

Отчёт по лабораторной работе№5

Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM.

Маев Даниил Егорович

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	18

Список иллюстраций

3.1	Открытие Midnight Commander	7
3.2	Создание папки для лабораторной работы	8
3.3	Папка lab05	9
3.4	Создание файла lab5-1.asm	9
3.5	Ввод текста	10
3.6	Проверка наличия текста	11
3.7	Транслирование текста, проверка работоспособности	12
3.8	Перенос файла в папку lab05	13
3.9	Создание копии	14
3.10	Внесение изменений в файл	15
3.11	Проверка наличия текста	16
3.12	Транслирование текста, проверка работоспособности	16
3.13	Проверка файла3	17
3.14	транслирование текста в файл	17

Список таблиц

1 Цель работы

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

2 Задание

1. Создайте копию файла lab5-1.asm. Внесите изменения в программу (без использования внешнего файла in_out.asm), так чтобы она работала по следующему алгоритму:
 - вывести приглашение типа “Введите строку:”;
 - ввести строку с клавиатуры;
 - вывести введённую строку на экран.
2. Получите исполняемый файл и проверьте его работу. На приглашение ввести строку введите свою фамилию.
3. Создайте копию файла lab5-2.asm. Исправьте текст программы с использование подпрограмм из внешнего файла in_out.asm, так чтобы она работала по следующему алгоритму:
 - вывести приглашение типа “Введите строку:”;
 - ввести строку с клавиатуры;
 - вывести введённую строку на экран.

3 Выполнение лабораторной работы

1. Создаем каталог для работы с программами на языке ассемблера NASM.

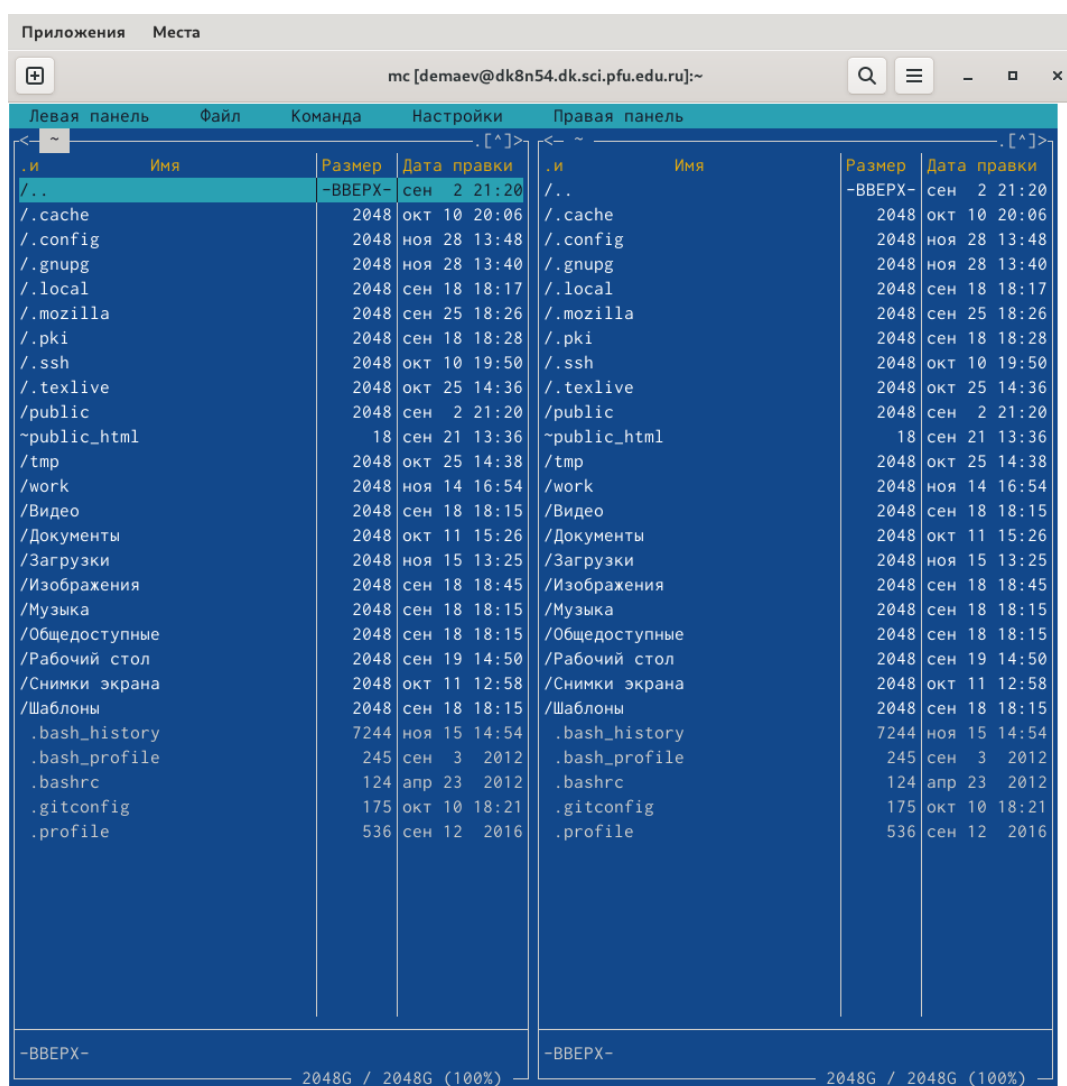


Рис. 3.1: Открытие Midnight Commander

2. С помощью функциональной клавиши F7 создаём папку lab05.

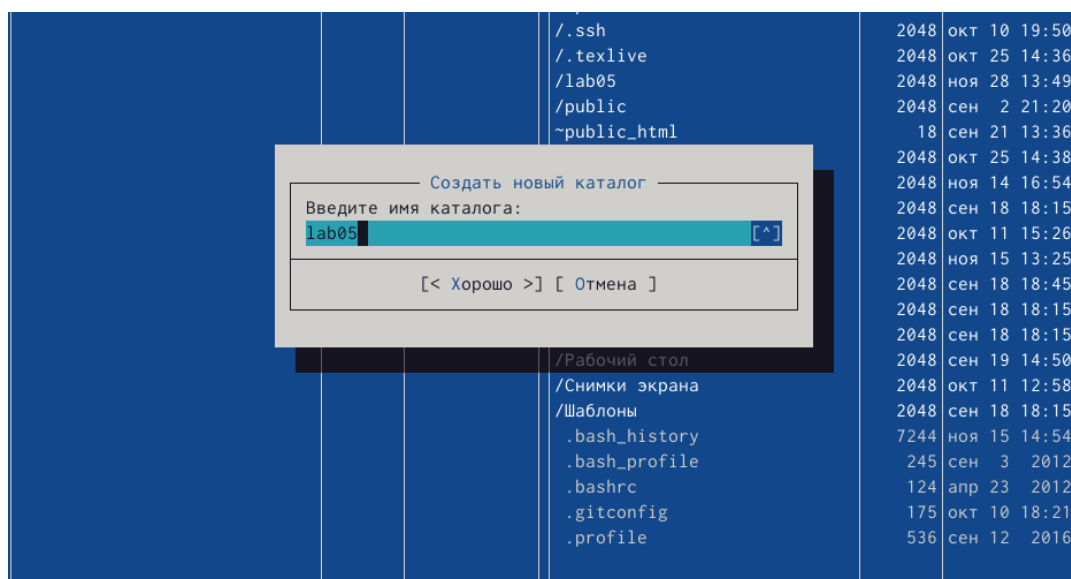


Рис. 3.2: Создание папки для лабораторной работы

3. Убедимся в правильном создании папки.

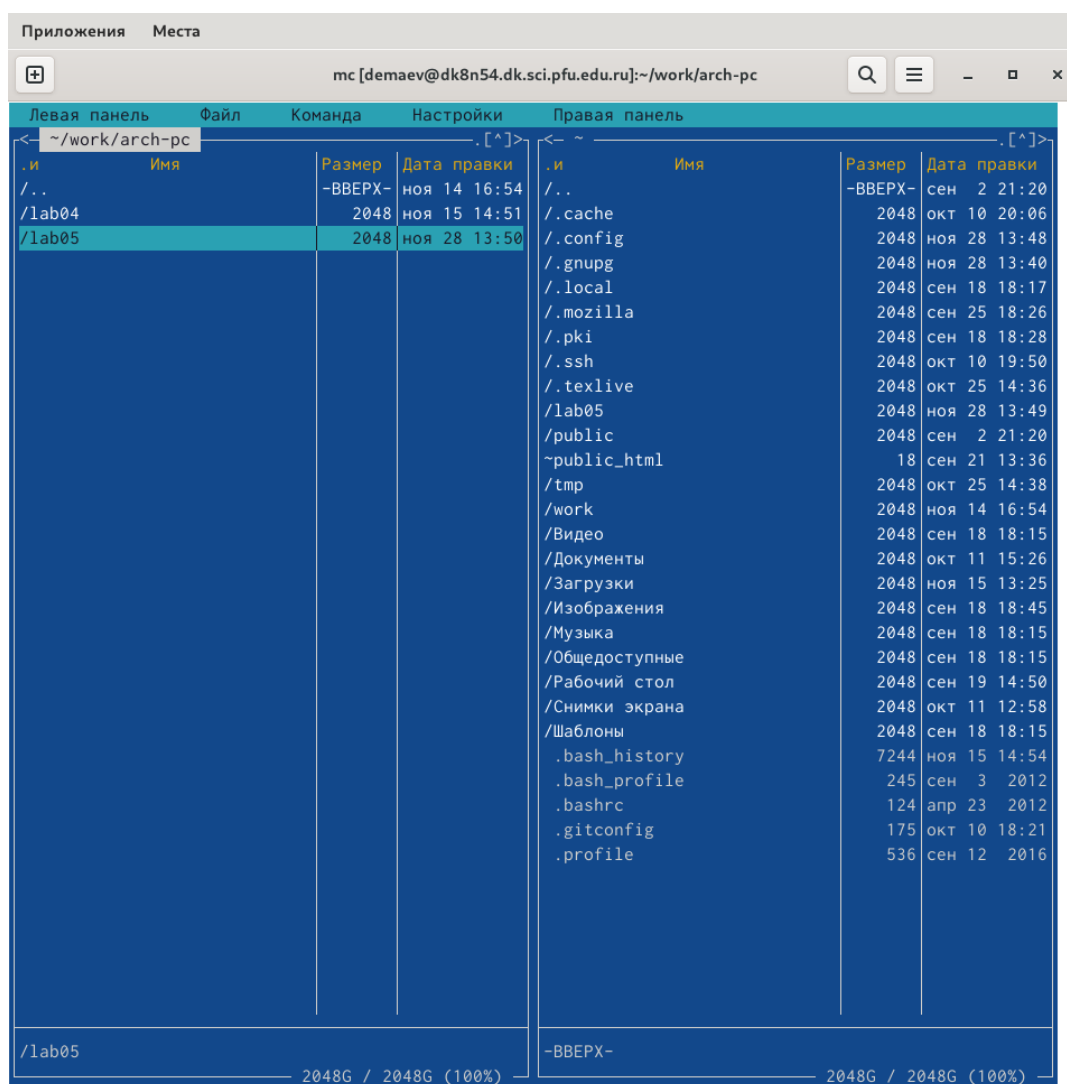


Рис. 3.3: Папка lab05

- Пользуясь строкой ввода и командой touch создадим файл lab5-1.asm.

Совет: Используйте C-x p для копирования текущего пути в командную строку.

```
demaev@dk8n54 ~/work/arch-pc $ touch lab5-1.asm
```

1Помощь 2Меню 3Просмотр 4Правка 5Копия 6Перенос 7НовКтлг 8

Рис. 3.4: Создание файла lab5-1.asm

- Откроем файл lab5-1.asm и введём текст из листинга 6.1.

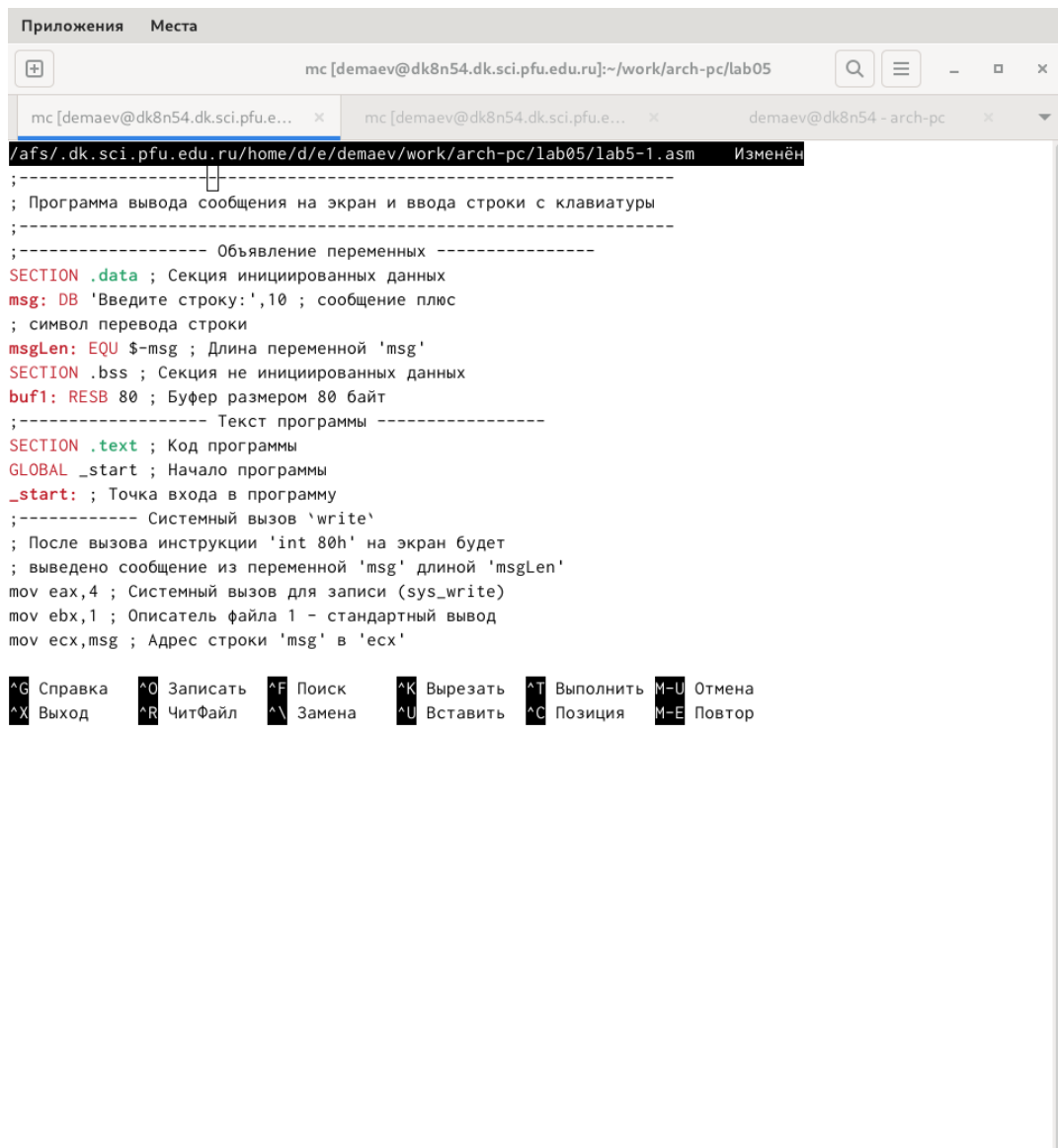


Рис. 3.5: Ввод текста

6. С помощью функциональной клавиши F3 откроем файл lab5-1.asm для проверки наличия текста.

```
mc [demaev@dk8n54.dk.sci.pfu.edu.ru:~/work/arch-pc/lab05
/afs/.dk.sci.pfu.edu.ru/home/d/e/demaev/work/arch-pc/lab05/lab5-1.asm 2432/2432 100%
; 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
;----- 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
```

1Помощь 2Разверн 3Выход 4Нех 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход

Рис. 3.6: Проверка наличия текста

7. Оттранслируем текст программы lab5-1.asm в объектный файл.

```
-----
demaev@dk8n54 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
demaev@dk8n54 ~/work/arch-pc/lab05 $ ls
lab5-1.asm  lab5-1.o
demaev@dk8n54 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
demaev@dk8n54 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Маев Даниил
demaev@dk8n54 ~/work/arch-pc/lab05 $
```

Рис. 3.7: Транслирование текста, проверка работоспособности

8. Скопируем файл `in_out.asm` в каталог с файлом `lab5-1.asm` с помощью функциональной клавиши F5.

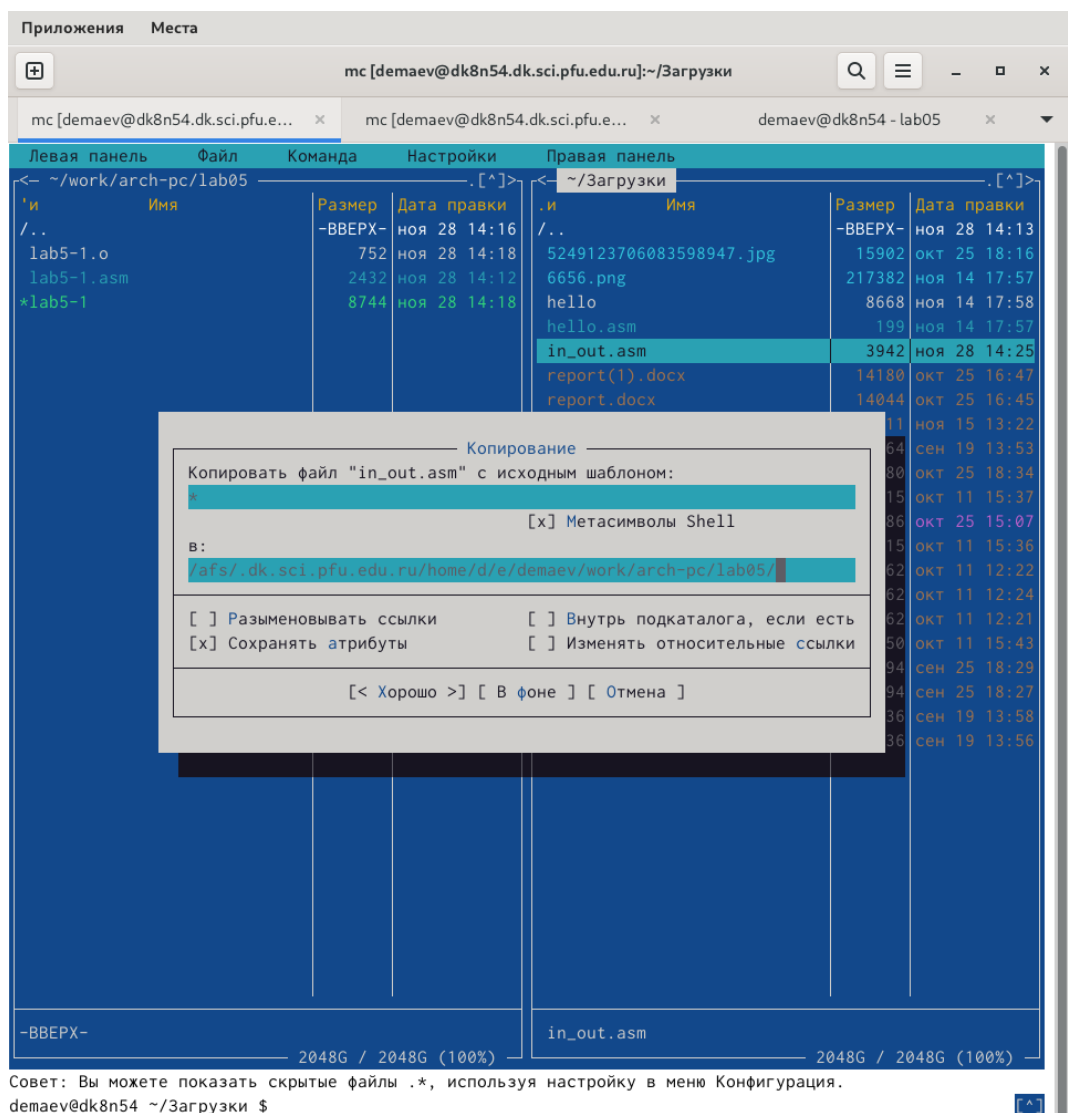


Рис. 3.8: Перенос файла в папку lab05

9. С помощью функциональной клавиши F6 создадим копию файла lab5-1.asm с именем lab5-2.asm.

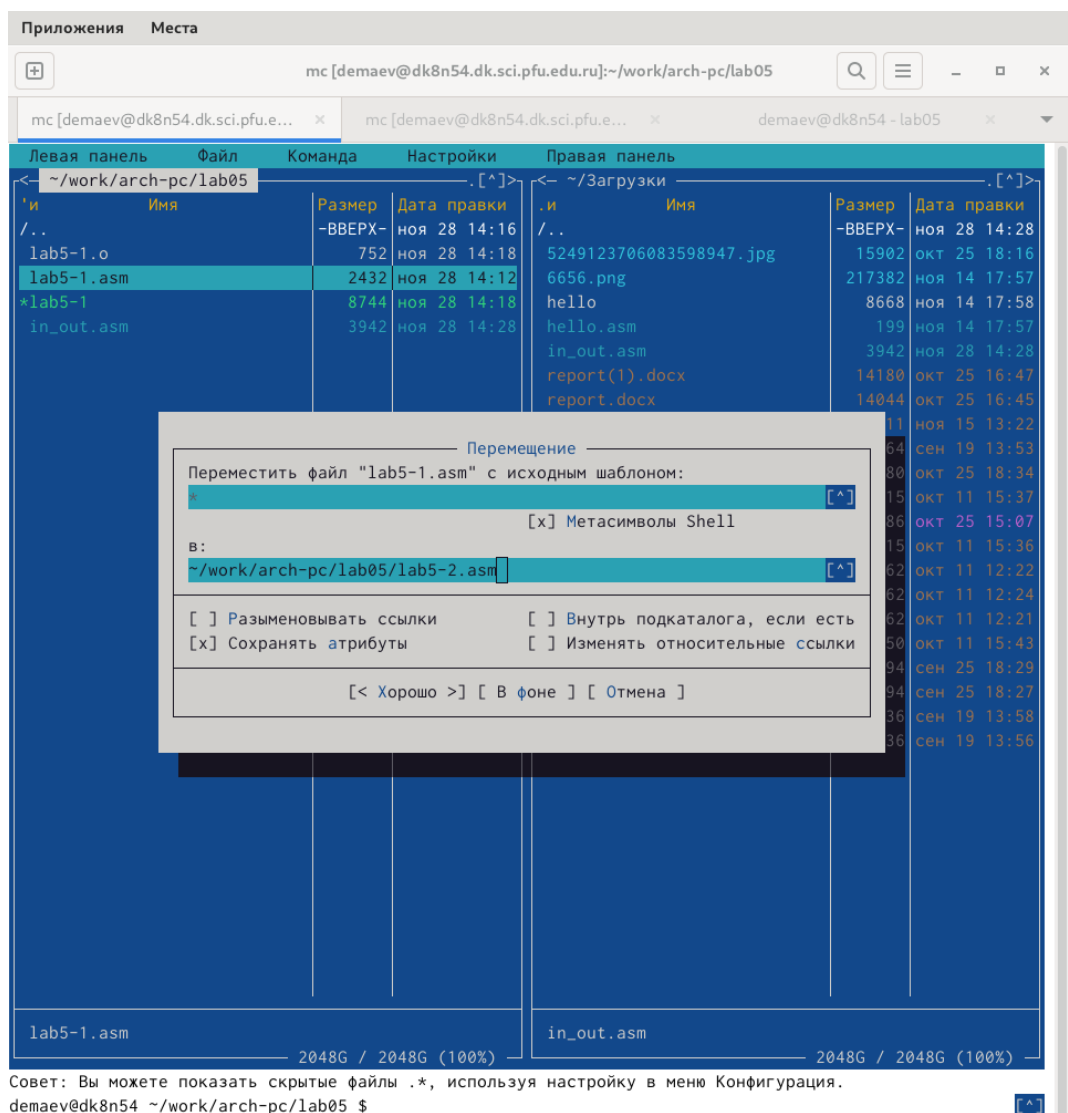


Рис. 3.9: Создание копии

10. Исправим текст программы в соответствии с листингом 6.2

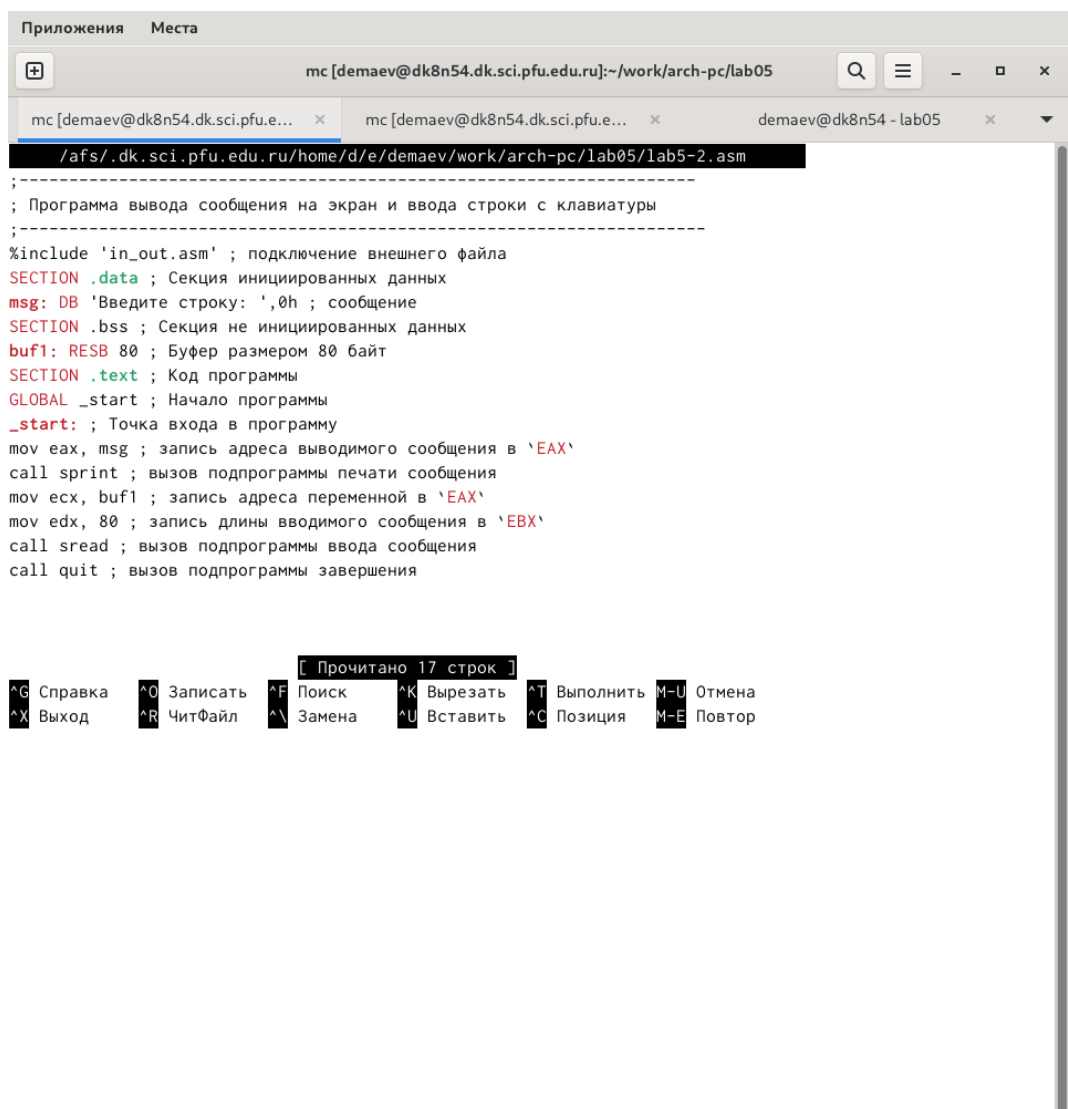


Рис. 3.10: Внесение изменений в файл

11. Проверим, что текст был изменен

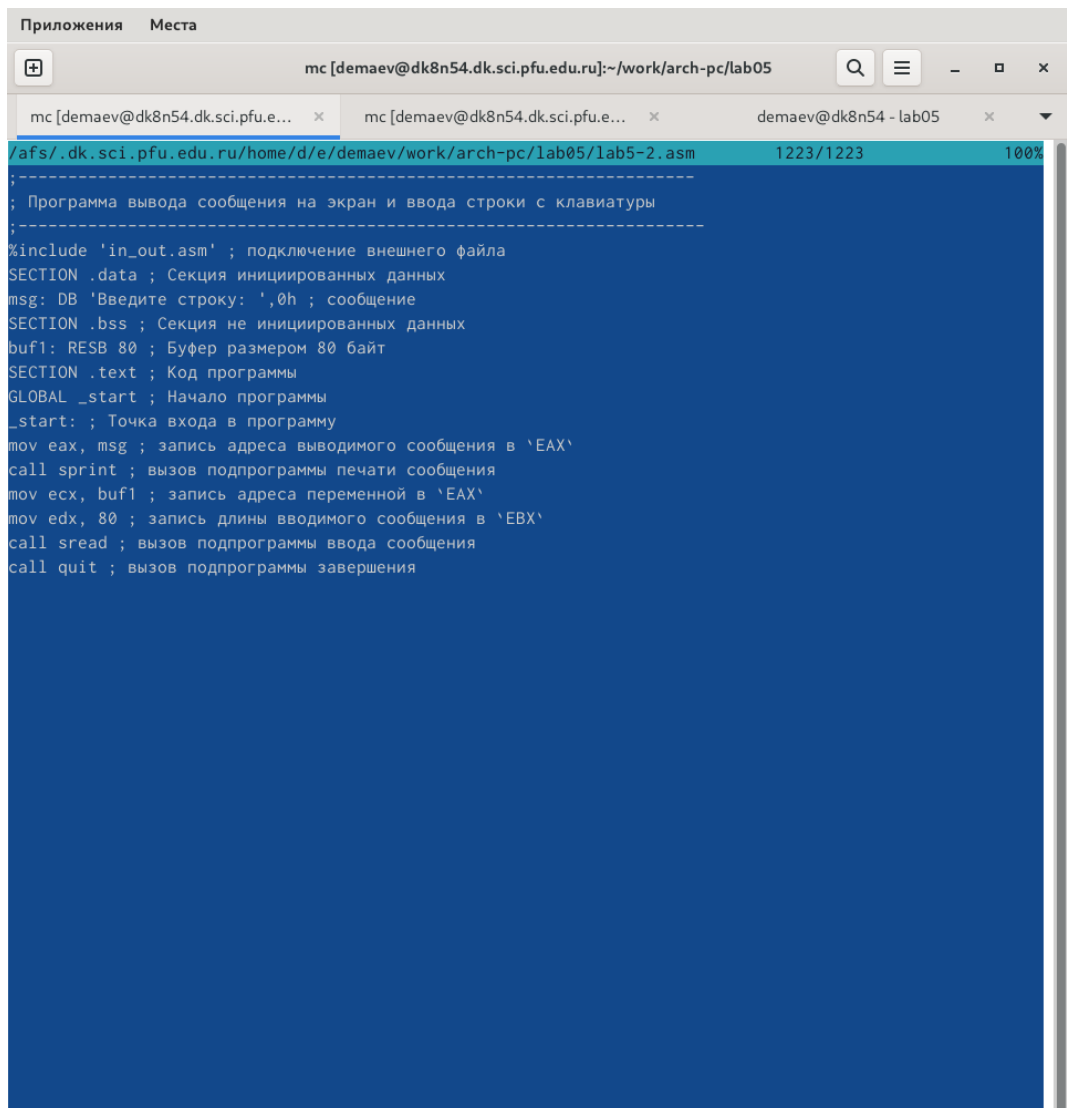


Рис. 3.11: Проверка наличия текста

12. Оттранслируем текст программы lab5-2.asm в объектный файл и проверим его работоспособность

```

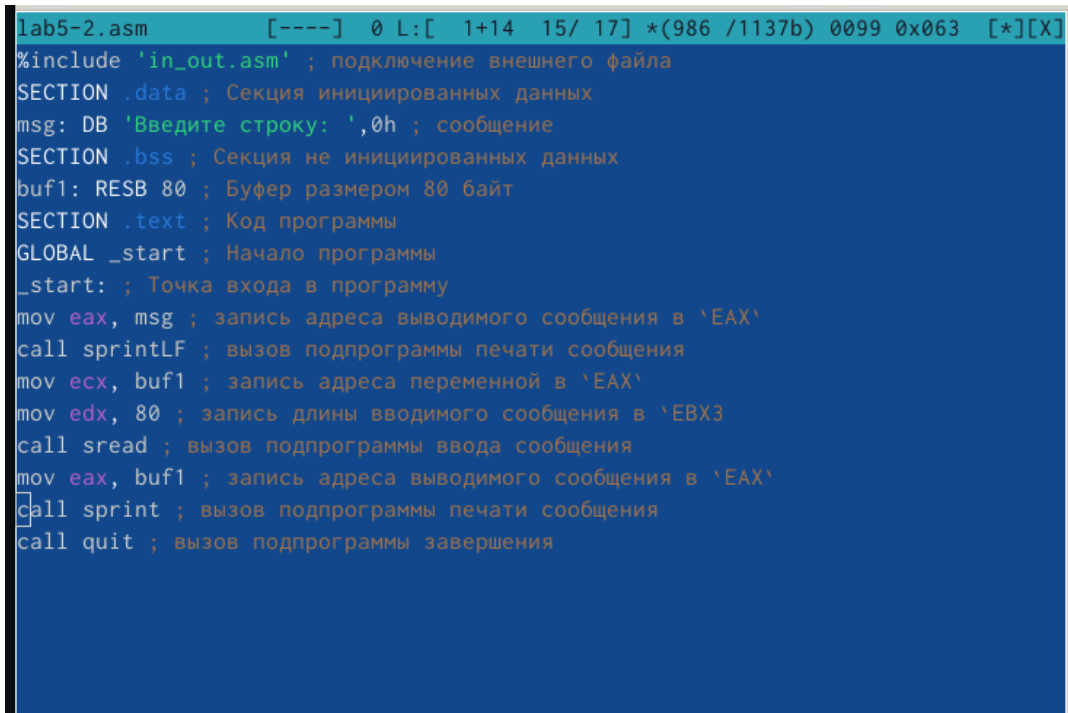
demaev@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
demaev@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
demaev@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
123

```

Рис. 3.12: Транслирование текста, проверка работоспособности

13. Исправляем текст программы, так чтобы она работала по следующему алгоритму:

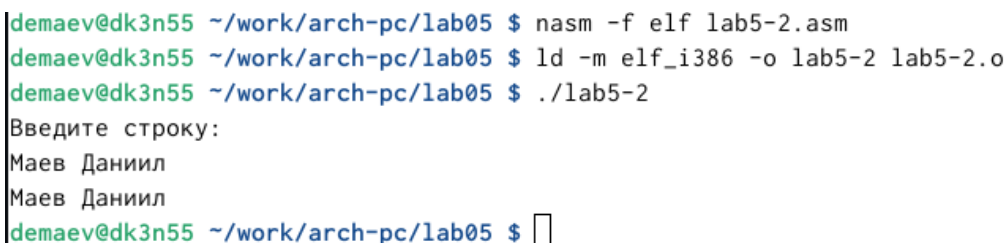
- вывести приглашение типа “Введите строку:”
- ввести строку с клавиатуры;
- вывести введенную строку на экран.



```
lab5-2.asm [----] 0 L:[ 1+14 15/ 17] *(986 /1137b) 0099 0x063 [*][X]
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не иницированных 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 sprintf ; 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, buf1 ; запись адреса выводимого сообщения в 'EAX'
call sprint ; вызов подпрограммы печати сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 3.13: Проверка файла3

14. Оттранслируем текст программы в объектный файл и проверяем его работу



```
demaev@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
demaev@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
demaev@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
Маев Даниил
Маев Даниил
demaev@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 3.14: транслирование текста в файл

4 Выводы

В ходе лабораторной работы мною были приобретены практические навыки работы в Midnight Commander, а также освоены инструкции языка ассемблера mov и int. Я научился работать с МС, и с его помощью работать с файлами (Создание, переименовывание, копирование, перемещение, удаление, и тд.)