Отчёт по лабораторной работе №4

Дисциплина: Архитектура Компьютера

Азарцова Вероника Валерьевна

Содержание

# 1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в MC (Midnight Commander), изучение инструкций языка ассемблера mov и int, написание и запуск программ с системными вызовами для обеспечения диалога с пользователем, подключение и использование в программе внешнего файла.

# 2 Задание

1. Изучение теоретического введения.
2. Выполнение лабораторной работы.
3. Выполнение заданий для самостоятельной работы.

# 3 Теоретическое введение

## 3.1 Основы работы с Midnight Commander

Midnight Commander (mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой.  
В Midnight Commander используются функциональные клавиши F1 — F10 , к которым привязаны часто выполняемые операции (табл. 1) Например, в табл. 1 приведено краткое описание стандартных каталогов Unix.

Таблица 1: Описание часто выполняемых операций mc

| F1-F10 | Описание выполняемой операции |
| --- | --- |
| F1 | вызов контекстно-зависимой подсказки |
| F2 | вызов меню, созданного пользователем |
| F3 | просмотр файла, на который указывает подсветка в активной панели |
| F4 | вызов встроенного редактора для файла, на который указывает подсветка в активной панели |
| F5 | копирование файла или группы отмеченных файлов из каталога, отображаемого в активной панели, в каталог, отображаемый на второй панели |
| F6 | перенос файла или группы отмеченных файлов из каталога, отображаемого в активной панели, в каталог, отображаемый на второй панели |
| F7 | создание подкаталога в каталоге, отображаемом в активной панели |
| F8 | удаление файла (подкаталога) или группы отмеченных файлов |
| F9 | вызов основного меню программы |
| F10 | выход из программы |

## 3.2 Структура программы на языке ассемблера NASM

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) — 1 байт; \* DW (define word) — 2 байта; \* DD (define double word) — 4 байта; \* DQ (define quad word) — 8 байт; \* DT (define ten bytes) — 10 байт.

## 3.3 Элементы программирования

### 3.3.1 Инструкция mov

Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике. В общем виде эта инструкция записывается в виде “mov dst,src”.  
Здесь операнд dst — приёмник, а src — источник.

### 3.3.2 Инструкция int

Инструкция языка ассемблера intпредназначена для вызова прерывания с указанным номером. В общем виде она записывается в виде int n Здесь n — номер прерывания, принадлежащий диапазону 0–255.

### 3.3.3 Системные вызовы для обеспечения диалога с пользователем

Вывести строку на экран можно используя системный вызов write под номером 4, поместив значение 4 в регистр eax. Первым аргументом write задаётся дескриптор файла. Для вывода на экран в качестве дескриптора файла нужно указать 1 (стандартный вывод).  
Вторым аргументом задаётся адрес выводимой строки.  
Последним аргументом задается максимальная длина выводимой строки.  
Для ввода строки с клавиатуры можно использовать аналогичный системный вызов read с такими же аргументами, как у вызова write, но дескриптором файла 0 (стандартный ввод).  
Системный вызов exit является обязательным в конце любой программы на языке ассемблер. Для обозначения конца программы перед вызовом инструкции int 80h необходимо поместить в регистр еах значение 1, а в регистр ebx код завершения 0.

# 4 Выполнение лабораторной работы

1. Ввожу команду mc. (рис. 1).

Команда mc

Рис. 1: Команда mc

Открывается Midnight Commander (рис. 2).

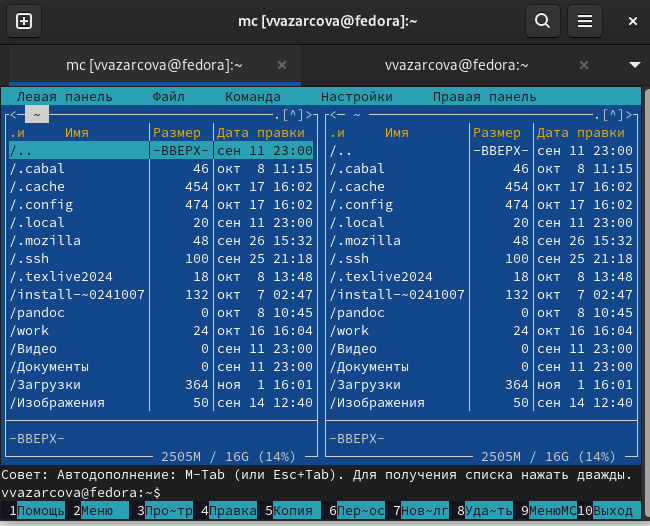


Рис. 2: Интерфейс Midnight Commander

1. Используя клавиши “вверх”, “вниз” и “Enter”, перехожу в каталог ~/work/arch-pc, созданный при выполнении лабораторной работы №4 (рис. 3).

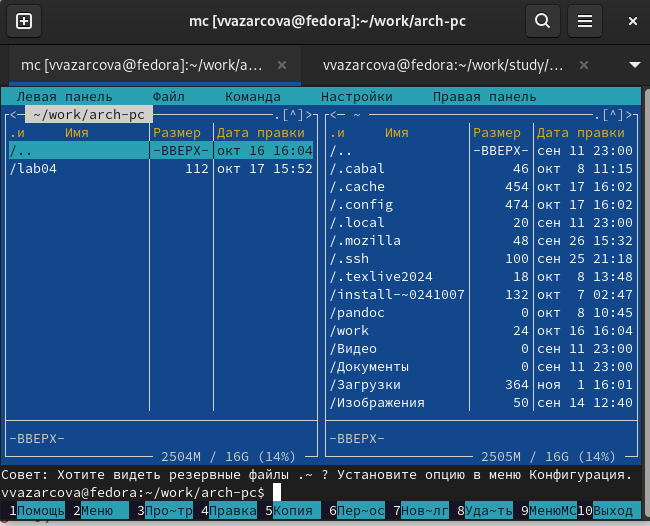


Рис. 3: Каталог arch-pc

1. С помощью функциональной клавиши F7 создаю папку lab05 и перехожу в созданный каталог (рис. 4).

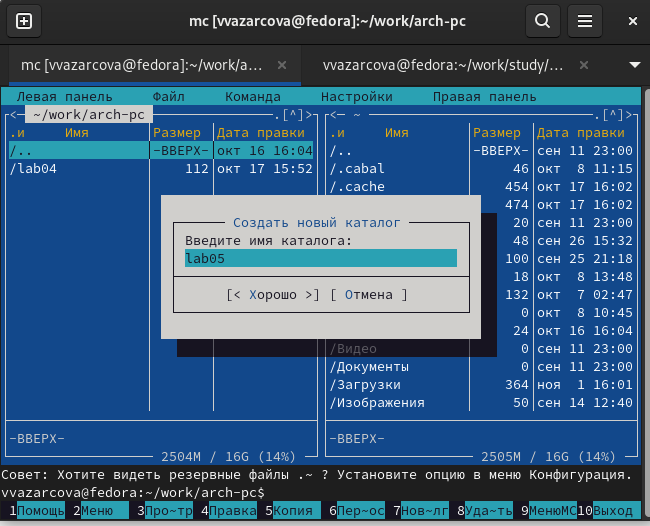


Рис. 4: Создание папки lab05

1. Пользуясь строкой ввода, создаю файл lab5-1.asm (рис. 5).

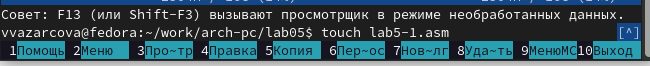


Рис. 5: Создание файла lab5-1.asm

Проверяю, что файл создан. (рис. 6).

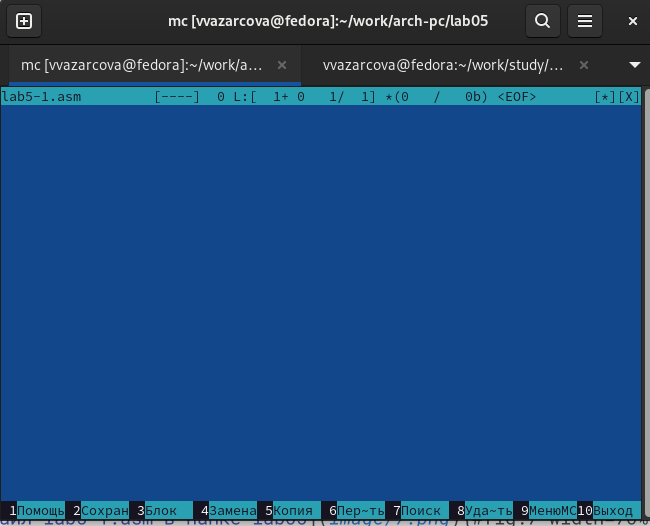


Рис. 6: Файл lab5-1.asm в папке lab05

1. С помощью функциональной клавиши “F4” открываю файл lab5-1.asm для редактирование во встроенном редакторе mcedit. (рис. 7).

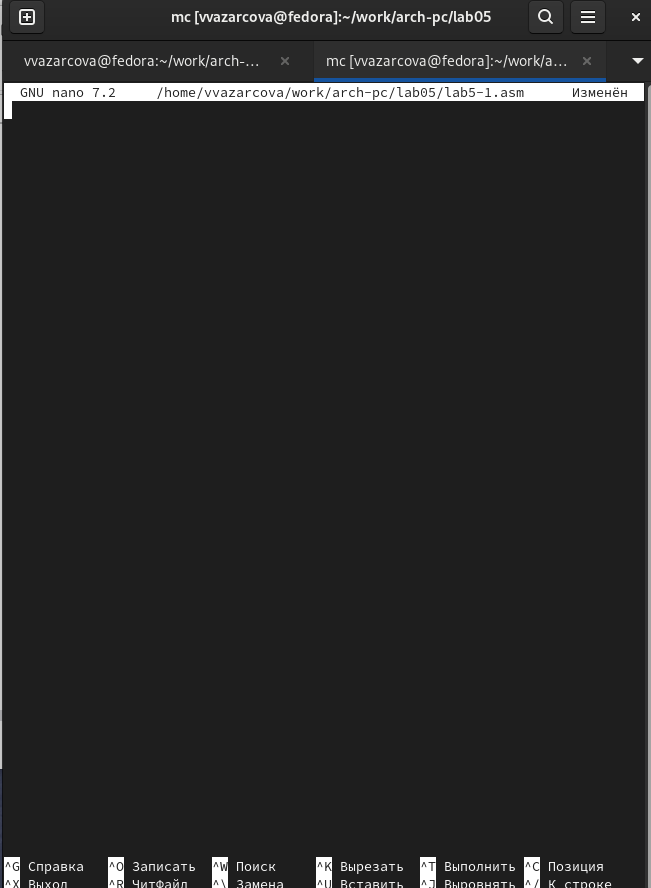


Рис. 7: Редактирование lab5-1.asm в mcedit

1. Ввожу текст программы. (рис. 8).

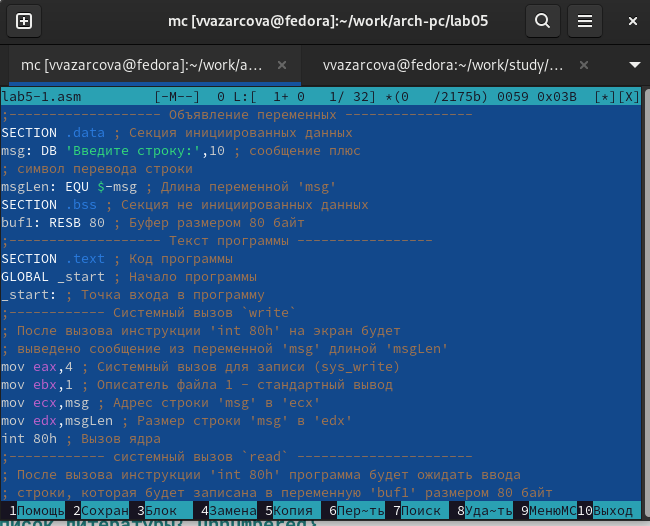


Рис. 8: Ввод текста программы в lab5-1.asm

Сохраняю изменения и закрываю файл (рис. 9).

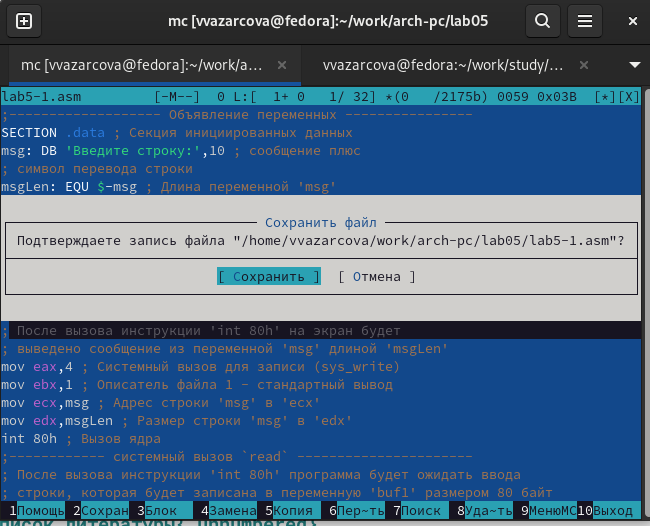


Рис. 9: Сохранение lab5-1.asm в mcedit

1. С помощью функциональной клавиши “F4” открываю файл lab5-1.asm, чтобы проверить, что он содержит файл программы (рис. 10).

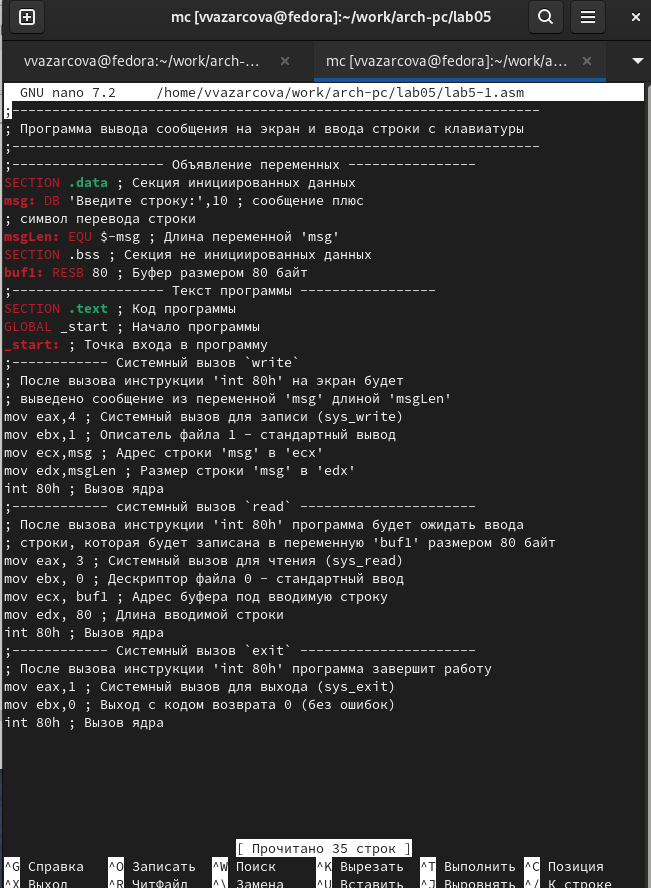


Рис. 10: Содержимое lab5-1.asm в mcedit после сохранения

1. Транслирую текст программы lab5-1.asm в обьектный файл lab5-1.o и проверяю результат с помощью ls. (рис. 11).

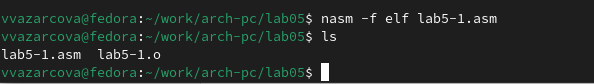


Рис. 11: Транслирование и наличие lab5-1-exe

Выполняю компановку обьектного файла lab5-1.o в исполняемый файл lab5-1\_exe и проверяю результат с помощью ls. (рис. 12).

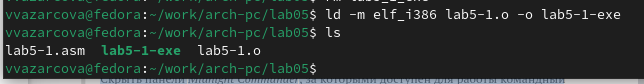


Рис. 12: Компановка и наличие lab5-1-exe

Запускаю получившийся файл. (рис. 13).

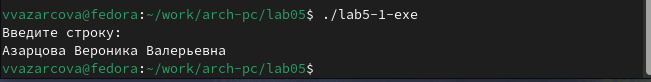


Рис. 13: Запуск lab5-1.asm

Файл работает корректно: выводит приглашение и запрашивает ввод.

## 4.1 Подключение внешнего файла in\_out.asm

1. Скачиваю файл in\_out.asm с курса в ТУИС в каталог Загрузки (рис. 14).

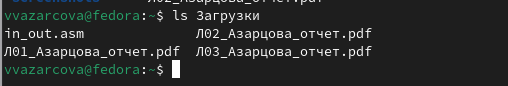


Рис. 14: Файл in\_out.asm в каталоге Загрузки

1. В одной панели mc открываю каталог с файлом lab5-1.asm, в другой каталог со скаченным файлом in\_out.asm. Копирую файл in\_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5 (рис. 15).

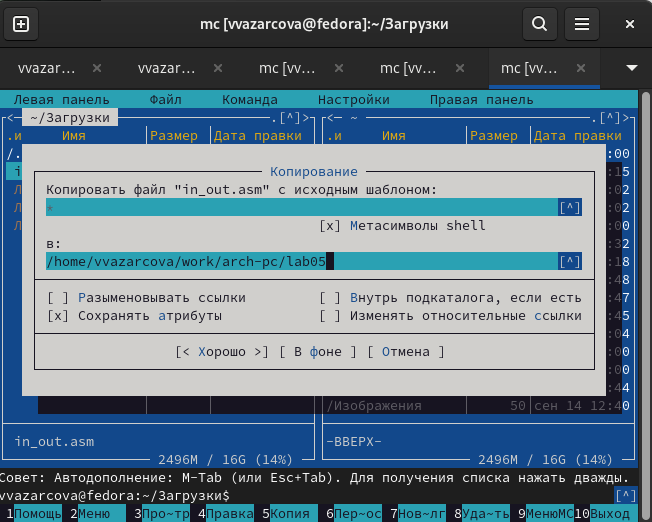


Рис. 15: Копирование 1in\_out.asm в Midnight Commander

Проверяю наличие файла в нужном каталоге (рис. 16).

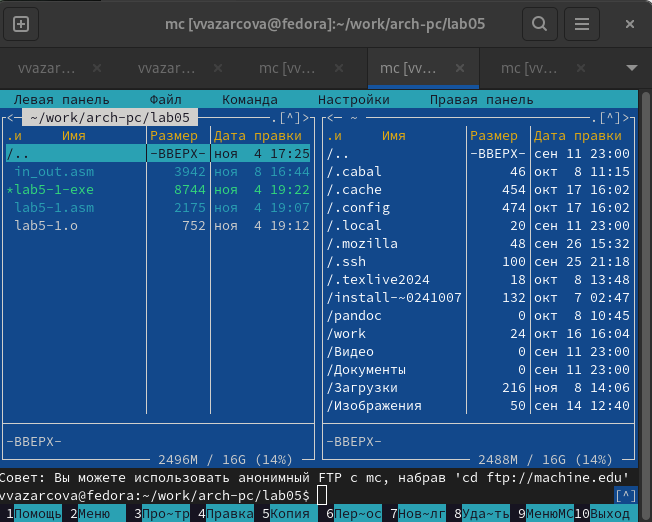


Рис. 16: Файл in\_out.asm в каталоге lab05

1. С помощью функциональной клавиши F6 создаю копию файла lab5-1.asm с именем lab5-2.asm (рис. 17).

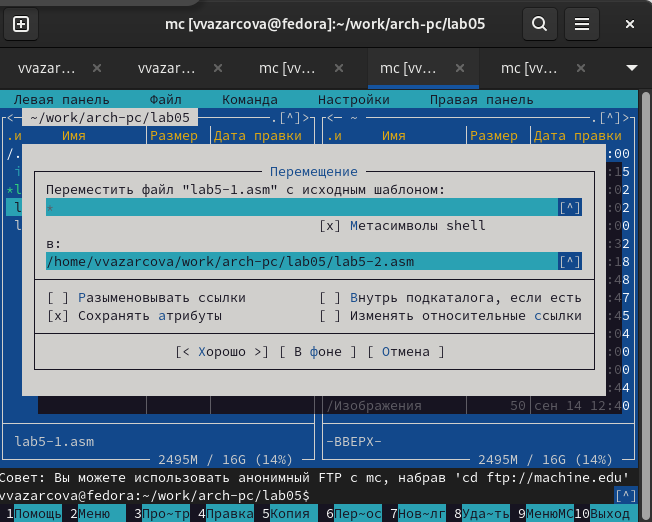


Рис. 17: Копирование файла lab5-1.asm

1. Исправляю текст программы в файле lab5-2.asm с использование подпрограмм из внешнего файла in\_out.asm (sprintLF, sread и quit) (рис. 18).

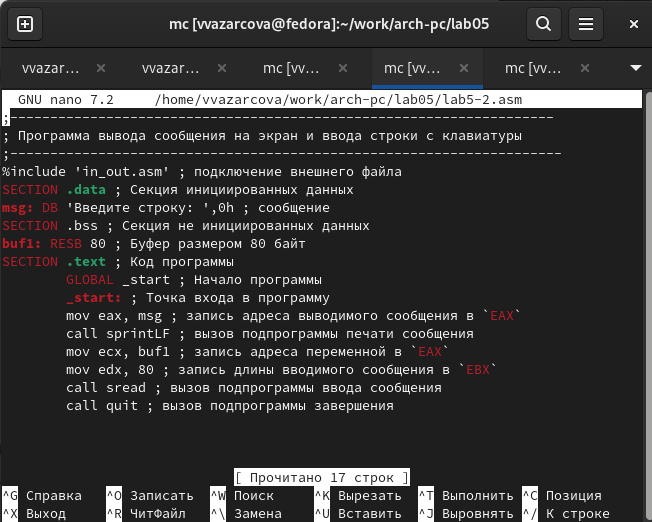


Рис. 18: Исправленный текст программы

Создаю исполняемый файл из lab5-2.asm и проверяю его работу (рис. 19).

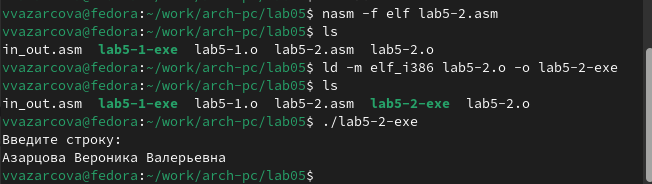


Рис. 19: Запуск файла

1. Заменяю подпрограмму sprintLF на sprint в файле lab5-1.asm (рис. 20).

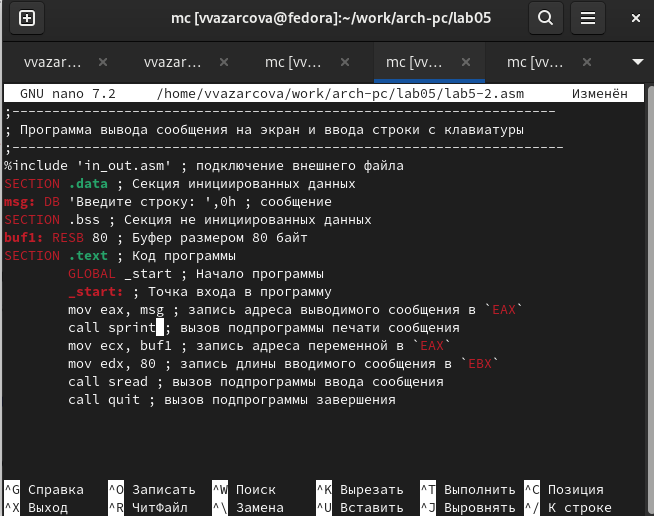


Рис. 20: Исправленный текст программы с sprint

Создаю исполняемый файл и проверяю его работу (рис. 21).

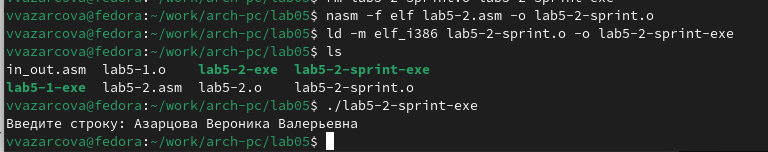


Рис. 21: Запуск файла с sprint

Разница между программой со sprintLF и программой со sprint состоит в том, что первая программа, после вывода запроса, запрашивает ввод на новой строке, в то время как вторая запрашивает ввод на той же строке, где выводит запрос.

# 5 Задания для самостоятельной работы

1. Создаю копию файла lab5-1.asm с помощью mc с названием lab5-3.asm (рис. 22).

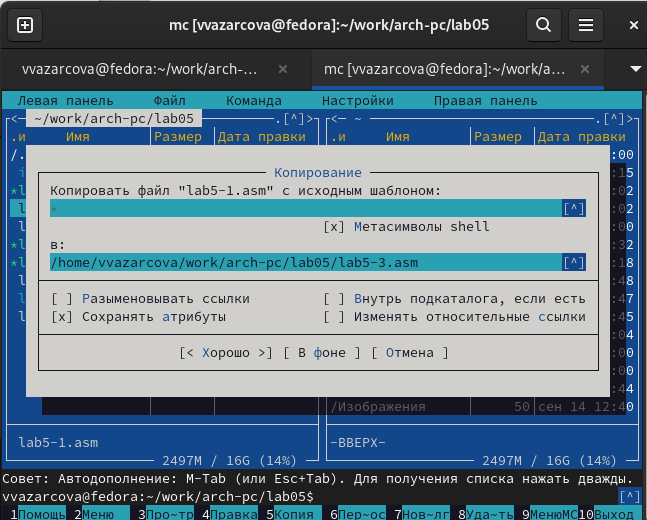


Рис. 22: Копирование lab5-1.asm в lab5-3.asm

Требуется изменить программу так, чтобы она работала по следующему алгоритму:  
Вывести приглашение типа “Введите строку:”;  
Ввести строку с клавиатуры;  
Вывести введённую строку на экран.  
Для этого добавляю в исходную программу переменную buf1Len, в которой будет записана длина строки buf1, введеной пользователем.  
Далее, используя эту переменную , пропишу системный вызов write, выводящий на экран содержимое переменной buf1 (рис. 23).

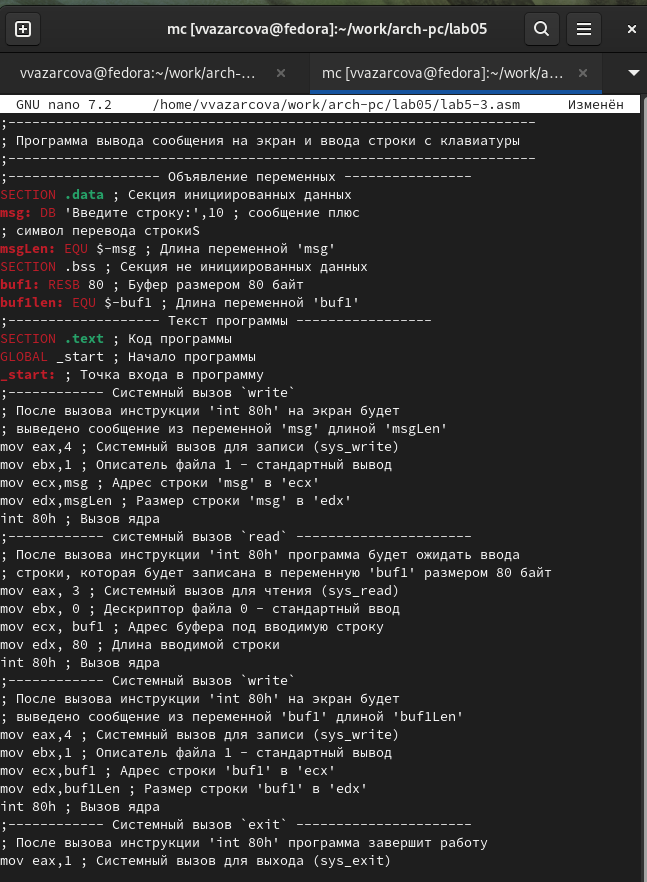


Рис. 23: Измененный текст программы в lab5-3.asm

1. Создаю исполняемый файл lab5-4-exe из lab5-3.asm и проверяю его работу (рис. 24).

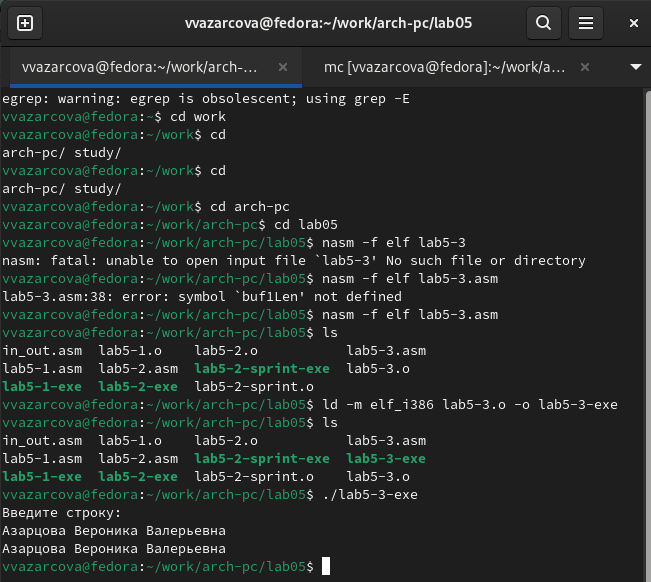


Рис. 24: Создание и запуск исполняемого файла lab5-3-exe

Программа выводит строку “Введите строку:”, запрашивает ввод с клавиатуры, и затем выводит введеную пользователем строку, т.е. ФИО. Значит, программа работает корректно и согласно алгоритму.

1. Создаю копию файла lab5-2.asm с помощью mc с названием lab5-4.asm (рис. 25).

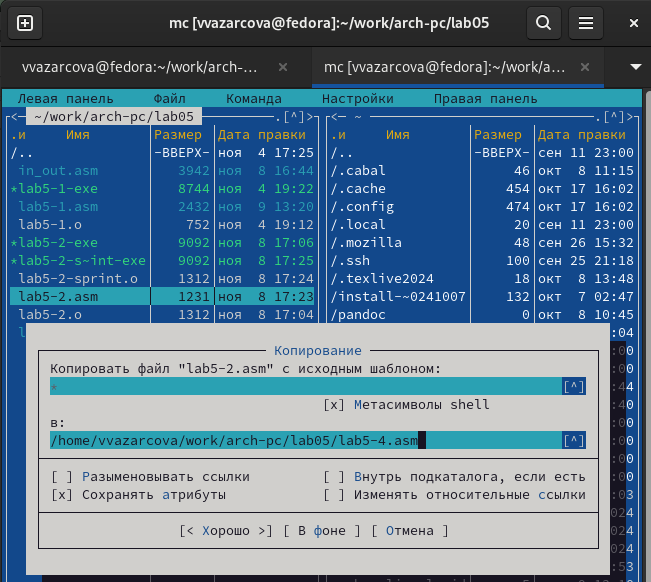


Рис. 25: Копирование lab5-2.asm в lab5-4.asm

Требуется изменить программу аналогично пункту номер 1, то есть:  
Вывести приглашение типа “Введите строку:”;  
Ввести строку с клавиатуры;  
Вывести введённую строку на экран.  
Для этого, во-первых, меняю в исходной программе sprint обратно на sprintLF для более понятного ввода и вывода, т.к. поменяла это во время выполнения пункта номер 13 лабораторной работы.  
Далее, используя подпрограммы из внешнего файла in\_out.asm, прописываю запись адреса введеного пользователем сообщения buf1 в EAX, и вызов подпрограммы печати сообщения sprintLF (рис. 26).

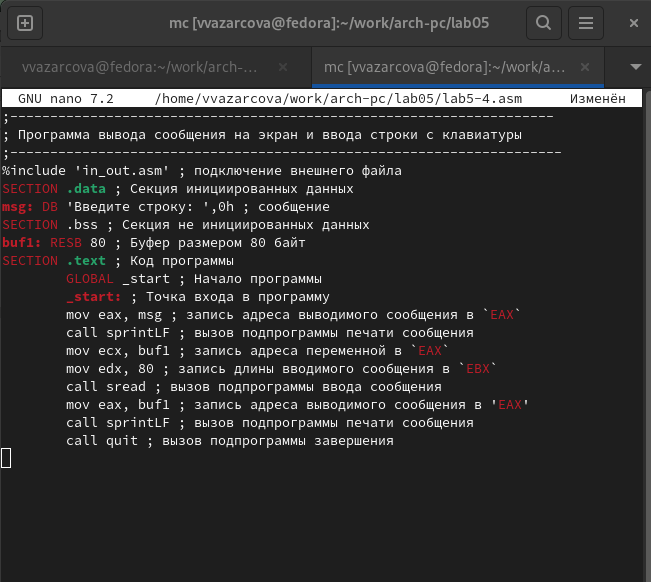


Рис. 26: Измененный текст программы в lab5-4-exe

1. Создаю исполняемый lab5-4-exe файл из lab5-4.asm и проверяю его работу (рис. 27).

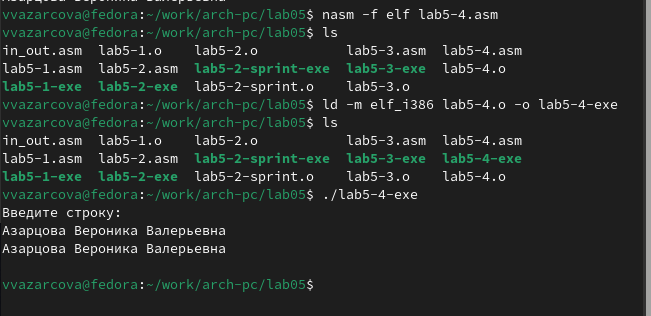


Рис. 27: Создание и запуск исполняемого файла lab5-4-exe

Программа выводит строку “Введите строку:”, запрашивает ввод с клавиатуры, и затем выводит введеную пользователем строку, т.е. ФИО, и при этом использует внешний файл in\_out.asm. Значит, программа работает корректно и согласно алгоритму.

# 6 Выводы

Подводя итоги данной лабораторной работы, я научилась пользоваться Midnight Commander и успешно написала и запустила несколько программ с системными вызовами для обеспечения диалога с пользователем, использовав в нескольких из них внешний файл.

# Список литературы