

Отчет по лабораторной работе 5

Основы работы с Midnight Commander

Власов Артем Сергеевич

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Порядок выполнения лабораторной работы	6
3.2	Задание для самостоятельной работы	16
4	Выводы	21

Список иллюстраций

3.1	Вводим в консоль команду <code>mc</code>	6
3.2	Переходим в каталог	7
3.3	Создаем каталог функциональной клавишей F7	8
3.4	Воспользуемся командой <code>touch</code>	9
3.5	Открываем файл функциональной клавишей, заполняем и сохраняем	10
3.6	Открываем файл и убеждаемся, что файл содержит текст программы	11
3.7	Проверяем, как работает данная программа	11
3.8	Скачиваем файл	12
3.9	Копируем скаченный файл	12
3.10	Создаем копию файла клавишей F6	13
3.11	Проверяем скопировался ли файл	14
3.12	Открываем и заполняем файл	15
3.13	Смотрим, как сработала программа	15
3.14	Редактируем файл	16
3.15	Смотрим, как сработала программа и сравниваем с прошлой . . .	16
3.16	Создаем копию файла <code>lab5-1.asm</code>	17
3.17	Редактируем файл	18
3.18	Проверяем правильность написания программы	18
3.19	Создаем копию файла <code>lab5-2.asm</code>	19
3.20	Редактируем файл	20
3.21	Проверяем правильность написания программы	20

1 Цель работы

Освоить инструкции языка ассемблера mov. Приобрести знания использования Midnight Commander.

2 Задание

Написать 2 программы по примеру и впоследствии изменить их по условию.

3 Выполнение лабораторной работы

3.1 Порядок выполнения лабораторной работы

Открываем Midnight Commander (рис. fig. 3.1).

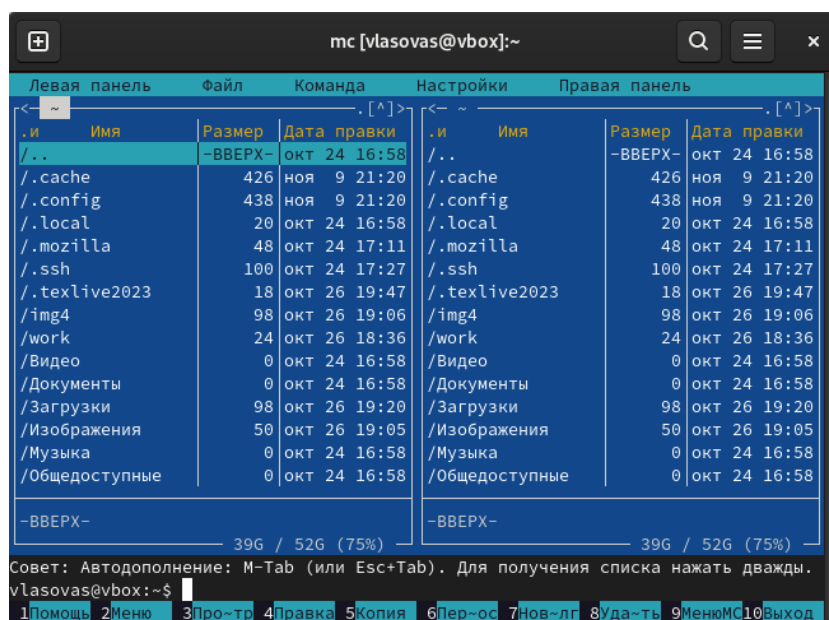


Рис. 3.1: Вводим в консоль команду mc

Переходим в каталог, созданный при выполнении 4 ЛБ (рис. fig. 3.2).

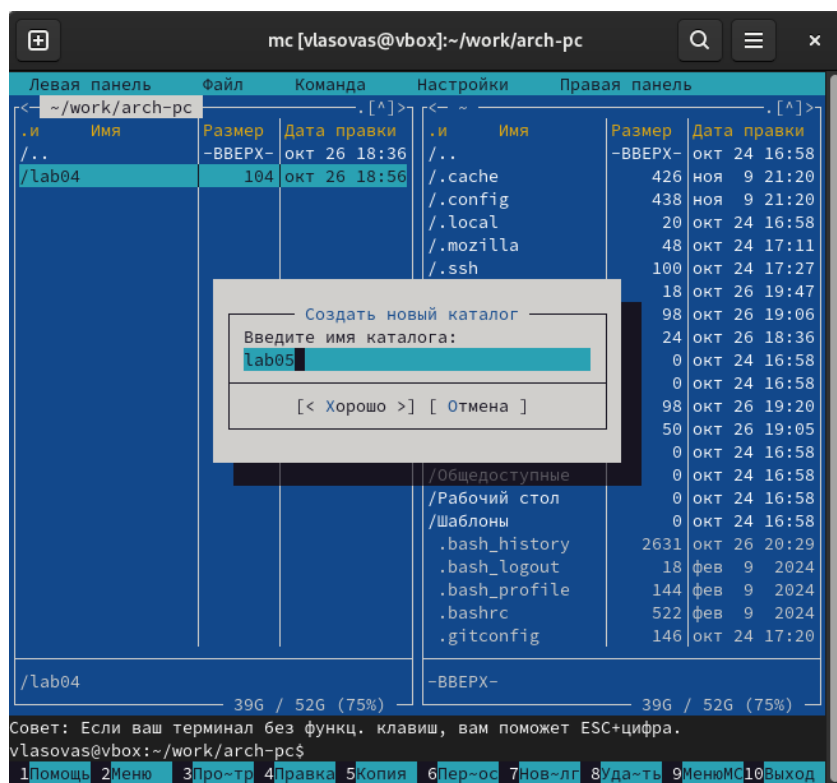


Рис. 3.3: Создаем каталог функциональной клавишей F7

Создаем файл lab5-1.asm (рис. fig. 3.4).

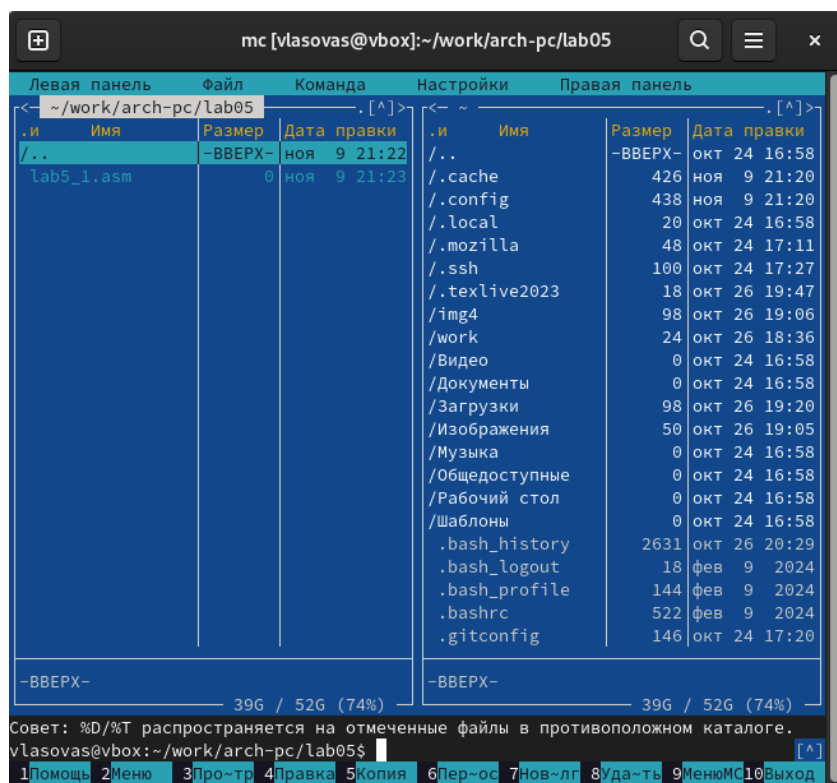
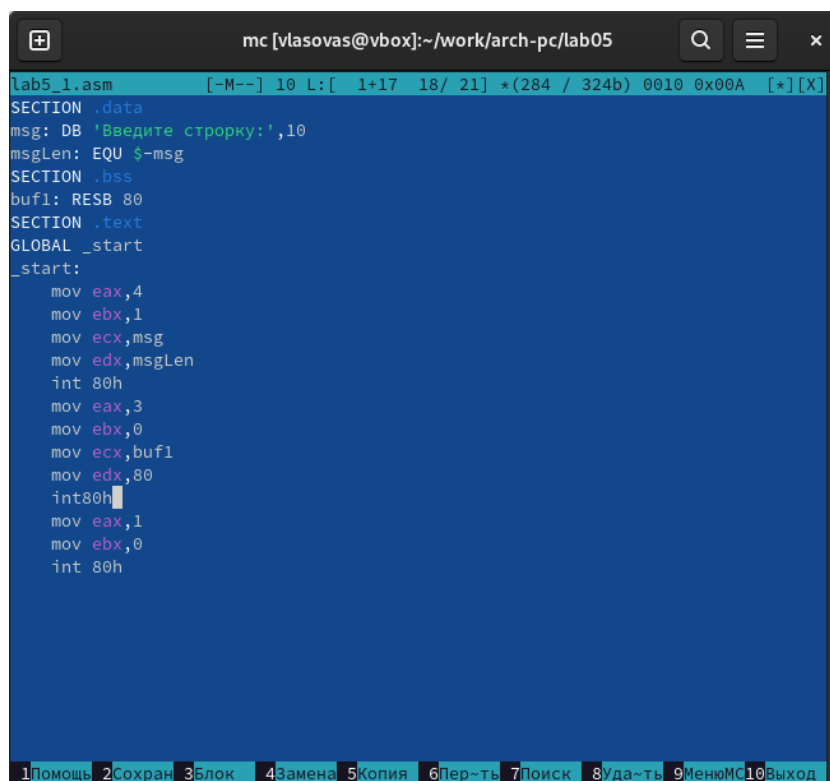


Рис. 3.4: Воспользуемся командой touch

Открываем файл для редактирования и заполняем его по листингу (рис. fig. 3.5).



```
mc [vlasovas@vbox]:~/work/arch-pc/lab05
lab5_1.asm [-M--] 10 L: [ 1+17 18/ 21] *(284 / 324b) 0010 0x00A [*] [X]
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80
SECTION .text
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
    int80h
    mov eax,1
    mov ebx,0
    int 80h
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

Рис. 3.5: Открываем файл функциональной клавишей, заполняем и сохраняем

Открываем файл для просмотра (рис. fig. 3.6).

```
GNU nano 7.2 /home/vlasovas/work/arch-pc/lab05/lab5_1.asm
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80
SECTION .text
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
    int80h
    mov eax,1
    mov ebx,0
    int 80h

[ Прочитана 21 строка ]
^G Справка  ^O Записать  ^W Поиск      ^K Вырезать   ^T Выполнить  ^С Позиция
^X Выход    ^R ЧитФайл   ^\ Замена     ^U Вставить   ^D Выводить   ^/ К строке
```

Рис. 3.6: Открываем файл и убеждаемся, что файл содержит текст программы

Транслируем текст программы и запускаем исполняемый файл (рис. fig. 3.7).

```
vlasovas@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
ld: неизвестный параметр «-0»
ld: используйте --help для получения информации о параметрах
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
vlasovas@vbox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Власов Артем Сергеевич
vlasovas@vbox:~/work/arch-pc/lab05$
```

Рис. 3.7: Проверяем, как работает данная программа

Скачиваем файл со страницы курса (рис. fig. 3.8).

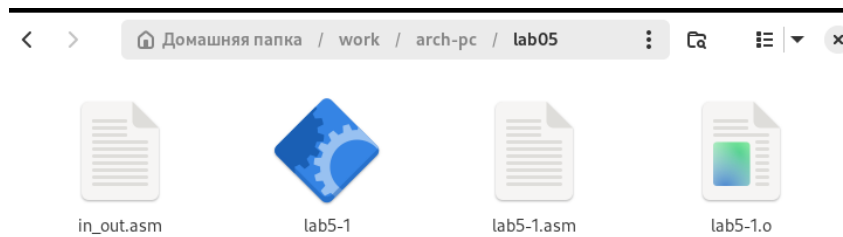


Рис. 3.8: Скачиваем файл

Копируем файл в нужную директорию (рис. fig. 3.9).

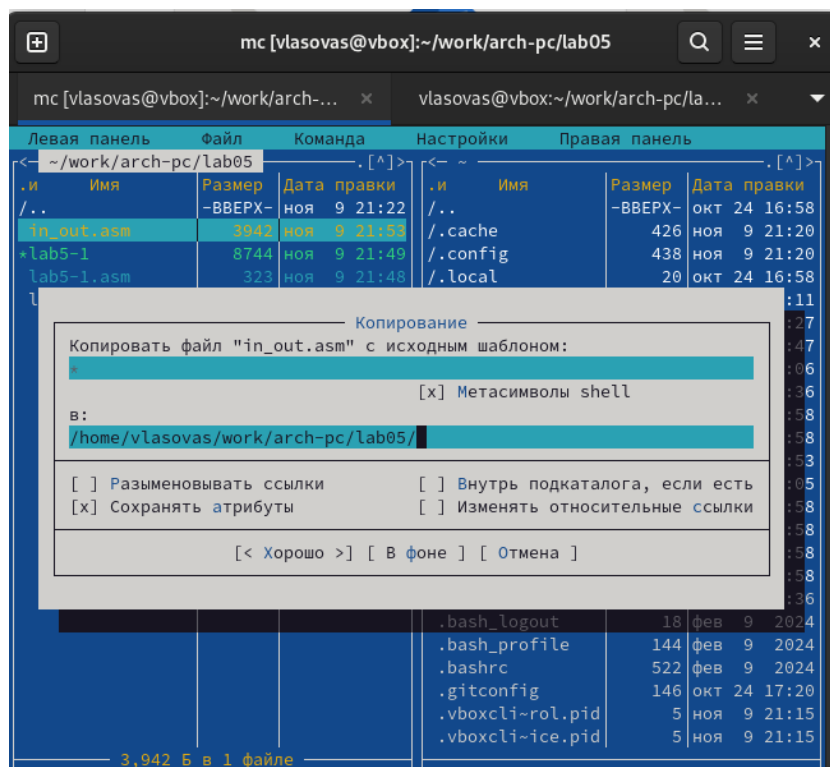


Рис. 3.9: Копируем скаченный файл

Создаем копию файла lab5-1.asm (рис. fig. 3.10).

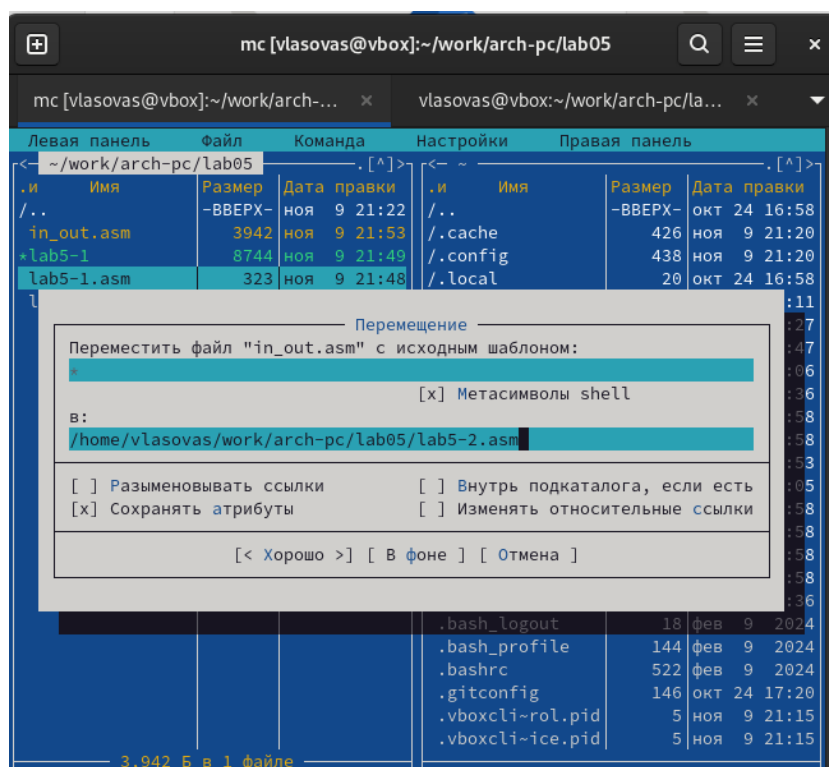


Рис. 3.10: Создаем копию файла клавишей F6

Проверяем созданный файл (рис. fig. 3.11).

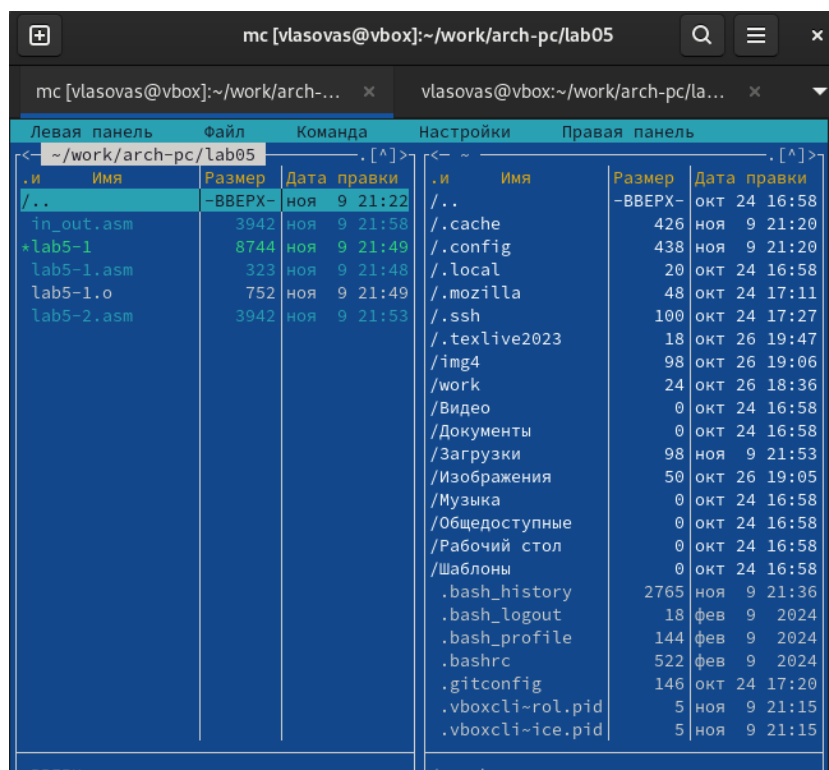
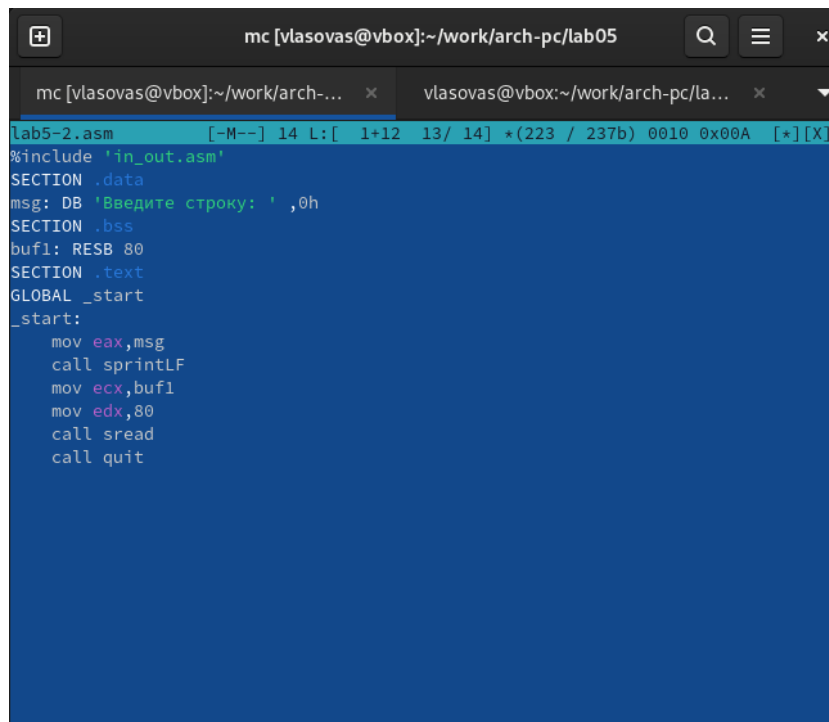


Рис. 3.11: Проверяем скопировался ли файл

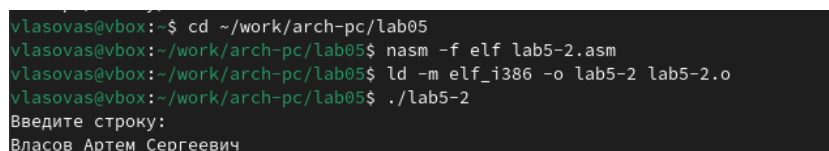
Открываем новый файл и заполняем его в соответствии с листингом (рис. fig. 3.12).



```
lab5-2.asm [-M--] 14 L: [ 1+12 13/ 14] *(223 / 237b) 0010 0x00A [*] [X]
#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
    call quit
```

Рис. 3.12: Открываем и заполняем файл

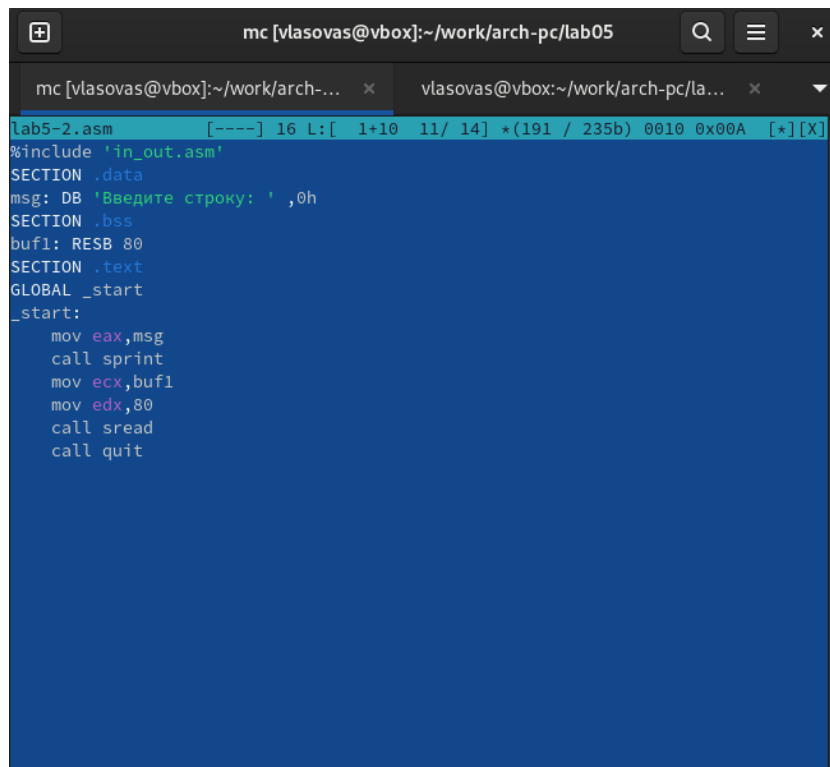
Транслируем и запускаем новый файл (рис. fig. 3.13).



```
vlasovas@vbox:~$ cd ~/work/arch-pc/lab05
vlasovas@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
vlasovas@vbox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Власов Артем Сергеевич
```

Рис. 3.13: Смотрим, как сработала программа

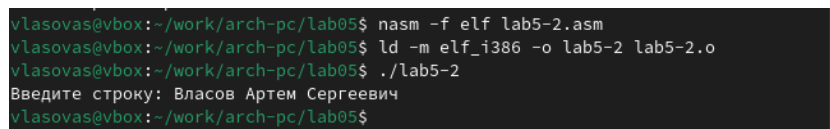
Снова открываем файл для редактирования и меняем sprintLF на sprint(рис. fig. 3.14).



```
lab5-2.asm [----] 16 L: [ 1+10 11/ 14] *(191 / 235b) 0010 0x00A [*][X]
#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 sprint
    mov ecx,buf1
    mov edx,80
    call sread
    call quit
```

Рис. 3.14: Редактируем файл

Транслируем и запускаем файл(рис. fig. 3.15).



```
vlasovas@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
vlasovas@vbox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Власов Артем Сергеевич
vlasovas@vbox:~/work/arch-pc/lab05$
```

Рис. 3.15: Смотрим, как сработала программа и сравниваем с прошлой

Таким образом можем понять, что команда `sprint` выводит текст в той же строке, а `sprintLF` переносит на новую строку.

3.2 Задание для самостоятельной работы

Создаем копию файла `lab5-1.asm` и называем его так же (рис. fig. 3.16).

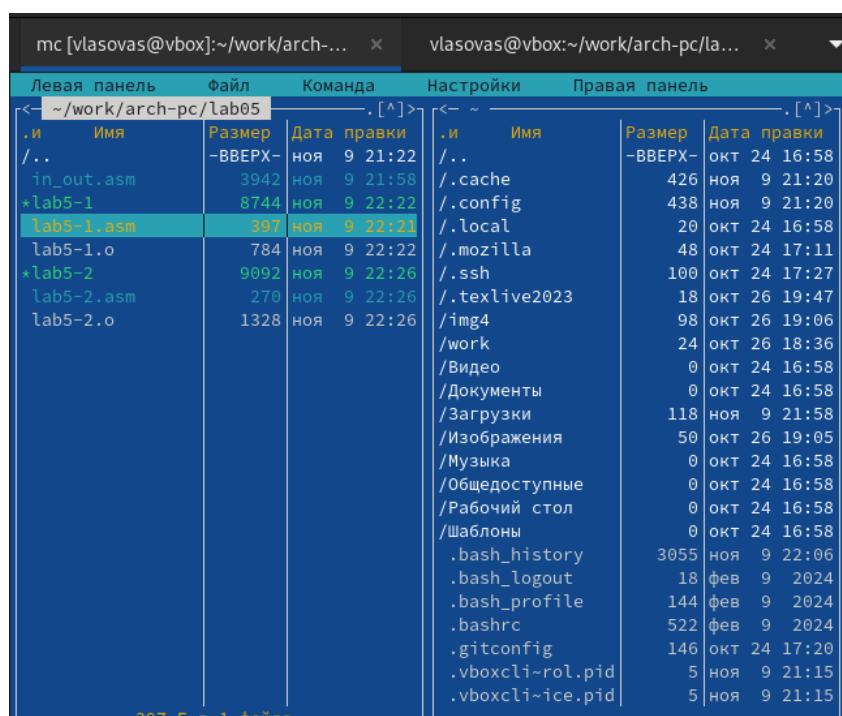
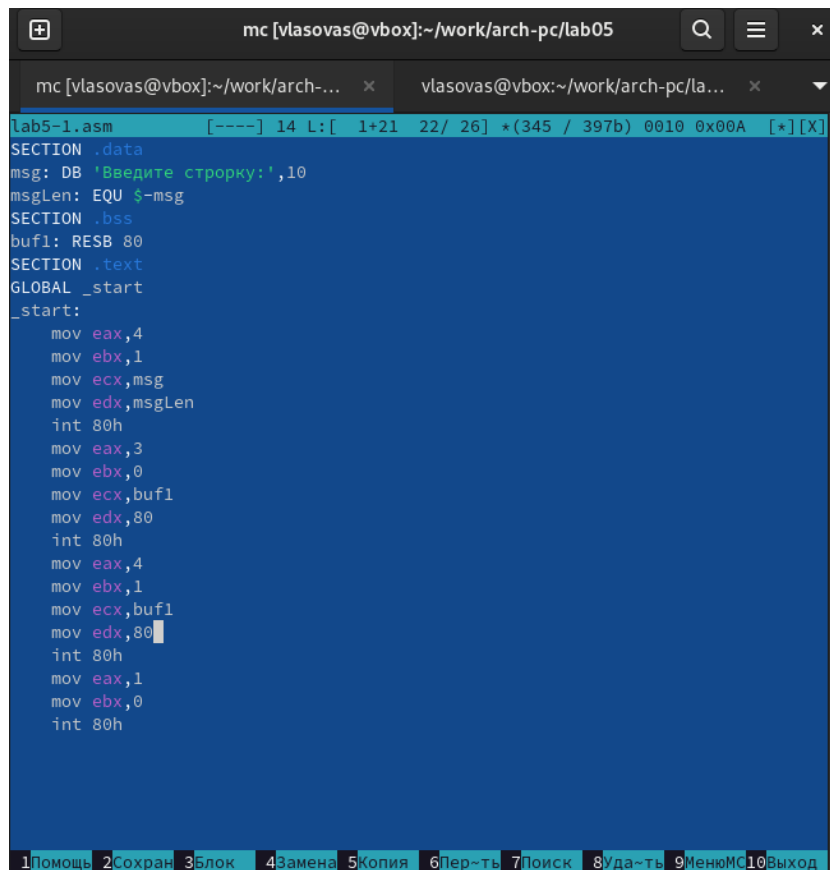


Рис. 3.16: Создаем копию файла lab5-1.asm

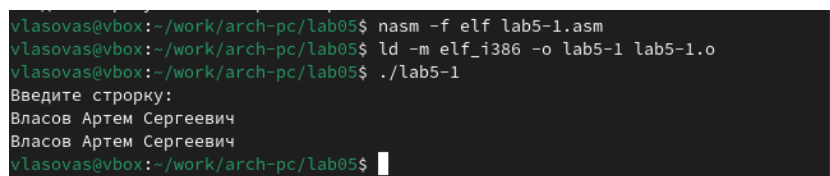
Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль (рис. fig. 3.17).



```
lab5-1.asm  [----] 14 L: [ 1+21 22/ 26] *(345 / 397b) 0010 0x00A [*][X]
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80
SECTION .text
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,80
    int 80h
    mov eax,1
    mov ebx,0
    int 80h
```

Рис. 3.17: Редактируем файл

Транслируем файл и запускаем программу (рис. fig. 3.18).



```
vlasovas@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
vlasovas@vbox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Власов Артем Сергеевич
Власов Артем Сергеевич
vlasovas@vbox:~/work/arch-pc/lab05$
```

Рис. 3.18: Проверяем правильность написания программы

Создаем копию файла lab5-2.asm и называем его так же (рис. fig. 3.19).

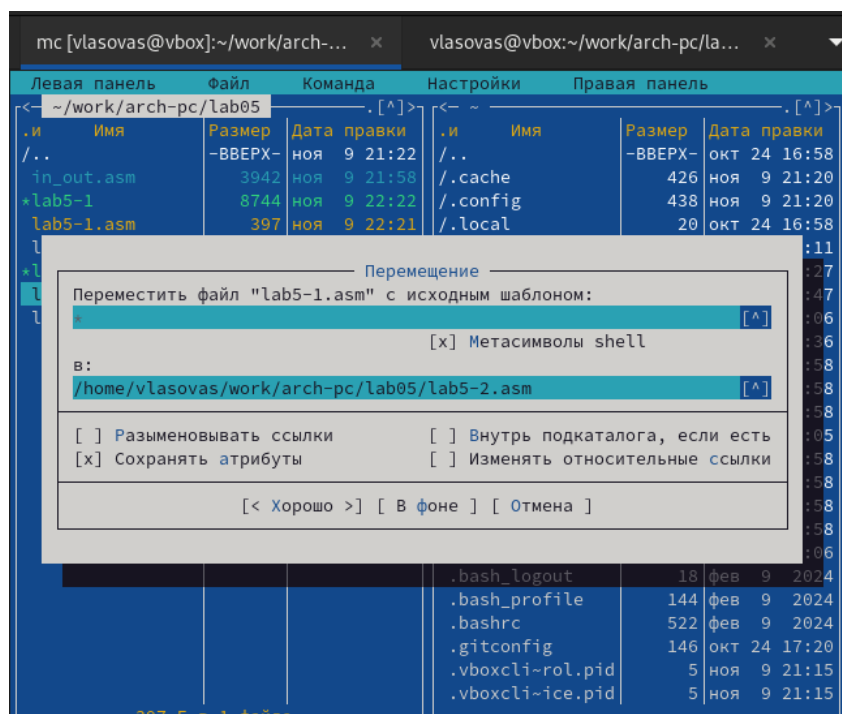
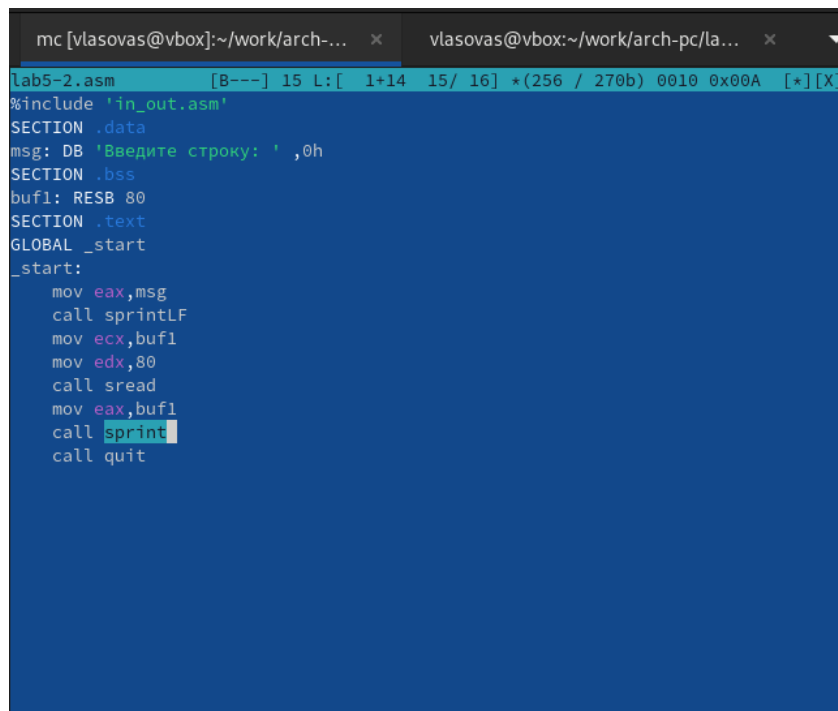


Рис. 3.19: Создаем копию файла lab5-2.asm

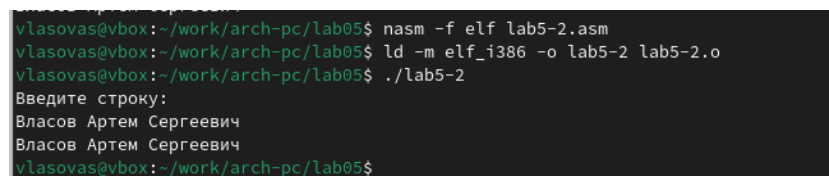
Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль (рис. fig. 3.20).



```
mc [vlasovas@vbox]:~/work/arch-... x  vlasovas@vbox:~/work/arch-pc/la... x
lab5-2.asm [B---] 15 L: [ 1+14 15/ 16] *(256 / 270b) 0010 0x00A [*][X]
#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 sprintf
    mov ecx,buf1
    mov edx,80
    call sread
    mov eax,buf1
    call sprint
    call quit
```

Рис. 3.20: Редактируем файл

Транслируем файл и запускаем программу (рис. fig. 3.21).



```
vlasovas@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
vlasovas@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
vlasovas@vbox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Власов Артем Сергеевич
Власов Артем Сергеевич
vlasovas@vbox:~/work/arch-pc/lab05$
```

Рис. 3.21: Проверяем правильность написания программы

4 Выводы

Мы приобрели навыки работы с Midnight Commander и освоили инструкцию mov.