ОТЧЕТ по лабораторной работе № 5

дисциплина: Архитектура компьютера

Студент: Идрисов Д.А

Содержание

# 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.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.

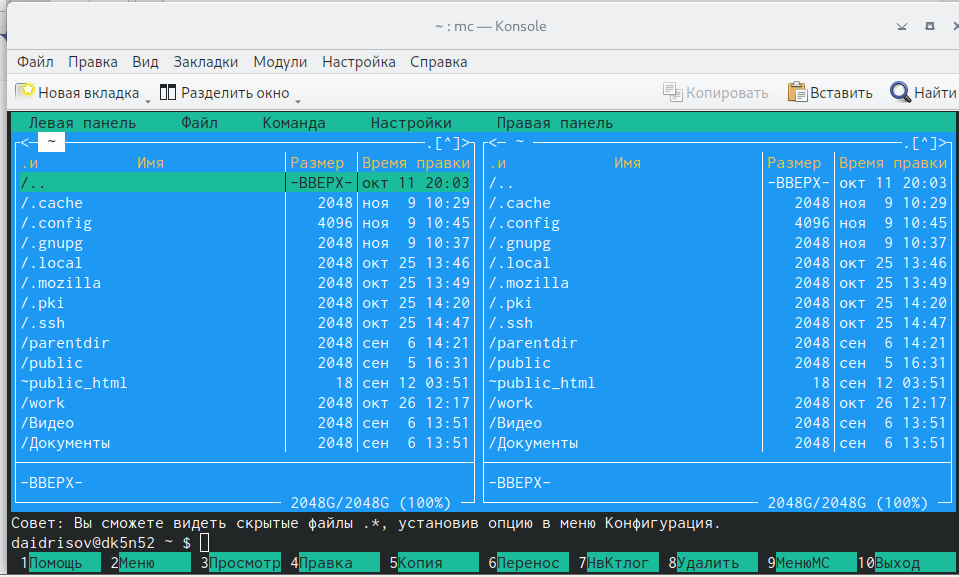
int n

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

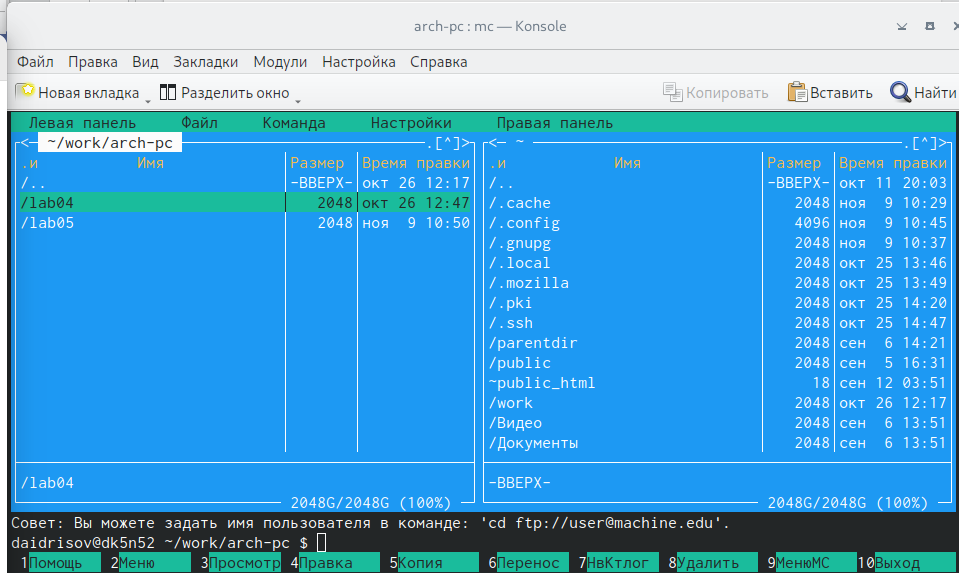
## 4.1 1 Основы работы с mc.

Я запускаю Midnight Commander, используя команду “mc”. (рис. ??).



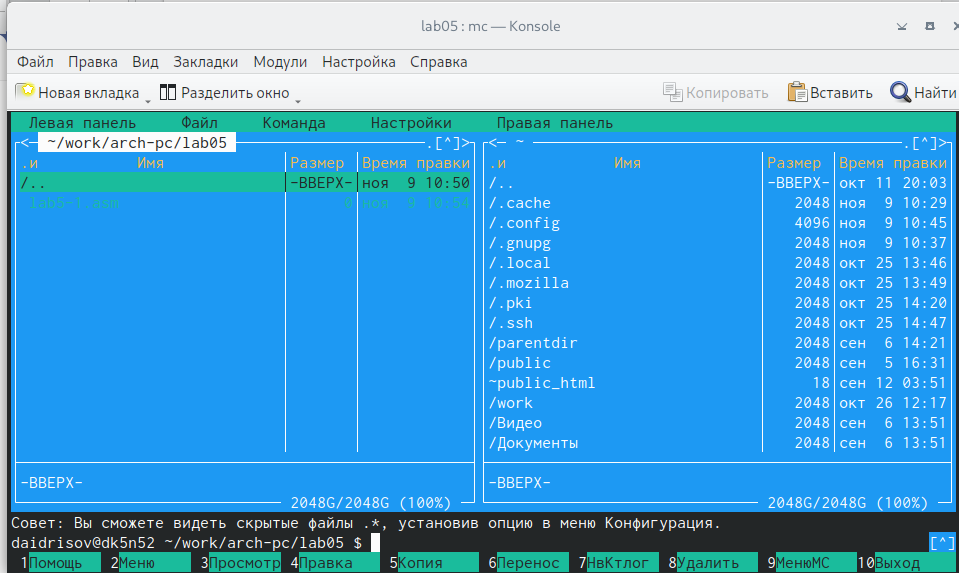
Открытый Midnight Commander

Перехожу в каталог ~/work/arch-pc (рис. ??).



Выполню переход

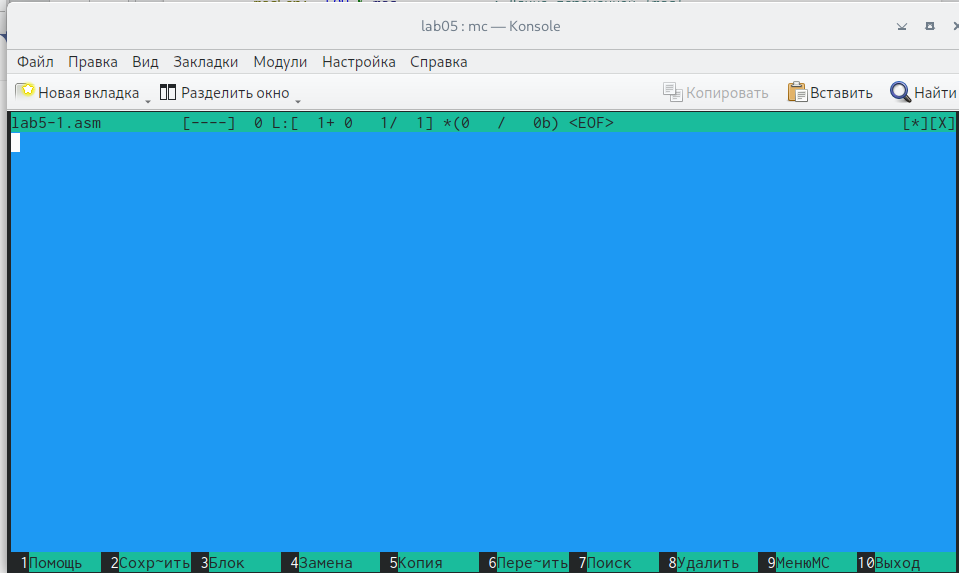
Создаю папку с именем “lab05” с помощью функциональной клавиши F7, и после этого выполняю команду “touch lab5-1.asm” (рис. ??).



Создание каталога

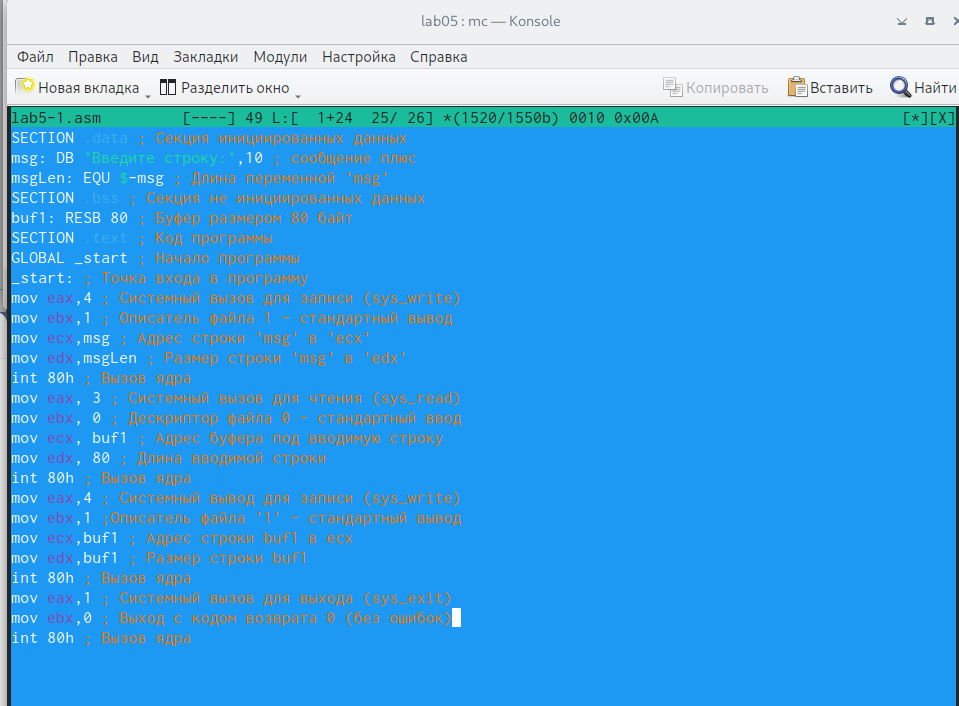
## 4.2 2 Структура программы на языке ассемблера NASM.

Используя функциональную клавишу F4, открою файл в текстовом редакторе nano для редактирования. (рис. ??).



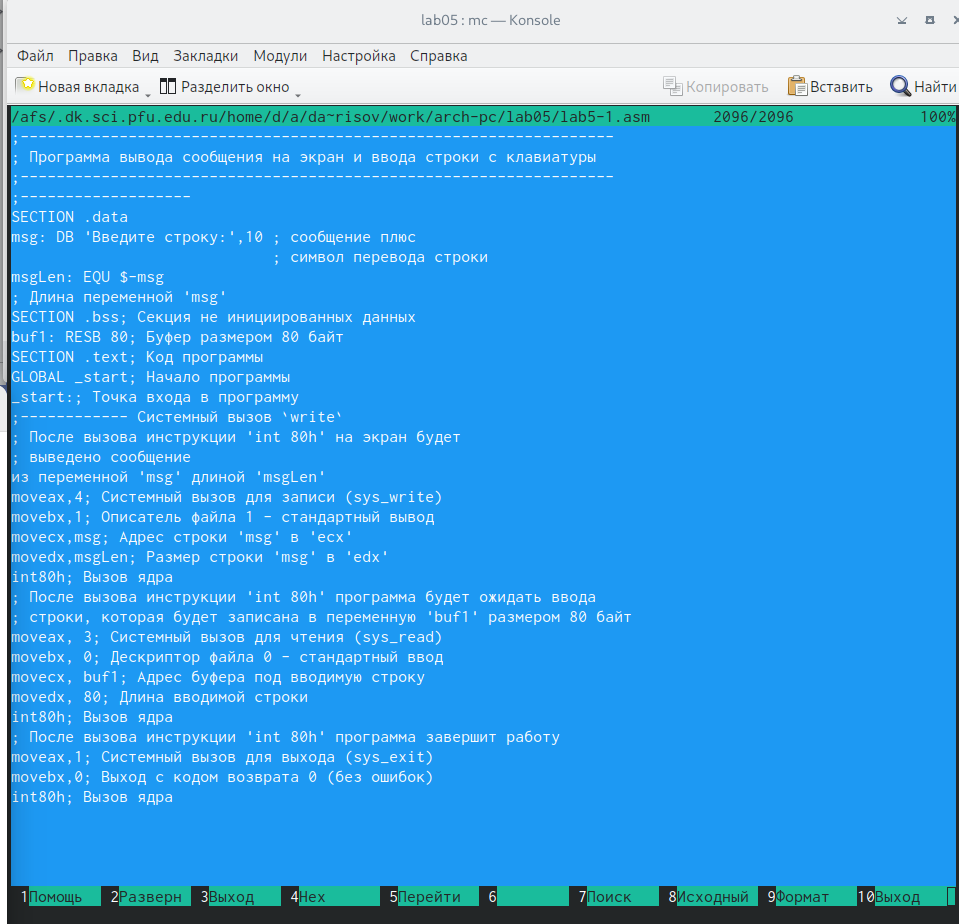
Открытие файла для редактирования

Добавлю в файл программный код, который будет запрашивать строку у пользователя. (рис. ??).



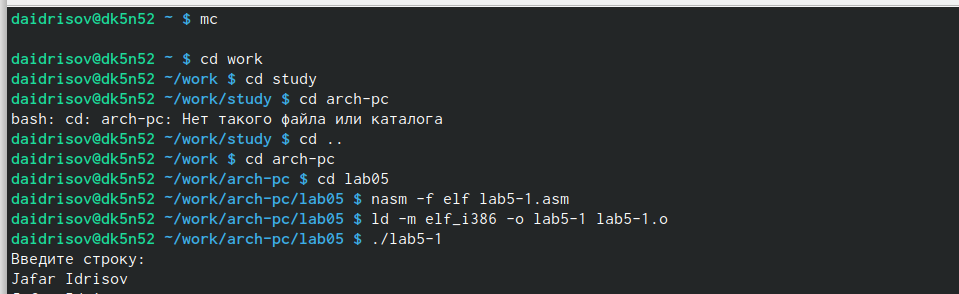
Редактирование файла

Используя функциональную клавишу F3, мы можем открыть файл для просмотра и проверить его содержимое, чтобы убедиться в наличии текста программы. (рис. ??).



Просмотр текста программы

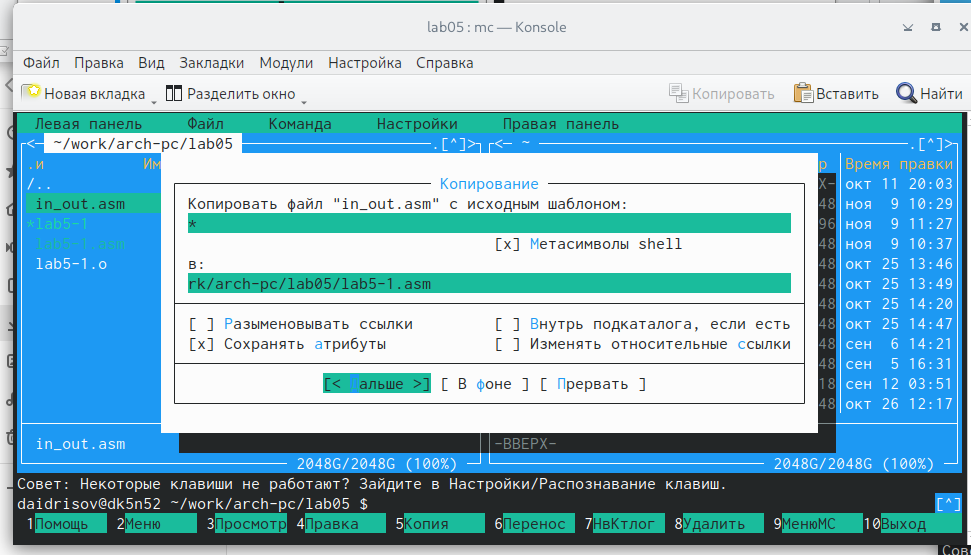
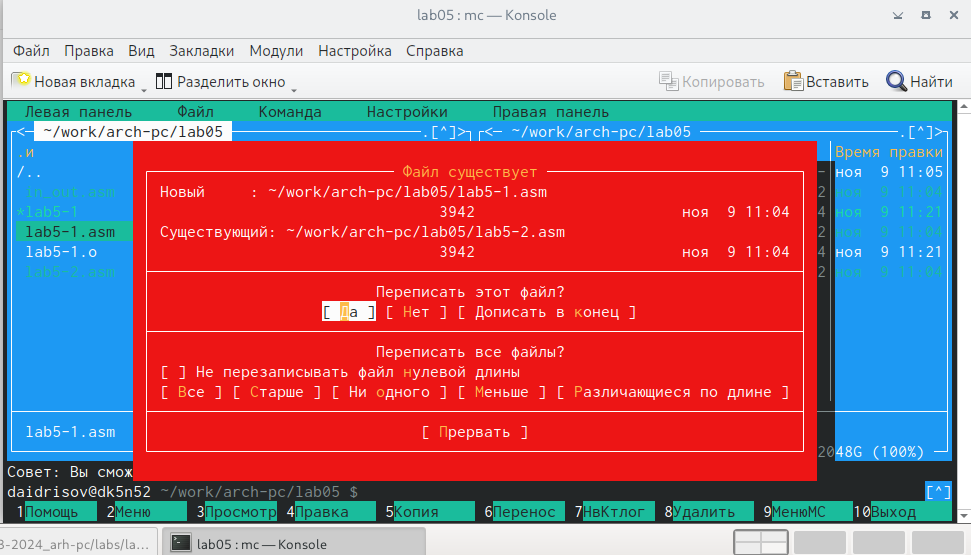
Я проведу процесс компиляции моей программы, начиная с трансляции текста программы в объектный файл с использованием команды “nasm -f elf lab5-1.asm”. Затем выполню компоновку файла с помощью команды “ld -m elf\_i386 -o lab5-1 lab5-1.o”. Наконец, запущу программу с помощью команды “./lab5-1”. (рис. ??).



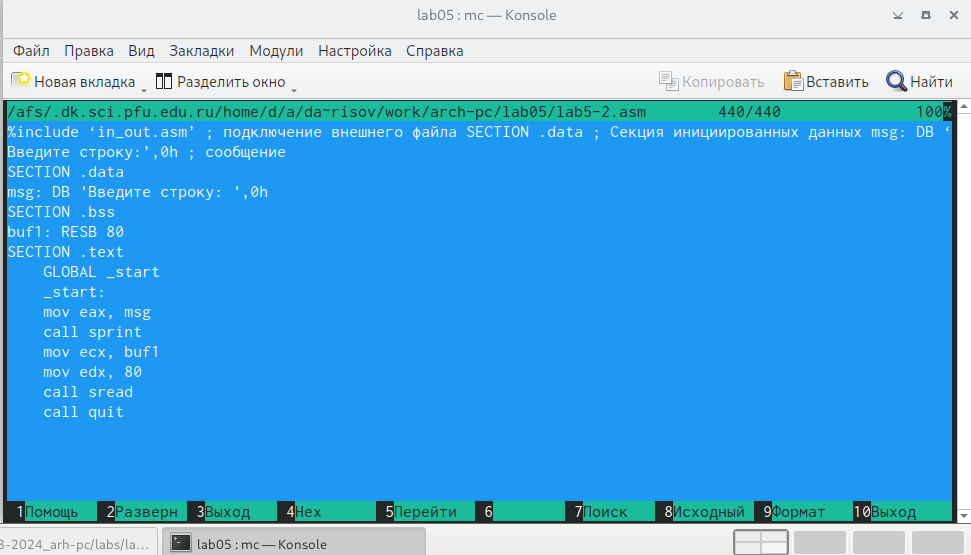
Трансляция и запуск

## 4.3 3.Подключение внешнего файла

Я загружаю файл на ТУИС и проверяю, что он был успешно загружен в папке, где лежит наша программа. Сразу же выполняю копирование этого файла в программу lab5-1.asm (рис. ??) (рис. ??).

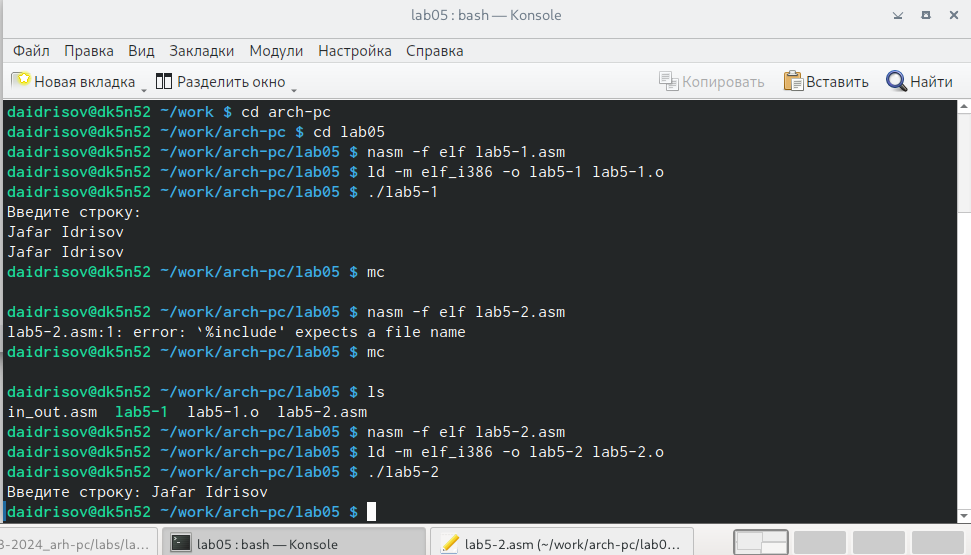
 

Изменяю текст программы и просмотриваю код (рис. ??).



Изменение кода

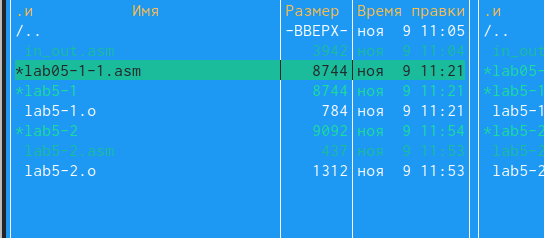
Запускаю программу (рис. ??).



Запуск программы

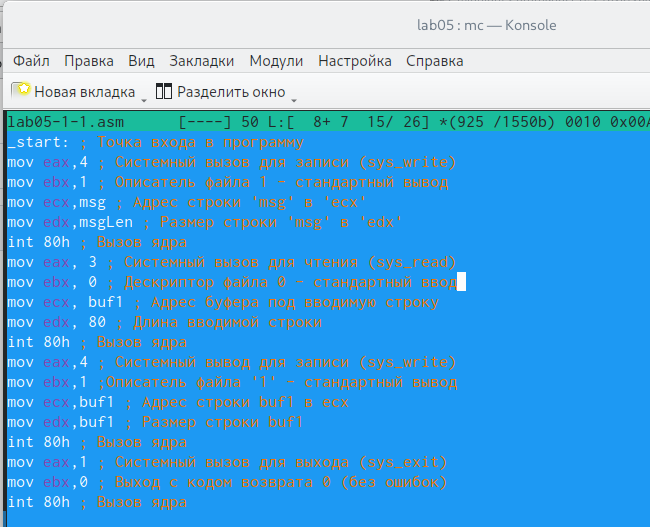
# 5 Выполнение заданий

№1 Я скопирую файл “lab5-1.asm”, используя клавишу F5, чтобы создать его дубликат. (рис. ??).



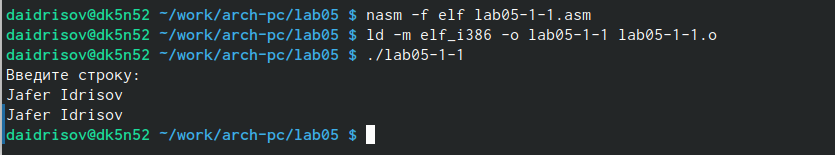
Создаю копию файла

Я внесу изменения в код, используя клавишу F4 для редактирования. (рис. ??).



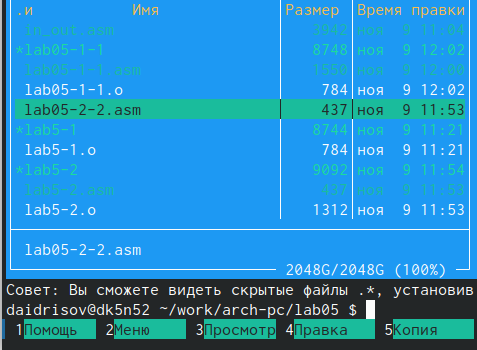
Изменения в коде

№2 Выполняем компоновку и запускаем программу. (рис. ??).



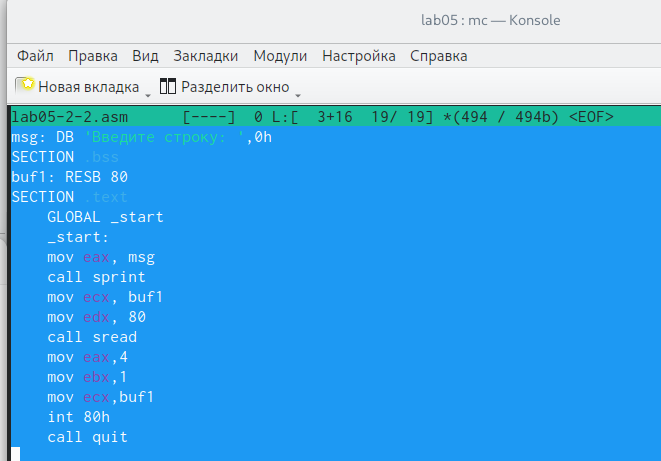
Компоновка и запуск

№3 Создам копию файла lab5-2.asm (использую клавишу F5). (рис. ??).



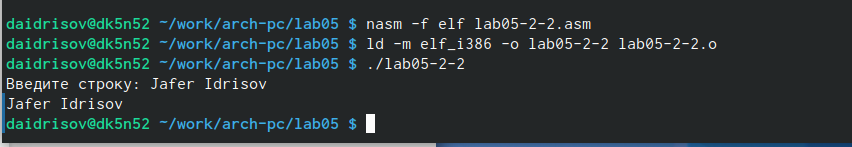
Создадим копию

Я исправлю код программы и интегрирую в него внешний файл “in\_out.asm” для его работы.(рис. ??).



Изменение кода

№4 Выполню компоновку и запускаю программу (рис. ??).



Компоновка и запуск программы

# 6 Выводы

В результате выполнения этой лабораторной работы, я получил практические навыки, которые наверняка будут полезными при работе с различными языками программирования.

# Список литературы

Лабораторная работа №5. Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM. Системные вызовы в ОС GNU Linux