Лабораторная работа №6

Основы работы Midnight Commander (mc). Структура программы на языке ассемблера NASM. Системные вызовы в OC GNU Linux

Медникова Екатерина Михайловна

Содержание

# 1 Цель работы

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. Для активации оболочки Midnight Commander достаточно ввести в командной строке mc и нажать клавишу Enter. В Midnight Commander используются функциональные клавиши F1 — F10 , к которым привязаны часто выполняемые операции.

Следующие комбинации клавиш облегчают работу с Midnight Commander:

• Tab используется для переключениями между панелями;

• Стрелки вверх и вниз используется для навигации, Enter для входа в каталог или открытия файла (если в файле расширений mc.ext заданы правила связи определённых расширений файлов с инструментами их запуска или обработки);

• Ctrl + u (или через меню Команда > Переставить панели ) меняет местами содержимое правой и левой панелей;

• Ctrl + o (или через меню Команда > Отключить панели ) скрывает или возвращает панели Midnight Commander, за которыми доступен для работы командный интерпретатор оболочки и выводимая туда информация.

• Ctrl + x + d (или через меню Команда > Сравнить каталоги ) позволяет сравнить содержимое каталогов, отображаемых на левой и правой панелях.

Дополнительную информацию о Midnight Commander можно получить по команде man mc и на странице проекта.

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 (учетверённое 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.

Синтаксис директив определения данных следующий:

DB [, ] [, ]

Для объявления неинициированных данных в секции .bss используются директивы resb, resw, resd и другие, которые сообщают ассемблеру, что необходимо зарезервировать заданное количество ячеек памяти.

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.

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 Выполнение лабораторной работы

1. Открыла Midnight Commander. Пользуясь клавишами стрелок вверх, вниз и Enter перешла в каталог ~/work/arch-pc созданный при выполнении лабораторной работы №5.

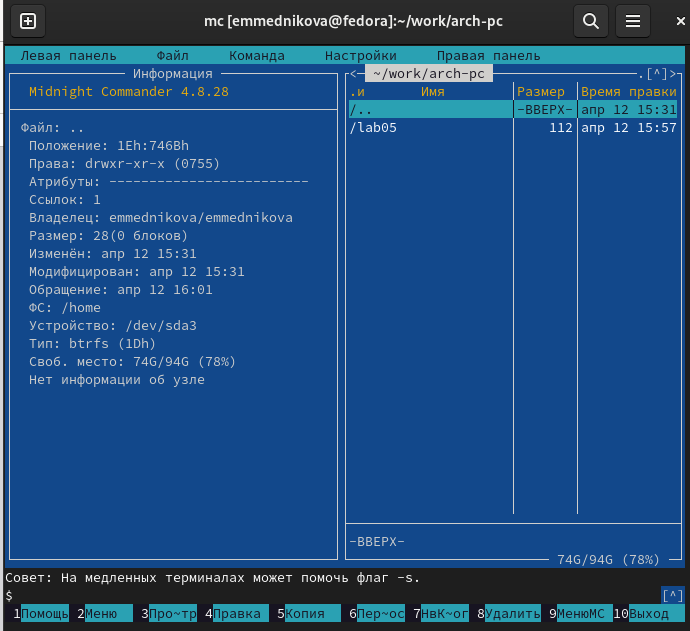


Figure 1: Переход в каталог

1. С помощью функциональной клавиши F7 создала папку lab06 и перешла в созданный каталог.

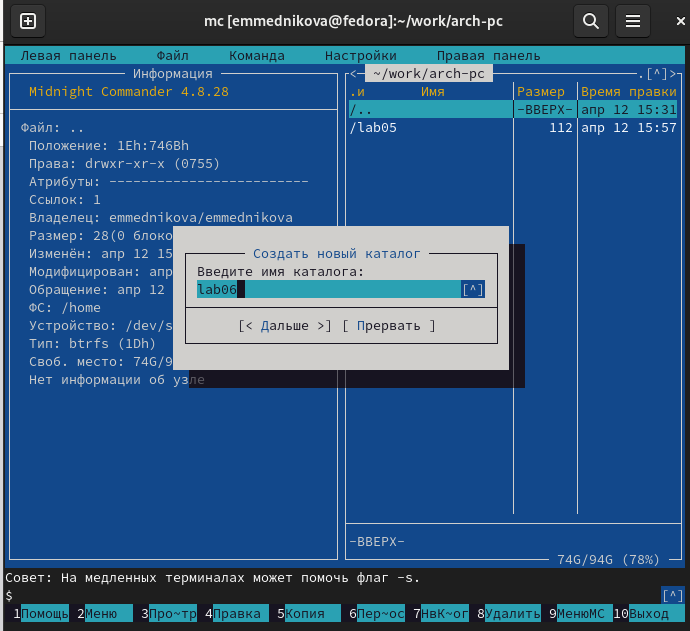


Figure 2: Создание папки

1. Пользуясь строкой ввода и командой touch создала файл lab6-1.asm.

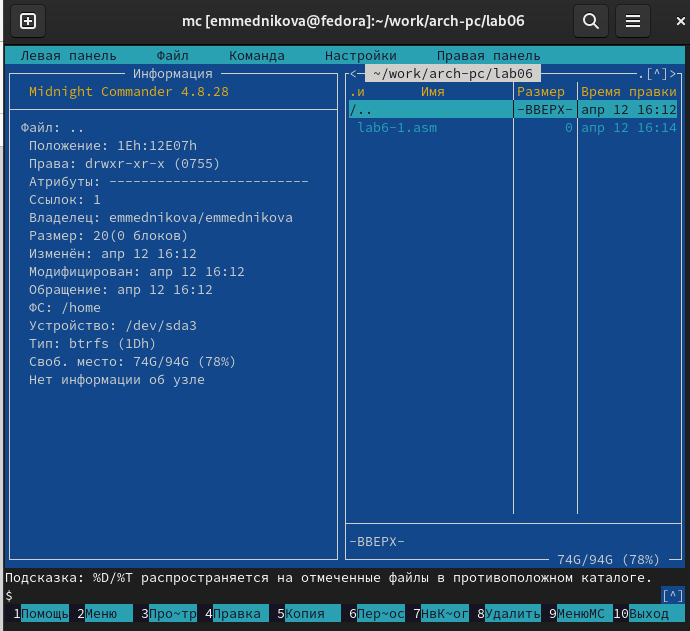


Figure 3: Создание файла

1. С помощью функциональной клавиши F4 открыла файл lab6-1.asm для редактирования во встроенном редакторе.



Figure 4: Встроенный редактор

1. Ввела текст программы из листинга 6.1, сохранила изменения и закрыла файл.

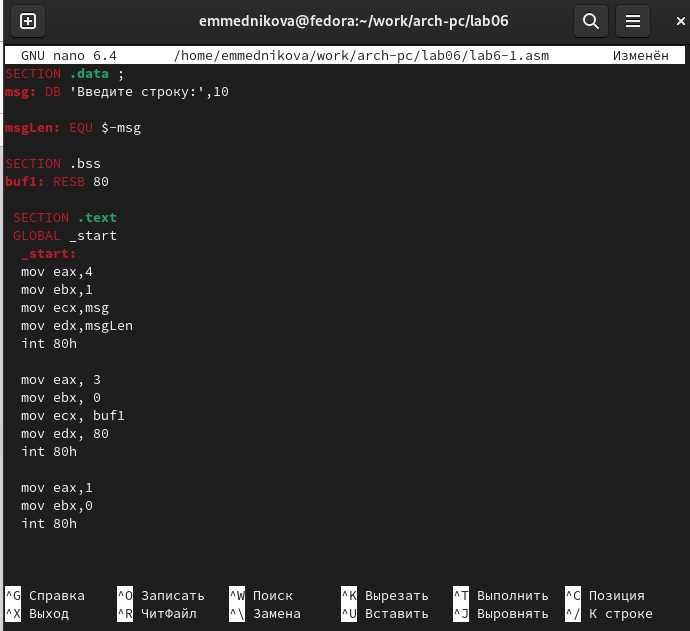


Figure 5: Ввод текста

1. С помощью функциональной клавиши F3 открыла файл lab6-1.asm для просмотра. Убедилась, что файл содержит текст программы.

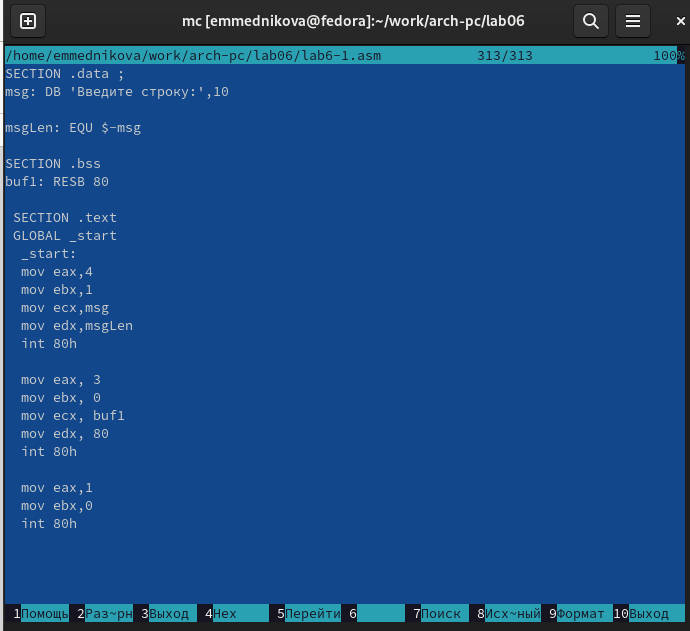


Figure 6: Открытие файла для просмотра

1. Оттранслировала текст программы lab6-1.asm в объектный файл. Выполнила компоновку объектного файла и запустите получившийся исполняемый файл.

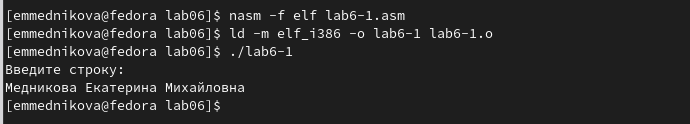


Figure 7: Компоновка файла

1. Скачала файл in\_out.asm со страницы ТУИС.

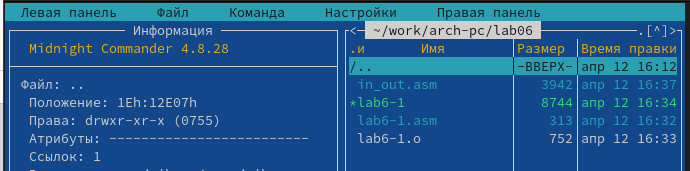


Figure 8: Скачивание файла

1. В одной из панелей mc открыла каталог с файлом lab6-1.asm. В другой панели каталог со скаченным файлом in\_out.asm. Скопировала файл in\_out.asm в каталог с файлом lab6-1.asm с помощью функциональной клавиши F5.

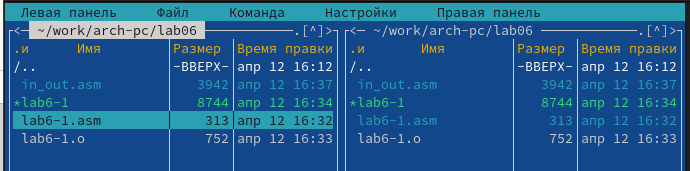


Figure 9: Копирование файла в каталог

1. С помощью функциональной клавиши F6 создала копию файла lab6-1.asm с именем lab6-2.asm. Выделила файл lab6-1.asm, нажала клавишу F6 , ввела имя файла lab6-2.asm и нажала клавишу Enter.

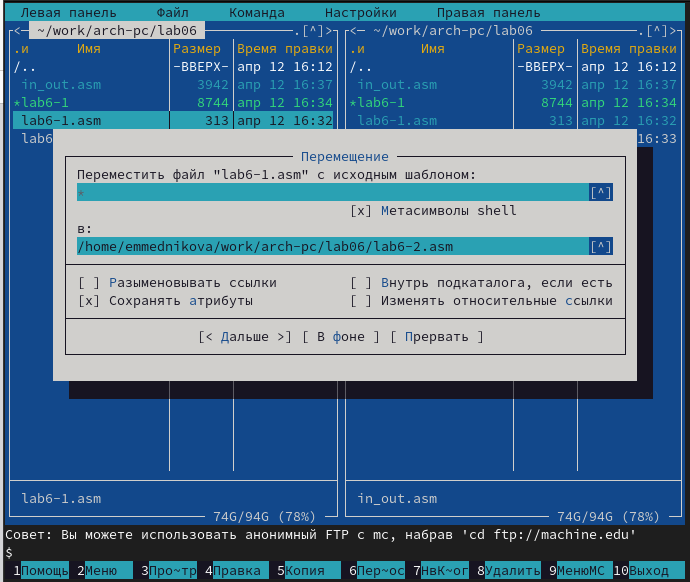


Figure 10: Создание копии файла

1. Исправила текст программы в файле lab6-2.asm с использование подпрограмм из внешнего файла in\_out.asm в соответствии с листингом 6.2. Создала исполняемый файл и проверила его работу.

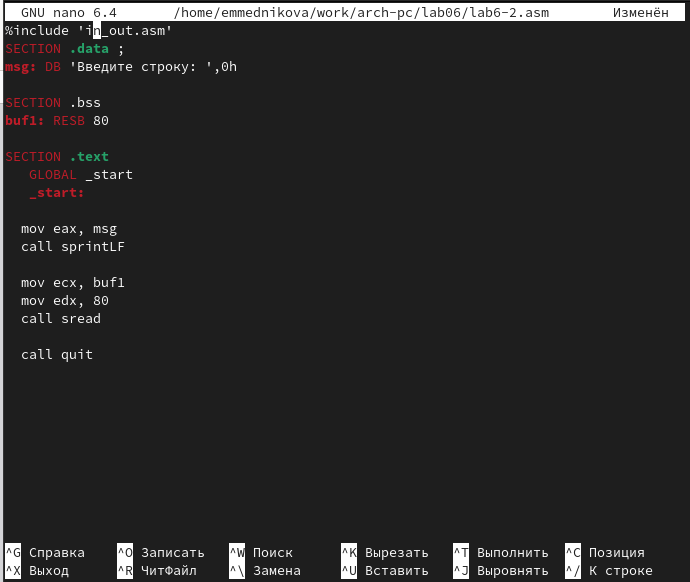


Figure 11: Исправление текста программы

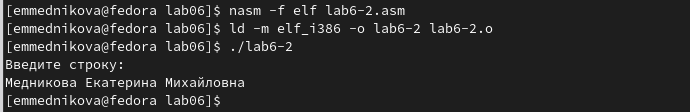


Figure 12: Проверка работы

1. В файле lab6-2.asm заменила подпрограмму sprintLF на sprint. Создала исполняемый файл и проверила его работу.

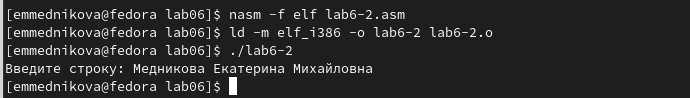


Figure 13: Проверка работы

# 4 Самостоятельная работа

1. Создала копию файла lab6-1.asm. Внесла изменения в программу (без использования внешнего файла in\_out.asm), так чтобы она работала по алгоритму, прописанному в задании.

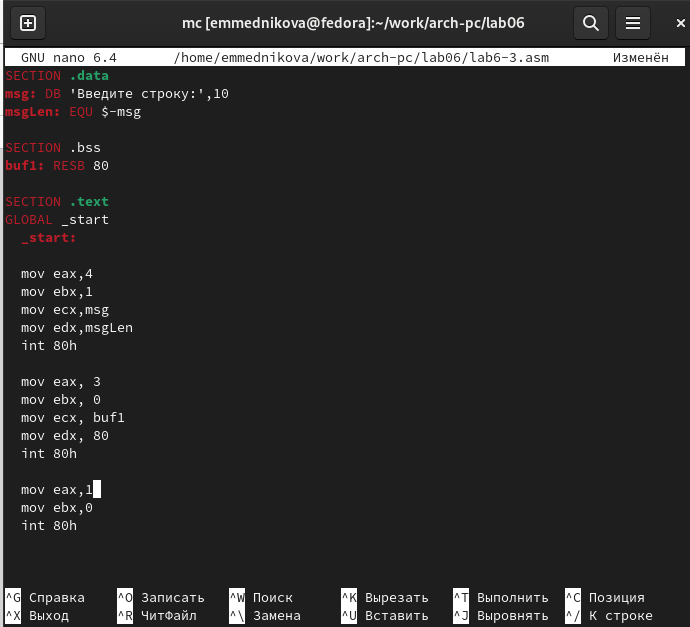


Figure 14: Копия файла и изменения в программе

1. Получила исполняемый файл и проверила его работу. На приглашение ввести строку ввела свои ФИ0.

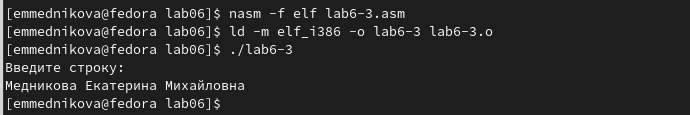


Figure 15: Проверка работы

1. Создала копию файла lab6-2.asm. Исправила текст программы с использованием подпрограмм из внешнего файла in\_out.asm, так чтобы она работала по алгоритму, прописанному в задании.

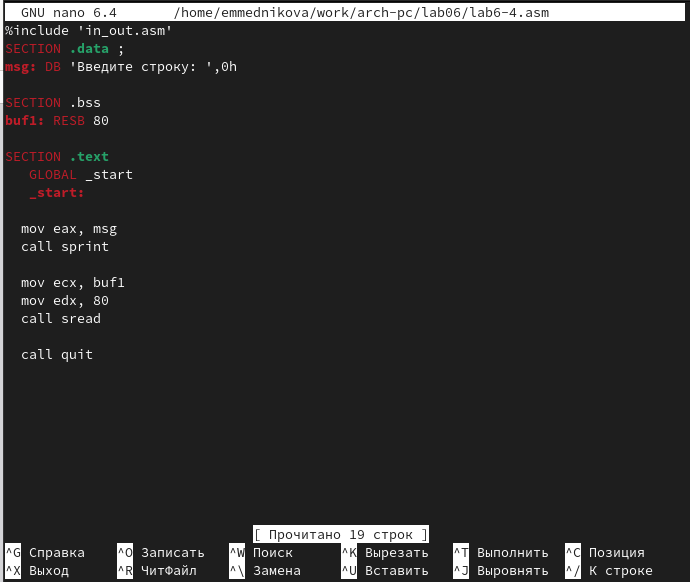


Figure 16: Копия файла и правка в программе

1. Создала исполняемый файл и проверила его работу.

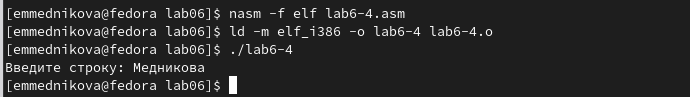


Figure 17: Создание файла и его проверка

# 5 Выводы

Приобрела практические навыки работы в Midnight Commander. Освоила инструкции языка ассемблера mov и int.