

# **Отчёт по лабораторной работе 5**

**Архитектура компьютера**

Цыкунова Екатерина Михайловна НКАбд-05-24

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
4.1	Знакомство с Midnight Commander . . . . .	8
4.2	Подключение внешнего файла in_out.asm . . . . .	13
4.3	Задание для самостоятельной работы . . . . .	19
<b>5</b>	<b>Выводы</b>	<b>23</b>

## Список иллюстраций

4.1	Запуск Midnight Commander . . . . .	8
4.2	Создание каталога . . . . .	9
4.3	Создание файла lab05-1.asm . . . . .	10
4.4	Программа lab05-1.asm . . . . .	11
4.5	Просмотр файла lab05-1.asm . . . . .	12
4.6	Запуск программы lab05-1.asm . . . . .	13
4.7	Копирование файла in_out.asm . . . . .	14
4.8	Копирование файла lab05-1.asm . . . . .	15
4.9	Программа lab05-2.asm . . . . .	16
4.10	Запуск программы lab05-2.asm . . . . .	17
4.11	Программа в файле lab05-2.asm . . . . .	18
4.12	Запуск программы lab05-2.asm . . . . .	18
4.13	Программа lab05-3.asm . . . . .	20
4.14	Запуск программы lab05-3.asm . . . . .	20
4.15	Программа lab05-4.asm . . . . .	21
4.16	Запуск программы lab05-4.asm . . . . .	22

## Список таблиц

# 1 Цель работы

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

## 2 Задание

1. Установить Midnight Commander
2. Изучить структуру программ
3. Изучить файл in\_out.asm
4. Дополнить программы по заданию.

### 3 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной.

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).

## 4 Выполнение лабораторной работы

### 4.1 Знакомство с Midnight Commander

Открываю Midnight Commander (рис. 4.1), с помощью клавиш со стрелками и Enter перехожу в каталог ~/work/arch-рс. Далее нажимаю F7 и создаю каталог lab05 (рис. 4.2).

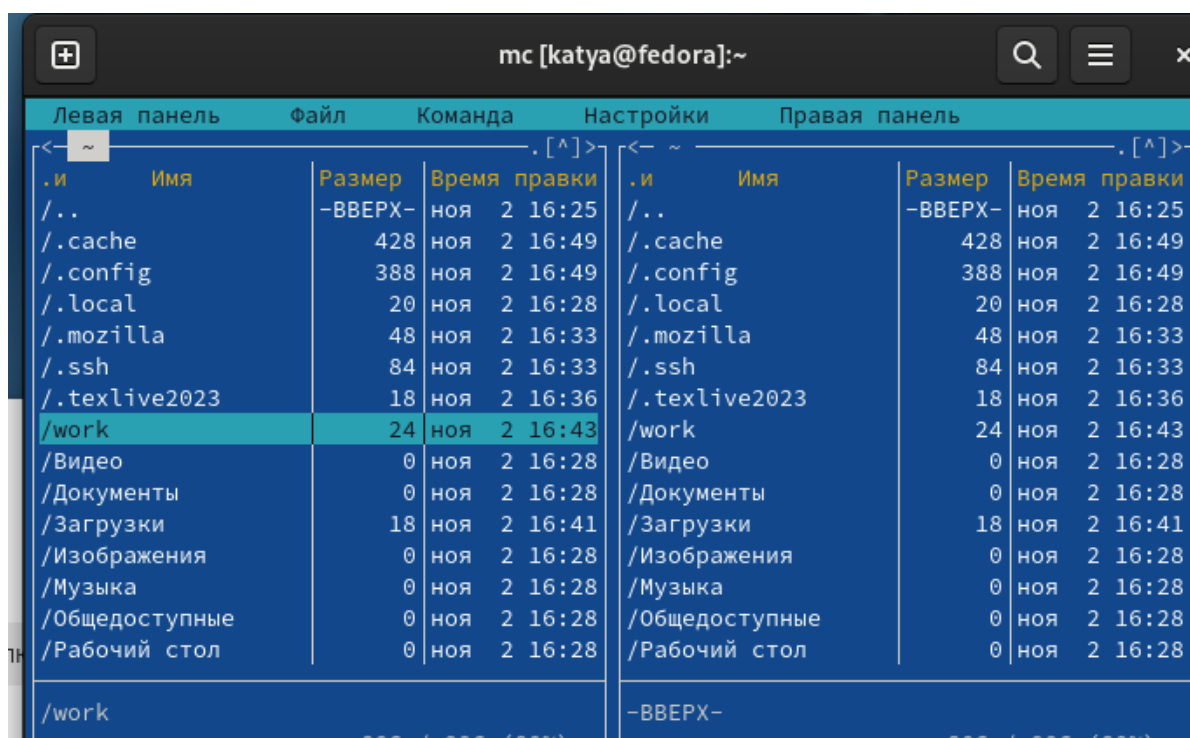


Рис. 4.1: Запуск Midnight Commander



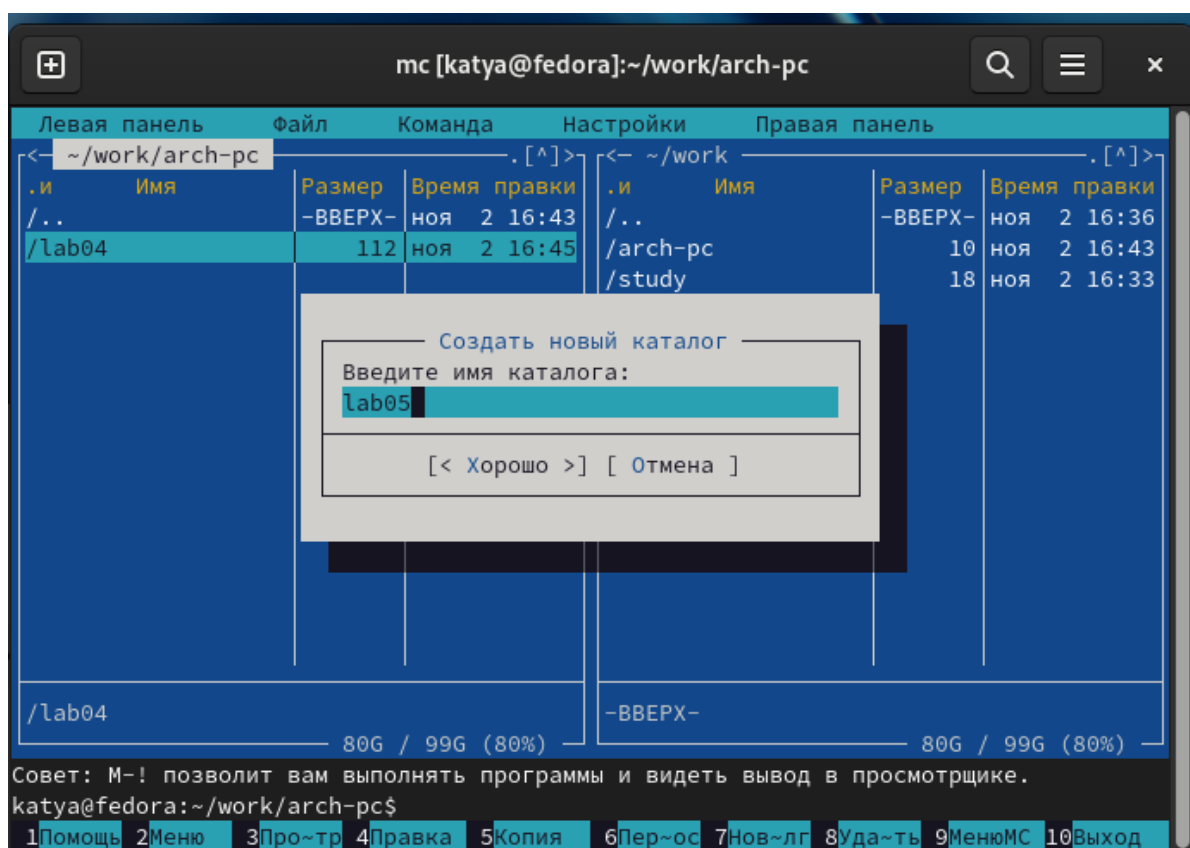


Рис. 4.2: Создание каталога

При помощи touch создаю файл lab05-1.asm (рис. 4.3)

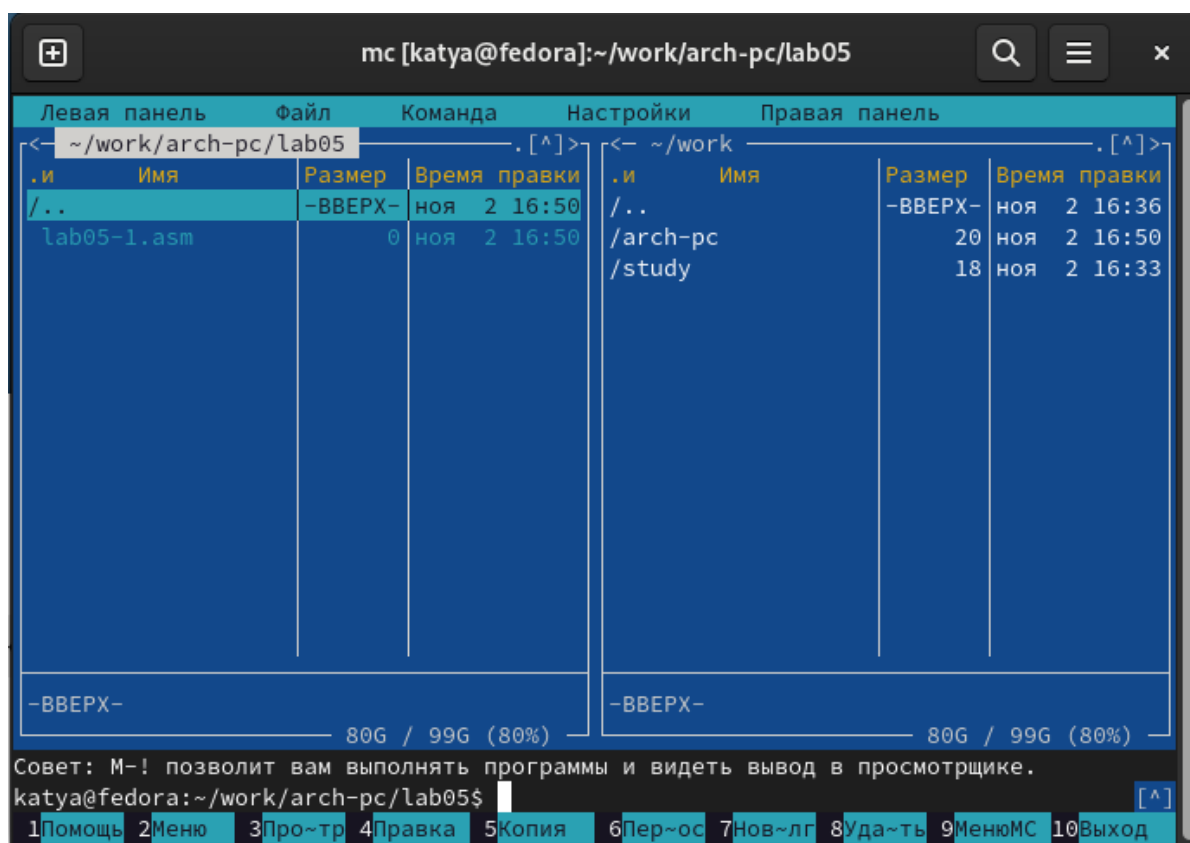
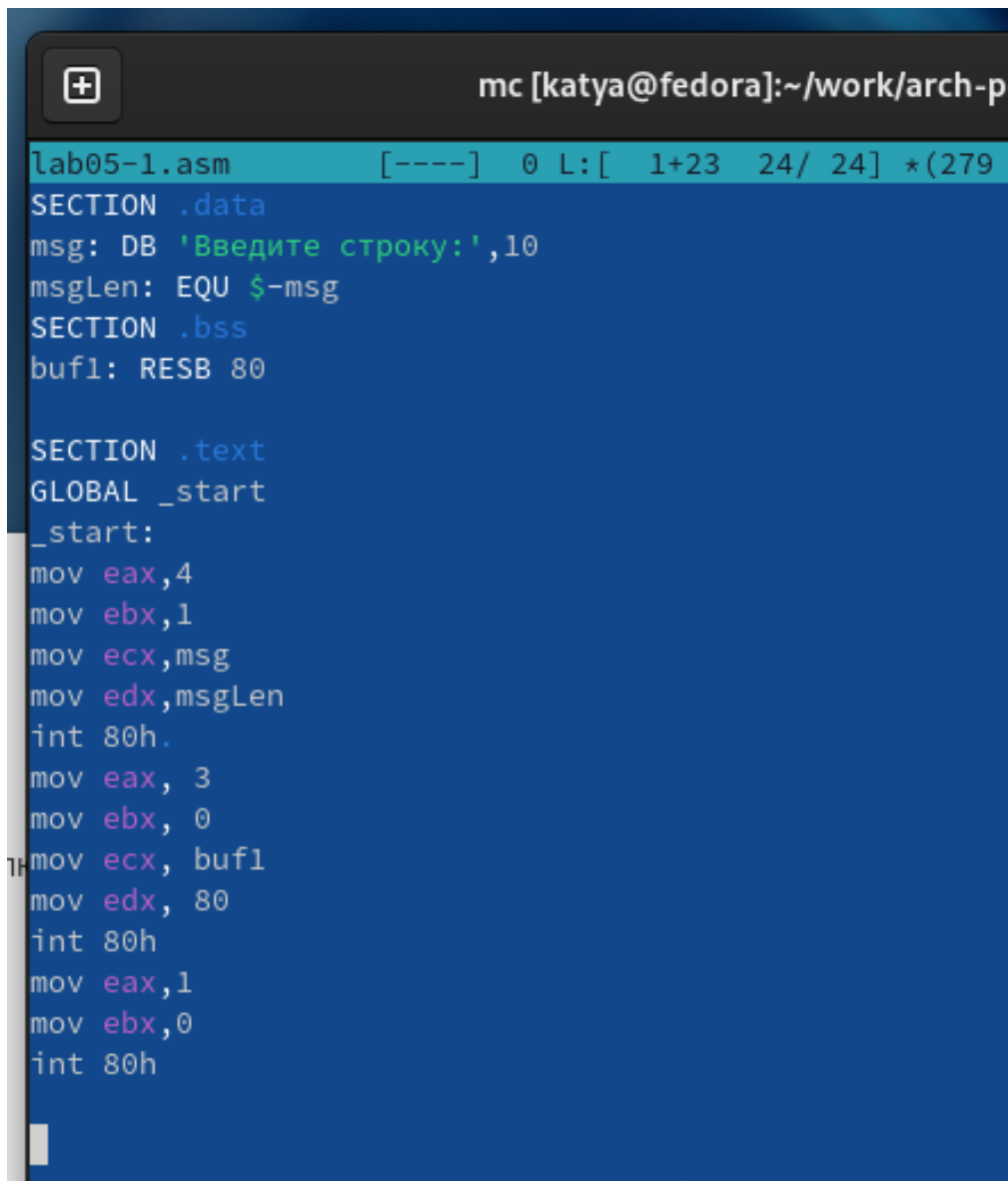


Рис. 4.3: Создание файла lab05-1.asm

Открываю файл на редактирование клавишей F4, выбираю редактор mcedit, пишу код программы из задания. (рис. 4.4)

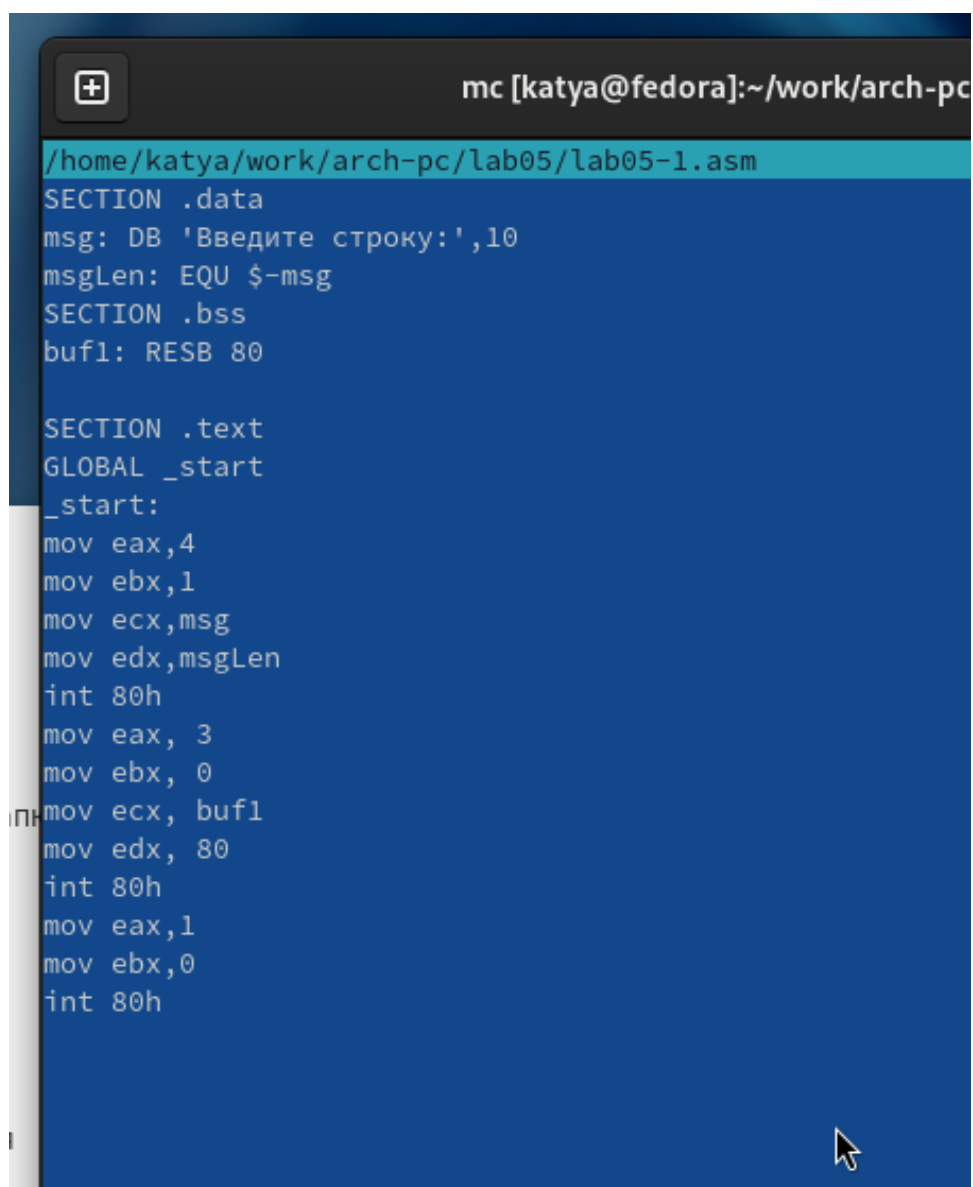


```
lab05-1.asm [----] 0 L:[ 1+23 24/ 24] *(279
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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,1
mov ebx,0
int 80h
```

Рис. 4.4: Программа lab05-1.asm

Открываю файл на просмотр клавишей F3 и проверяю, что он содержит набранный код. (рис. 4.5)



```
mc [katya@fedora]:~/work/arch-pc
/home/katya/work/arch-pc/lab05/lab05-1.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
int 80h
mov eax,1
mov ebx,0
int 80h
```

Рис. 4.5: Просмотр файла lab05-1.asm

Транслирую файл программы в объектный файл, выполняю компоновку объектного файла, получился исполняемый файл программы. (рис. 4.6)

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



Рис. 4.6: Запуск программы lab05-1.asm

## 4.2 Подключение внешнего файла in\_out.asm

Скачиваю файл in\_out.asm и размещаю его в рабочем каталоге. (рис. 4.7) Для копирования используется клавиша F5. Для перемещения используется клавиша F6.

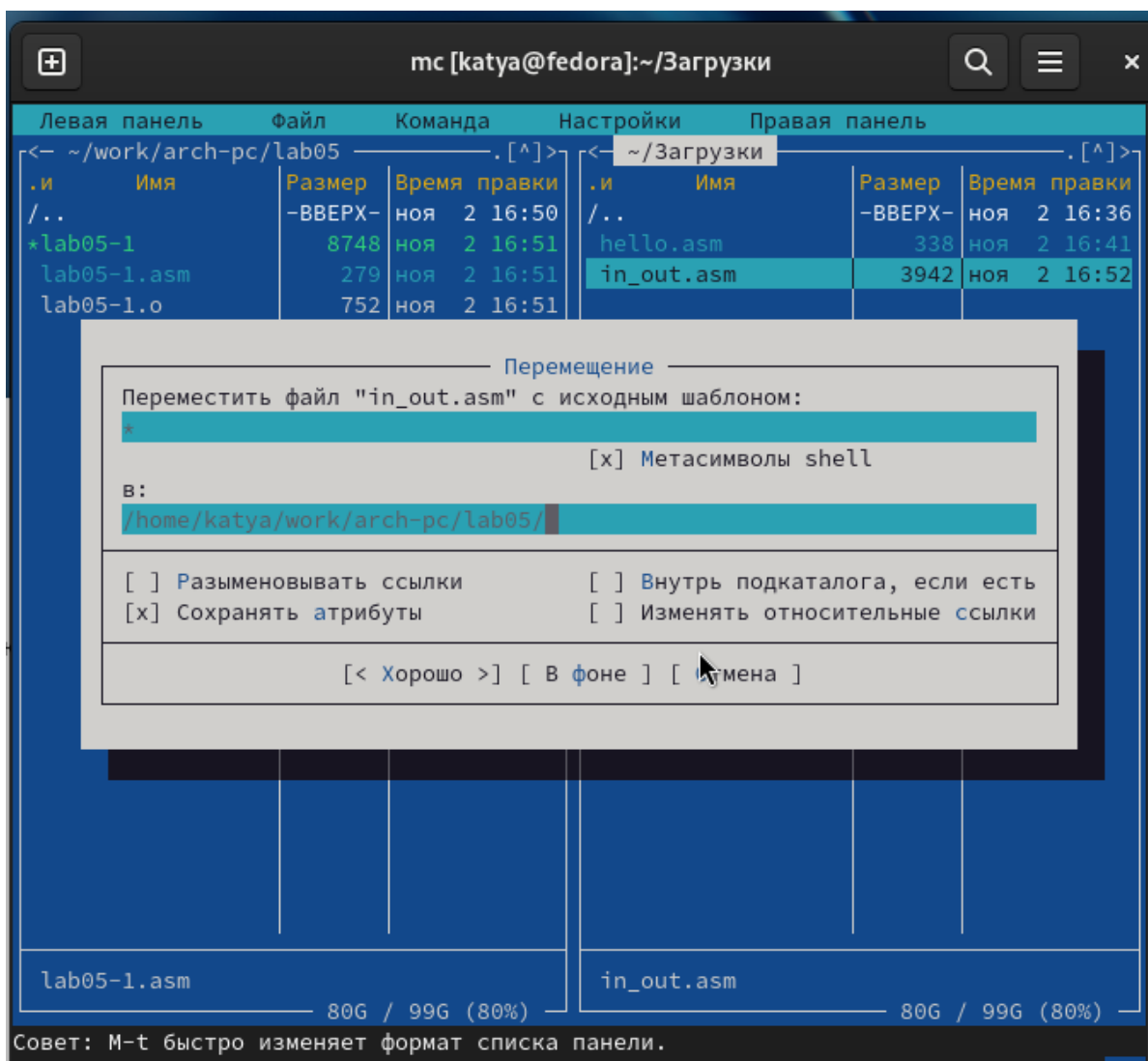


Рис. 4.7: Копирование файла in\_out.asm

Скопировала lab05-1.asm в lab05-2.asm. (рис. 4.8)

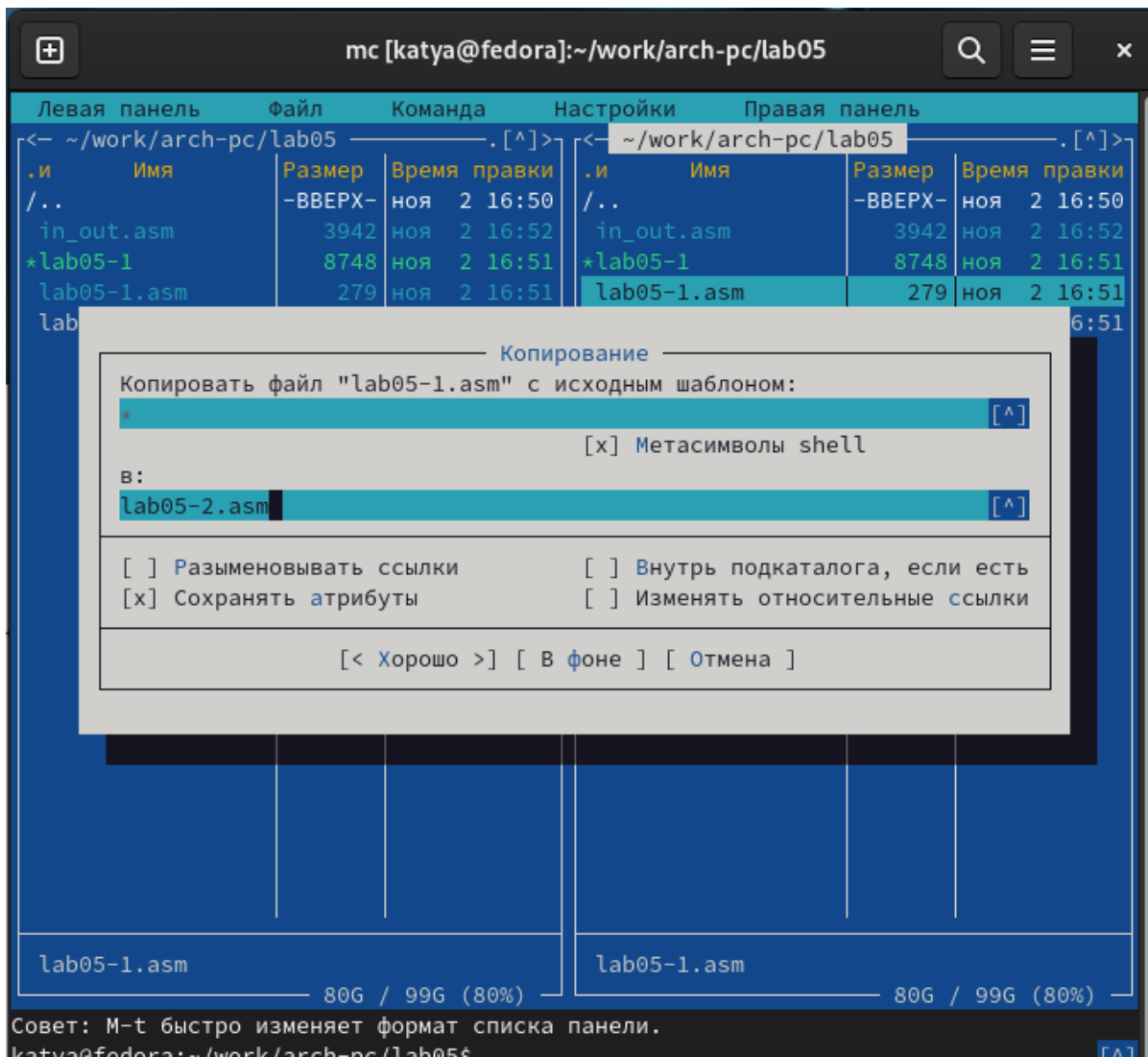
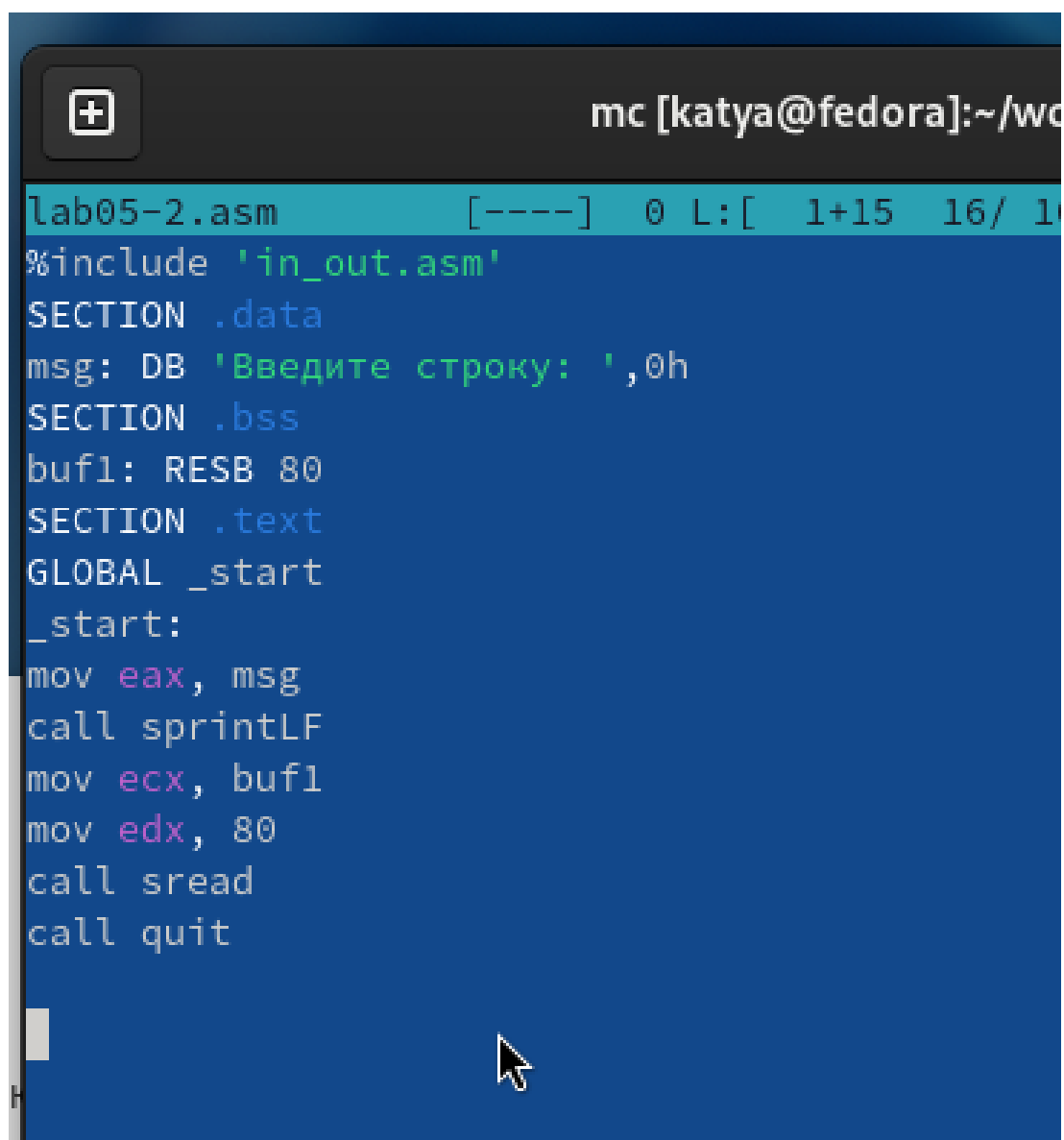


Рис. 4.8: Копирование файла lab05-1.asm

Пишу код программы lab05-2.asm с использованием подпрограмм из внешнего файла in\_out.asm. (рис. 4.9)



```
lab05-2.asm      [-----]  0  L:[  1+15  16/ 10
#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. 4.9: Программа lab05-2.asm

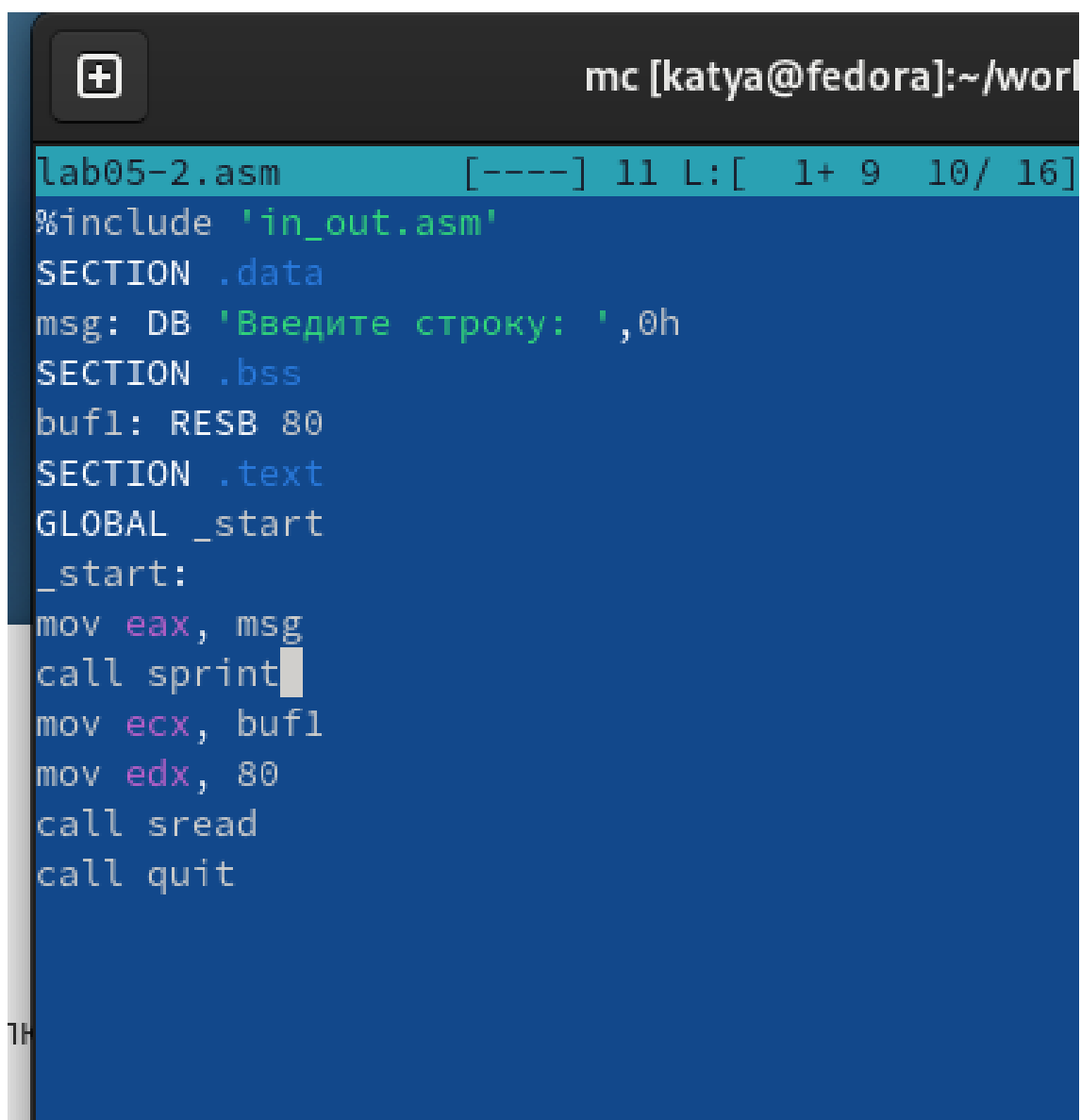
Скомпилирую программу и проверю запуск. (рис. 4.10)



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

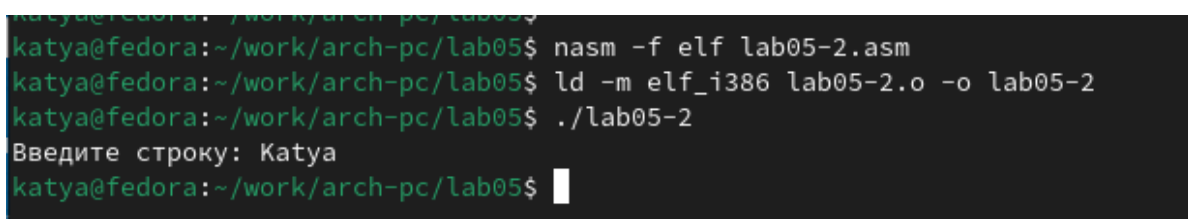
Рис. 4.10: Запуск программы lab05-2.asm

В файле lab5-2.asm заменила подпрограмму sprintLF на sprint. Заново собрала исполняемый файл. (рис. 4.11) (рис. 4.12)



```
mc [katya@fedora]:~/work
lab05-2.asm [----] 11 L: [ 1+ 9 10/ 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 sprint
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 4.11: Программа в файле lab05-2.asm



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

Рис. 4.12: Запуск программы lab05-2.asm

Теперь после вывода строки она не завершается символом перехода на новую

строку.

### **4.3 Задание для самостоятельной работы**

Скопировала программу lab05-1.asm и изменила код, так чтобы она работала по следующему алгоритму: (рис. 4.13) (рис. 4.14)

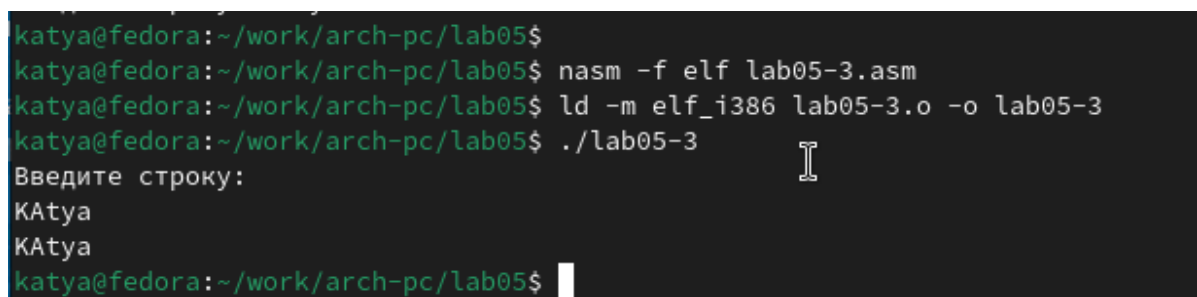
- вывести приглашение типа “Введите строку:”;
- ввести строку с клавиатуры;
- вывести введённую строку на экран.



```
mc [katya@fedora]:~/work/arch-pc/
lab05-3.asm [----] 7 L: [ 1+23 24/ 27] *(302
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
```

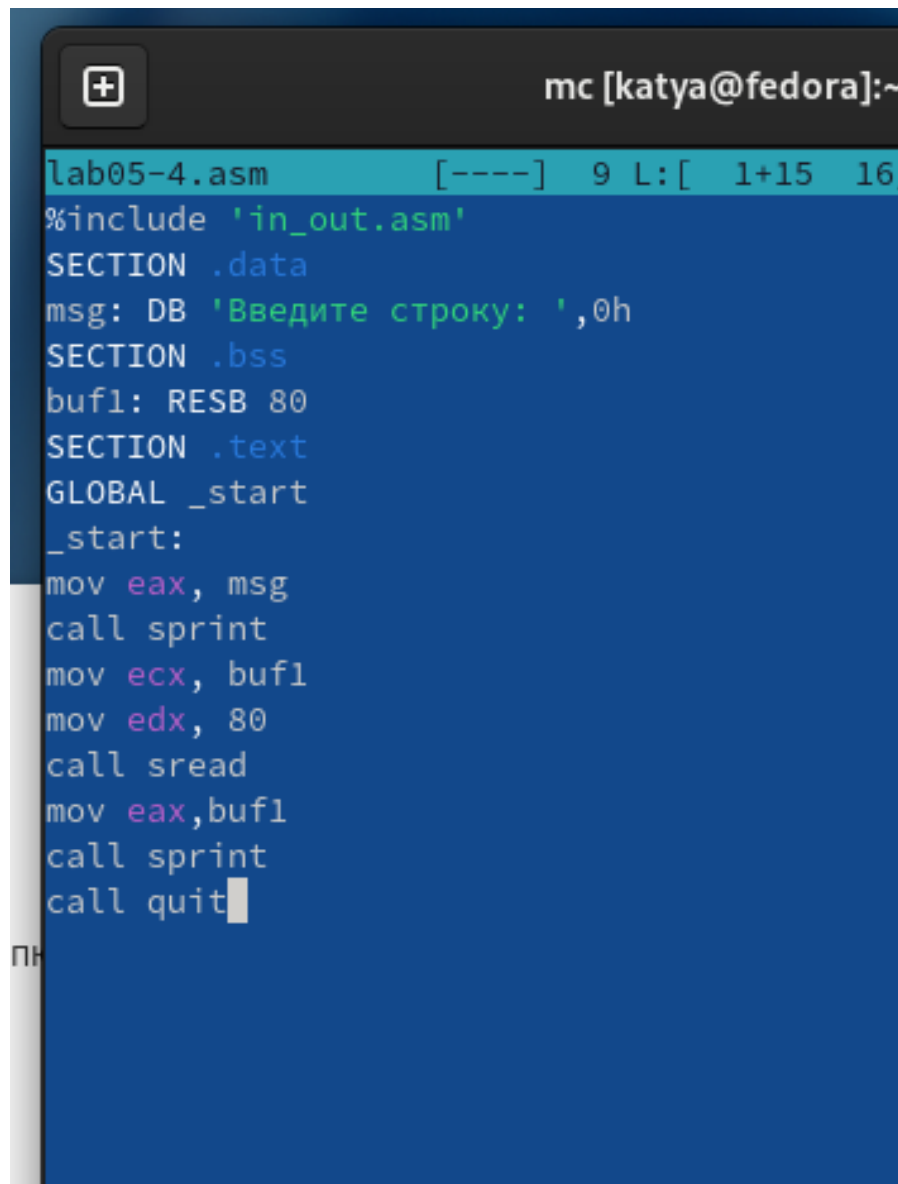
Рис. 4.13: Программа lab05-3.asm



```
katya@fedora:~/work/arch-pc/lab05$
katya@fedora:~/work/arch-pc/lab05$ nasm -f elf lab05-3.asm
katya@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
katya@fedora:~/work/arch-pc/lab05$ ./lab05-3
Введите строку:
KAtya
KAtya
katya@fedora:~/work/arch-pc/lab05$
```

Рис. 4.14: Запуск программы lab05-3.asm

Аналогично скопировала программу lab05-2.asm и изменила код, но теперь использовал подпрограммы из файла in\_out.asm. (рис. 4.15) (рис. 4.16)



```
mc [katya@fedora]:~
lab05-4.asm [----] 9 L: [ 1+15 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 sprint
mov ecx, buf1
mov edx, 80
call sread
mov eax, buf1
call sprint
call quit
```

Рис. 4.15: Программа lab05-4.asm

```
katya@fedora:~/work/arch-pc/lab05$  
katya@fedora:~/work/arch-pc/lab05$ nasm -f elf lab05-4.asm  
katya@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-4.o -o lab05-4  
katya@fedora:~/work/arch-pc/lab05$ ./lab05-4  
Введите строку: Katya  
Katya  
katya@fedora:~/work/arch-pc/lab05$
```

Рис. 4.16: Запуск программы lab05-4.asm

## 5 Выводы

Научились писать базовые ассемблерные программы. Освоили ассемблерные инструкции `mov` и `int`.