

Отчёт по лабораторной работе №5

Выполнил студент НКАбд-02-25

Арина Андреевна Дрекина

Содержание

1 Цель работы	3
2 Порядок выполнения лабораторной работы	4
3 Подключение внешнего файла in_out.asm	9
4 Задание для самостоятельной работы	13
5 Вывод	16

1 Цель работы

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

2 Порядок выполнения лабораторной работы

Для начала с помощью команды «mc» я открыла Midnight Commander.(Рисунок 2.1 и Рисунок 2.2)

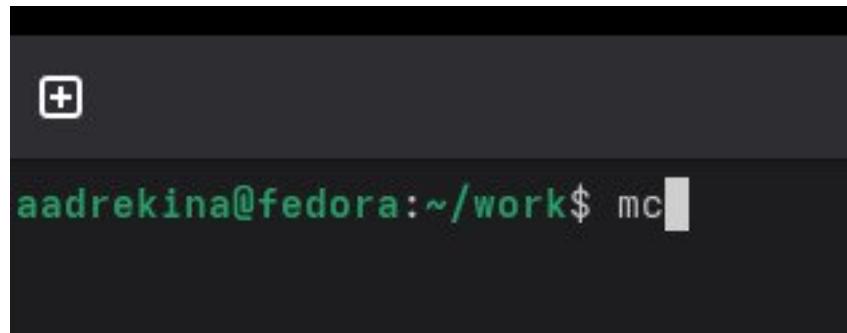


Рисунок 2.1: Выполнение команды mc.

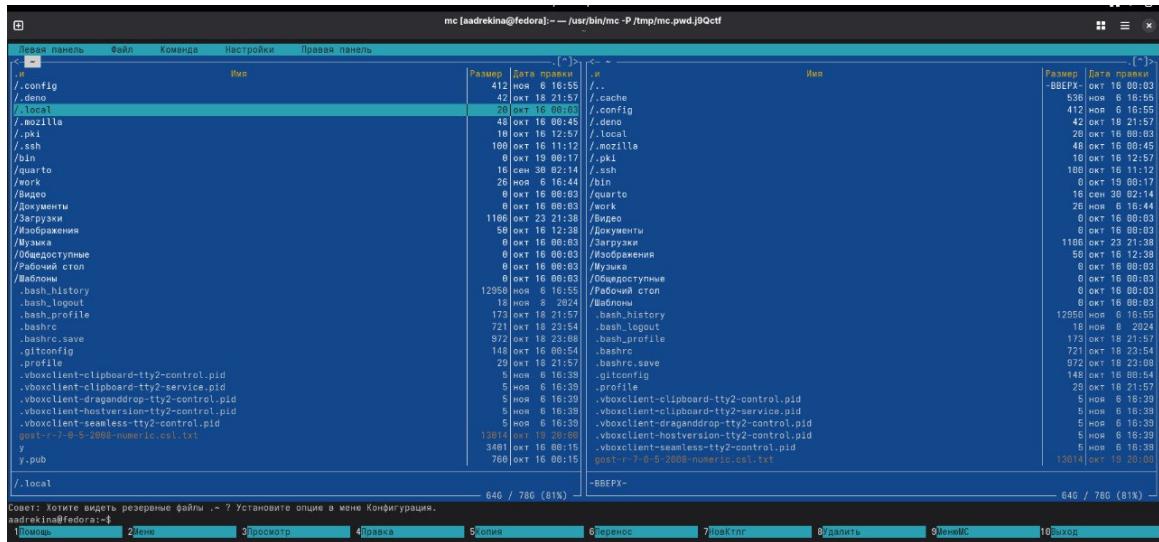


Рисунок 2.2: Открытие Midnight Commander.

Далее с помощью нужных клавиш я перешла в каталог «~/work/arch-pc»(Рисунок 2.3)

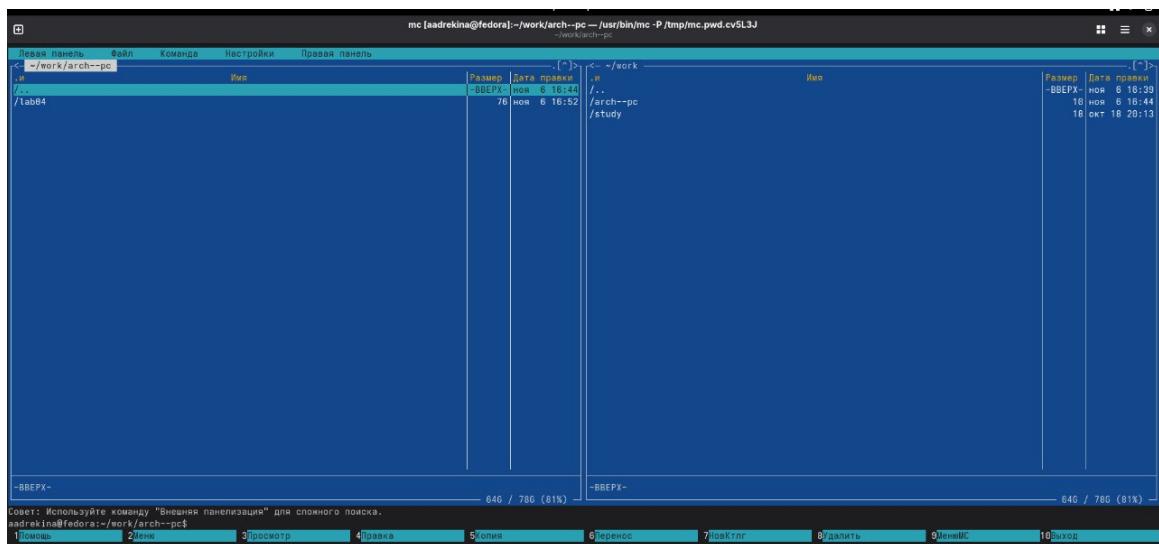


Рисунок 2.3: Переход в нужный каталог для дальнейшей работы.

Далее с помощью клавиши F7 я создала папку «lab05»(Рисунок 2.4)

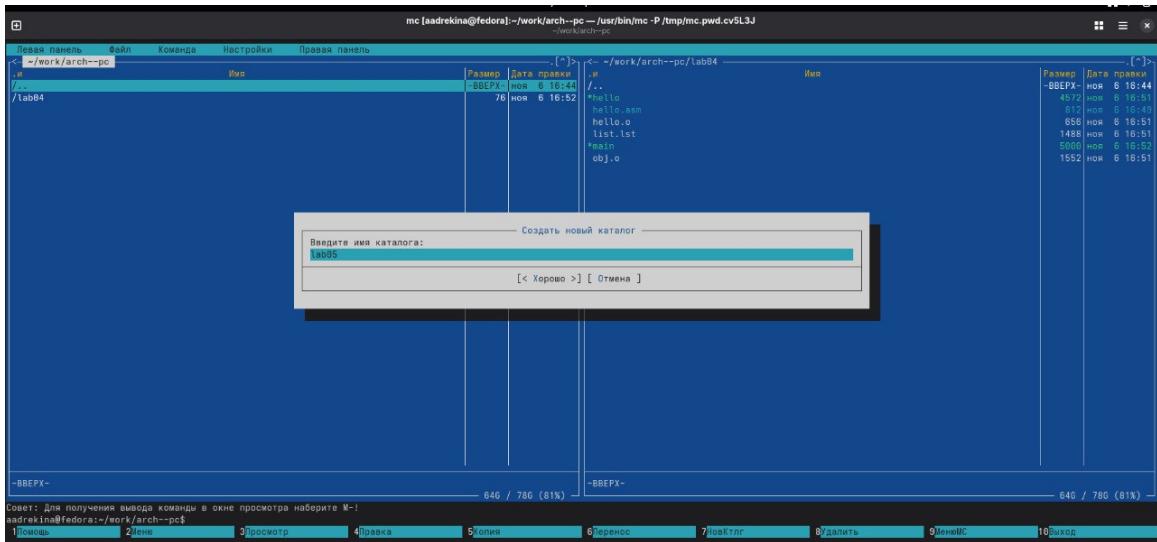


Рисунок 2.4: Создание папки «lab05».

Далее я вошла в созданную папку и с помощью команды touch создала текстовый файл «lab5-1.asm» (Рисунок 2.5)

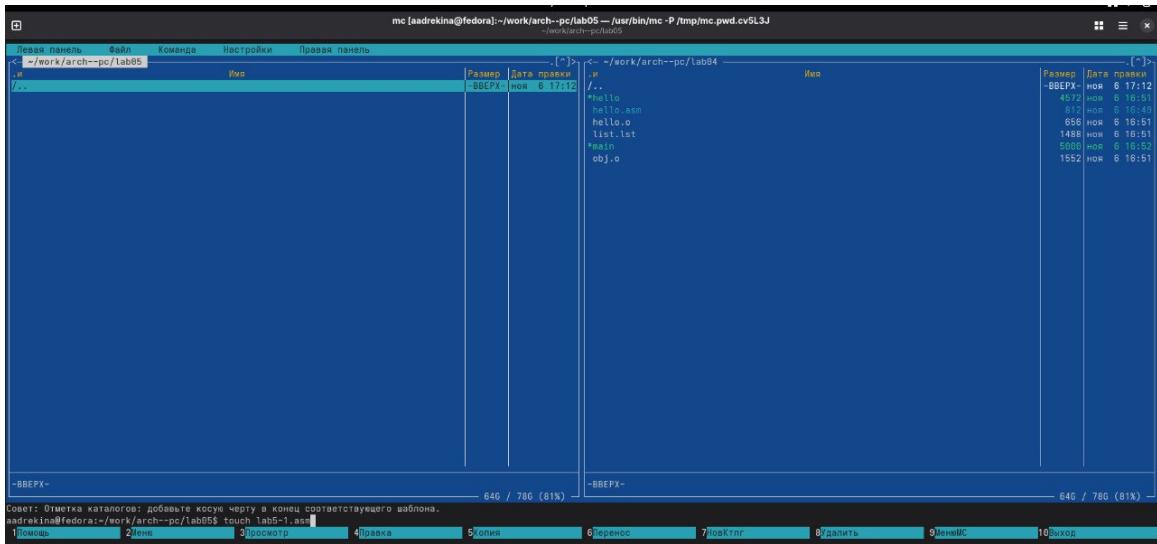


Рисунок 2.5: Создание текстового файла «lab5-1.asm».

Далее с помощью клавиши F4 я открыла созданный файл и вставила туда Листинг 5.1, который взяла из лекции.(Рисунок 2.6)

```

mc [aadrekina@fedora]:~/work/arch--pc/lab05 -- /usr/bin/mc -P /tmp/mc.pwd.cv5L3J
[...]
lab5-1.asm [ -H -] 26 L: 1+34 35/ 35 *(2431/2431b) <EOF>
[...]
    Программа вывода сообщения на экран и ввода строки с клавиатуры
    ..... объявление переменных .....
SECTION data ; Секция инициализированных данных
msg: DB '$->' ; Введите строку: ,10 ; сообщение для msg
SECTION bss ; Секция не инициализированных данных
buf1: RESB 80 ; буфер размером 80 байт
buf2: RESB 80 ; буфер размером 80 байт
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
        ; Системный 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, 1 ; Описатель файла 0 - стандартный ввод
        ; mov ecx, buf1 ; Адрес буфера под вводимую строку
        ; mov edx, 80 ; Длина вводимой строки
        int 80h ; Вызов ядра
        ; Системный вызов 'exit'
        ; После вызова инструкции 'int 80h' программа завершит работу
        ; mov eax, 1 ; Системный вызов для выхода (sys_exit)
        ; mov ebx, 0 ; Выход с кодом возврата 0 (без ошибок)
        int 80h ; Вызов ядра
[...]

```

Рисунок 2.6: Ввод текста из Листинг 5.1 в файл.

После этого с помощью клавиши F2 я сохранила изменения и после этого закрыла файл с помощью клавиши F10. (Рисунок 2.7)

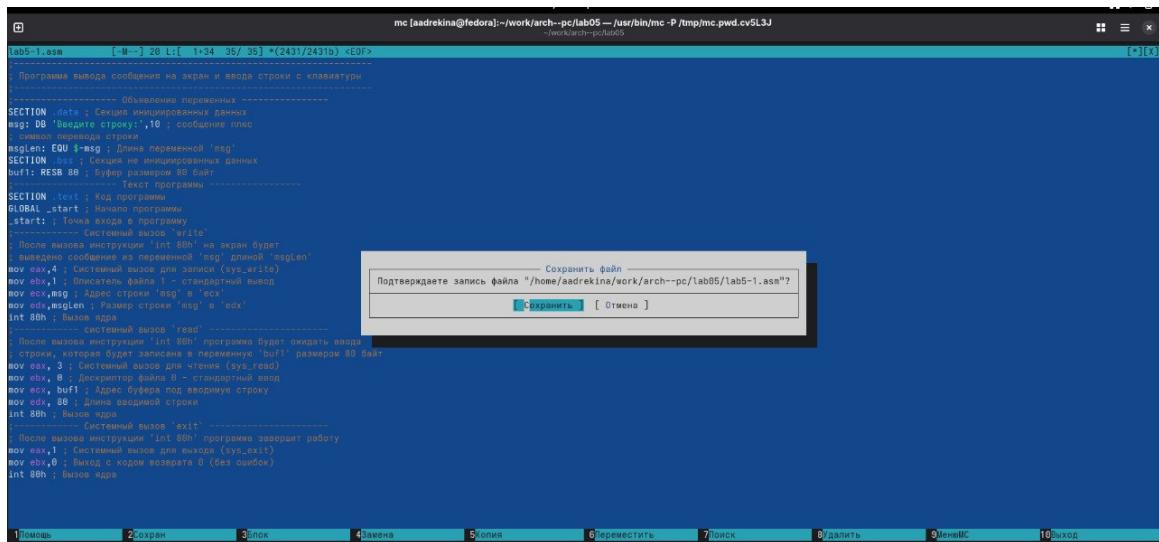


Рисунок 2.7: Сохранение изменений в файле.

Далее я заново открыла файл и убедилась, что данные сохранились. (Рисунок 2.8)

```
mc [aadrekina@fedora]:~/work/arch--pc/lab05 -- /usr/bin/mc -P /tmp/mc.pwd.cv5L3J
/home/aadrekina/work/arch--pc/lab05/lab5-1.asm
----- Объявление переменных -----
SECTION data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение для ввода строки
msgLen: EQU $-msg ; Длина переменной "msg"
SECTION bss ; Секция не инициализированных данных
buf: RESB 80 ; Буфер размером 80 байт
----- Текст программы -----
SECTION text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный 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 ; Консольный файл descriptor для стандартного вывода
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 'buf' размером 80 байт
mov eax, 3 ; Системный вызов для чтения (sys_read)
mov ebx, 0 ; Дескриптор файла 0 - стандартный ввод
mov ecx, buf ; Адрес буфера под якорную строку
mov edx, 80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов "exit"
; После выполнения инструкции 'int 80h' программа завершит работу
mov eax, 1 ; Системный вызов для выхода (sys_exit)
mov ebx, 0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рисунок 2.8: Повторное открытие файла и проверка изменений.

Далее я оттранслировала текст программы в объектный файл, выполнила компоновку и запустила исполняемый файл.(Рисунок 2.9)

```
aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-1
bash: ./lab5-1: Нет такого файла или каталога
aadrekina@fedora:~/work/arch--pc/lab05$ nasm -f elf lab5-1.asm
aadrekina@fedora:~/work/arch--pc/lab05$ d -m elf_i386 -o lab5-1 lab5-1.o
bash: d: команда не найдена...
aadrekina@fedora:~/work/arch--pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-1
Введите строку:
Дрекина Арина Андреевна
aadrekina@fedora:~/work/arch--pc/lab05$
```

Рисунок 2.9: Транслирование текста, выполнение компоновки и запуск исполняемого файла.

3 Подключение внешнего файла

in_out.asm

Для начала я скачала файл из туис. После этого в левой панели я открыла каталог с файлом lab5-1.asm, а в правой панели я открыла каталог со скачанным файлом.

(Рисунок 3.1)

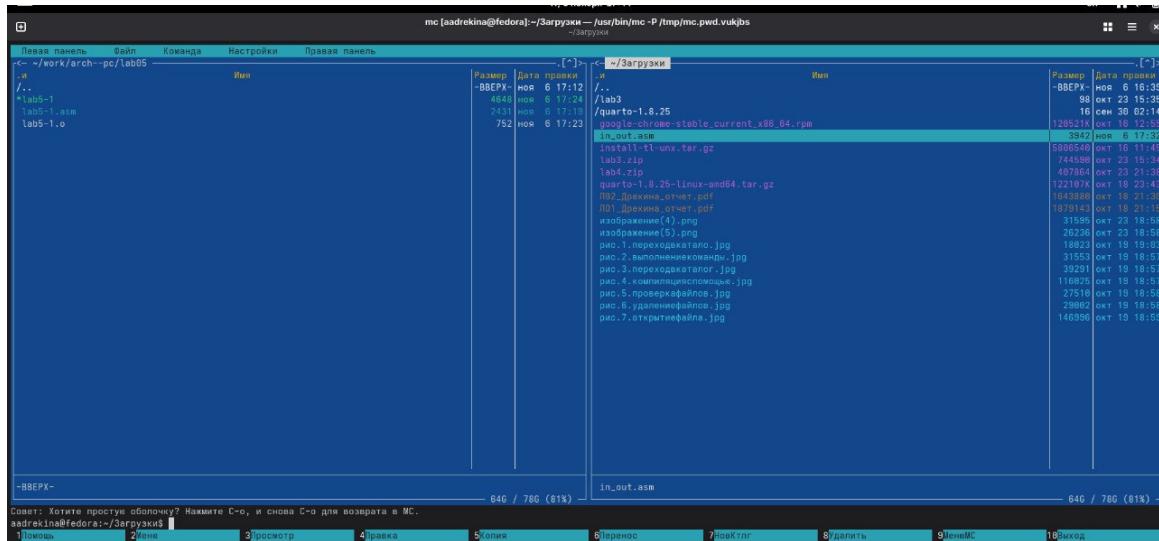


Рисунок 3.1: Открытие левой и правой панели.

После этого я скопировала файл in_out.asm и переместила в каталог lab05 (Рисунок 3.2)

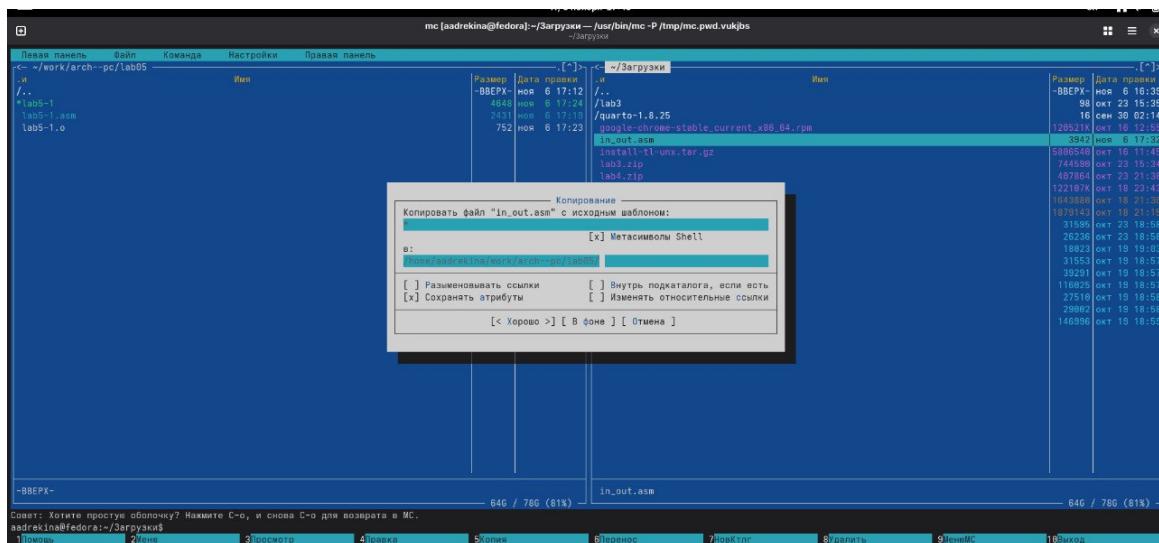


Рисунок 3.2: Перемещение файла в каталог lab05.

После этого я убедилась, что файл появился в каталоге lab05 (Рисунок 3.3)

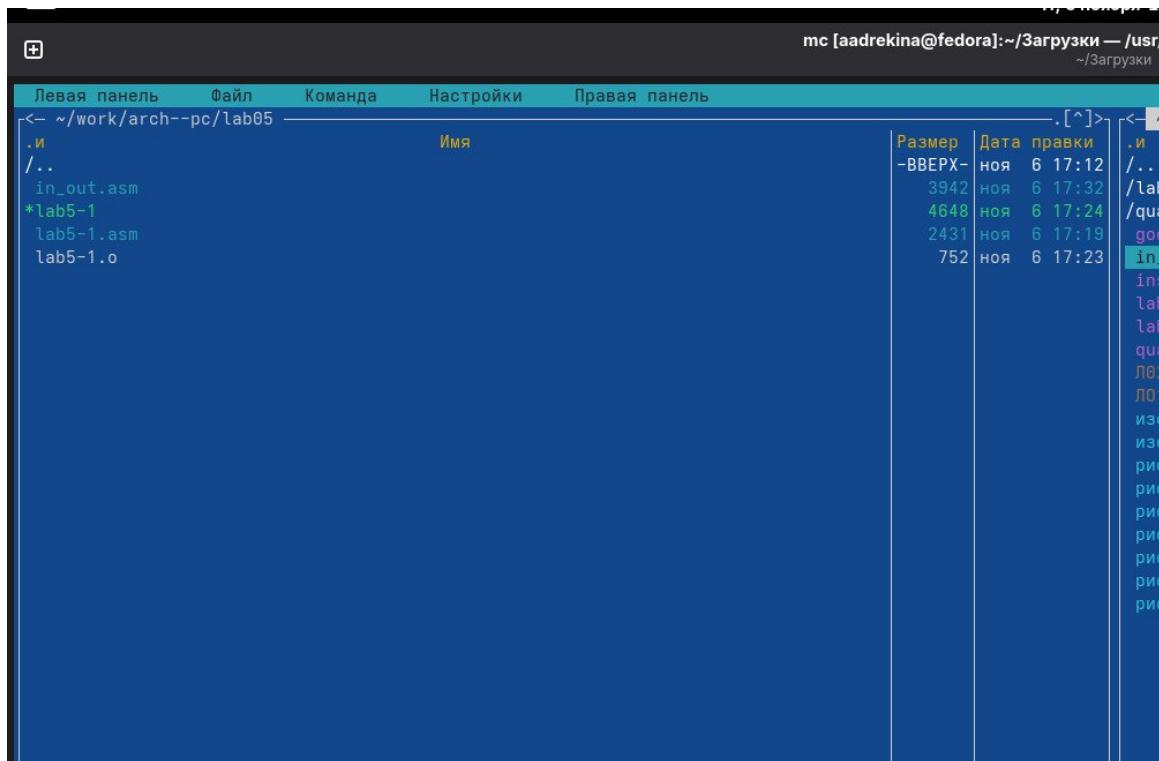


Рисунок 3.3: Проверка наличия файла в каталоге lab05.

Далее я создала копию файла lab5-1.asm и назвала его «lab5-2.asm». (Рисунок 3.4)

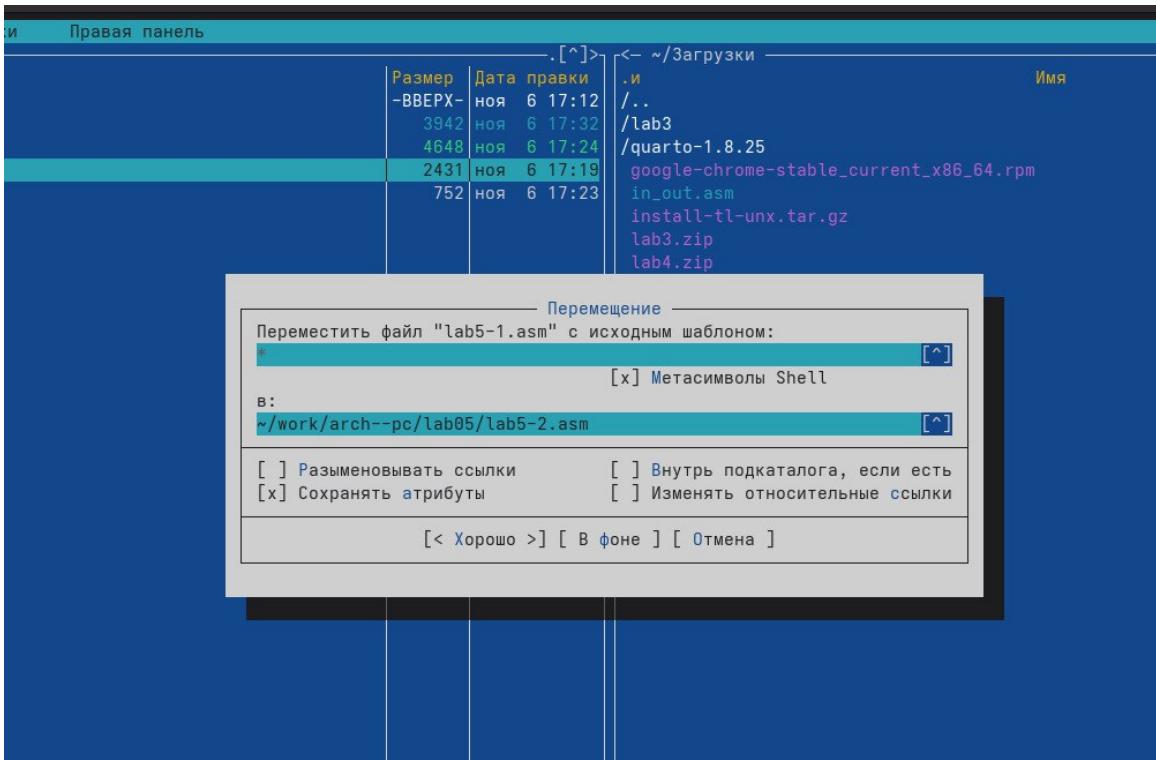


Рисунок 3.4: Создание копии файла lab5-1.asm.

Потом я заменила текст и вставила туда текст из Листинга 5.2. (Рисунок 3.5)

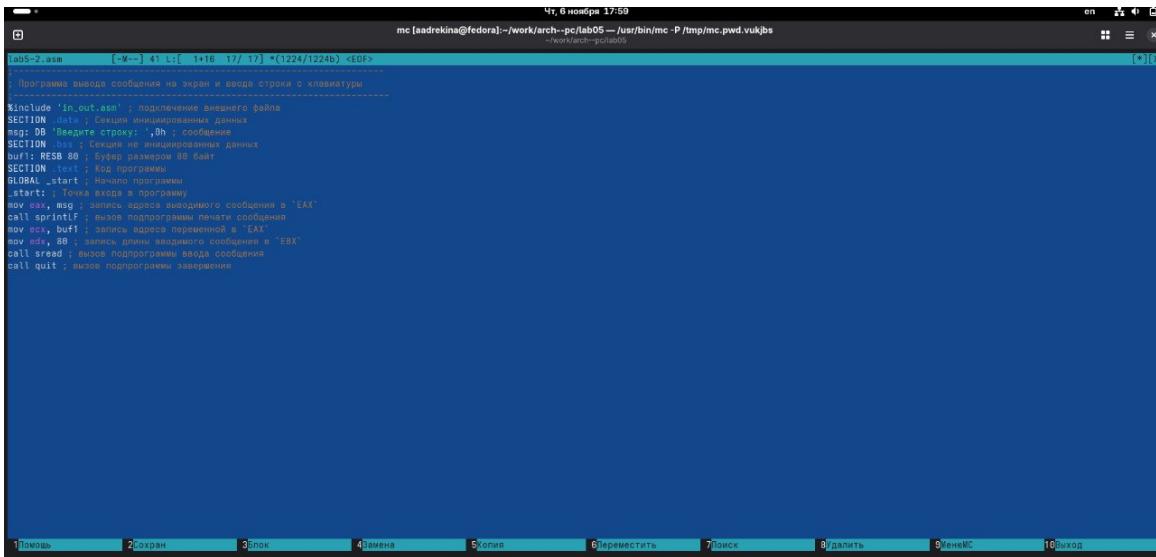
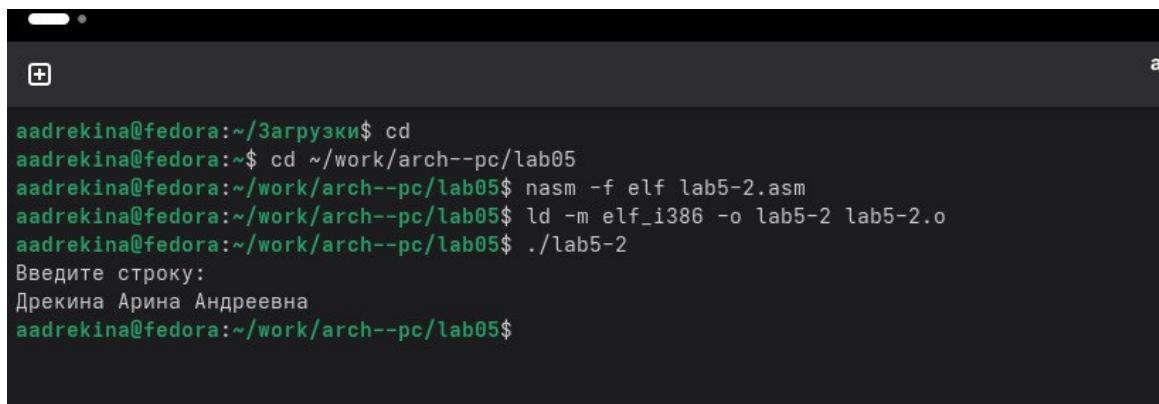


Рисунок 3.5: Изменение текста файла.

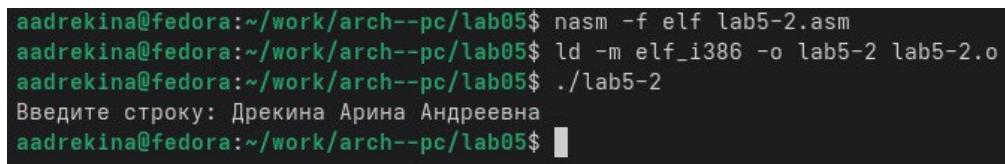
Затем я создала исполняемый файл и проверила его работу.(Рисунок 3.6)



```
aadrekina@fedora:~/Загрузки$ cd
aadrekina@fedora:~$ cd ~/work/arch--pc/lab05
aadrekina@fedora:~/work/arch--pc/lab05$ nasm -f elf lab5-2.asm
aadrekina@fedora:~/work/arch--pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-2
Введите строку:
Дрекина Арина Андреевна
aadrekina@fedora:~/work/arch--pc/lab05$
```

Рисунок 3.6: Создание исполняемого файла и проверка его работы.

Далее в этом же файле я заменила подпрограмму sprintLF на sprint. И проверила, что изменится при выводе. (Рисунок 3.7)



```
aadrekina@fedora:~/work/arch--pc/lab05$ nasm -f elf lab5-2.asm
aadrekina@fedora:~/work/arch--pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-2
Введите строку: Дрекина Арина Андреевна
aadrekina@fedora:~/work/arch--pc/lab05$ █
```

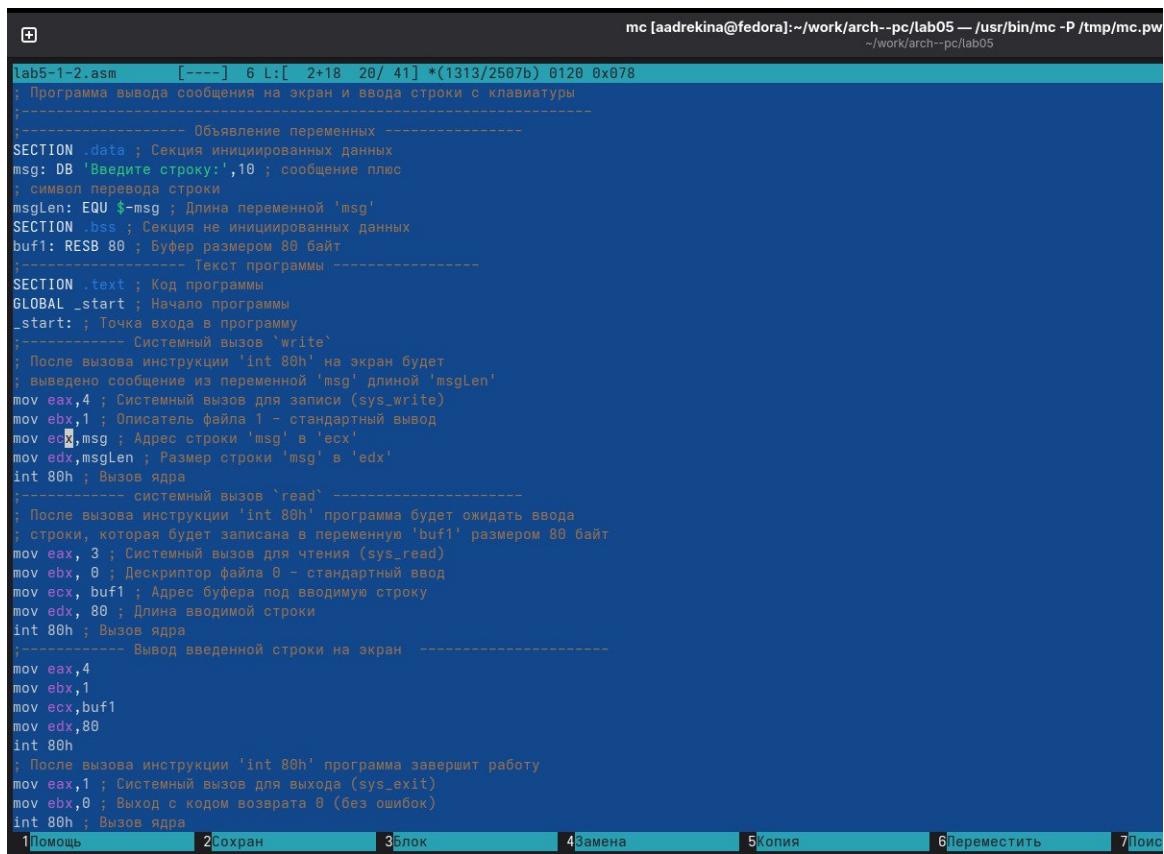
Рисунок 3.7: Проверка вывода.

Разница между выводами очевидна: первый просит ввод со следующий строки, а второй просит ввод с той же строки.

4 Задание для самостоятельной работы

После этого я создала копию файла lab5-1.asm и назвала новый файл «lab5-1-2.asm».

Затем внесла изменения в программу, так чтобы она работала по сказанному алгоритму. (Рисунок 4.1)



The screenshot shows a terminal window with the command `mc [aadrekina@fedora]:~/work/arch--pc/lab05 — /usr/bin/mc -P /tmp/mc.pw`. The assembly code in the editor is as follows:

```
lab5-1-2.asm      [---] 6 L:[ 2+18 20/ 41] *(1313/2507b) 0120 0x078
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный 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
;----- Вывод введенной строки на экран -----
mov eax,4
mov ebx,1
mov ecx,buf1
mov edx,80
int 80h
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

The menu bar at the bottom includes: Помощь (Help), Сохран (Save), Блок (Block), Замена (Replace), Копия (Copy), Переместить (Move), and Поиск (Search).

Рисунок 4.1: Измененный текст.

Далее я сделала lab5-1-2.asm исполняемым и проверила его работу. (Рисунок 4.2)

```
aadrekina@fedora:~/work/arch--pc/lab05$ nasm -f elf lab5-1-2.asm
aadrekina@fedora:~/work/arch--pc/lab05$ ld -m elf_i386 -o lab5-1-2 lab5-1-2.o
aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-1-2
Введите строку:
Дрекина
Дрекина
aadrekina@fedora:~/work/arch--pc/lab05$
```

Рисунок 4.2: Проверка работы программы.

После этого я создала копию файла lab5-2.asm и назвала его lab5-2-2.asm. (Рисунок 4.3)

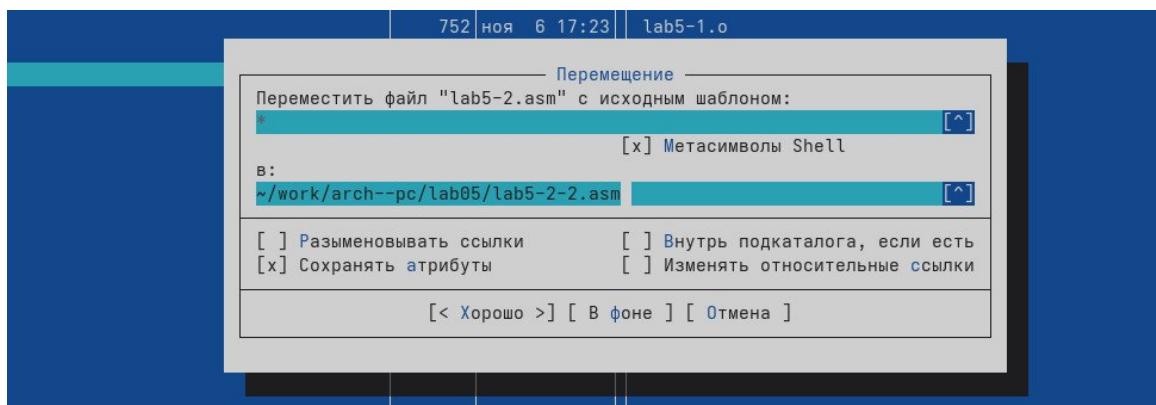
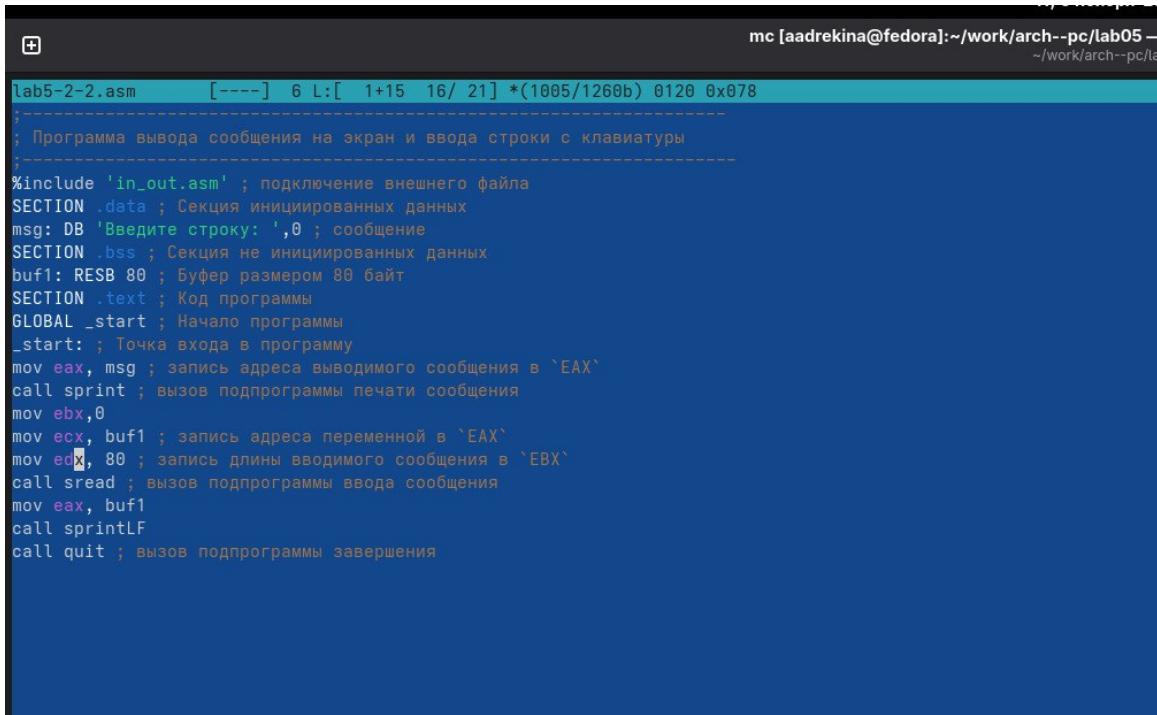


Рисунок 4.3: Создание копии с названием lab5-2-2.asm.

Потом внесла изменения чтобы программа выводила введенную строку на экран (Рисунок 4.4)



mc [aadrekina@fedora:~/work/arch--pc/lab05 --
~/work/arch--pc/la

```
lab5-2-2.asm      [---] 6 L:[ 1+15 16/ 21] *(1005/1260b) 0120 0x078
;-----;
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----;
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ',0 ; сообщение
SECTION .bss ; Секция не инициализированных 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 ebx,0
    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, buf1
    call sprintLF
    call quit ; вызов подпрограммы завершения
```

Рисунок 4.4: Измененный текст.

Затем я сделала файл исполняемым и проверила его работу. (Рисунок 4.5)



```
[root] ~$ cd /opt/nasm
[aadrekina@fedora:~/work/arch--pc/lab05$ nasm -f elf lab5-2-2.asm
[aadrekina@fedora:~/work/arch--pc/lab05$ ld -m elf_i386 -o lab5-2-2 lab5-2-2.o
[aadrekina@fedora:~/work/arch--pc/lab05$ ./lab5-2-2
Введите строку: Дрекина
Дрекина
[aadrekina@fedora:~/work/arch--pc/lab05$
```

Рисунок 4.5: Транслирование файла, и проверка работы программы.

5 Вывод

В ходе выполнения лабораторной работы я приобрела практические навыки работы в Midnight Commander. И освоила инструкцию языка ассемблера mov и int.