Отчёт по лабораторной работе №5

Дисциплина: Архитектура компьютера

Кирьянова Екатерина Андреевна

Содержание

# 1 Цель работы

Приобрести практические навыки работы в Midnight Commander и освоить инструкции 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 Midnight Commander

Открываю Midnight Commander и перехожу в нужный каталог (рис. 1).

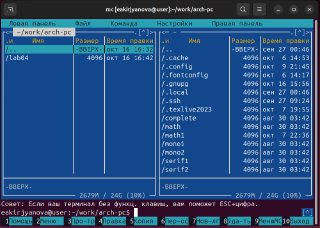


Рис. 1: Midnight Commander

Создаю новый каталог с помощью клавиши F7 и перехожу в него (рис. 2)

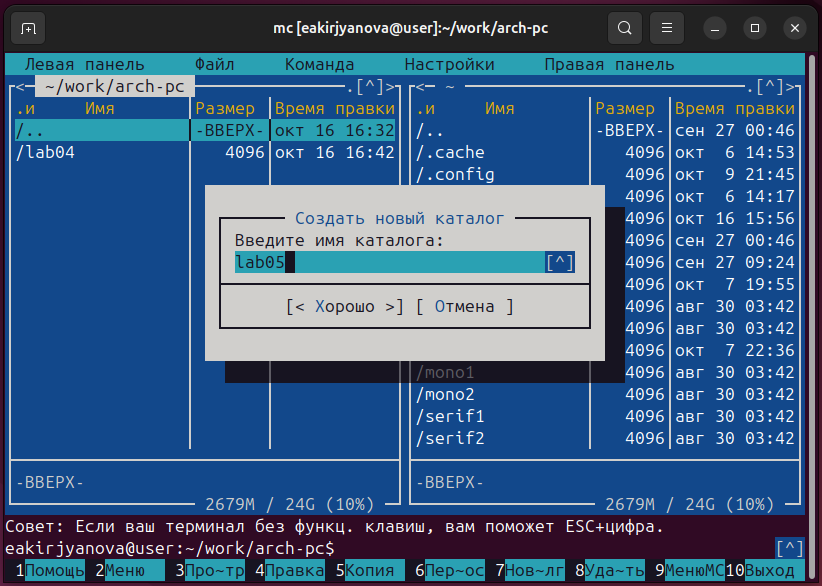


Рис. 2: Новый каталог

Создаю новый файл lab5-1.asm (рис. 3).

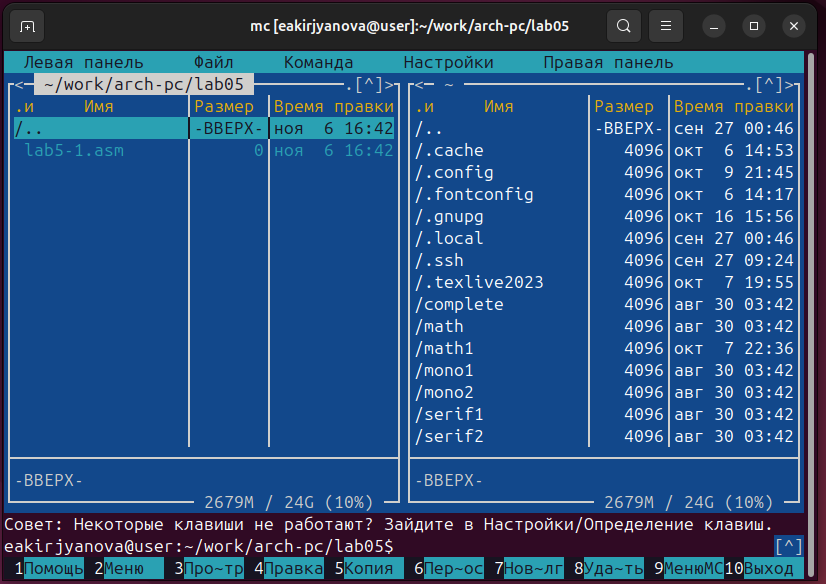


Рис. 3: Новый файл

## 4.2 Cтруктура программы на языке ассемблера NASM

Открываю файл в текстовом редакторе nano (рис. 4).

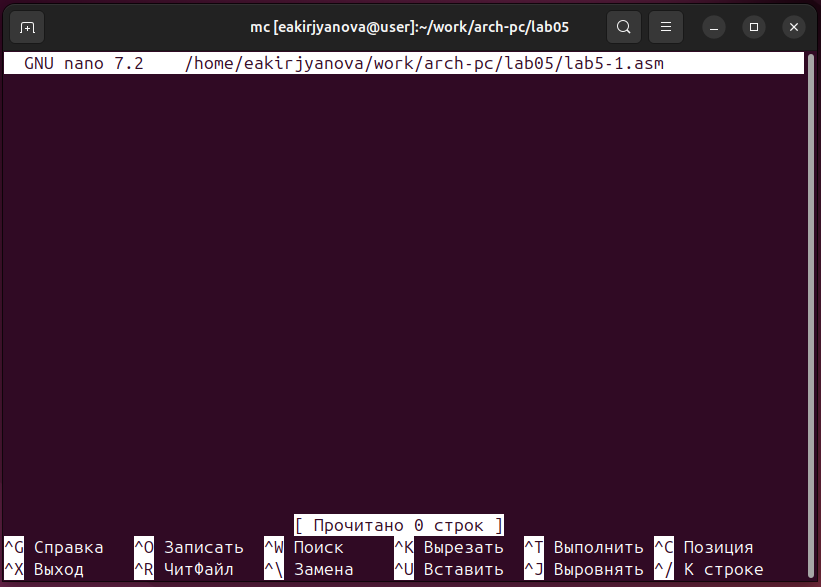


Рис. 4: Nano

Ввожу текст программы из листинга 5.1 (рис. 5).

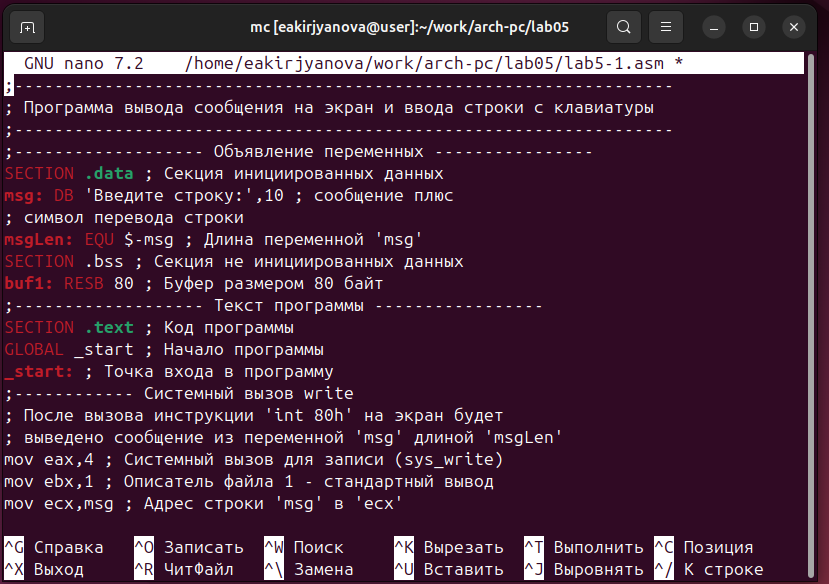


Рис. 5: Программа

Выхожу из файла с помощью клавиш Ctrl+X и сохраняю с помощью клавиш Y, Enter. Далее открываю файл клавишей F3 и проверяю успешность выполнения (рис. 6).

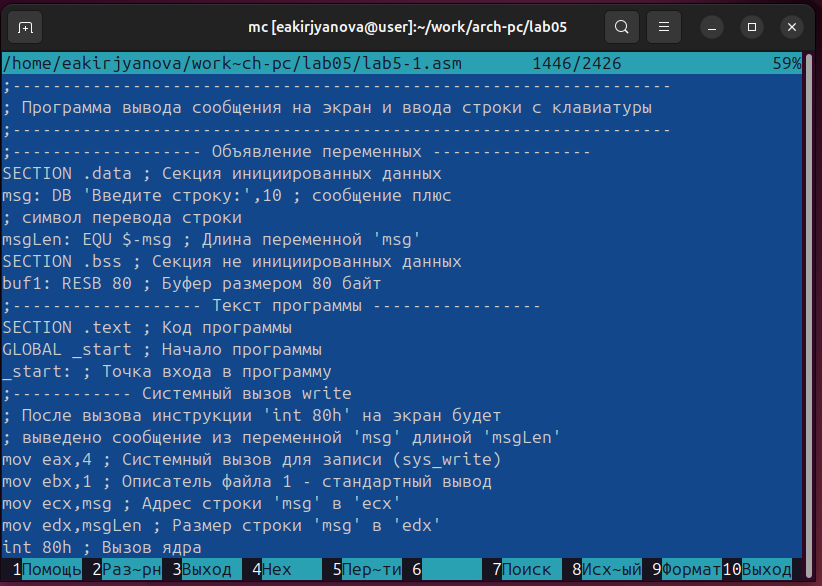


Рис. 6: Сохранение

Транслирую текст программы в объектный файл, выполняю компоновку объектного файла и запускаю исполняемый файл. После строки “Введите строку:” ввожу ФИО (рис. 7).

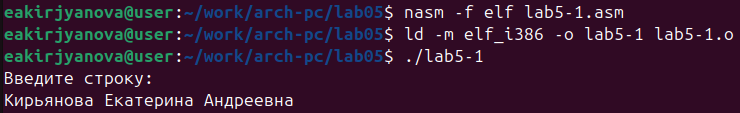


Рис. 7: Запуск программы

## 4.3 Подключение внешнего файла

Открываю две панели (рис. 8).

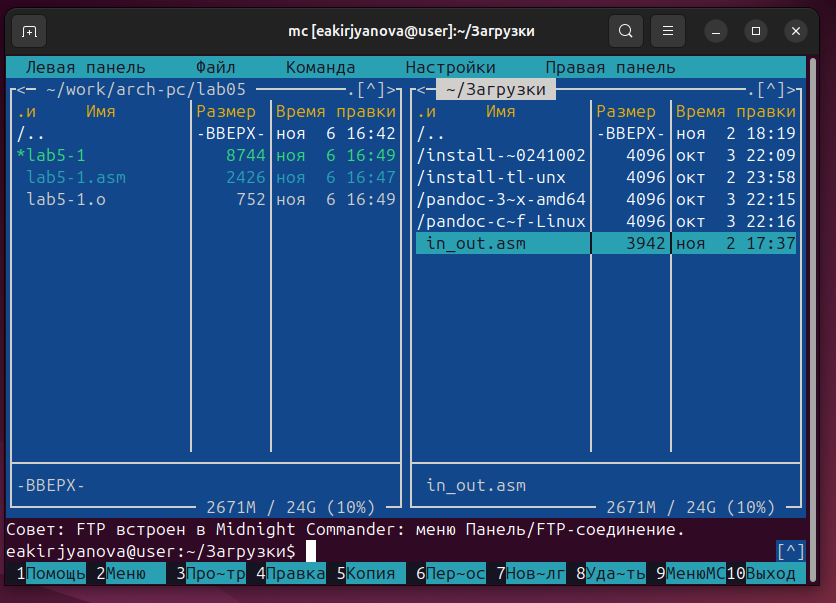


Рис. 8: Обе панели

Копирую внешний файл в нужный каталог с помощью клавиши F5 (рис. 9).

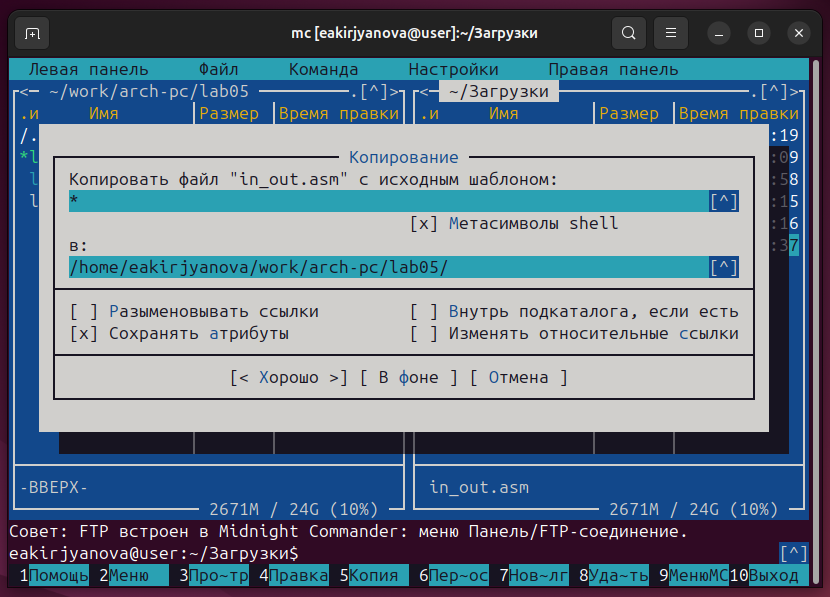


Рис. 9: Перемещение файла

Копирование прошло успешно (рис. 10).

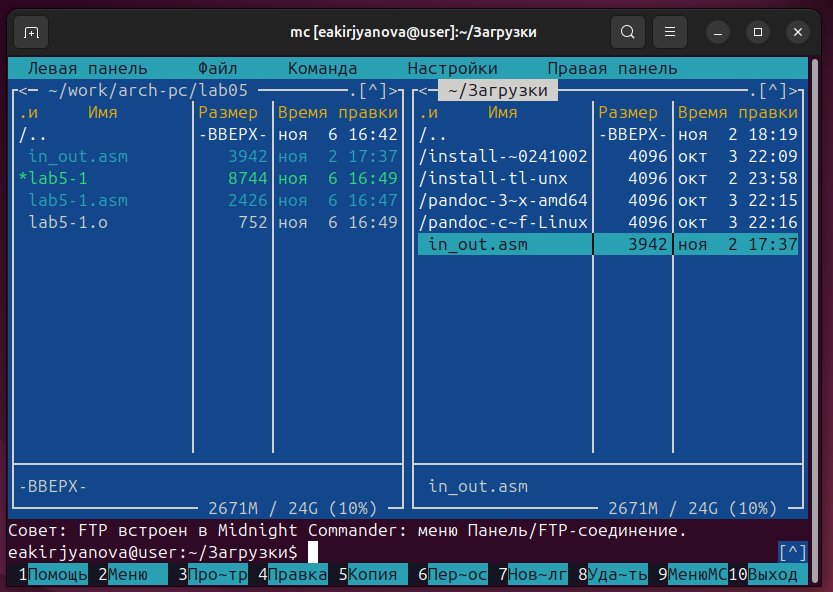


Рис. 10: Результат копирования

Создаю копию файла lab5-1.asm с именем lab5-2.asm с помощью клавиши F5 (рис. 11).

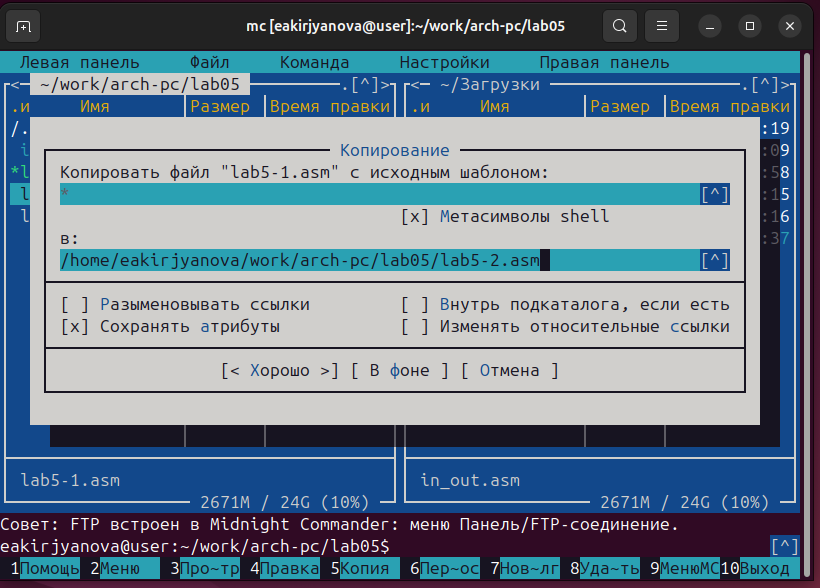


Рис. 11: Создание копии

Демонстрирую успешность выполнения (рис. 12).

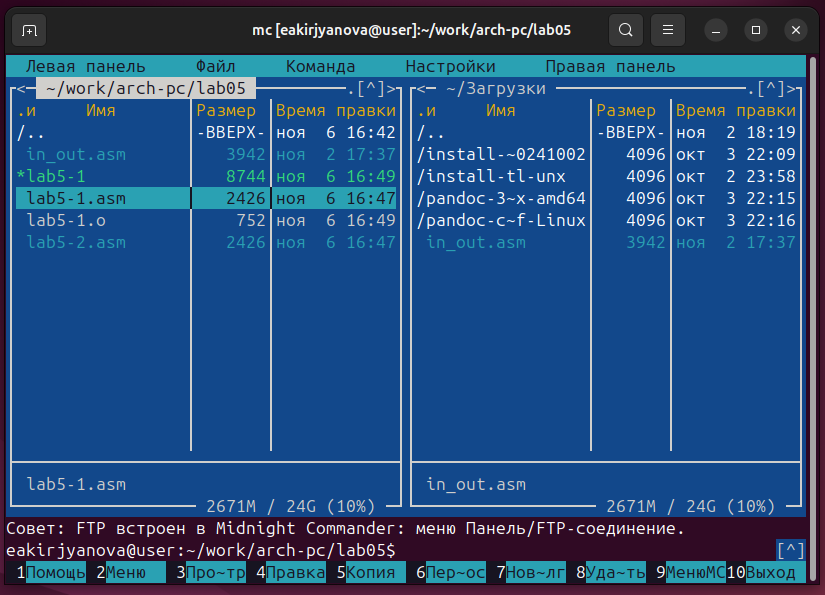


Рис. 12: Копия файла

Редактирую файл согласно листингу 5.2 (рис. 13).

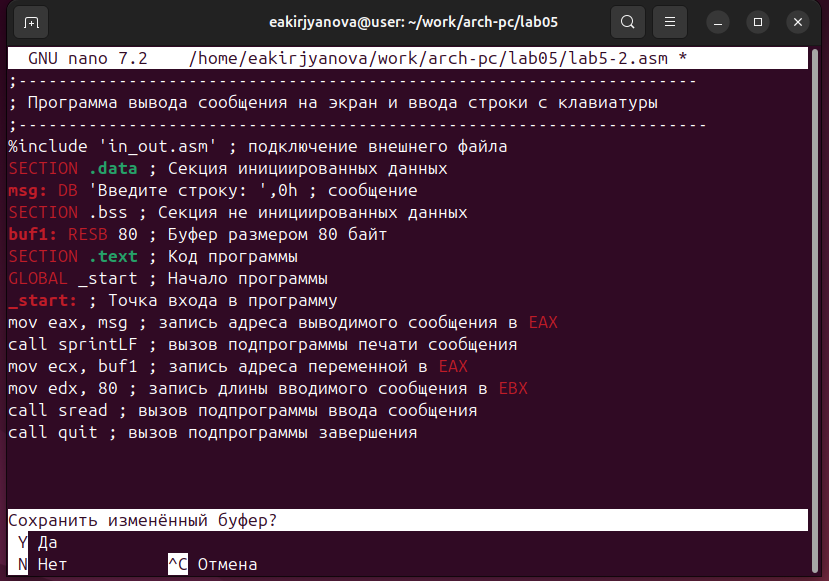


Рис. 13: Текст программы

Транслирую текст программы в объектный файл, выполняю компоновку объектного файла и запускаю исполняемый файл (рис. 14).

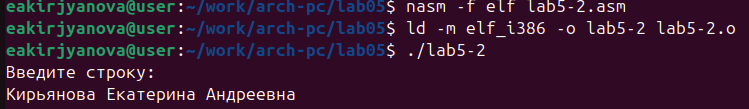


Рис. 14: Запуск файла

Создаю файл lab5-22.asm на основе файла lab5-2.asm и меняю sprintLF на sprint (рис. 15).

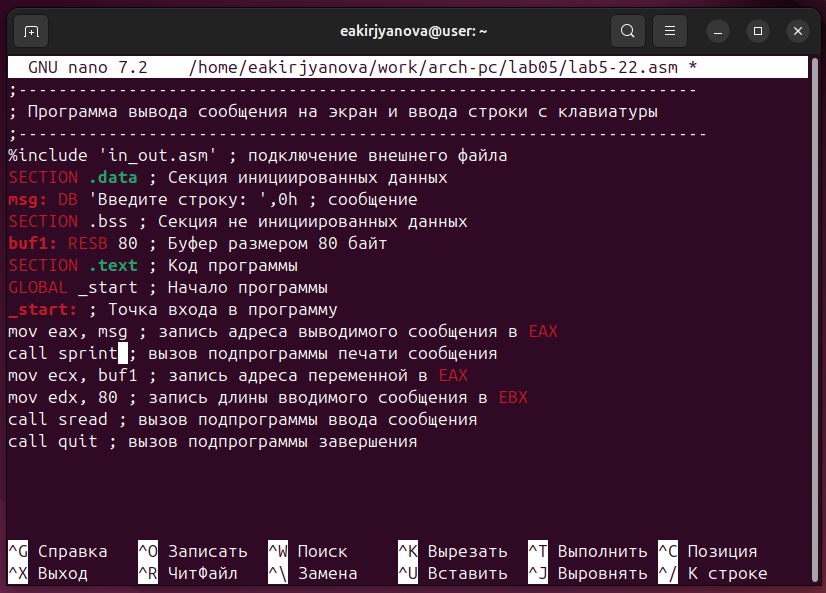


Рис. 15: Изменение программы

Снова транслирую в объектный файл, выполняю его компоновку и запускаю (рис. 16).

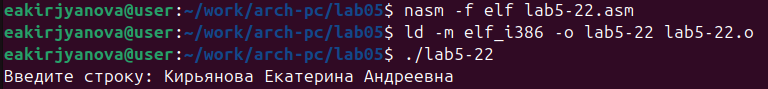


Рис. 16: Запуск

Разница между первым исполняемым файлом lab5-2 и вторым lab5-22 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.

## 4.4 Задание для самостоятельной работы

Создаю копию файла lab5-1.asm с именем lab5-11.asm с помощью клавиши F5 (рис. 17).

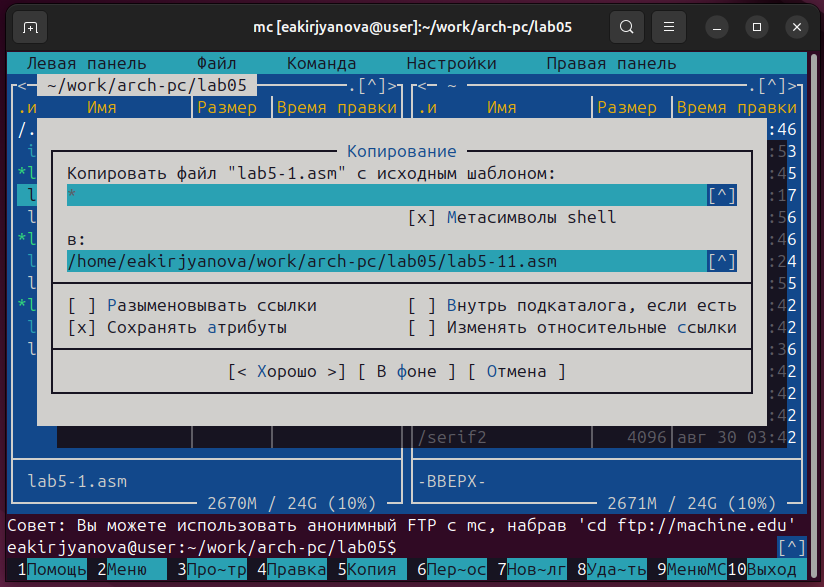


Рис. 17: Копирование файла

Изменяю программу, чтобы кроме запроса ввода она выводила заданную строку (рис. 18).

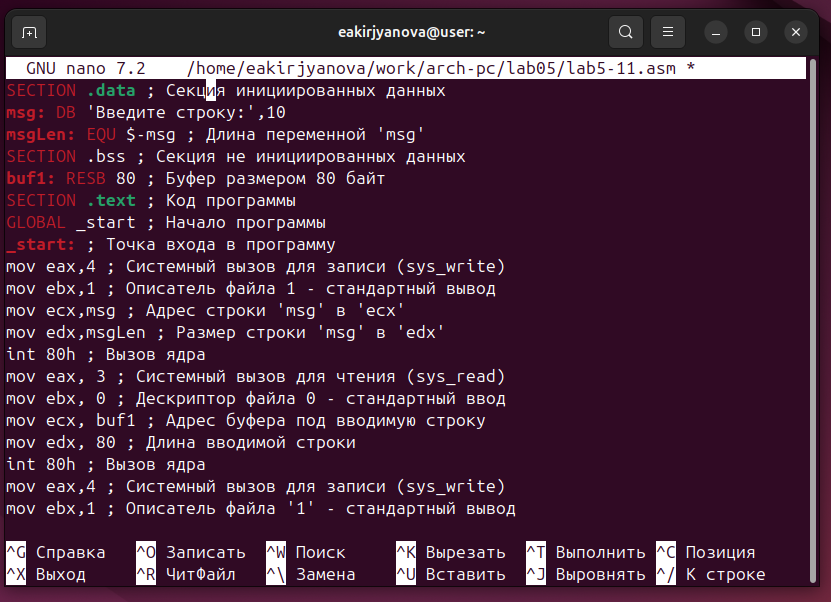


Рис. 18: Редактирование файла

Транслирую текст программы в объектный файл, выполняю компоновку объектного файла и запускаю исполняемый файл. Программа выводит заданную мною строку (рис. 19).

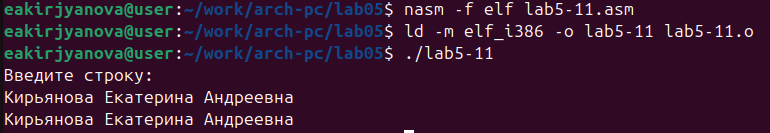


Рис. 19: Запуск программы

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

Создаю копию файла lab5-2.asm с именем lab5-222.asm с помощью клавиши F5 (рис. 20).

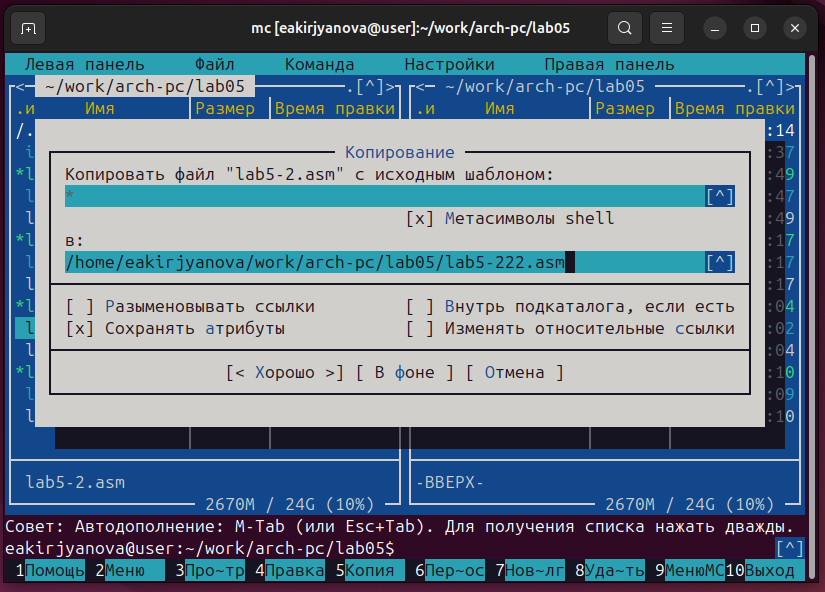


Рис. 20: Копирование файла

Изменяю программу, чтобы кроме запроса ввода она выводила заданную строку (рис. 21).

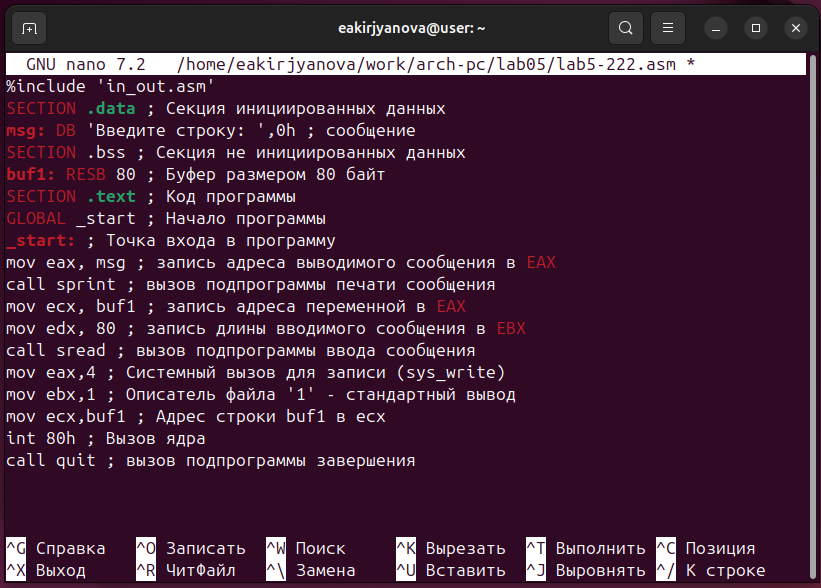


Рис. 21: Редактирование файла

Транслирую текст программы в объектный файл, выполняю компоновку объектного файла и запускаю исполняемый файл. Программа выводит заданную мною строку (рис. 22).

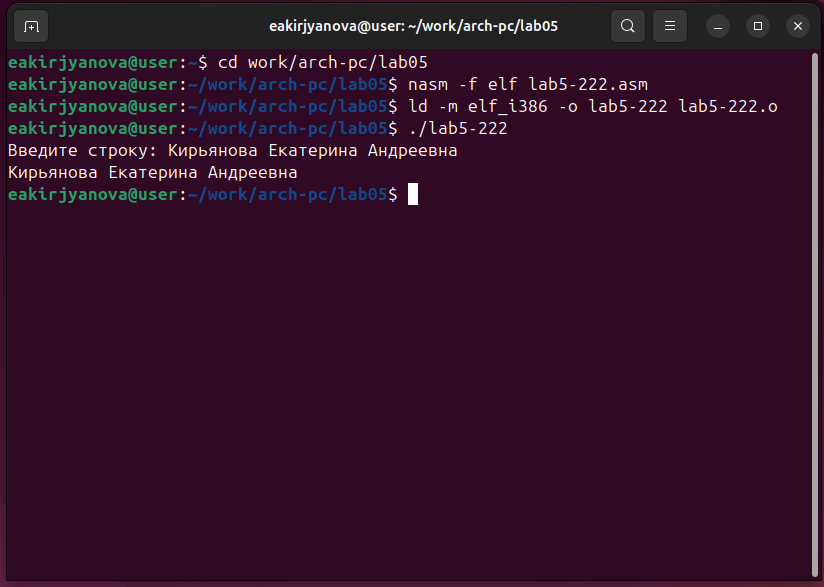


Рис. 22: Запуск программы

Код программы из пункта 2:

%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

# 5 Выводы

В ходе выполнения данной лабораторной работы я приобрела практические навыки работы в Midnight Commander и освоила инструкции языка ассемблера mov и int.

# 6 Список литературы

1. [Лабораторная работа №5](https://esystem.rudn.ru/pluginfile.php/2089538/mod_resource/content/0/%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%965.%20%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D1%8B%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B%20%D1%81%20Midnight%20Commander%20%28%29.%20%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D1%8B%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM.%20%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D1%8B%D0%B5%20%D0%B2%D1%8B%D0%B7%D0%BE%D0%B2%D1%8B%20%D0%B2%20%D0%9E%D0%A1%20GNU%20Linux.pdf)