Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Бызова Мария Олеговна

Содержание

# 1 Цель работы

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

# 2 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

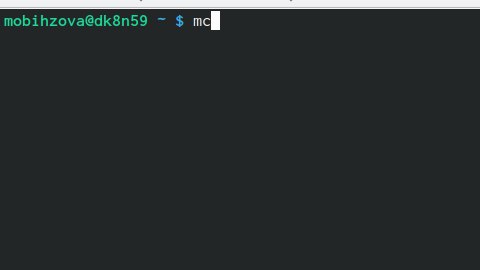
Здесь операнд dst — приёмник, а src — источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непосредственные значения (const). Инструкция языка ассемблера intпредназначена для вызова прерывания с указанным номером.

int n

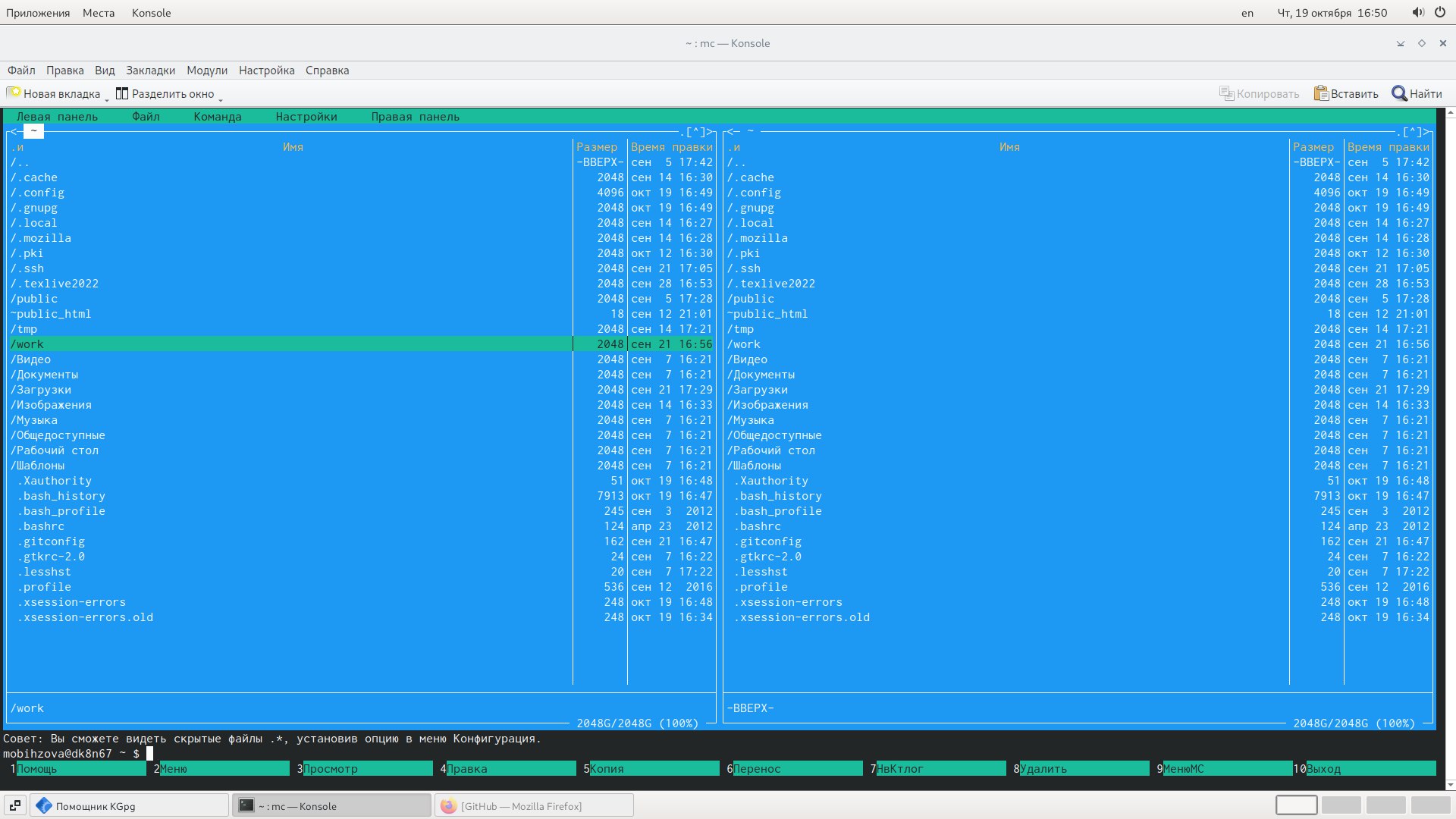
Здесь n — номер прерывания, принадлежащий диапазону 0–255. При программировании в Linux с использованием вызовов ядра sys\_calls n=80h (принято задавать в шестнадцатеричной системе счисления).

# 3 Выполнение лабораторной работы

1. Открываем Midnight Commander, введя в терминал mc (рис. [??], [??]).

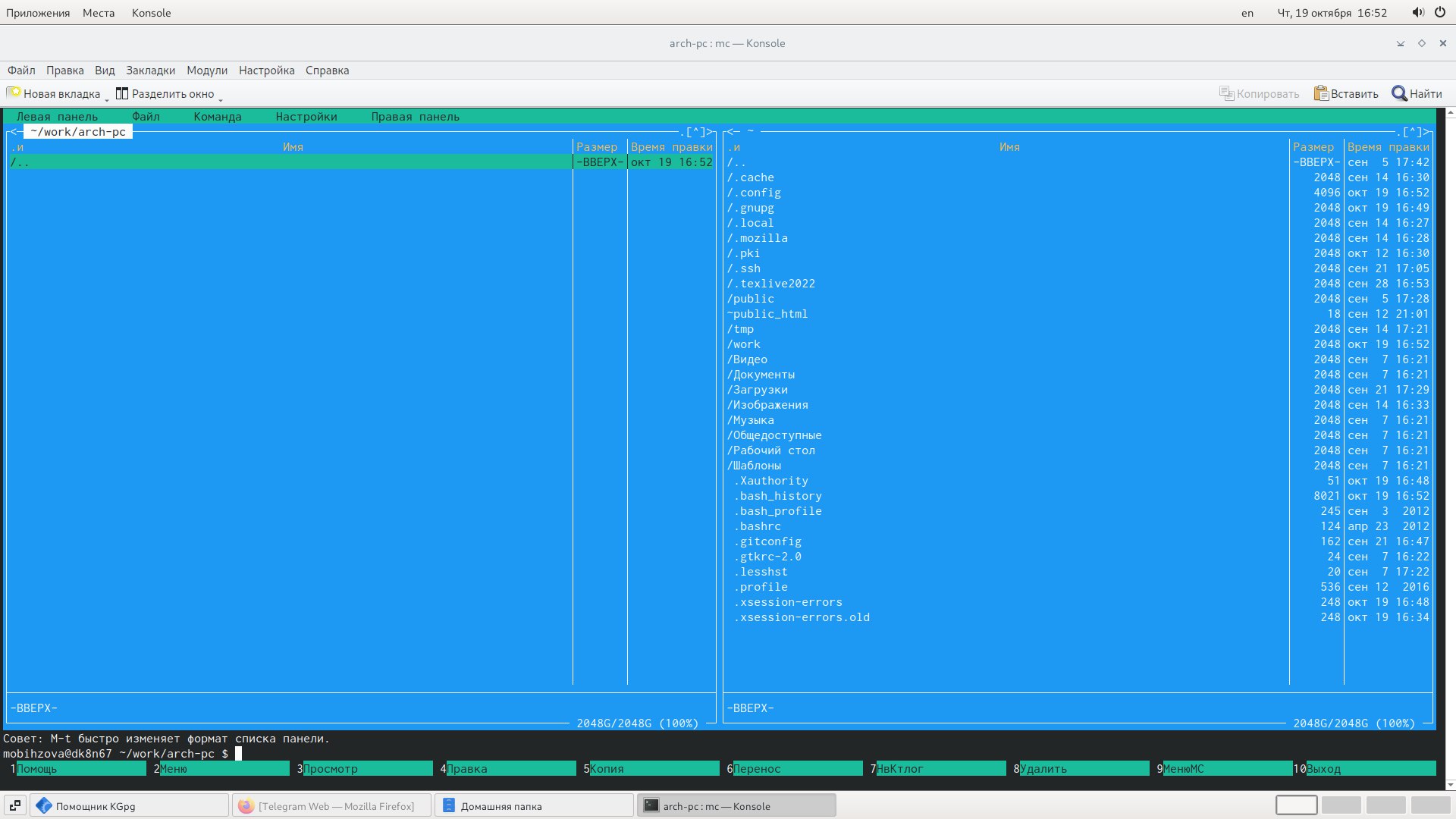


Введение команды



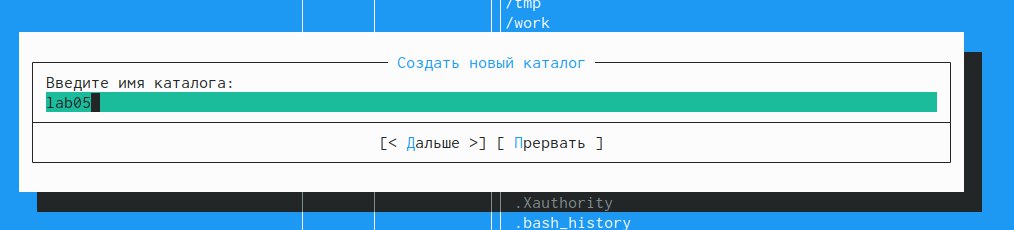
Открытый mc

1. Пользуясь клавишами клавиатуры, переходим в каталог ~/work/arch-pc, созданный при выполнении предыдущей лаборатороной работы рис. [??])



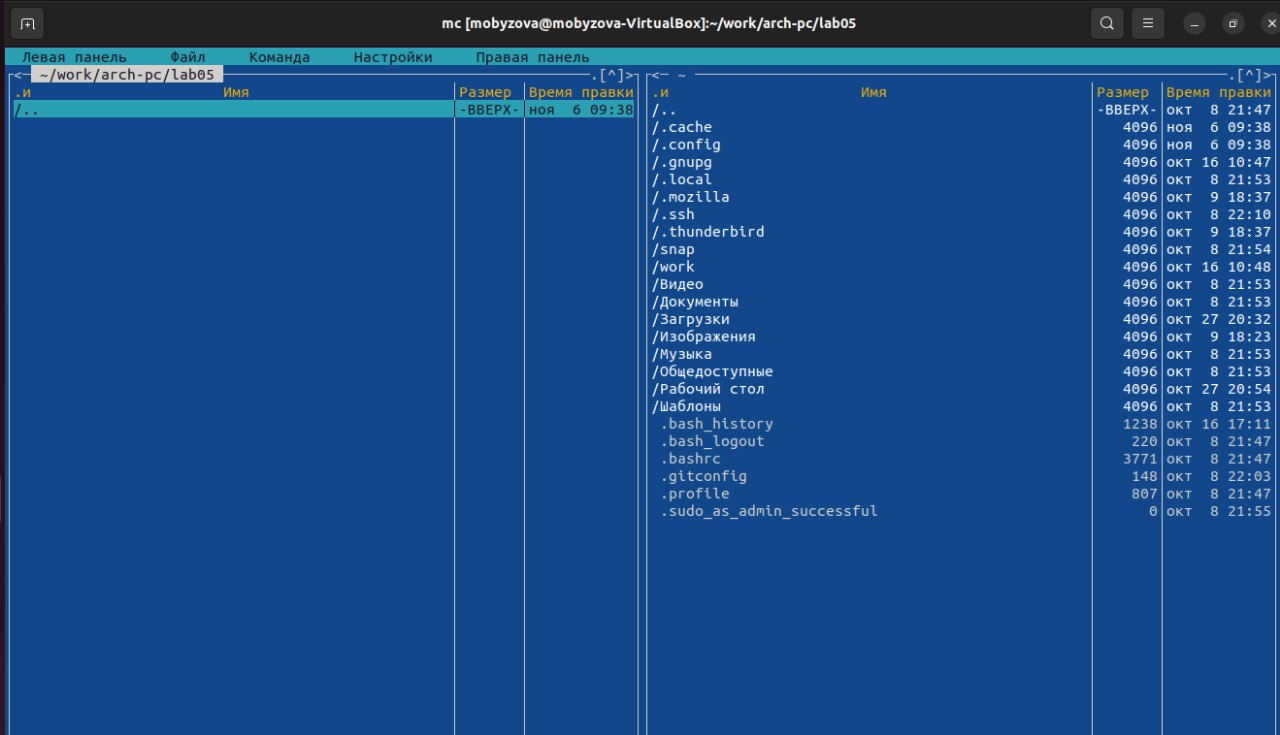
Перемещение между директориями

1. С помощью функциональной клавиши F7 создаем каталог lab05 (рис. [??]).



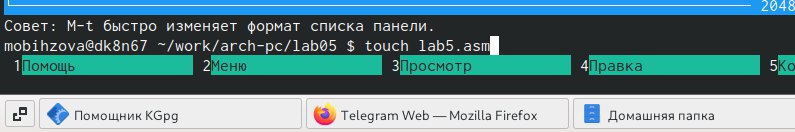
Создание каталога

Переходим в созданный каталог (рис. [??]).



Перемещение между директориями

1. В строке ввода прописываем команду touch lab5-1.asm, чтобы создать файл, в котором будем работать (рис. [??]).



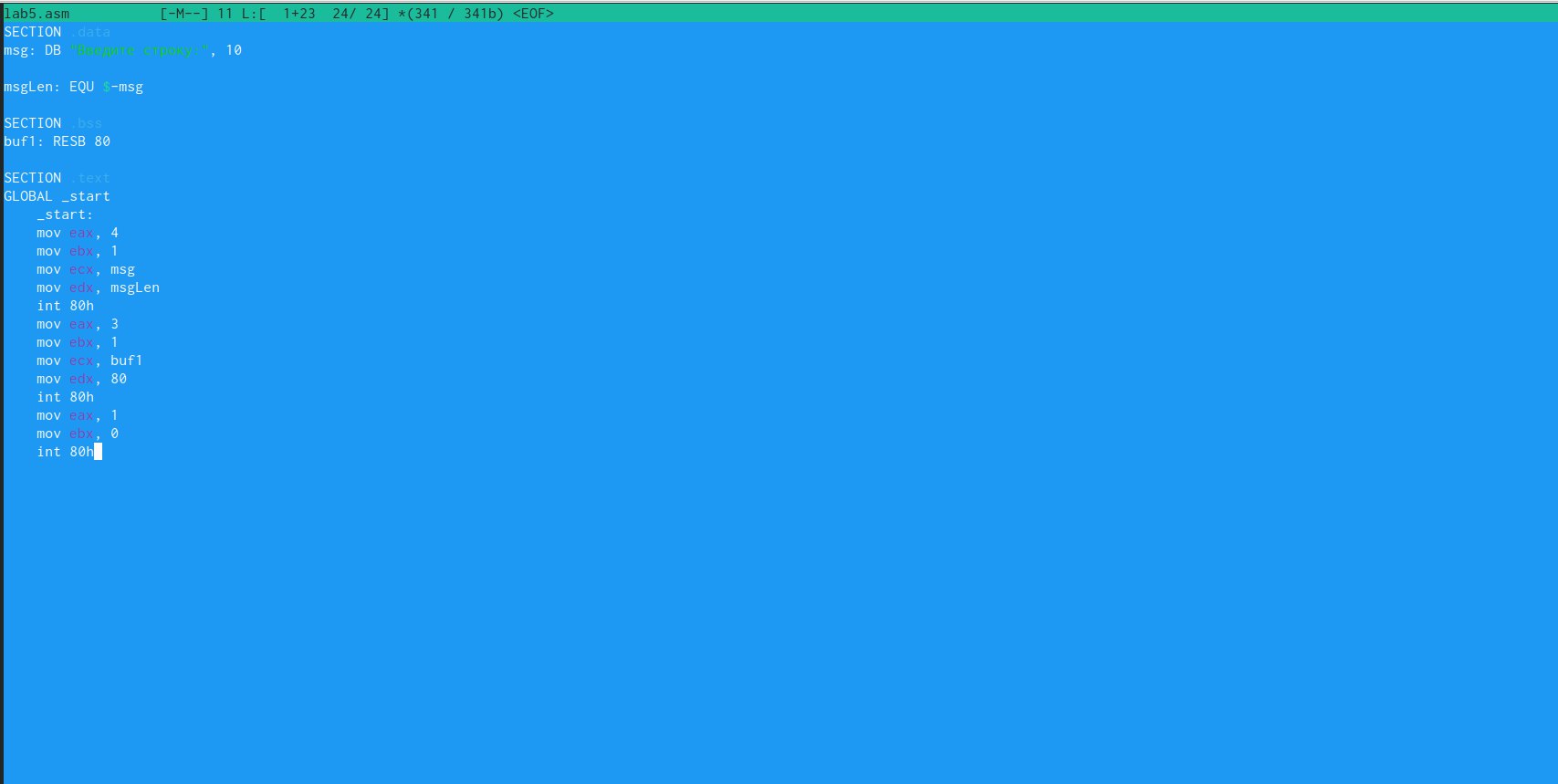
Создание файла

1. С помощью функциональной клавиши F4 открываем созданный файл для редактирования во встроенном редакторе (рис. [??]). Как правило в качестве встроенного редактора Midnight Commander используется редакторы nano или mcedit.



Открытие файла для редактирования

1. Вводим в файл код программы для запроса строки у пользователя (рис. [??]). Далее выходим из файла (Ctrl+X), сохраняя изменения (Y, Enter).



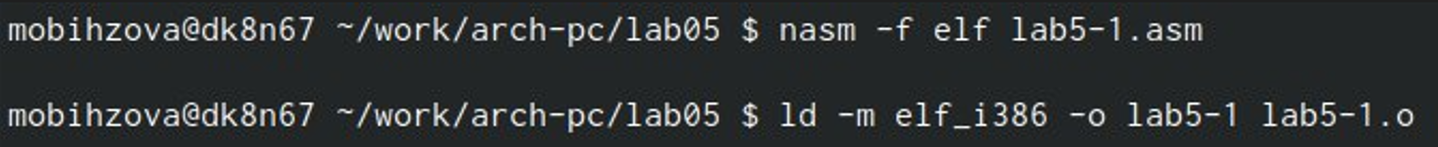
Редактирование файла

1. С помощью функциональной клавиши F3 открываем файл для просмотра, чтобы проверить, содержит ли файл текст программы (рис. [??]).



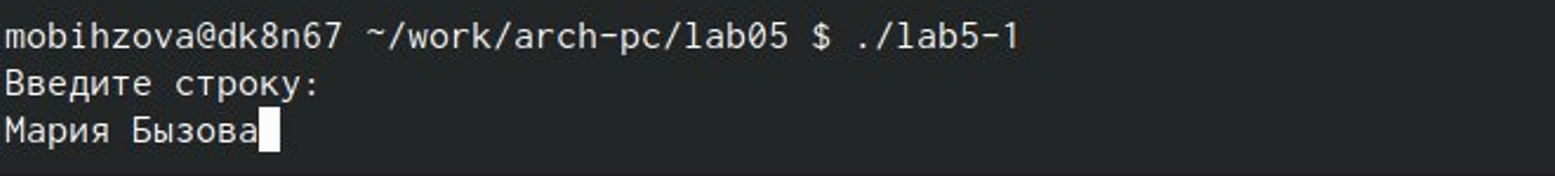
Открытие файла для просмотра

1. Транслируем текст программы файла в объектный файл командой nasm -f elf lab5-1.asm. Создается объектный файл lab5-1.o. Выполняем компоновку объектного файла с помощью команды ld -m elf\_i386 -o lab5-1 lab5-1.o (рис. [??]). Создается исполняемый файл lab5-1.



Компиляция файла и передача на обработку компоновщику

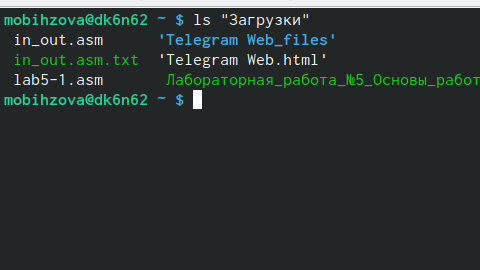
Запускаем исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, вводим свои ФИО, на этом программа заканчивает свою работу (рис. [??]).



Исполнение файла

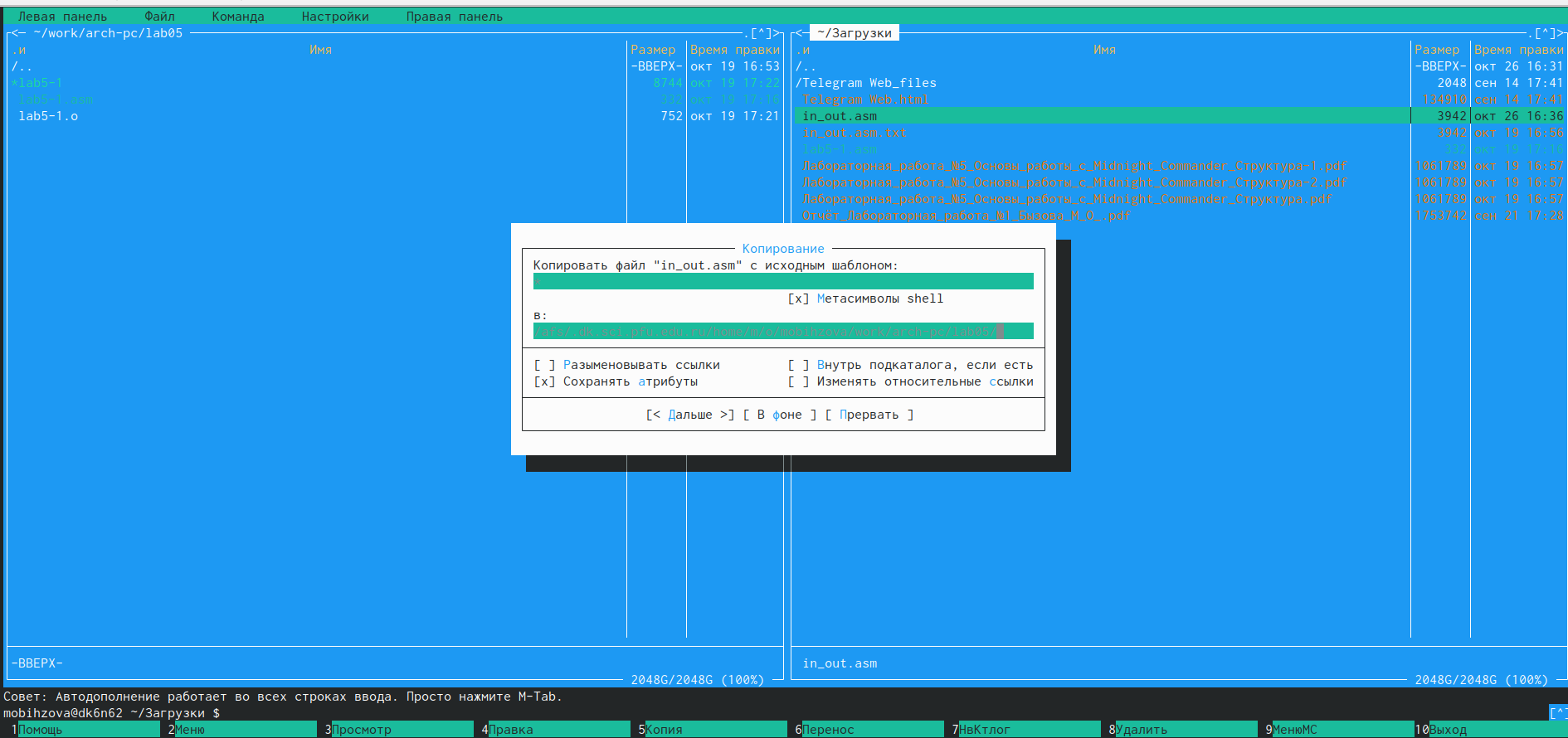
## 3.1 Подключение внешнего файла

1. Скачиваем файл in\_out.asm со страницы курса в ТУИС. Он сохраняется в каталог “Загрузки” (рис. [??]).



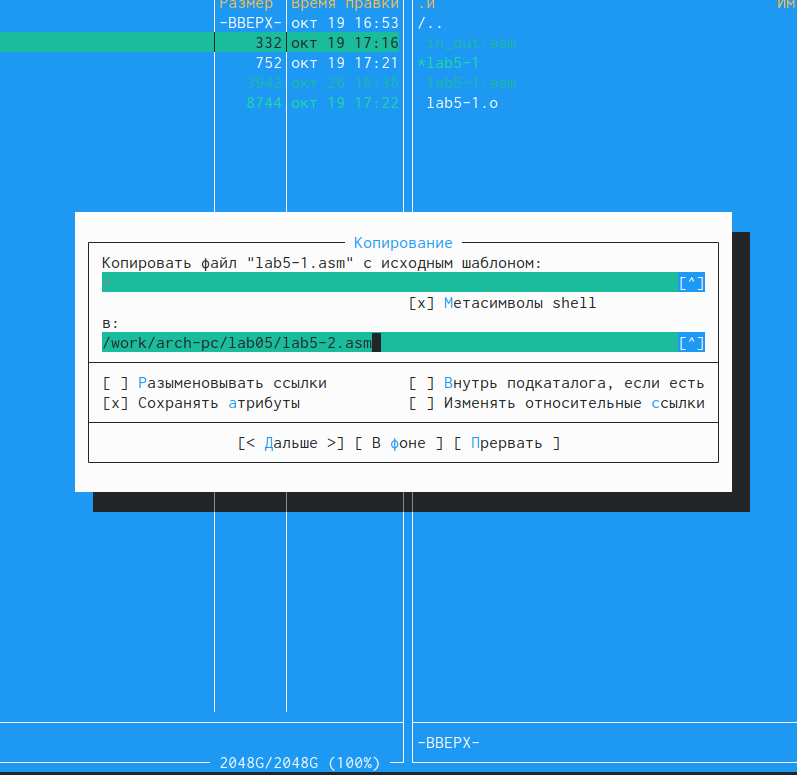
Скачанный файл

1. С помощью функциональной клавиши F5 копируем файл in\_out.asm из каталога Загрузки в созданный каталог lab05 (рис. [??]).



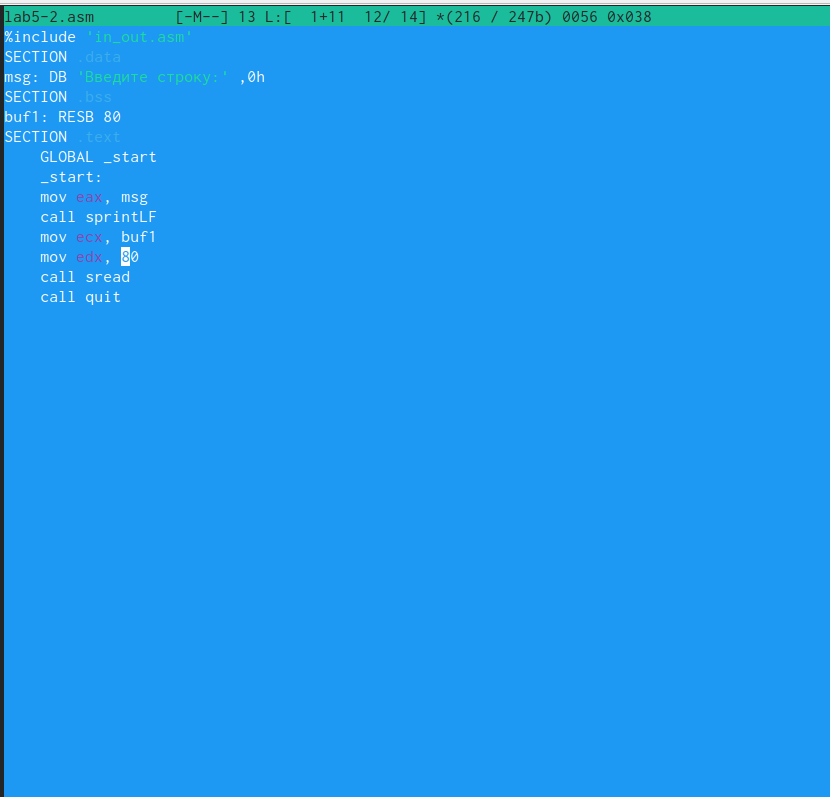
Копирование файла

1. С помощью функциональной клавиши F5 копируем файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываем имя для копии файла (рис. [??]).



Копирование файла

1. Изменяем содержимое файла lab5-2.asm во встроенном редакторе (рис. [??]), чтобы в программе использовались подпрограммы sprintLF, sread и quit из внешнего файла in\_out.asm.

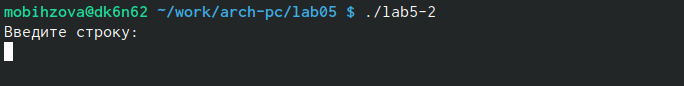


Редактирование файла

Транслируем текст программы файла в объектный файл командой nasm -f elf lab5-2.asm. Создается объектный файл lab5-2.o. Выполняем компоновку объектного файла с помощью команды ld -m elf\_i386 -o lab5-2 lab5-2.o Создается исполняемый файл lab5-2. Запускаем исполняемый файл (рис. [??], [??],).

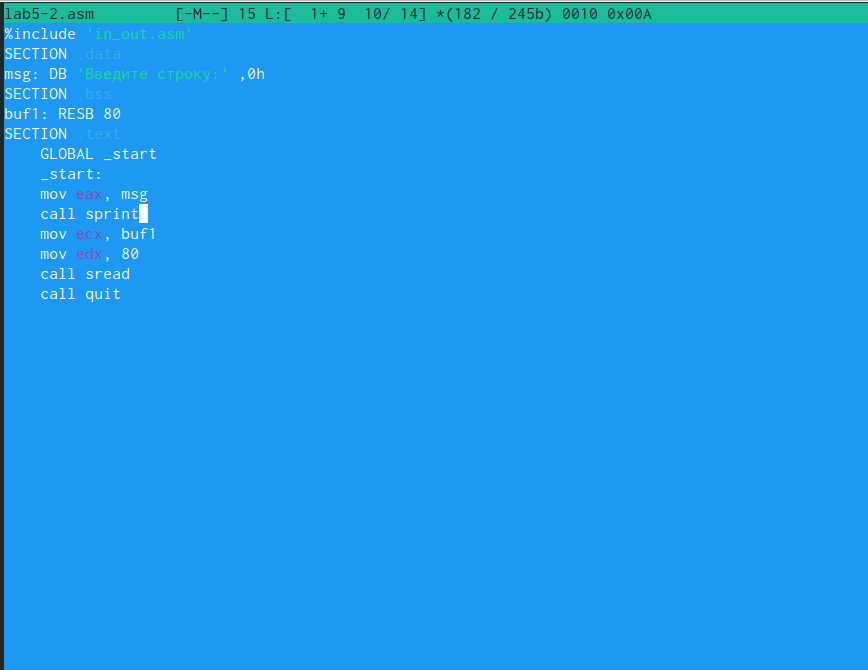
Компиляция файла и передача на обработку компоновщику

Компиляция файла и передача на обработку компоновщику



Исполнение файла

1. Открываем файл lab5-2.asm для редактирования во встроенном редакторе функциональной клавишей F4. Изменяем в нем подпрограмму sprintLF на sprint. Сохраняем изменения и открываем файл для просмотра, чтобы проверить сохранение действий (рис. [??]).

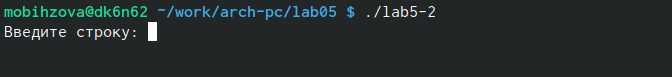


Отредактированный файл

Снова транслируем файл, выполняем компоновку созданного объектного файла. Транслируем текст программы файла в объектный файл командой nasm -f elf lab5-2.asm. Создается объектный файл lab5-2.o. Выполняем компоновку объектного файла с помощью команды ld -m elf\_i386 -o lab5-2 lab5-2.o Создается исполняемый файл lab5-2. Запускаем новый исполняемый файл (рис. [??], [??]).

Компиляция файла и передача на обработку компоновщику

Компиляция файла и передача на обработку компоновщику

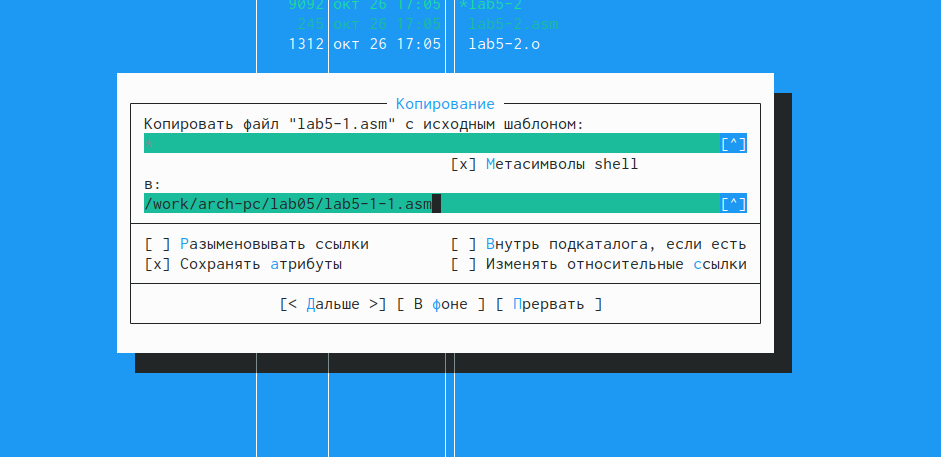


Исполнение файла

Разница между первым исполняемым файлом и вторым 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.

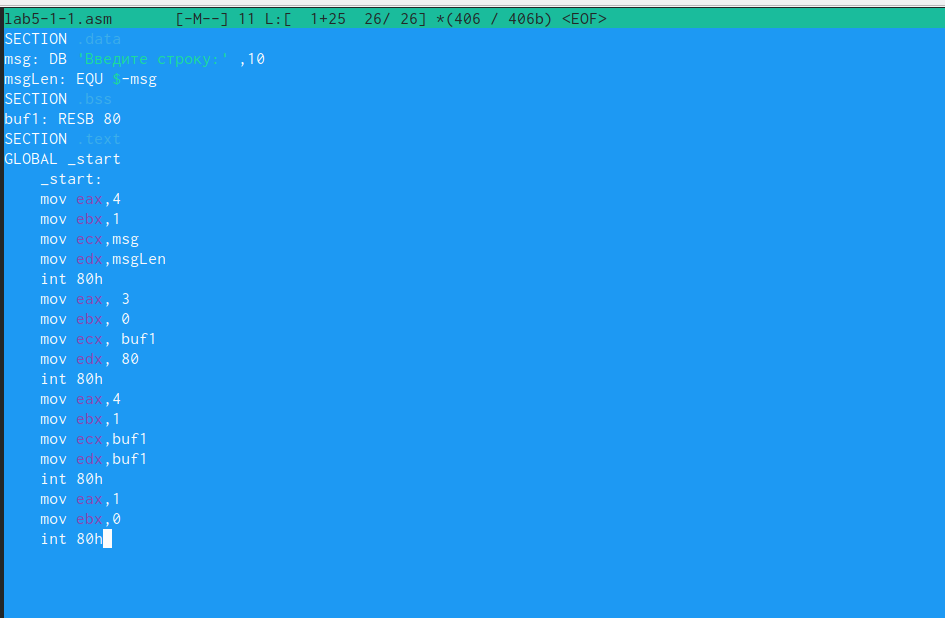
## 3.2 Выполнение заданий для самостоятельной работы

1. Создаем копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5 (рис. [??]).



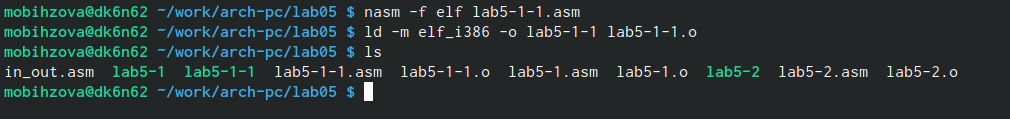
Копирование файла

С помощью функциональной клавиши F4 открываем созданный файл для редактирования. Изменяем программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [??]).

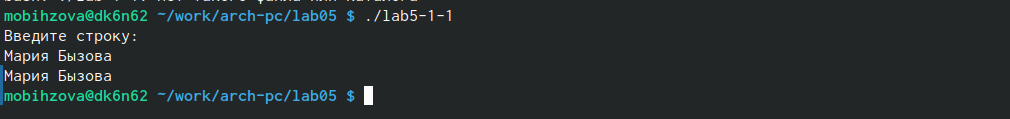


Редактирование файла

1. Создаем объектный файл lab5-1-1.o, отдаем его на обработку компоновщику, получаем исполняемый файл lab5-1-1, запускаем полученный исполняемый файл. Программа запрашивает ввод, вводим свои ФИО, далее программа выводит введенные нами данные (рис. [??], [??]).



Компиляция файла и передача на обработку компоновщику

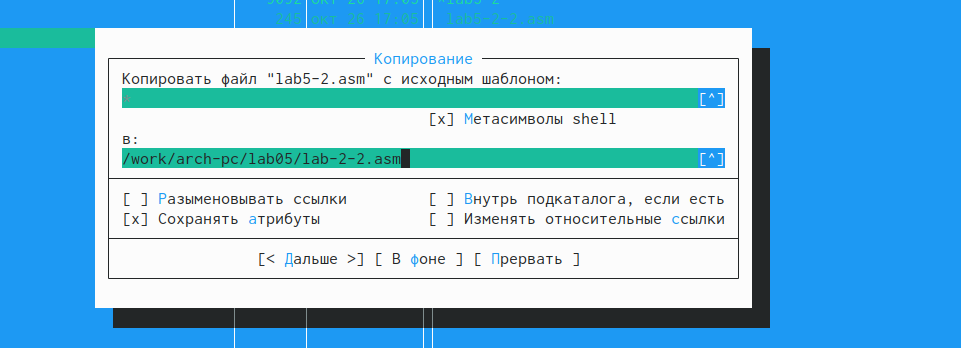


Исполнение файла

Код программы из пункта 1:

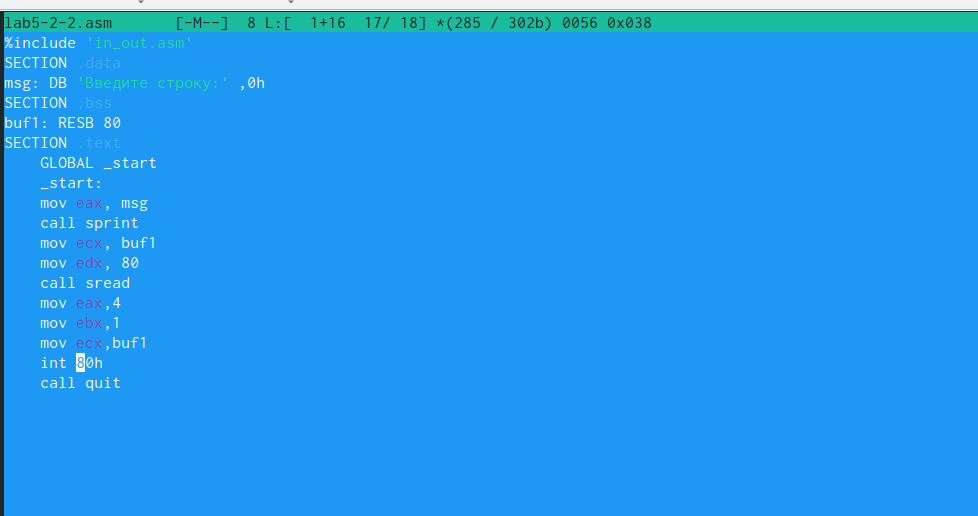
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку:',10  
msgLen: EQU $-msg ; Длина переменной 'msg'  
SECTION .bss ; Секция не инициированных данных  
buf1: RESB 80 ; Буфер размером 80 байт  
SECTION .text ; Код программы  
GLOBAL \_start ; Начало программы  
 \_start: ; Точка входа 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. Создаем копию файла lab5-2.asm с именем lab5-2-2.asm с помощью функциональной клавиши F5 (рис. [??]).



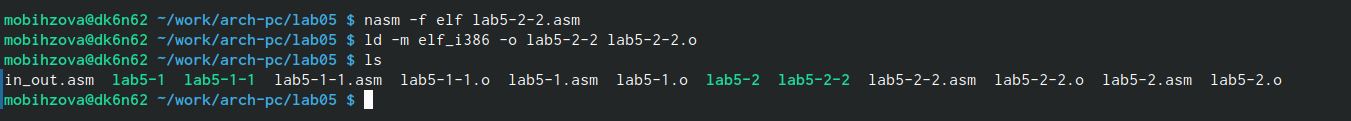
Копирование файла

С помощью функциональной клавиши F4 открываем созданный файл для редактирования. Изменяем программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [??]).

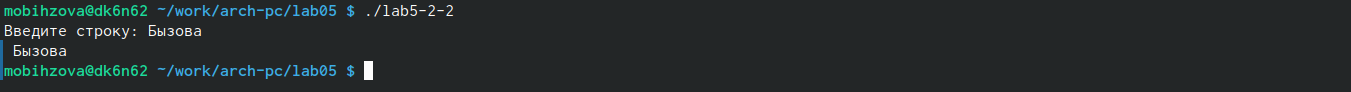


Редактирование файла

1. Создаем объектный файл lab5-2-2.o, отдаем его на обработку компоновщику, получаеи исполняемый файл lab5-2-2, запускаем полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, вводим свои ФИО, далее программа выводит введенные нами данные (рис. [??], [??]).



Компиляция файла и передача на обработку компоновщику



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

# 4 Выводы

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