Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Канева Екатерина Павловна

Содержание

# 1 Цель работы

Целью работы является приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

# 2 Задание

Создать файлы, выводящие строку “Введите строку:” и выводящие введённую строку на экран, с использованием файла in\_out.asm и без него.

# 3 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управ- лению 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.

Таблица 1: Функциональные клавиши Midnight Commander

| Функциональные клавиши | Выполняемое действие |
| --- | --- |
| 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 |

Следующие комбинации клавиш облегчают работу с Midnight Commander:

* Tab используется для переключениями между панелями;
* и 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.

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:

<имя> DB <операнд> [, <операнд>] [, <операнд>]

# 4 Выполнение лабораторной работы

## 4.1 Лабораторная работа

Откроем Midnight Commander (рис. 1):

mc

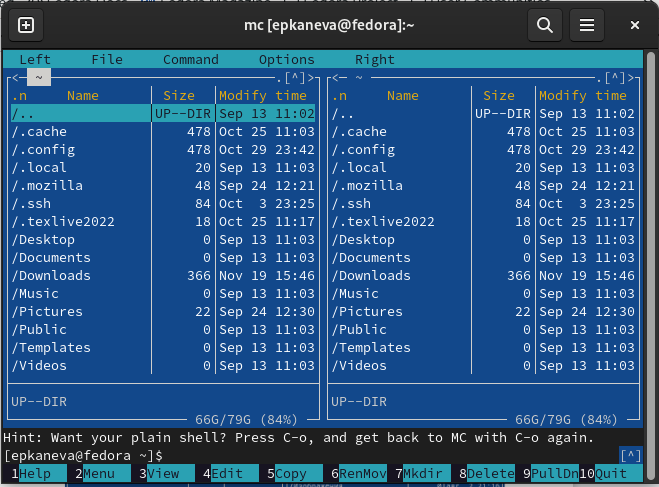


Рис. 1: Отрываем Midnight Commander.

Пользуясь клавишами , и Enter, перейдём в каталог ~/work/study/2022-2023/"Архитектура компьютера"/arh-pc, созданный при выполнении предыдущей лабораторной работы (рис. 2):

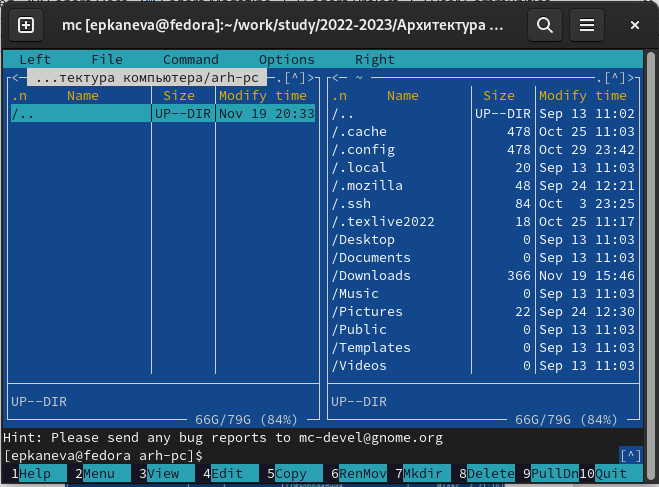


Рис. 2: Переходим в нужный каталог.

С помощью клавиши F7 создадим каталог lab06 и перейдём в него (рис. 3):

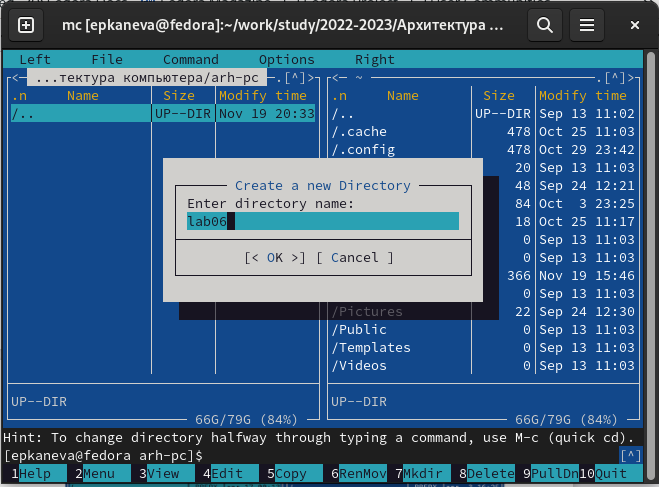


Рис. 3: Создание каталога lab06.

Создадим файл lab6-1.asm (рис. 4):

touch lab6-1.asm

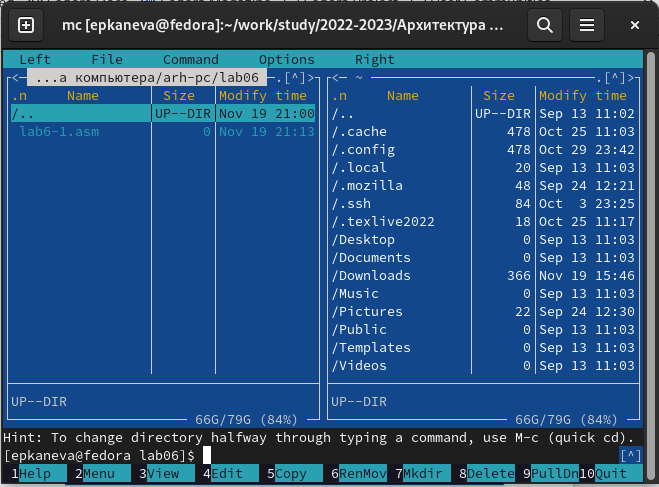


Рис. 4: Создание файла lab6-1.asm

Откроем файл для редактирования с помощью функциональной клавиши F4 и введём в него текст программы, данный в тексте лабораторной работы (рис. 5):

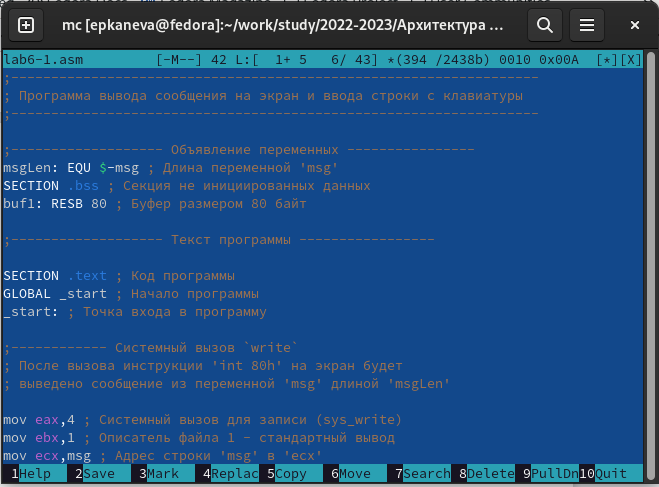


Рис. 5: Ввод текста программы.

Сохраним и закроем файл. Откроем его снова уже с помощью клавиши F3, чтобы проверить, что программа введена - всё действительно корректно.

Теперь оттранслируем текст программы lab6-1.asm в объектный файл, выполним компоновку объектного файла и запустим исполняемый файл (рис. 6):

nasm -f elf lab6-1.asm  
ld -m elf\_i386 -o lab6-1 lab6-1.o  
./lab6-1

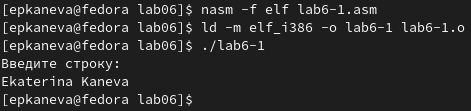


Рис. 6: Трансляция текста программы, компоновка объектного файла, запуск исполняемого файла.

Скачаем файл in\_out.asm из ТУИС, поместим его в один каталог с программами (рис. 7):

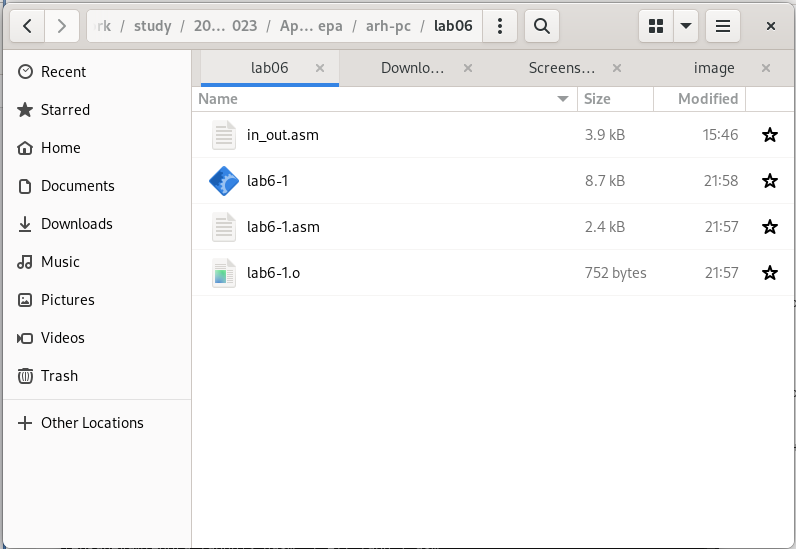


Рис. 7: Файл in\_out.asm в одном каталоге с программами

С помощью функциональной клавиши F6 скопируем файл lab6-1.asm как файл lab6-2.asm (рис. 8):

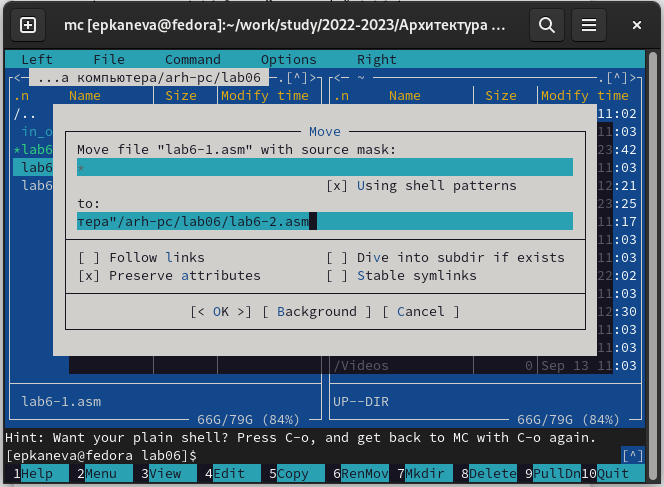


Рис. 8: Копирование файла с изменением имени.

Исправим текст программы в файле lab6-2.asm в соответствии с заданием (рис. 9):

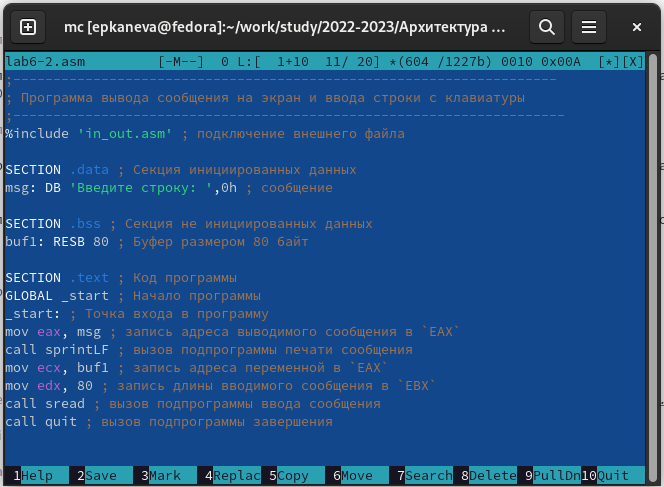


Рис. 9: Редактирование текста программы.

Создадим объектный файл, исполняемый файл, запустим программу (рис. 10):

nasm -f elf lab6-2.asm  
ld -m elf\_i386 -o lab6-2 lab6-2.o  
./lab6-2

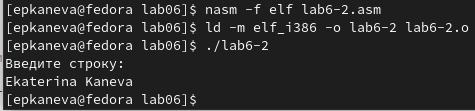


Рис. 10: Запуск программы lab6-2 с sprintLF.

Заменим sprintLF на sprint, создадим объектный и исполняемый файлы, запустим программу (рис. 11):

nasm -f elf lab6-2.asm  
ld -m elf\_i386 -o lab6-2 lab6-2.o  
./lab6-2

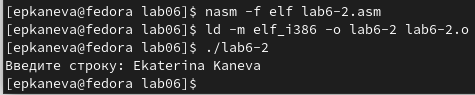


Рис. 11: Запуск программы lab6-2 с sprint.

Видим, что теперь не происходит переход на новую строку для ввода текста.

## 4.2 Самостоятельная работа

Создадим копию файла lab6-1.asm с именем lab6-11.asm (рис. 12):

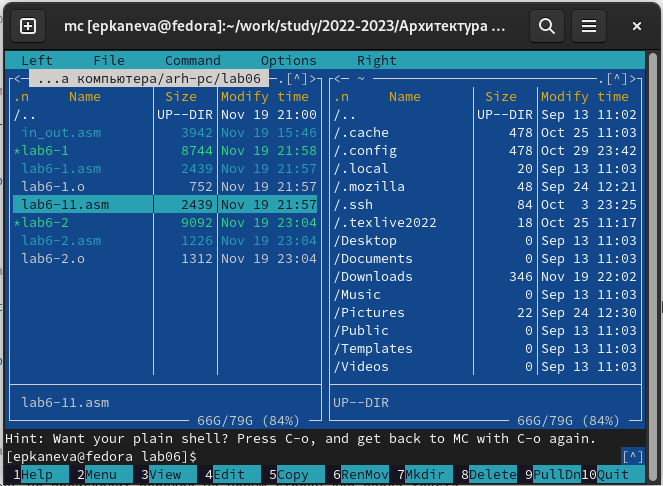


Рис. 12: Копирование файла с изменением имени.

Внесём изменения в программу в соответствии с поставленным заданием (рис. 13):

SECTION .data  
msg: DB 'Введите строку:',10  
  
msgLen: EQU $-msg  
  
SECTION .bss  
buf1 RESB 80  
  
SECTION .txt  
GLOBAL \_start  
\_start:  
  
mov eax,4  
mov ebx,1  
mov ecx,msg  
mov edx,msgLen  
int 80h  
  
mov eax,3  
mov ebx,0  
mov ecx,buf1  
mov edx,80  
int 80h  
  
mov eax,4  
mov ebx,1  
mov ecx,buf1  
mov edx,msgLen  
int 80h  
  
mov eax,1  
mov ebx,0  
int 80h

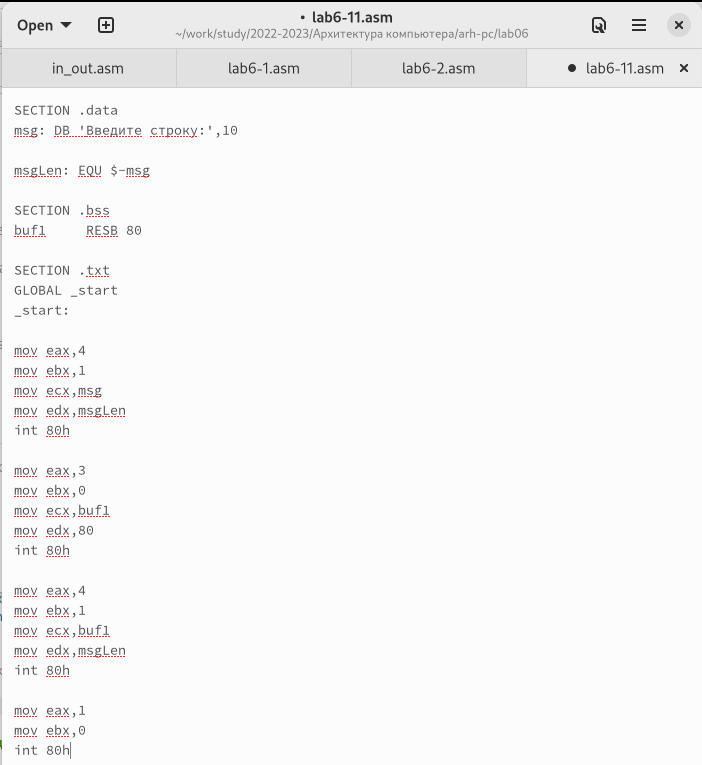


Рис. 13: Редактирование текста программы.

Создадим объектный файл, исполняемый файл, запустим программу (рис. 14):

nasm -f elf lab6-11.asm  
ld -m elf\_i386 -o lab6-11 lab6-11.o  
./lab6-11

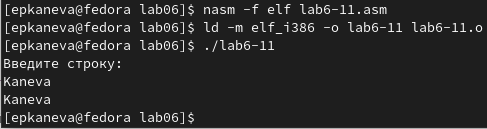


Рис. 14: Запуск программы lab6-11.

Создадим копию файла lab6-2.asm с именем lab6-21.asm (рис. 15):

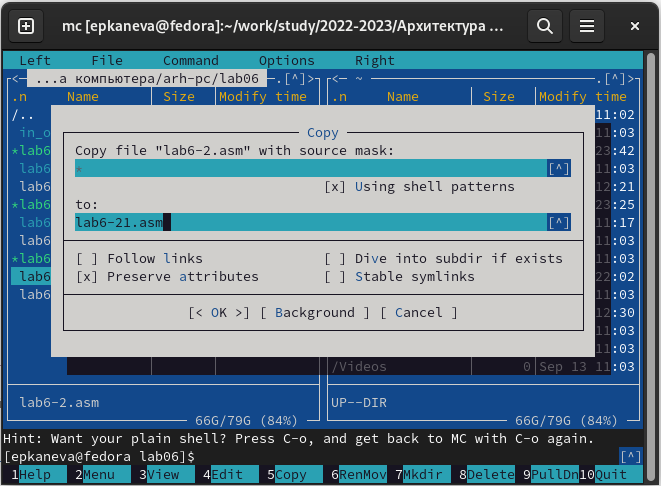


Рис. 15: Копирование файла с изменением имени.

Изменим код программы, чтобы он работал в соответствии с заданием (рис. 16):

%include 'in\_out.asm'  
  
SECTION .data  
msg: DB 'Введите строку:',0h  
  
SECTION .bss  
buf1: RESB 80  
  
SECTION .text  
GLOBAL \_start  
\_start:  
  
mov eax, msg  
call sprintLF  
  
mov ecx, buf1  
mov edx, 80  
call sread  
  
mov eax, buf1  
call sprint  
  
call quit

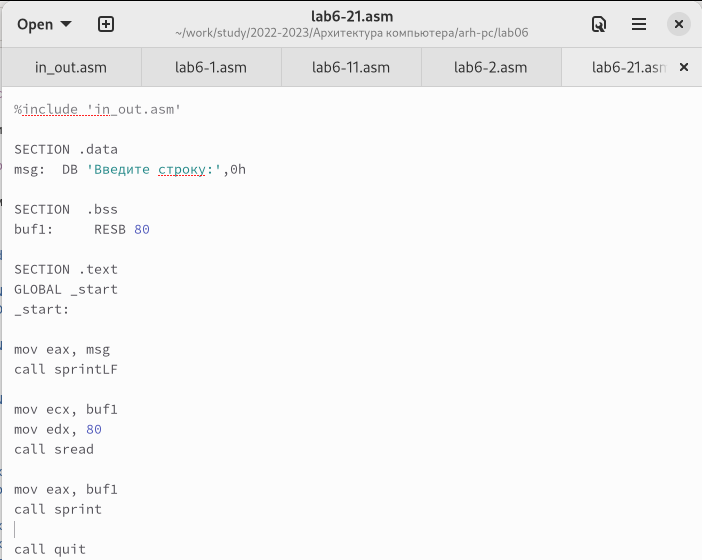


Рис. 16: Редактирование текста программы.

Создадим объектный файл, исполняемый файл, запустим программу (рис. 17):

nasm -f elf lab6-21.asm  
ld -m elf\_i386 -o lab6-21 lab6-21.o  
./lab6-21

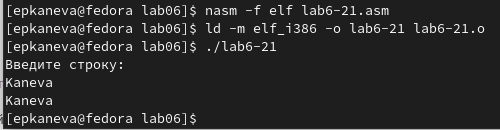


Рис. 17: Запуск программы lab6-21.

Как видим, программа работает корректно.

Созданные файлы \*.asm перенесём в каталог с отчётом, файлы загрузим на GitHub.

# 5 Выводы

Приобретели практические навыки работы в Midnight Commander. Освоили инструкции языка ассемблера mov и int.

# Список литературы