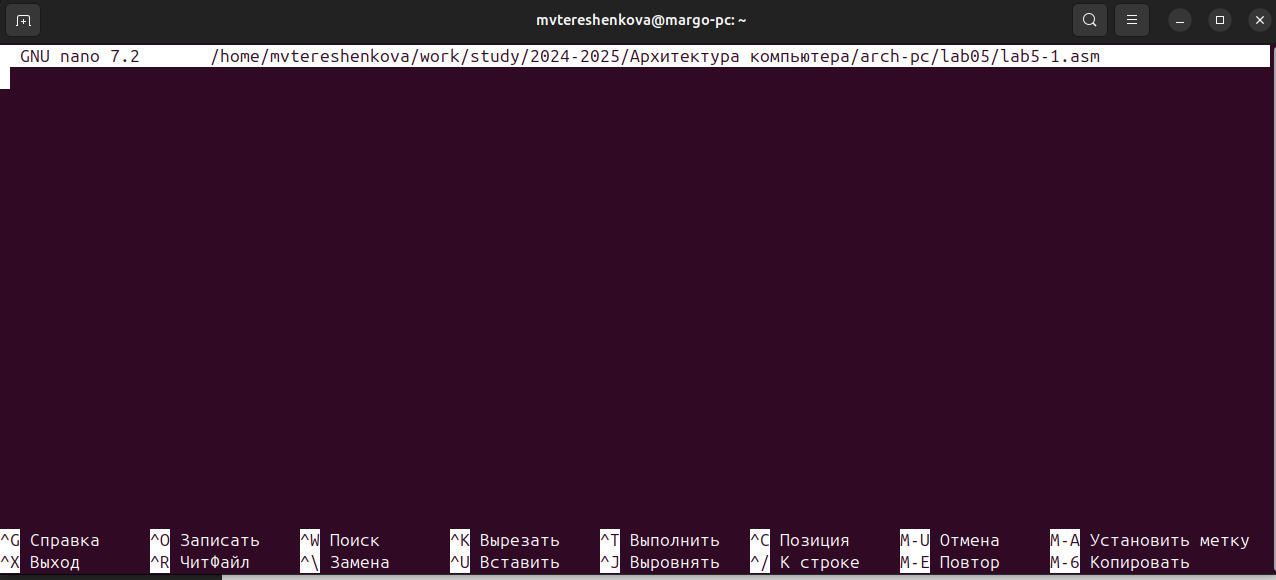


Рис. 6: *Открытие файла для редактирования*

Ввожу в файл код программы для запроса строки у пользователя. Далее выхожу из файла **Ctrl+X**, сохраняя изменения **Y, Enter**.

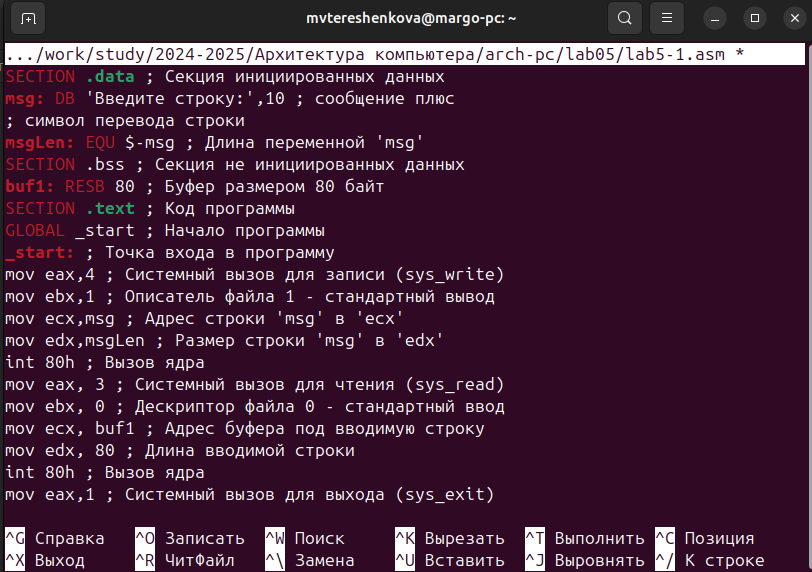


Рис. 7: *Редактирование файла*

С помощью функциональной клавиши **F3** открываю файл для просмотра для проверки.

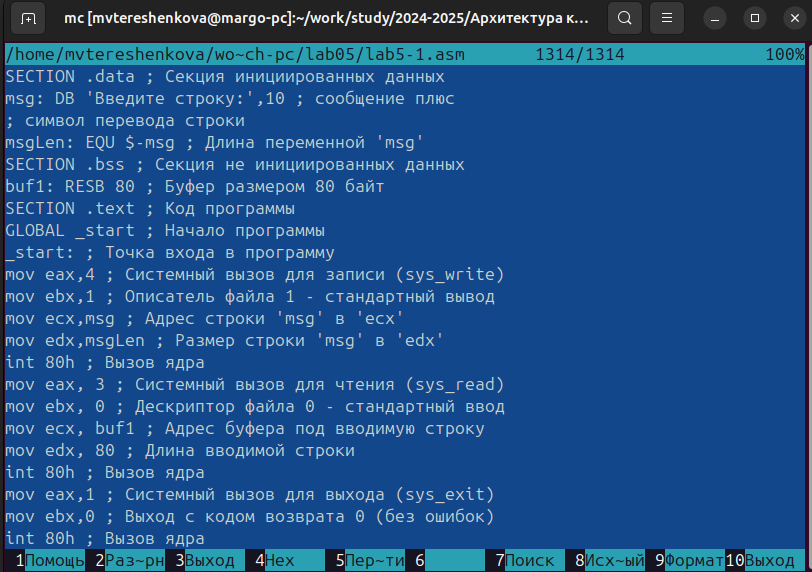


Рис. 8: *Открытие файла для просмотра*

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**. Создался исполняемый файл lab5-1.Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои имя и фамилию, на этом программа заканчивает свою работу.

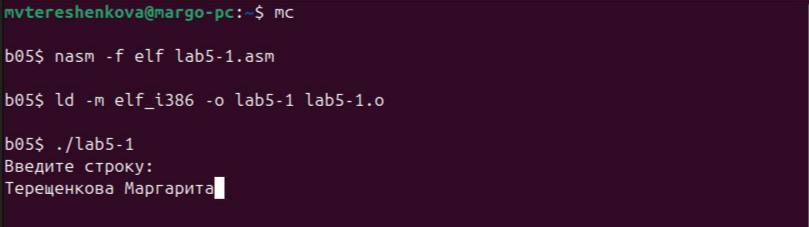


Рис. 9: *Компиляция файла, передача на обработку компоновщику и исполнение файла*

## 4.3 Подключение внешнего файла

Скачиваю файл in\_out.asm со страницы курса в ТУИС. Он сохранился в каталог “Загрузки”. С помощью функциональной клавиши **F5** копирую файл in\_out.asm из каталога Загрузки в созданный каталог lab05.

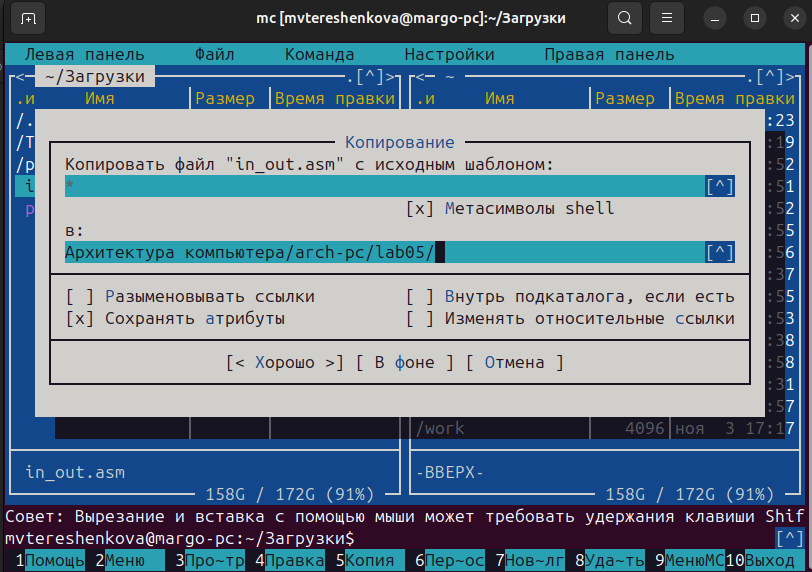


Рис. 10: *Копирование файла*

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.

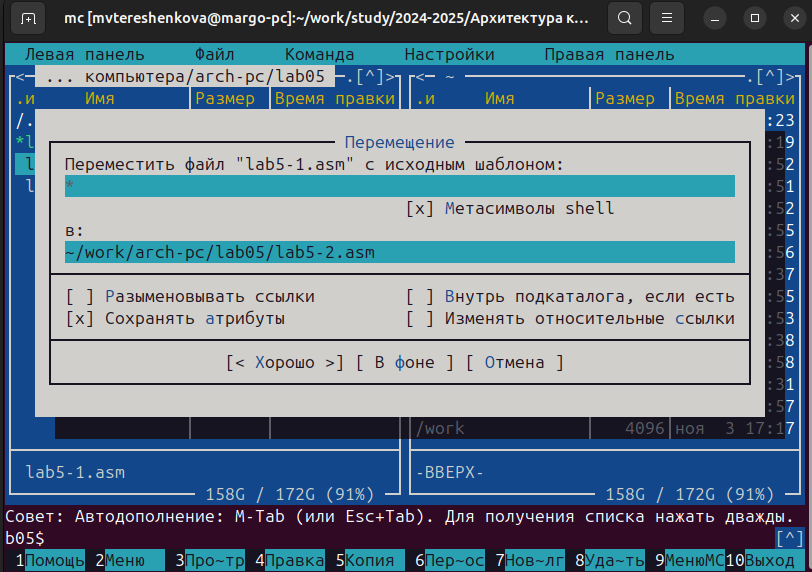


Рис. 11: *Копирование файла*

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.

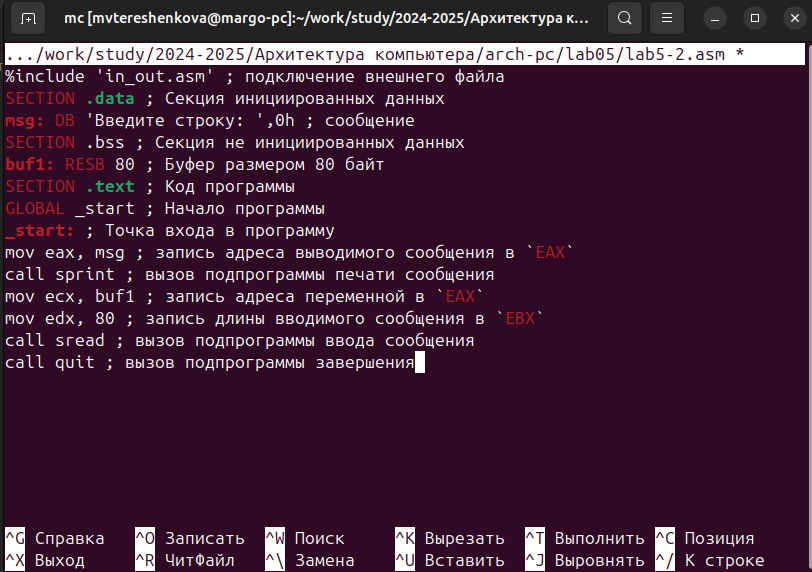


Рис. 12: *Редактирование файла*

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.

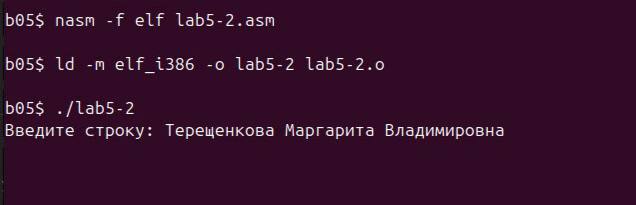


Рис. 13: *Исполнение файла*

Открываю файл lab5-2.asm для редактирования в nano функциональной клавишей **F4**. Изменяю в нем подпрограмму sprint на sprintLF. Сохраняю изменения и открываю файл для просмотра клавишей **F3**, чтобы проверить сохранение действий.

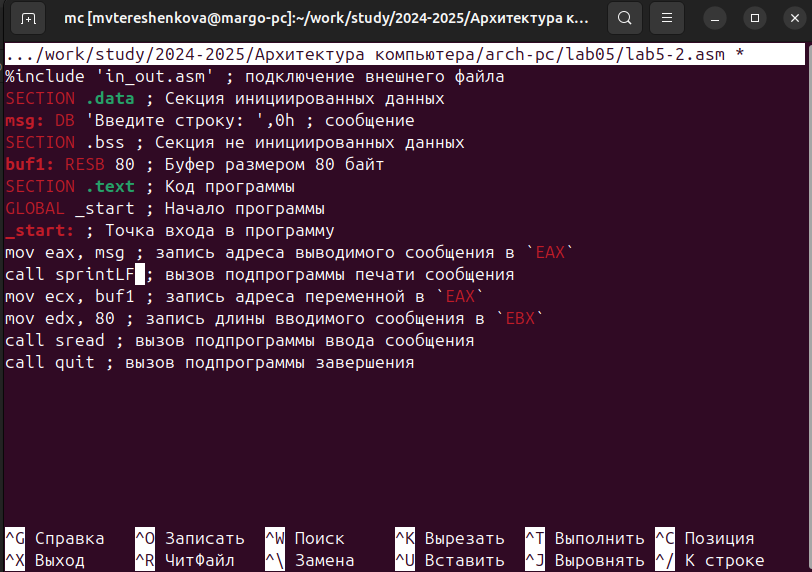


Рис. 14: *Редактирование файла*

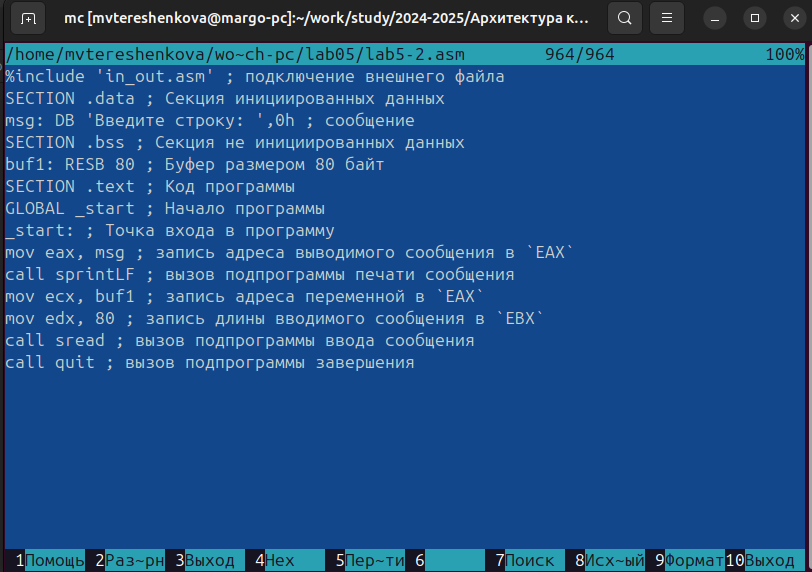


Рис. 15: *Проверка редактирования файла*

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл.

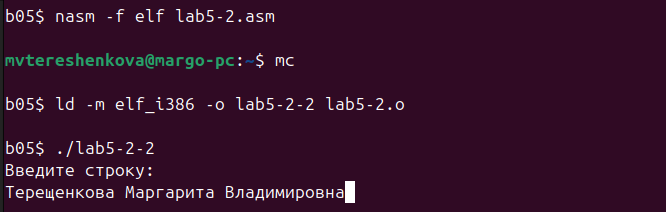


Рис. 16: *Исполнение файла*

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

## 4.4 Выполнение самостоятельной работы

1. Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши **F5**.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.

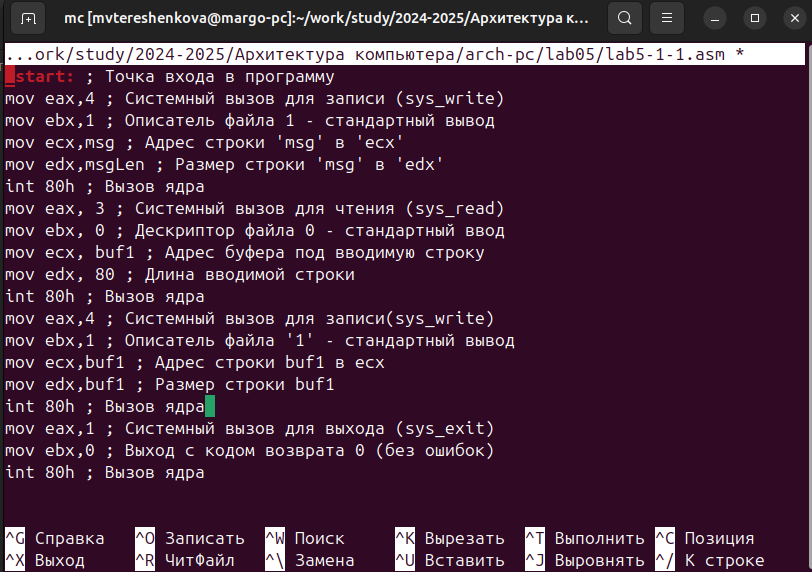


Рис. 17: *Редактирование файла*

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.

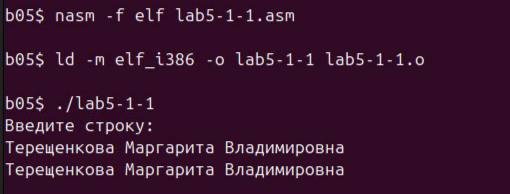


Рис. 18: *Исполнение файла*

1. Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши **F5**.

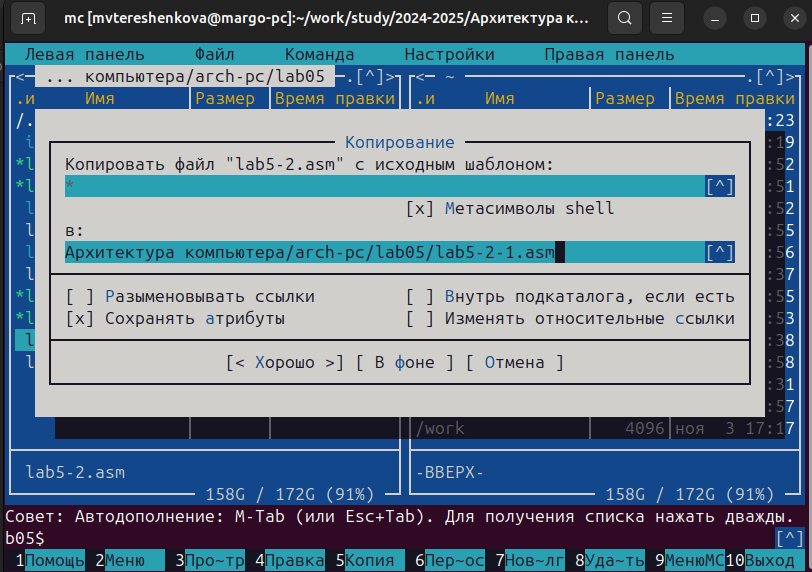


Рис. 19: *Копирование файла*

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.

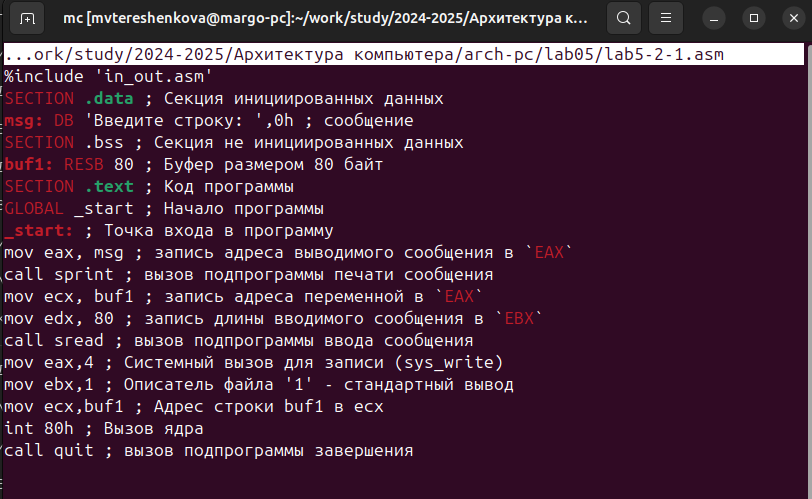


Рис. 20: *Редактирование файла*

С помощью функциональной клавиши **F3** открываю файл для просмотра для проверки.

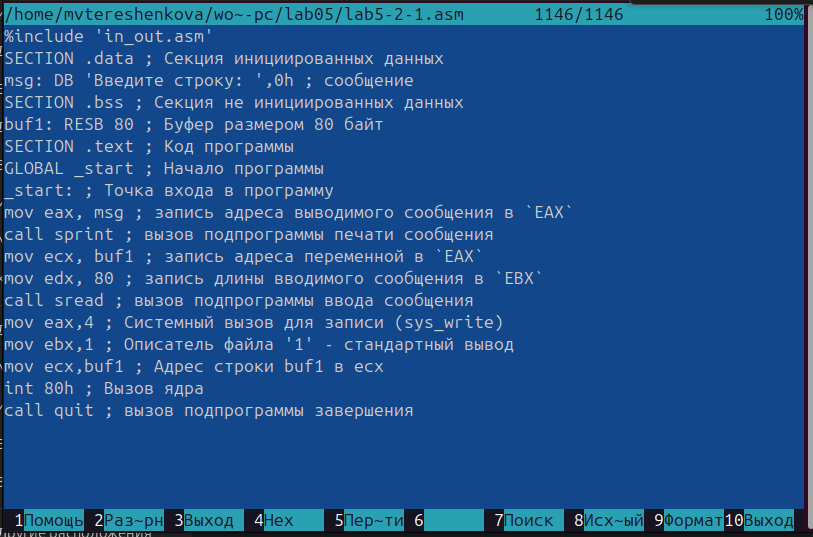


Рис. 21: *Проверка редактирования файла*

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.

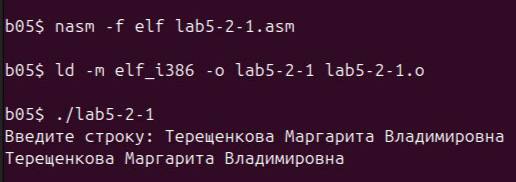


Рис. 22: *Исполнение файла*

# 5 Выводы

Благодаря данной лабораторной работе приобрела практические навыки работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

# Список литературы

Архитектура ЭВМ