

Отчёт по лабораторной работе 5

Дисциплина: архитектура компьютера

Синюшко Элза Камировна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Знакомство с Midnight Commander	8
4.2	Подключение внешнего файла in_out.asm	14
4.3	Задание для самостоятельной работы	18
5	Выводы	22
6	Источники	23

Список иллюстраций

4.1	Запуск Midnight Commander	8
4.2	Перехожу в рабочую папку	9
4.3	Создание каталога	10
4.4	Создание файла lab05-1.asm	11
4.5	Программа lab05-1.asm	12
4.6	Просмотр файла lab05-1.asm	13
4.7	Запуск программы lab05-1.asm	14
4.8	Копирование файла in_out.asm	15
4.9	Копирование файла lab05-1.asm	16
4.10	Программа lab05-2.asm	17
4.11	Запуск программы lab05-2.asm	17
4.12	Программа в файле lab05-2.asm	18
4.13	Запуск программы lab05-2.asm	18
4.14	Программа lab05-3.asm	19
4.15	Запуск программы lab05-3.asm	20
4.16	Программа lab05-4.asm	20
4.17	Запуск программы lab05-4.asm	21

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, с помощью клавиш со стрелками и Enter перехожу в каталог ~/work/arch-рс. Далее нажимаю F7 и создаю каталог lab05.

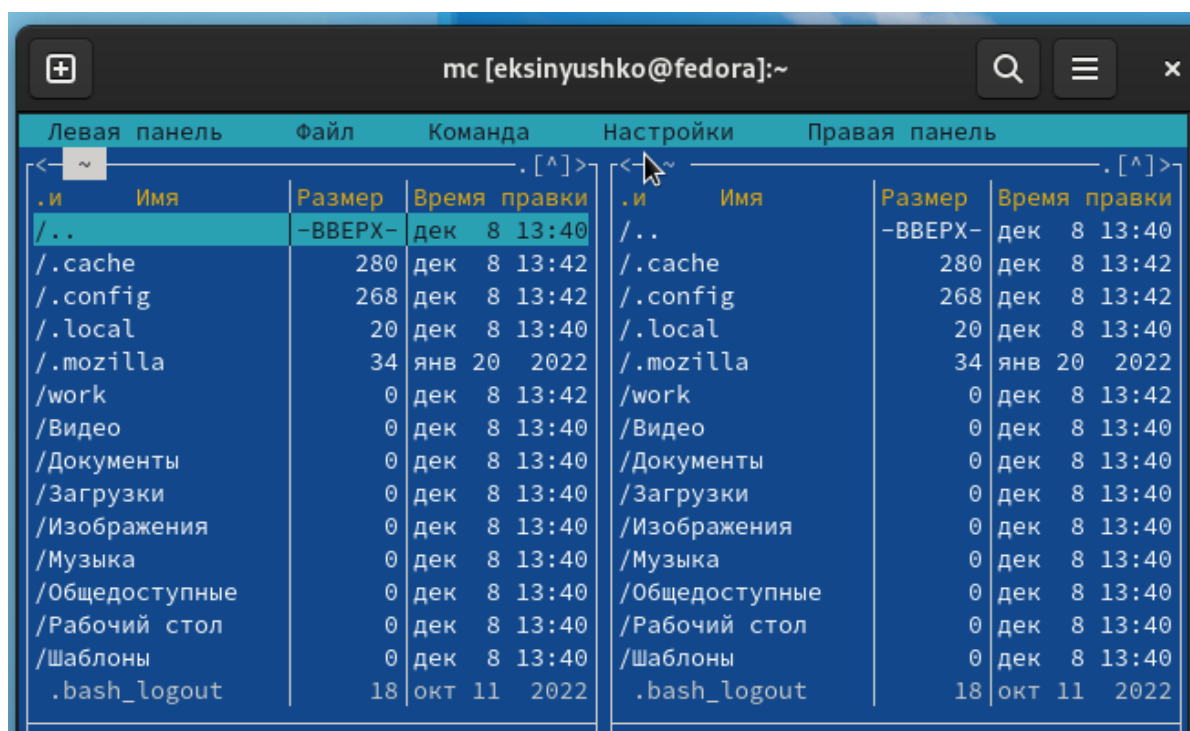


Рис. 4.1: Запуск Midnight Commander

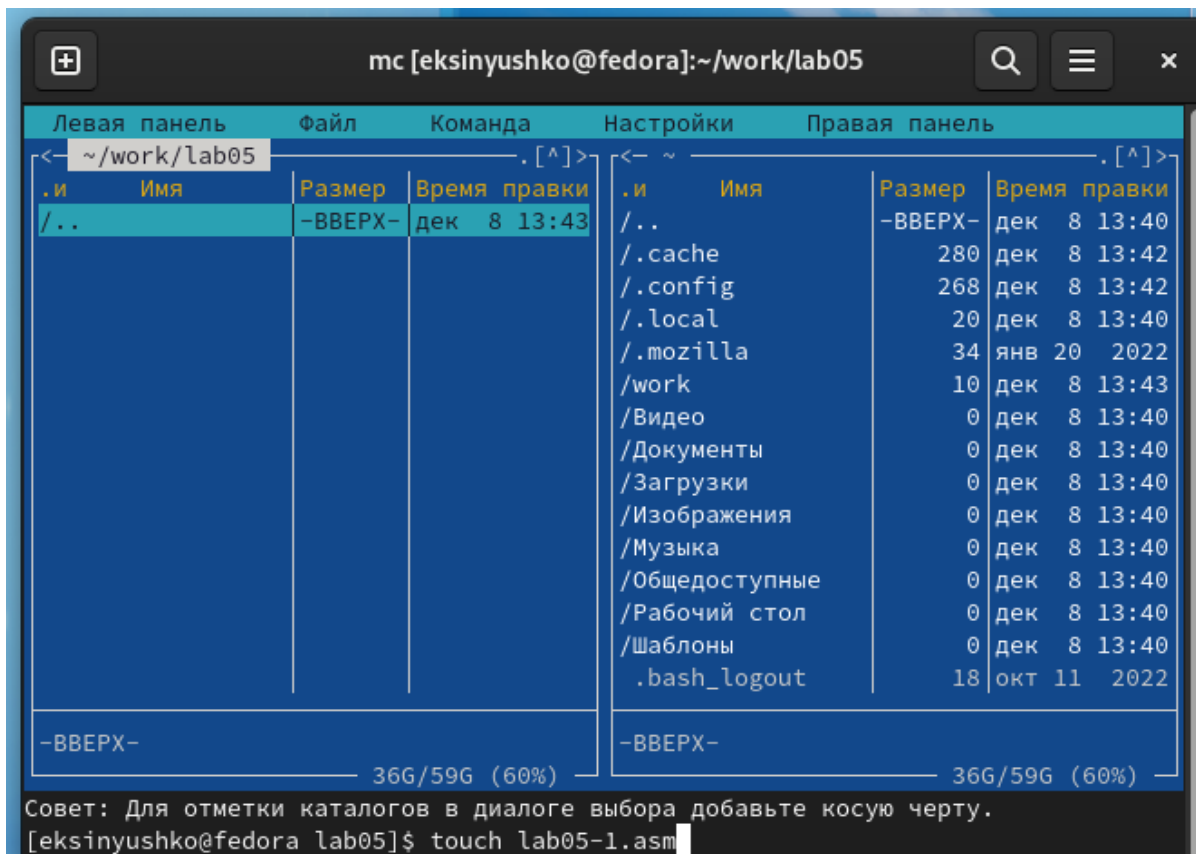
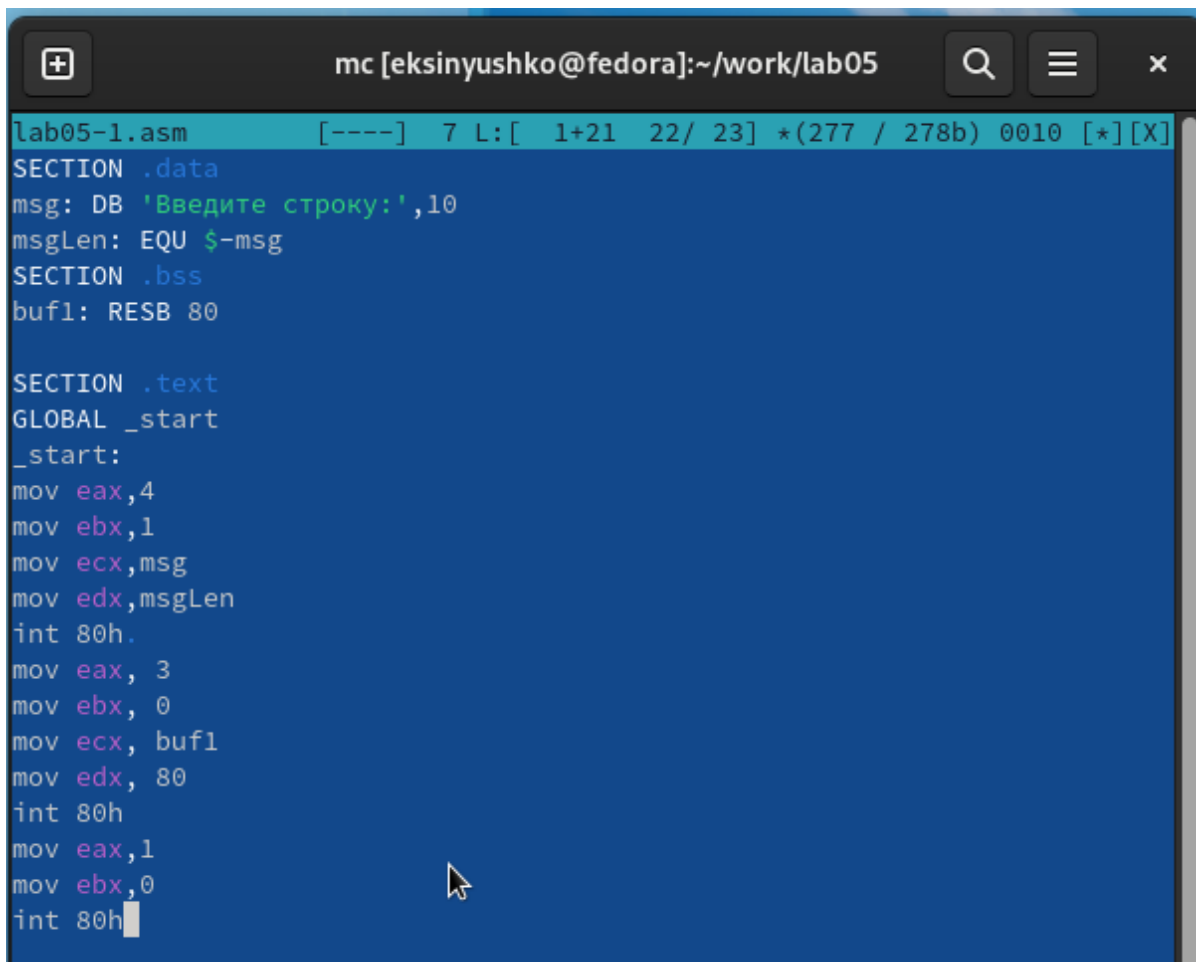


Рис. 4.4: Создание файла lab05-1.asm

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

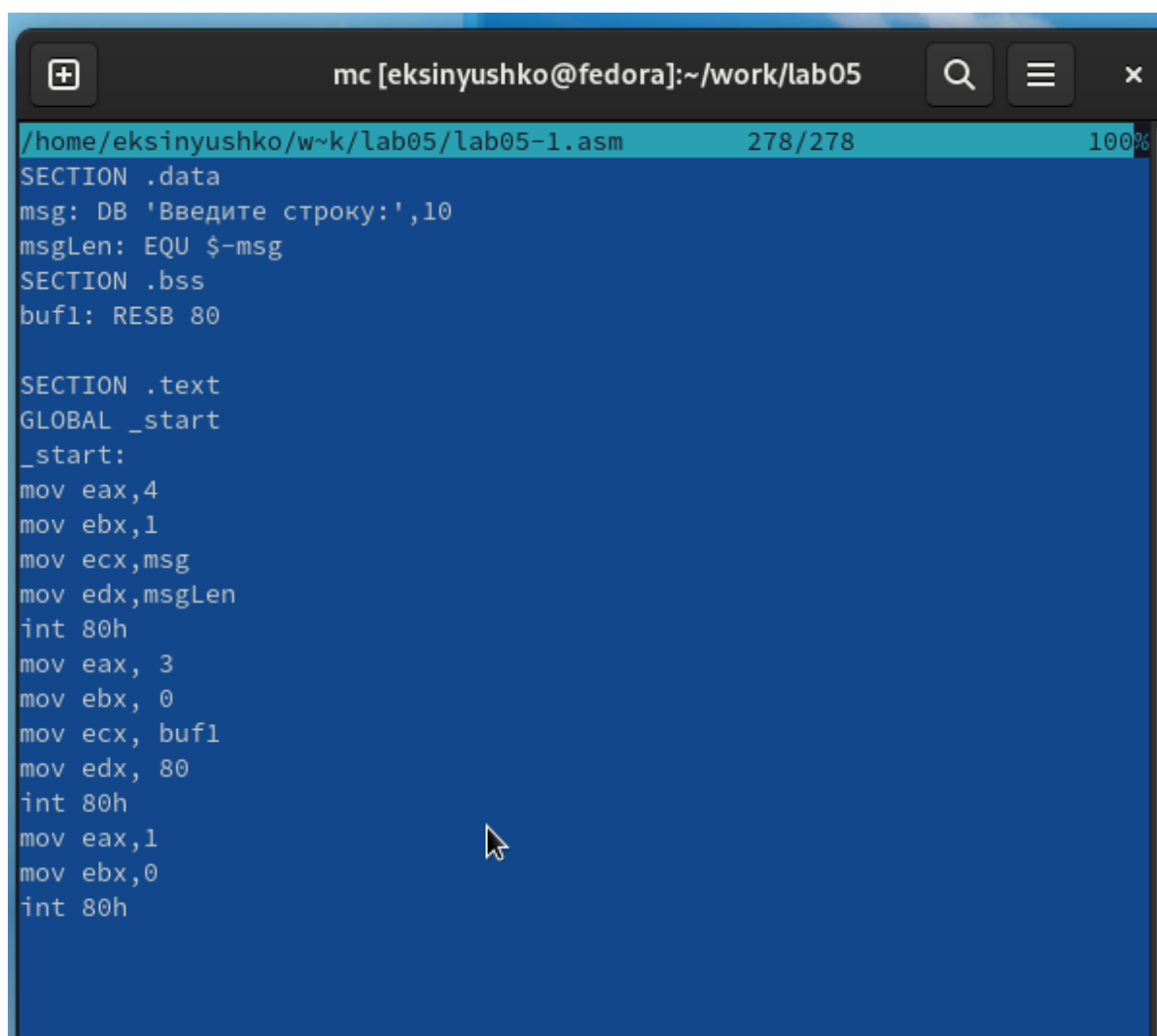


```
lab05-1.asm [----] 7 L: [ 1+21 22/ 23] *(277 / 278b) 0010 [*] [X]
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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

Рис. 4.5: Программа lab05-1.asm

Открываю файл на просмотр клавишей F3 и проверяю, что он содержит набранный код.

A screenshot of a code editor window titled 'mc [eksinyushko@fedora]:~/work/lab05'. The editor displays the contents of the file '/home/eksinyushko/w~k/lab05/lab05-1.asm', which is 278 lines long and shown at 100% zoom. The code is written in assembly language and is divided into three sections: .data, .bss, and .text. The .data section contains a message string and its length. The .bss section reserves space for a buffer. The .text section contains the main logic of the program, starting with a global _start function. The program prints the message 'Введите строку:' and then enters a loop where it reads user input into a buffer and prints it back. The code uses standard x86-64 assembly instructions like mov, int, and equ.

```
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. 4.6: Просмотр файла lab05-1.asm

Транслирую файл программы в объектный файл, выполняю компоновку объектного файла, получился исполняемый файл программы.

```
[eksinyushko@fedora lab05]$ nasm -f elf lab05-1.asm
[eksinyushko@fedora lab05]$ ld -m elf_i386 lab05-1.o -o lab05-1
[eksinyushko@fedora lab05]$ ./lab05-1
Введите строку:
Elza
[eksinyushko@fedora lab05]$
```

Рис. 4.7: Запуск программы lab05-1.asm

4.2 Подключение внешнего файла in_out.asm

Скачиваю файл in_out.asm и размещаю его в рабочем каталоге. Для копирования используется клавиша F5. Для перемещения используется клавиша F6.

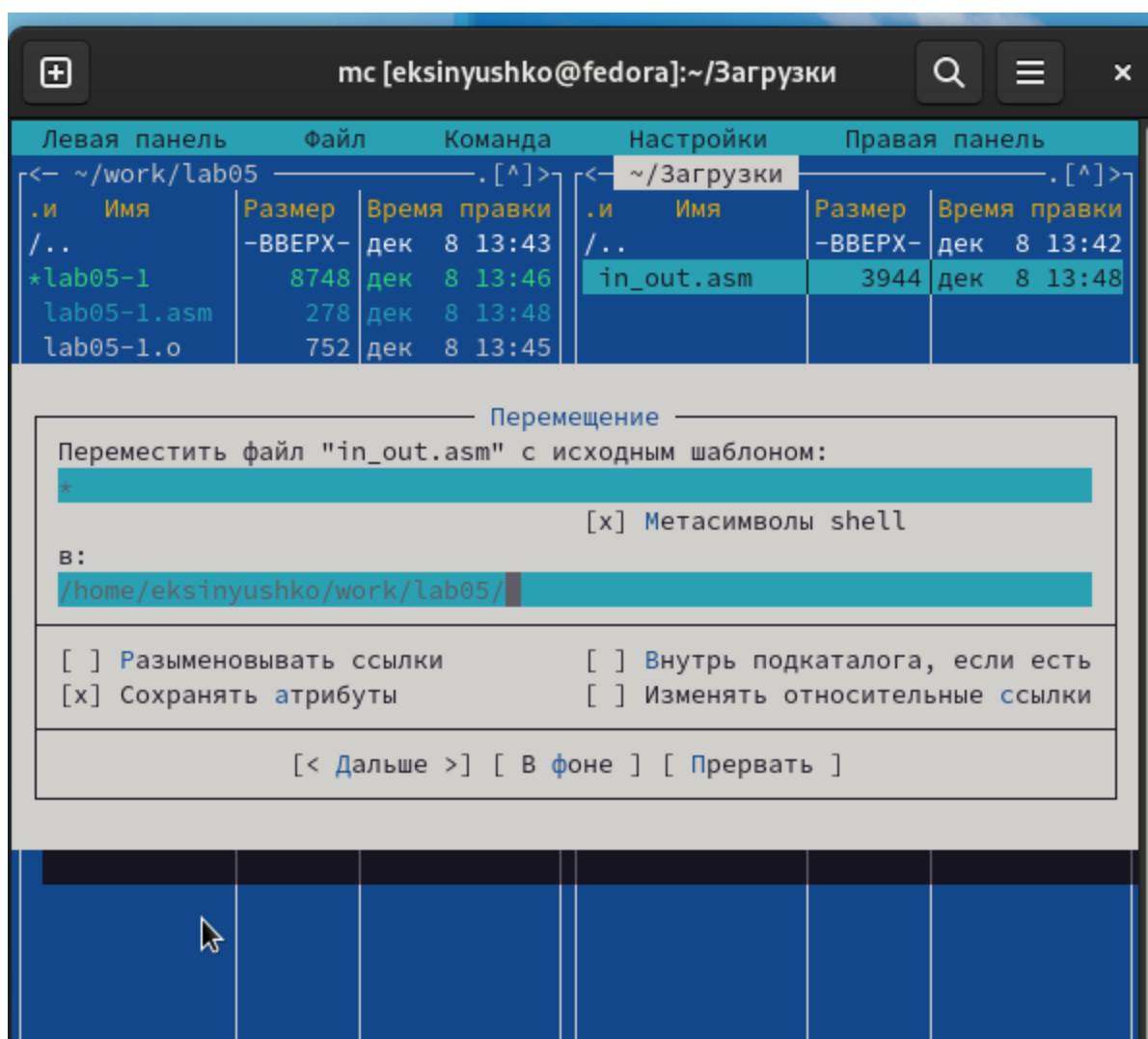
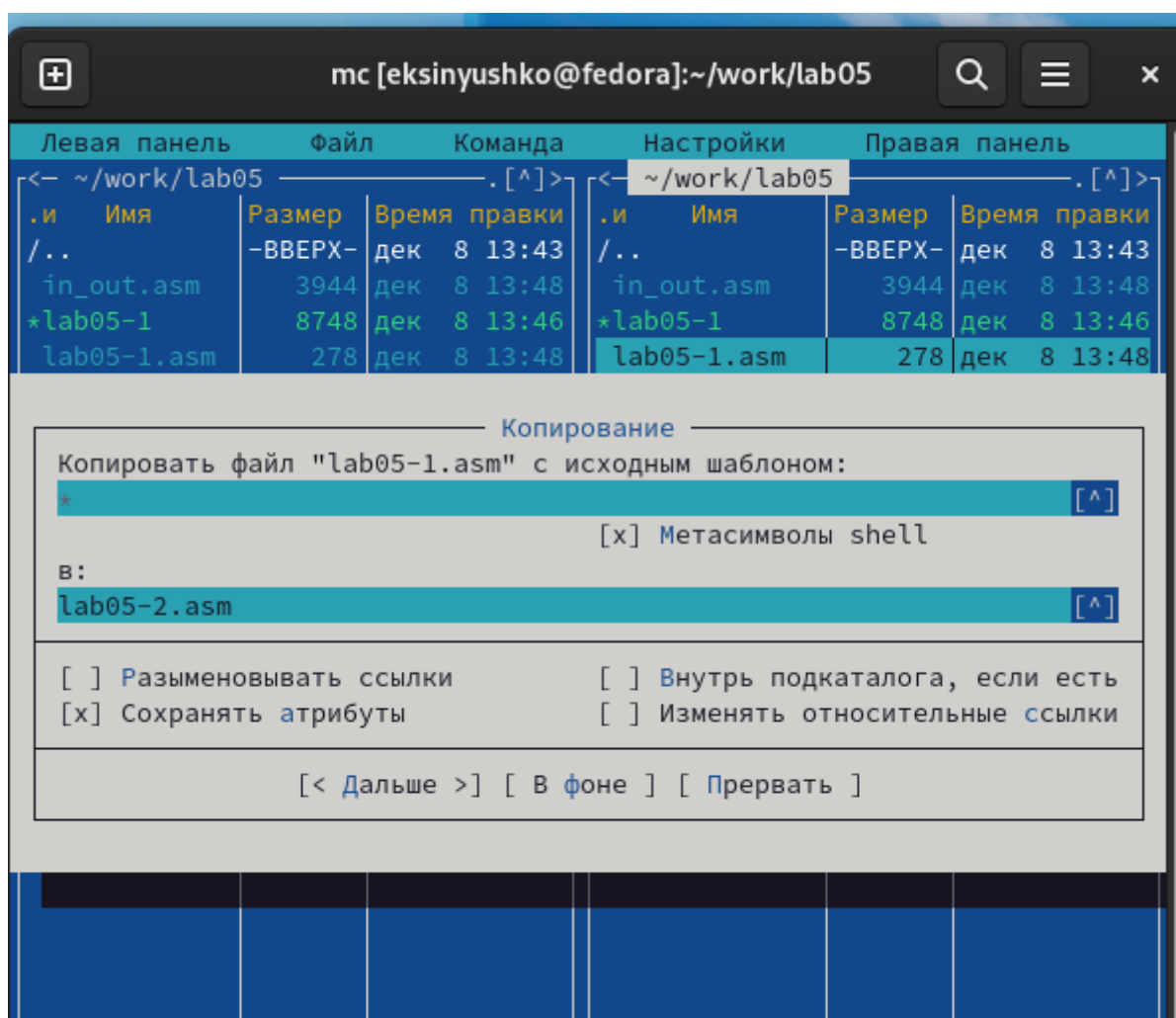


Рис. 4.8: Копирование файла in_out.asm

Скопировала lab05-1.asm в lab05-2.asm.



Копирование

Копировать файл "lab05-1.asm" с исходным шаблоном:

*

[x] Метасимволы shell

в:

lab05-2.asm

[] Разыменовывать ссылки

[] Внутрь подкаталога, если есть

[x] Сохранять атрибуты

[] Изменять относительные ссылки

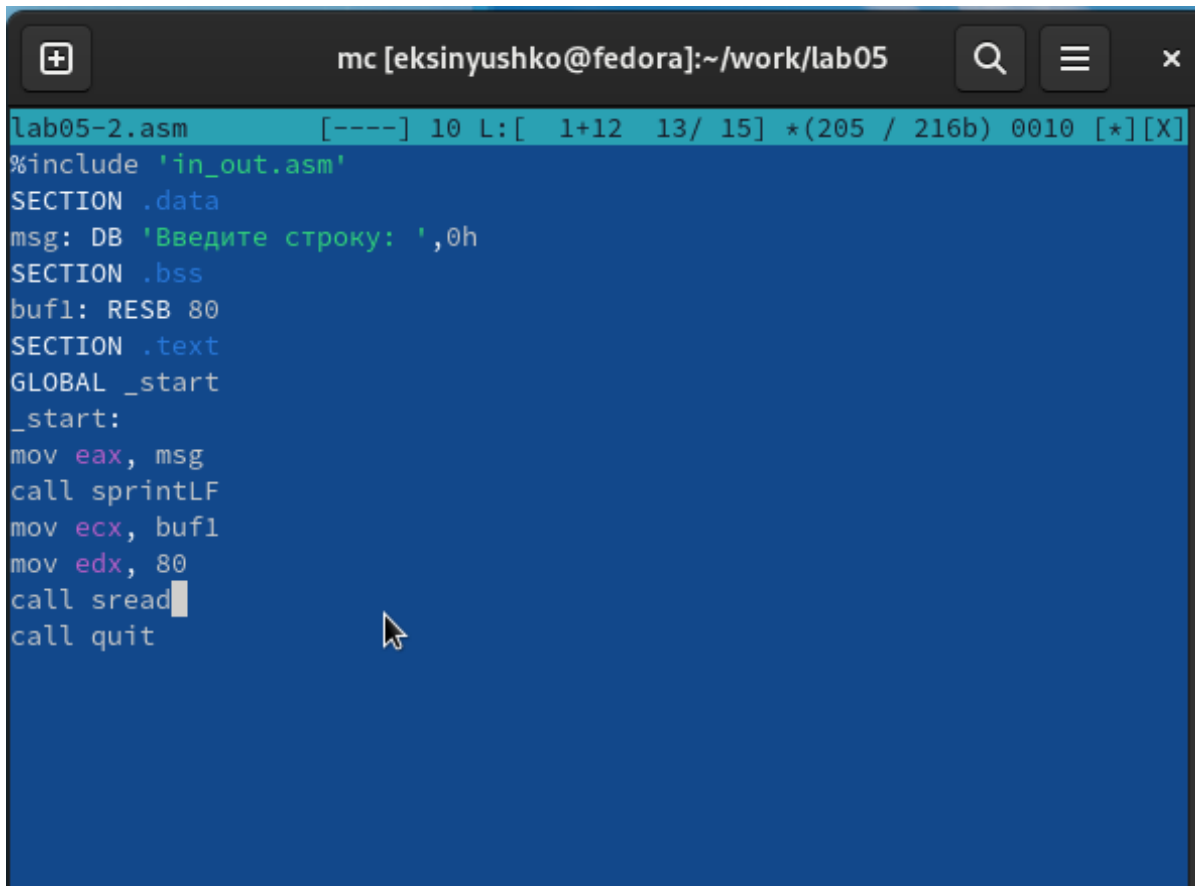
[< Дальше >]

[В фоне]

[Прервать]

Рис. 4.9: Копирование файла lab05-1.asm

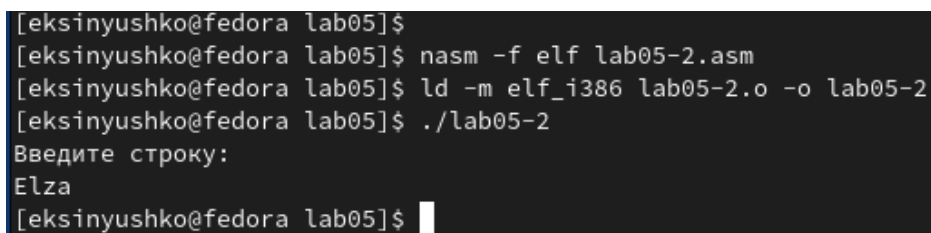
Пишу код программы lab05-2.asm с использованием подпрограмм из внешнего файла in_out.asm.



```
lab05-2.asm [----] 10 L: [ 1+12 13/ 15] *(205 / 216b) 0010 [*] [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. 4.10: Программа lab05-2.asm

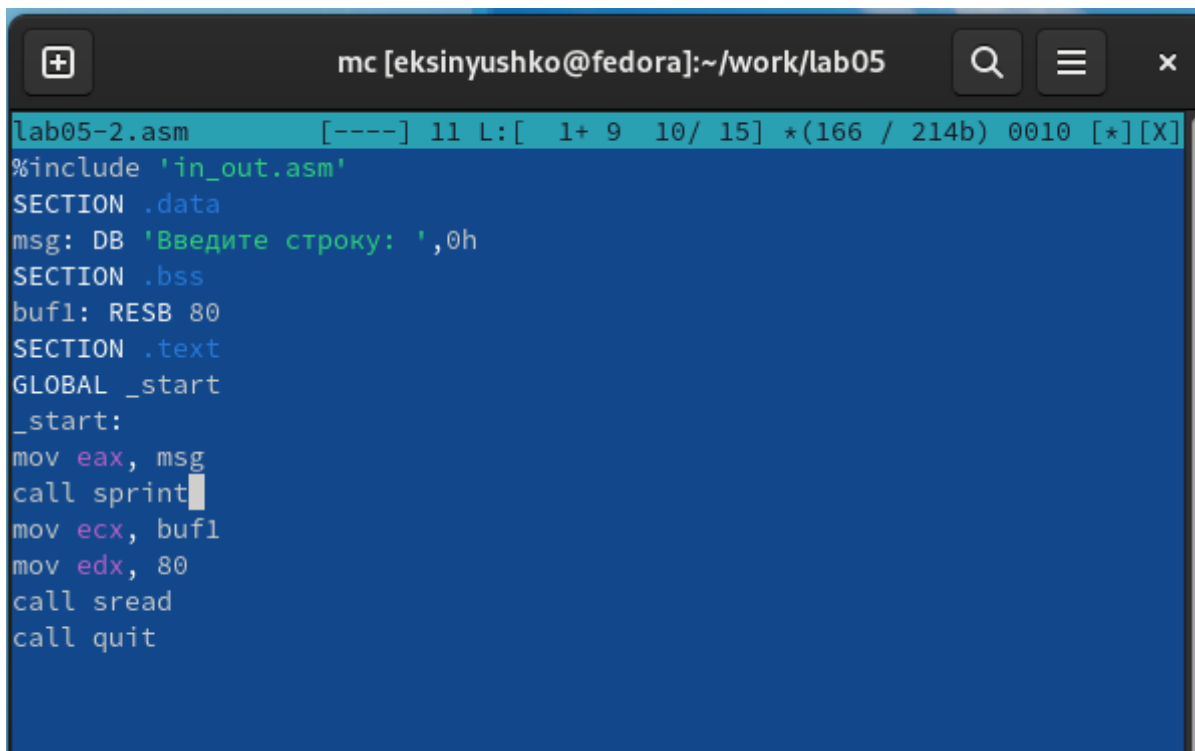
Скомпилирую программу и проверю запуск.



```
[eksinyushko@fedora lab05]$
[eksinyushko@fedora lab05]$ nasm -f elf lab05-2.asm
[eksinyushko@fedora lab05]$ ld -m elf_i386 lab05-2.o -o lab05-2
[eksinyushko@fedora lab05]$ ./lab05-2
Введите строку:
Elza
[eksinyushko@fedora lab05]$
```

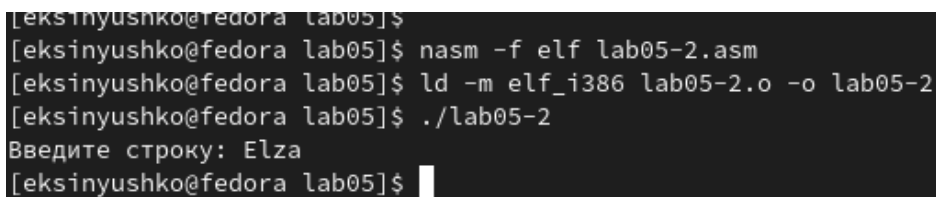
Рис. 4.11: Запуск программы lab05-2.asm

В файле lab5-2.asm заменила подпрограмму sprintLF на sprint. Заново собрала исполняемый файл.



```
lab05-2.asm [----] 11 L:[ 1+ 9 10/ 15] *(166 / 214b) 0010 [*][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. 4.12: Программа в файле lab05-2.asm



```
[eksinyushko@fedora lab05]$
[eksinyushko@fedora lab05]$ nasm -f elf lab05-2.asm
[eksinyushko@fedora lab05]$ ld -m elf_i386 lab05-2.o -o lab05-2
[eksinyushko@fedora lab05]$ ./lab05-2
Введите строку: Elza
[eksinyushko@fedora lab05]$
```

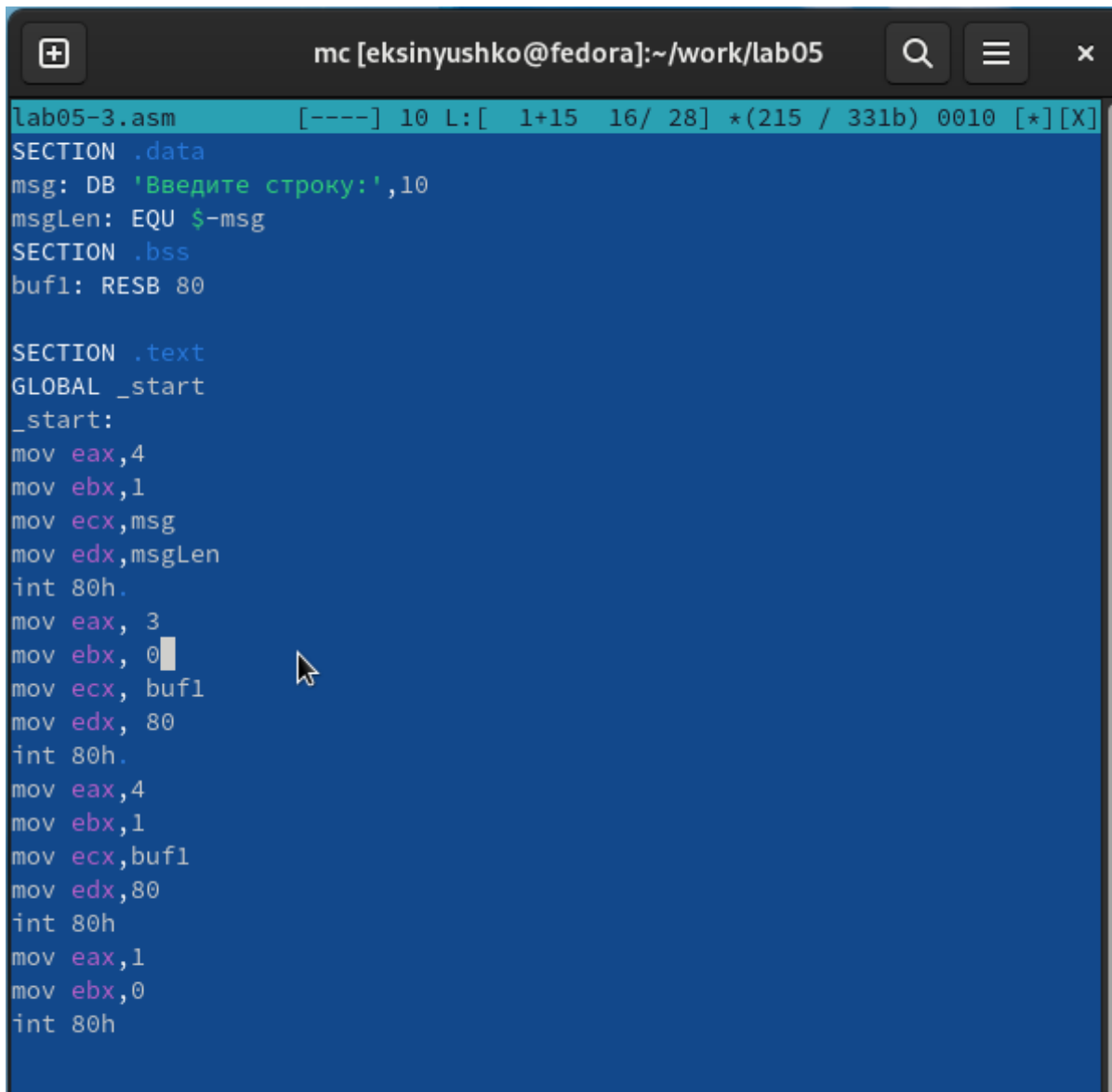
Рис. 4.13: Запуск программы lab05-2.asm

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:

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



```
mc [eksinyushko@fedora]:~/work/lab05
lab05-3.asm [----] 10 L:[ 1+15 16/ 28] *(215 / 331b) 0010 [*][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. 4.14: Программа lab05-3.asm

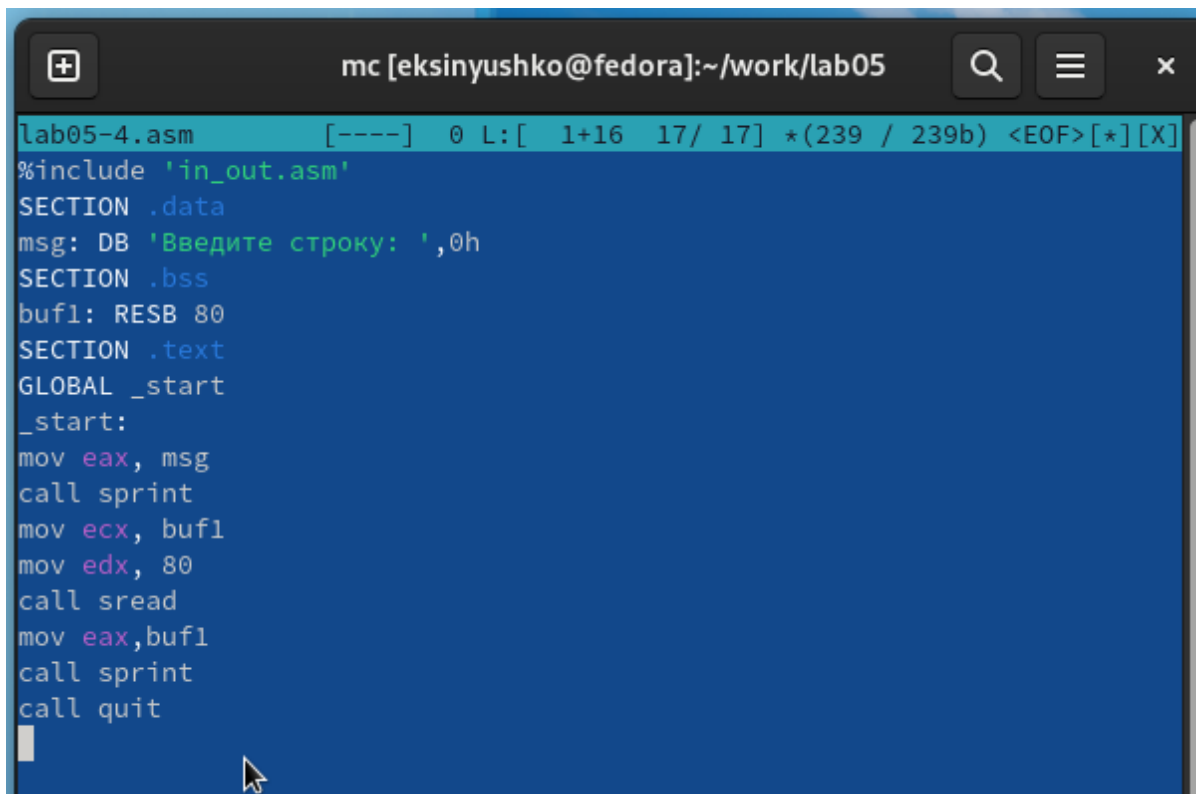
```

[eksinyushko@fedora lab05]$ nasm -f elf lab05-3.asm
[eksinyushko@fedora lab05]$ ld -m elf_i386 lab05-3.o -o lab05-3
[eksinyushko@fedora lab05]$ ./lab05-3
Введите строку:
Elza
Elza
[eksinyushko@fedora lab05]$
[eksinyushko@fedora lab05]$

```

Рис. 4.15: Запуск программы lab05-3.asm

Аналогично скопировала программу lab05-2.asm и изменила код, но теперь использовал подпрограммы из файла in_out.asm.



```

lab05-4.asm  [----]  0 L: [ 1+16 17/ 17] *(239 / 239b) <EOF>[*] [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
mov eax, buf1
call sprint
call quit

```

Рис. 4.16: Программа lab05-4.asm

```
[eksinyushko@fedora lab05]$ nasm -f elf lab05-4.asm
[eksinyushko@fedora lab05]$ ld -m elf_i386 lab05-4.o -o lab05-4
[eksinyushko@fedora lab05]$ ./lab05-4
Введите строку: Elza
Elza
[eksinyushko@fedora lab05]$
[eksinyushko@fedora lab05]$
```

Рис. 4.17: Запуск программы lab05-4.asm

5 Выводы

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

6 Источники

1. Архитектура ЭВМ - Материалы курса