Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Астраханцева Анастасия Александровна

Содержание

# 1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

# 2 Выполнение лабораторной работы

Для начала открою Midnight Commander с помощью команды mc(рис. 1):

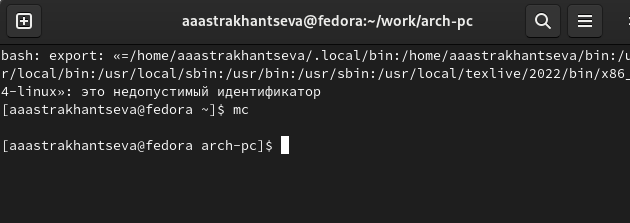


Рис. 1: Открытие Midnight Commander

Далее с помощью клавиши F7 создаю папку lab06 и перехожу в нее (рис. 2):

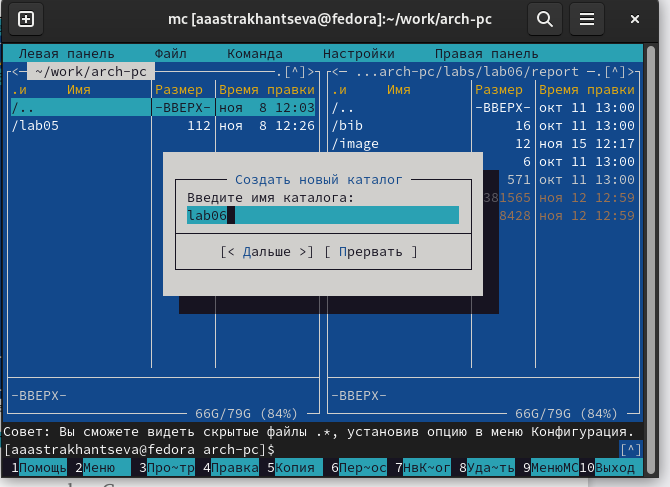


Рис. 2: Создание папки lab06

После этого, пользуясь строкой ввода и командой touch создаю файл lab06-1.asm (рис. 3):

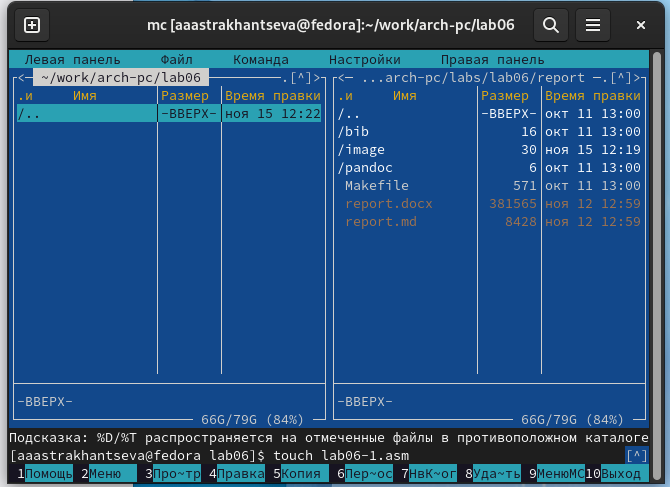


Рис. 3: Создание lab06-1.asm

Далее с помощью клавиши F4 открываю созданный файл для редактирование во встроенном редакторе, записываю текст программы, сохраняю его (рис. 4):

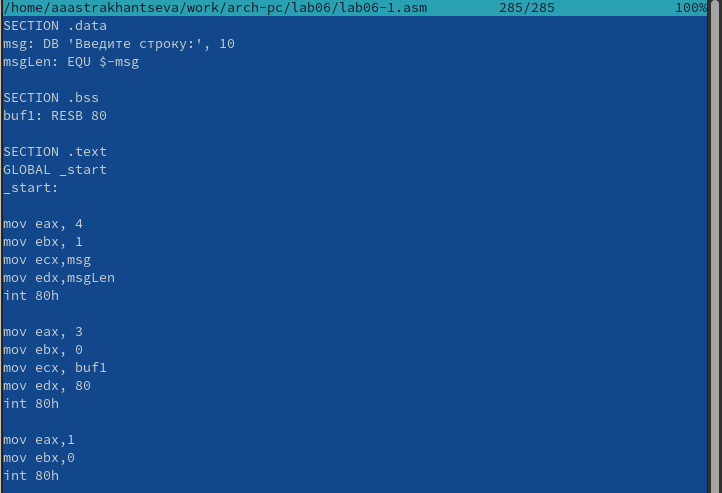


Рис. 4: Запись текста программы

Код программы 1:

;------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;------------------------------------------------------------------  
;------------------- Объявление переменных ----------------  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку:',10 ; сообщение плюс  
; символ перевода строки  
msgLen: EQU $-msg ; Длина переменной 'msg'  
SECTION .bss ; Секция не инициированных данных  
buf1: RESB 80 ; Буфер размером 80 байт  
;------------------- Текст программы -----------------  
SECTION .text ; Код программы  
GLOBAL \_start ; Начало программы  
\_start: ; Точка входа в программу  
;------------ Cистемный вызов `write`  
; После вызова инструкции 'int 80h' на экран будет  
; выведено сообщение из переменной 'msg' длиной 'msgLen'  
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  
;------------ системный вызов `read` ----------------------  
; После вызова инструкции 'int 80h' программа будет ожидать ввода  
; строки, которая будет записана в переменную 'buf1' размером 80  
байт↪  
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

После этого нужно создать объектный файл lab06-1.о. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл с помощью команд (рис. 5):

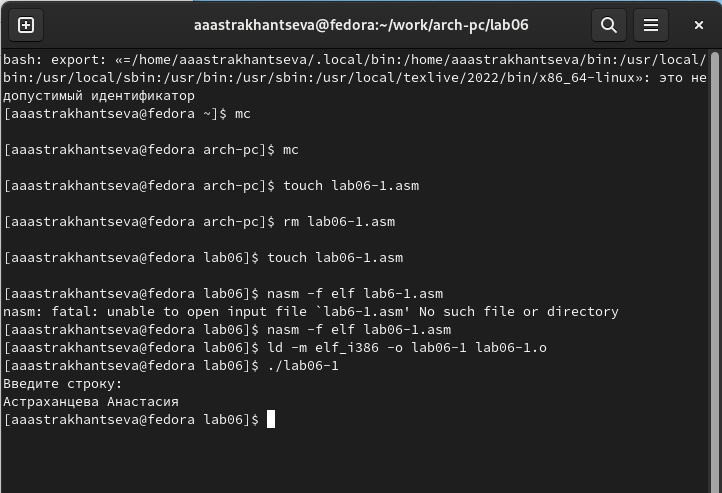


Рис. 5: Комановка и запуск lab06-1.asm

## 2.1 Подключение внешнего файла in\_out.asm

Для упрощения написания программ часто встречающиеся одинаковые участки кода (такие как, например, вывод строки на экран или выход их программы) можно оформить в виде подпрограмм и сохранить в отдельные файлы, а во всех нужных местах поставить вызов нужной подпрограммы. Это позволяет сделать основную программу более удобной для написания и чтения.

Далее необходим скачать файл in\_out.asm со страницы курса в ТУИС. Подключаемый файл in\_out.asm должен лежать в том же каталоге, что и файл с программой, в которой он используется, поэтому с помощью клавиши F5 я копирую скачанный файл в папку lab06 (рис. 6):

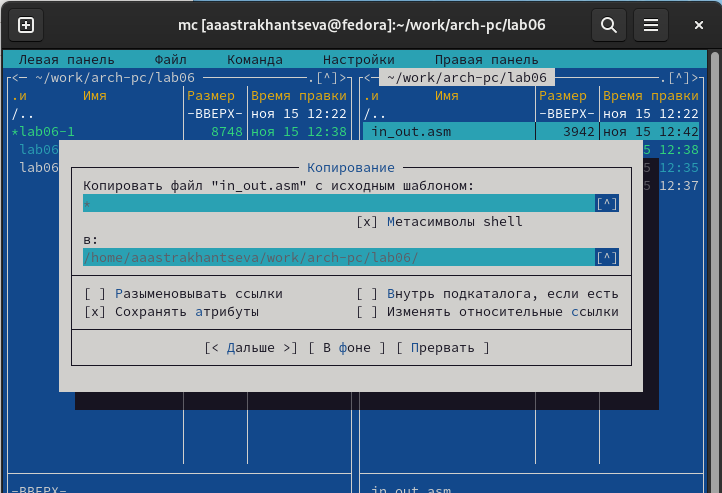


Рис. 6: Перенос in\_out.asm

С помощью клавиши F6 создаю копию файла lab06-1 с именем lab06-2 (рис. 7):

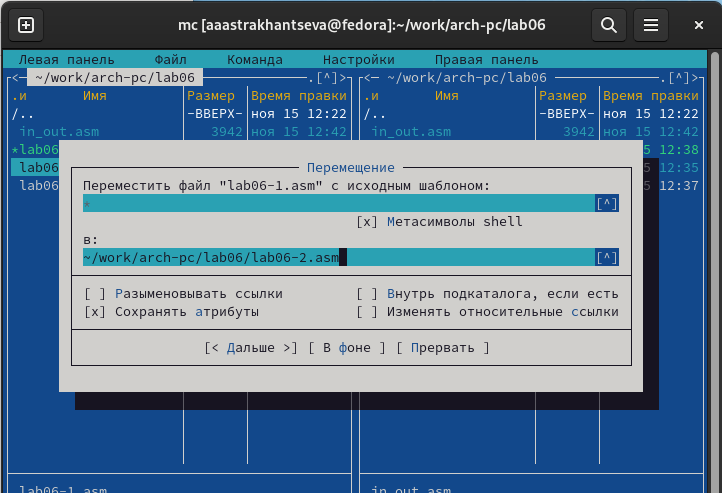


Рис. 7: Создание lab06-2

После этого я исправляю текст программы в файле lab06-2(рис. 8):

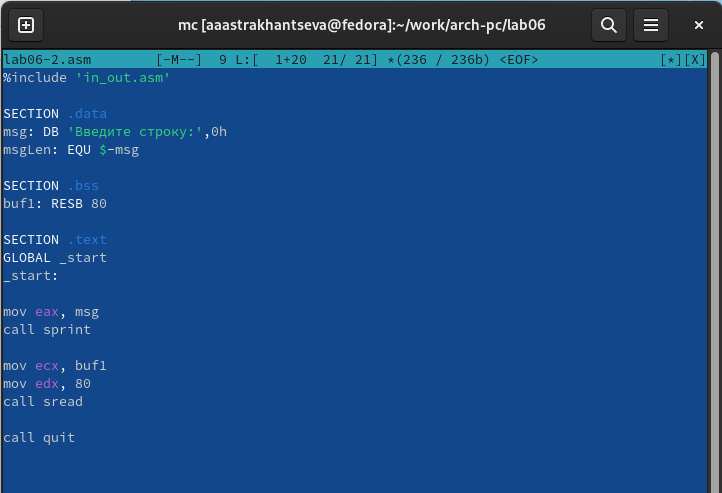


Рис. 8: Правка текста программы

Код программы 2:

;--------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;---------------------------------------------------------------------  
%include 'in\_out.asm' ; подключение внешнего файла  
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 sprintLF ; 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  
call quit ; вызов подпрограммы завершения

После этого нужно оттранслировать текст программы lab06-2.asm в объектный файл. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл с помощью команд: (рис. 9):

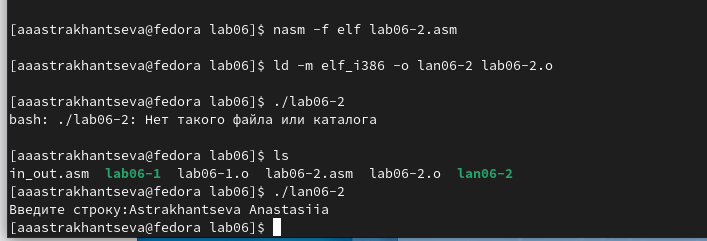


Рис. 9: Запуск lab06-2.asm

## 2.2 Задания для самостоятельной работы

Для начала создаю копию файла lab06-1.asm с именем lab06-1-1.asm (рис. 10):

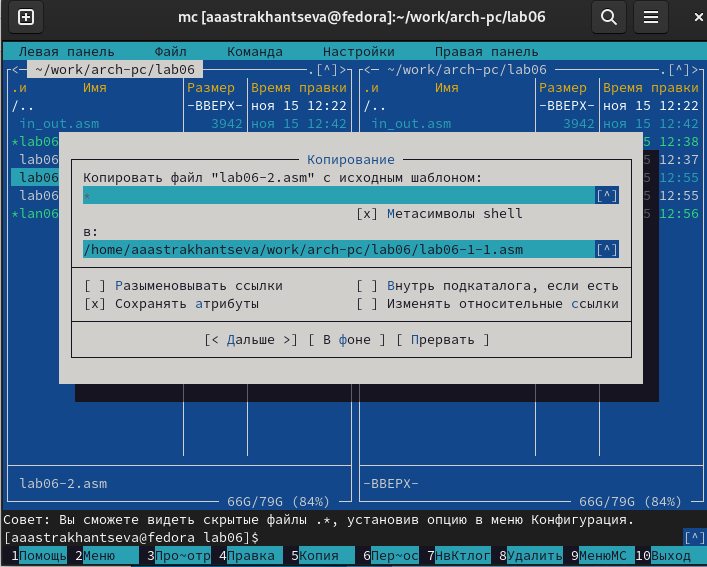


Рис. 10: Запуск lab06-2.asm

После этого вношу изменения в программу (рис. 11):

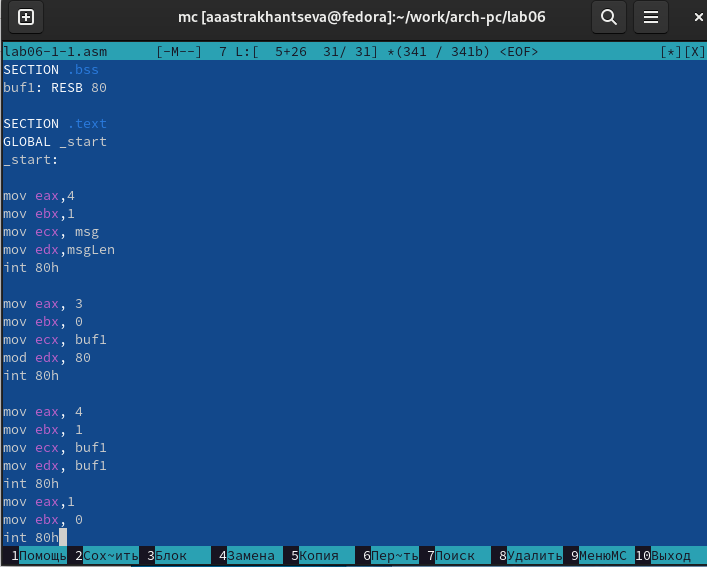


Рис. 11: Изменение lad06-1.asm

Код программы 3:

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

После этого нужно создать объектный файл lab06-1-1.о. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл с помощью команд (рис. 12):

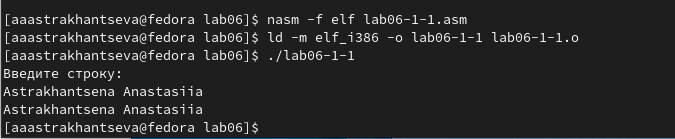


Рис. 12: Изменение lad06-1.asm

Далее копирую файл lab06-2.asm c именем lab06-2-1.asm (рис. 13):

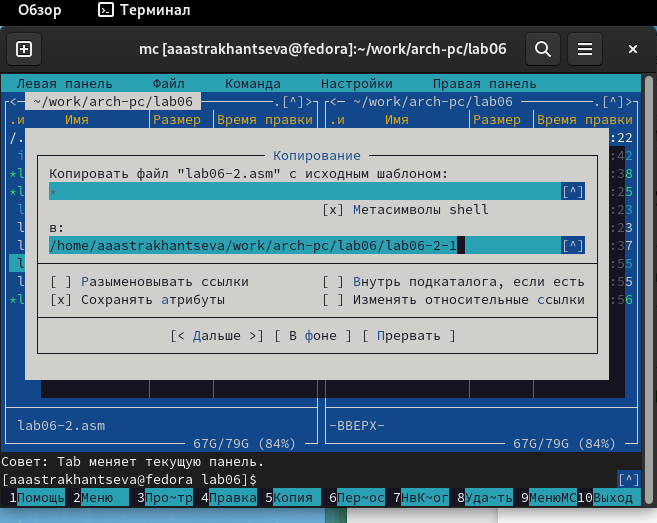


Рис. 13: Создание lab06-2-1.asm

Вношу изменения в программу, чтобы она выводила сторчку на экран (рис. 14):

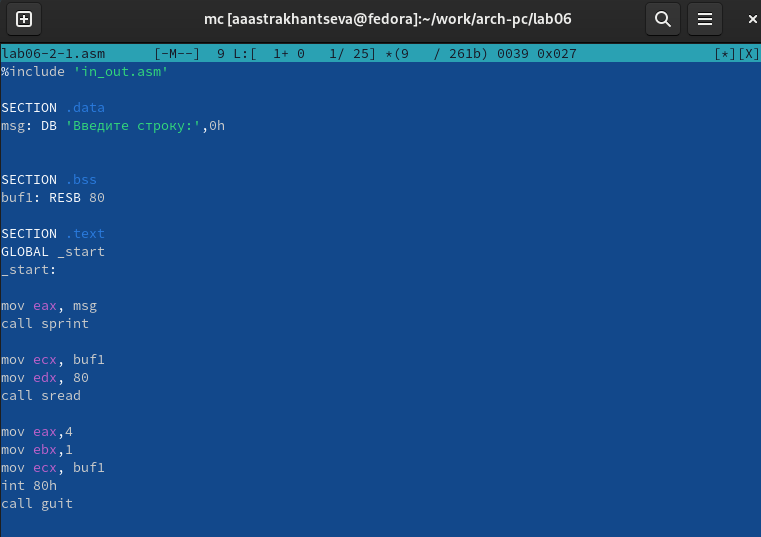


Рис. 14: Изменения в lab06-2-1.asm

Код программы 4:

%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

После этого нужно создать объектный файл lab06-2-1.о. Выполняю компоновку объектного файла и запускаю получившийся исполняемый файл с помощью команд (рис. 15):

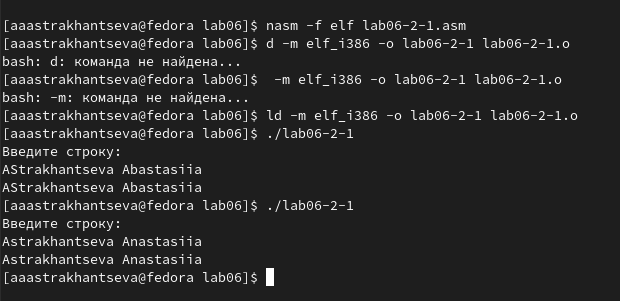


Рис. 15: Изменения в lab06-2-1.asm

# 3 Выводы

Я приобрела практические навыки работы в Midnight Commander и освоила инструкции языка ассемблера mov и int.