

Отчёт по лабораторной работе 5

Архитектура компьютеров и операционные системы

Абдулфазова Лейла Али гызы

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	12
4.3	Выполнение заданий для самостоятельной работы	17
5	Выводы	20

Список иллюстраций

4.1	окно Midnight Commander	8
4.2	Создание каталога	9
4.3	Создала файл lab05-1.asm	10
4.4	Редактирование файла lab05-1.asm	11
4.5	Проверка кода lab05-1.asm	12
4.6	Тестирование программы lab05-1.asm	12
4.7	Копирование файла in_out.asm	13
4.8	Копирование файла lab05-1.asm	14
4.9	Редактирование файла lab05-2.asm	15
4.10	Тестирование программы lab05-2.asm	15
4.11	Редактирование файла lab05-2.asm	16
4.12	Тестирование программы lab05-2.asm	16
4.13	Редактирование файла lab05-3.asm	17
4.14	Тестирование программы lab05-3.asm	18
4.15	Редактирование файла lab05-4.asm	18
4.16	Тестирование программы lab05-4.asm	19

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. Изучение возможностей Midnight Commander
2. Изучение файла in_out.asm
3. Выполнение заданий, рассмотрение примеров
4. Выполнение заданий для самостоятельной работы

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

Открыла Midnight Commander и с помощью клавиш со стрелками и Enter перешла в каталог ~/work/arch-рс. Затем нажала F7 и создала каталог с названием lab05.

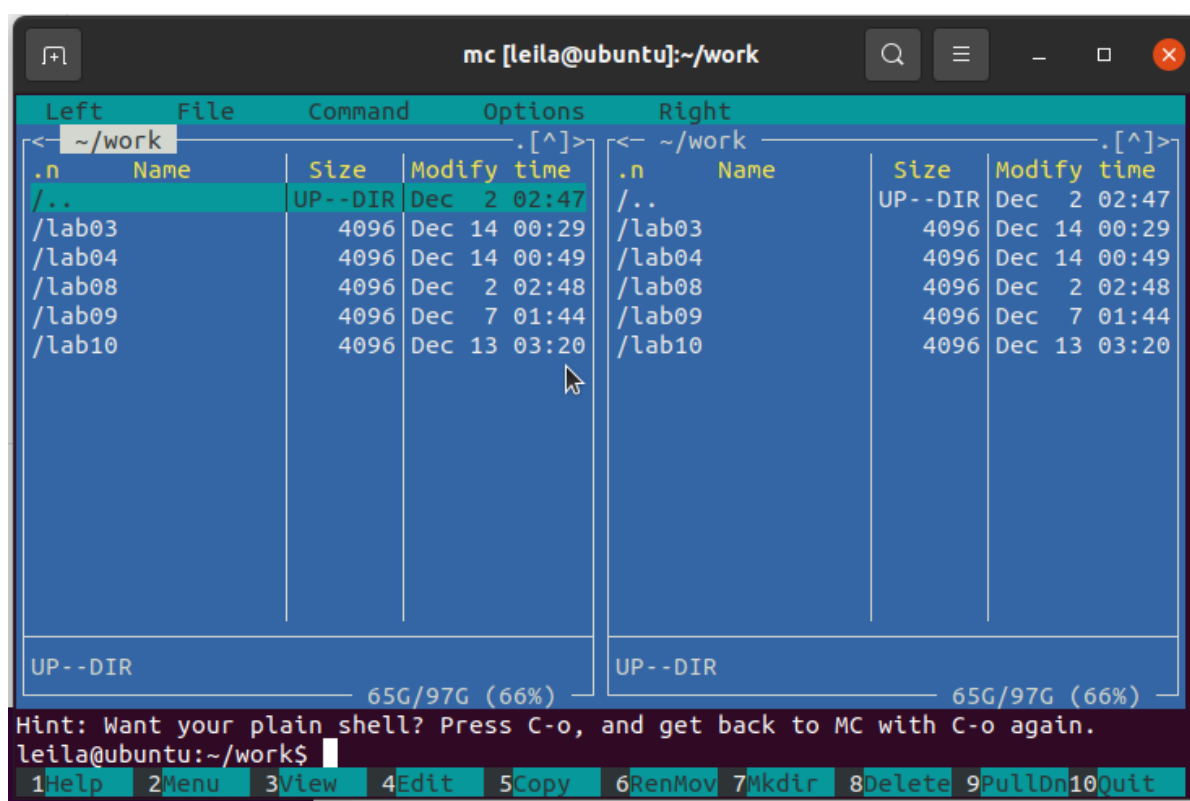


Рис. 4.1: окно Midnight Commander

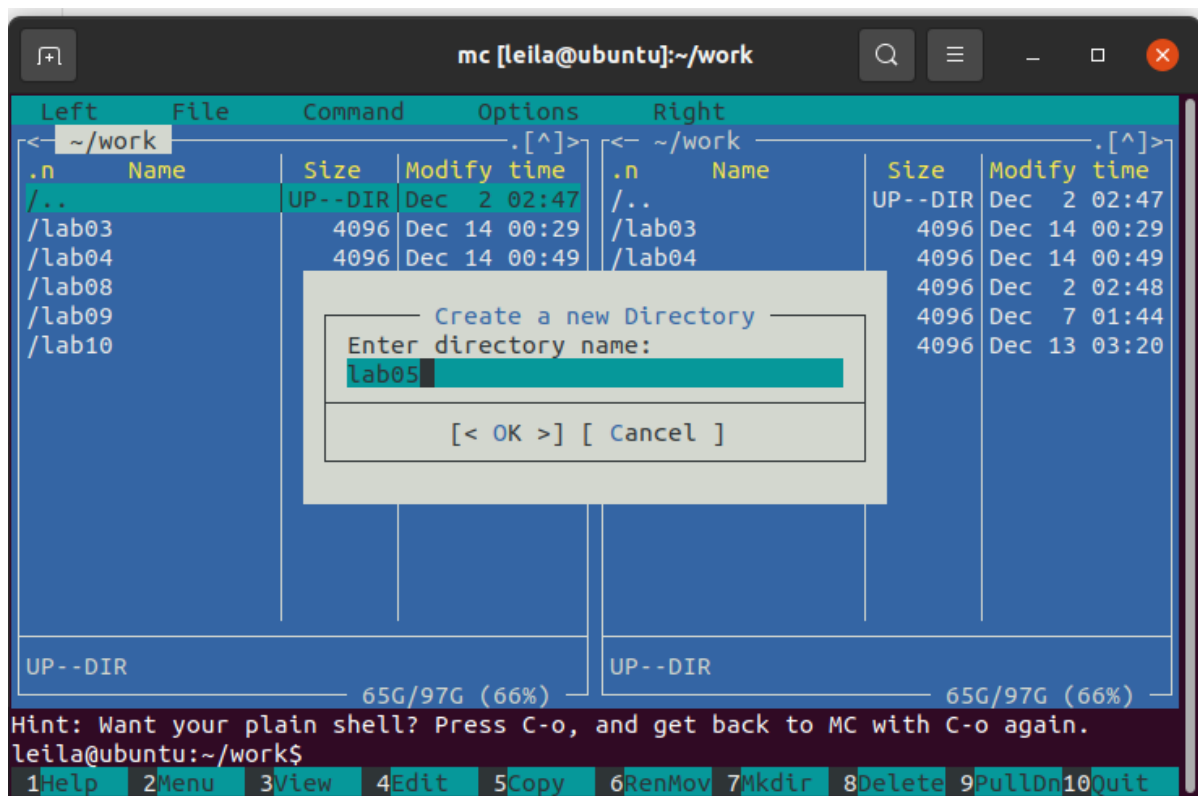


Рис. 4.2: Создание каталога

Используя команду touch, создала файл с именем lab05-1.asm.

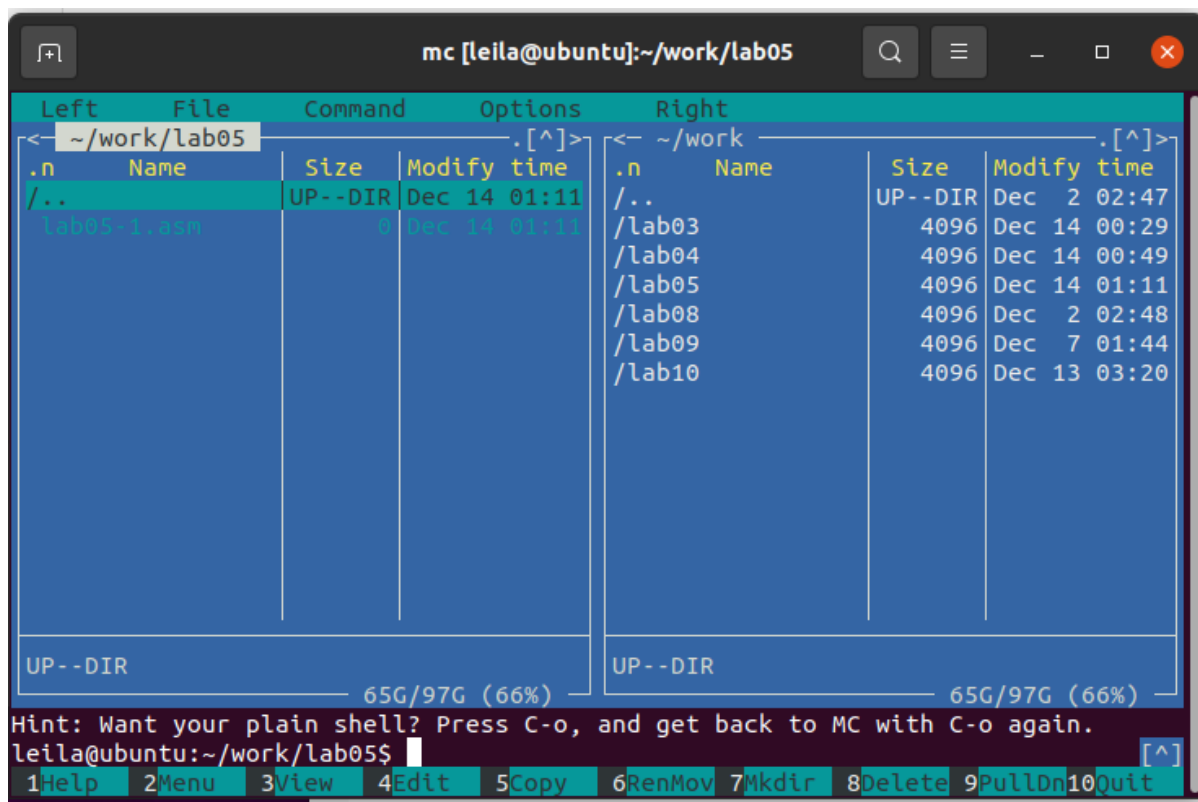
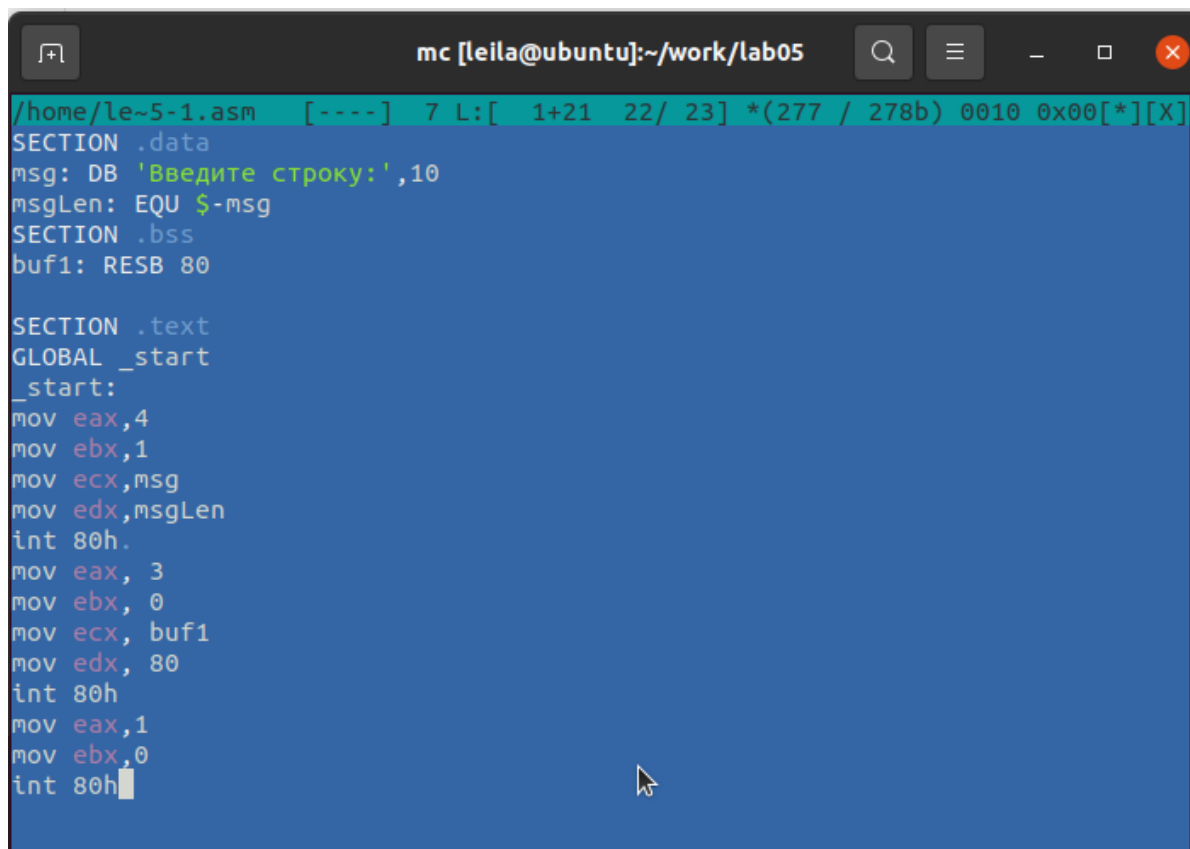


Рис. 4.3: Создала файл lab05-1.asm

Затем открыла файл для редактирования, нажав клавишу F4, и выбрала редактор mceditor. Написала код программы, соответствующий заданию.

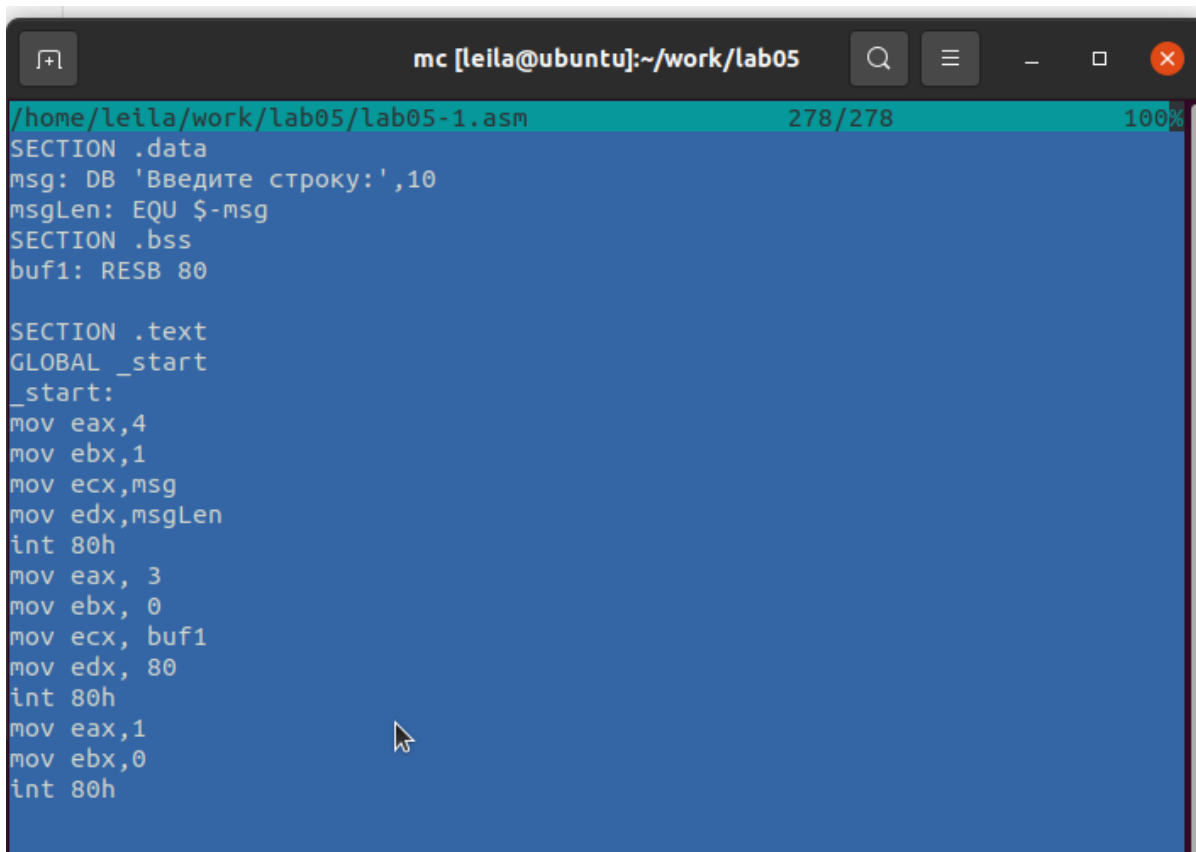


```
mc [leila@ubuntu]:~/work/lab05
/home/le~5-1.asm [----] 7 L: [ 1+21 22/ 23] *(277 / 278b) 0010 0x00[*][X]
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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. 4.4: Редактирование файла lab05-1.asm

Далее открыла файл для просмотра, нажав клавишу F3, и убедилась, что он содержит написанный код.

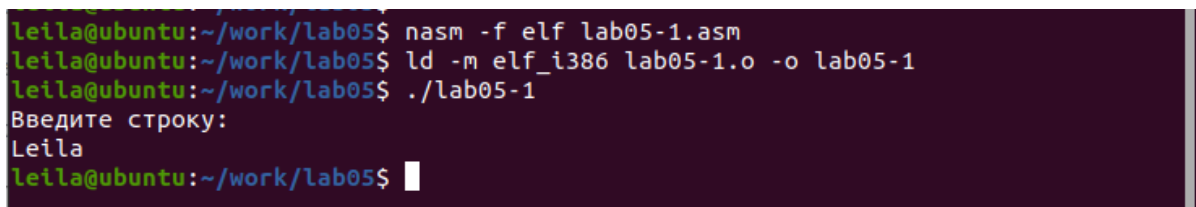
A screenshot of a code editor window titled 'mc [leila@ubuntu]:~/work/lab05'. The editor shows the assembly file 'lab05-1.asm' with 278 lines of code. The code is as follows:

```
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: Проверка кода lab05-1.asm

С помощью трансляции файла программы в объектный файл, выполнения компоновки объектного файла и получения исполняемого файла, проверила работу программы.

A screenshot of a terminal window showing the compilation and execution of the assembly program. The commands and output are as follows:

```
leila@ubuntu:~/work/lab05$ nasm -f elf lab05-1.asm
leila@ubuntu:~/work/lab05$ ld -m elf_i386 lab05-1.o -o lab05-1
leila@ubuntu:~/work/lab05$ ./lab05-1
Введите строку:
leila
```

Рис. 4.6: Тестирование программы lab05-1.asm

4.2 Подключение внешнего файла in_out.asm

Скачала файл “in_out.asm” и разместила его в рабочем каталоге.

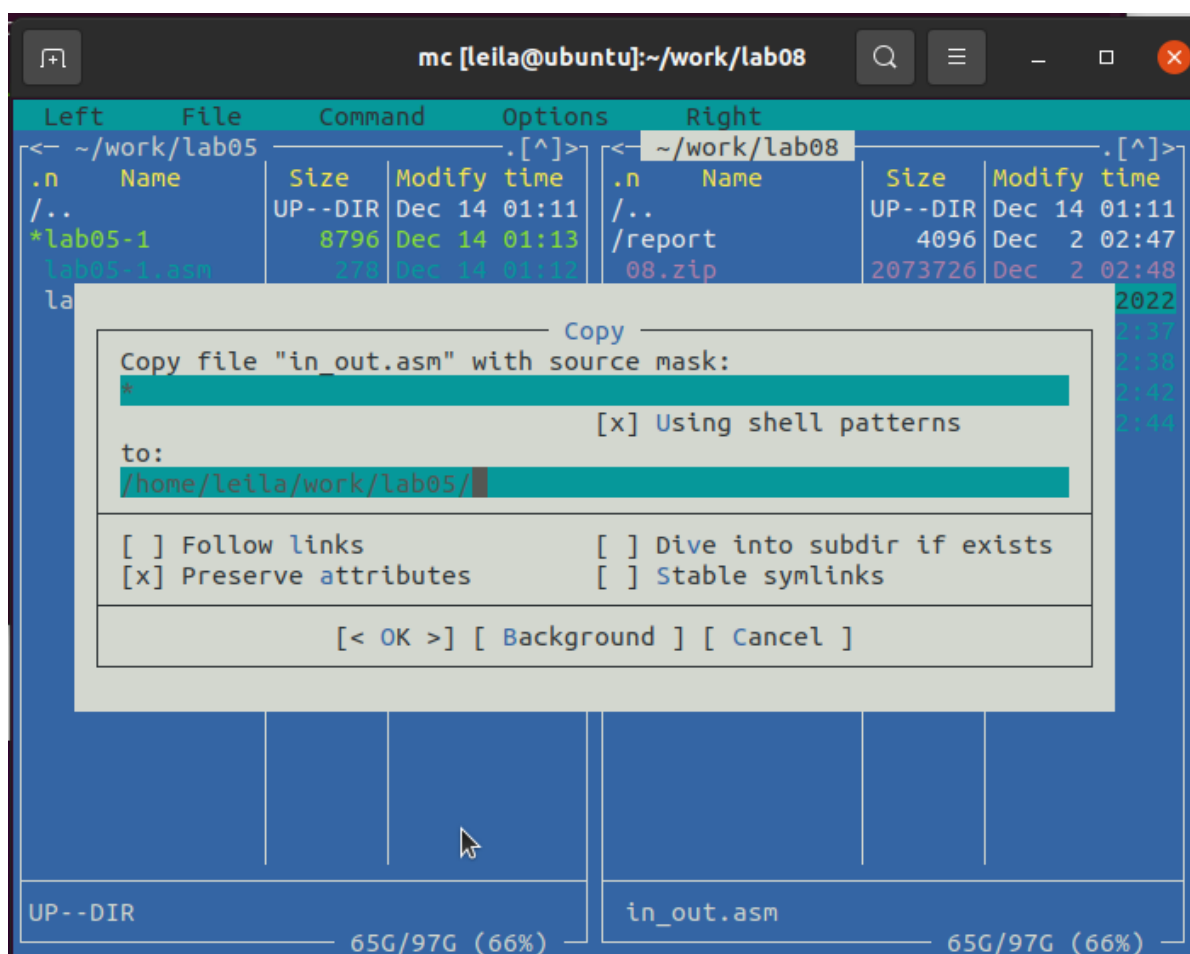


Рис. 4.7: Копирование файла in_out.asm

С помощью клавиши F5 скопировала содержимое файла lab05-1.asm в файл lab05-2.asm.

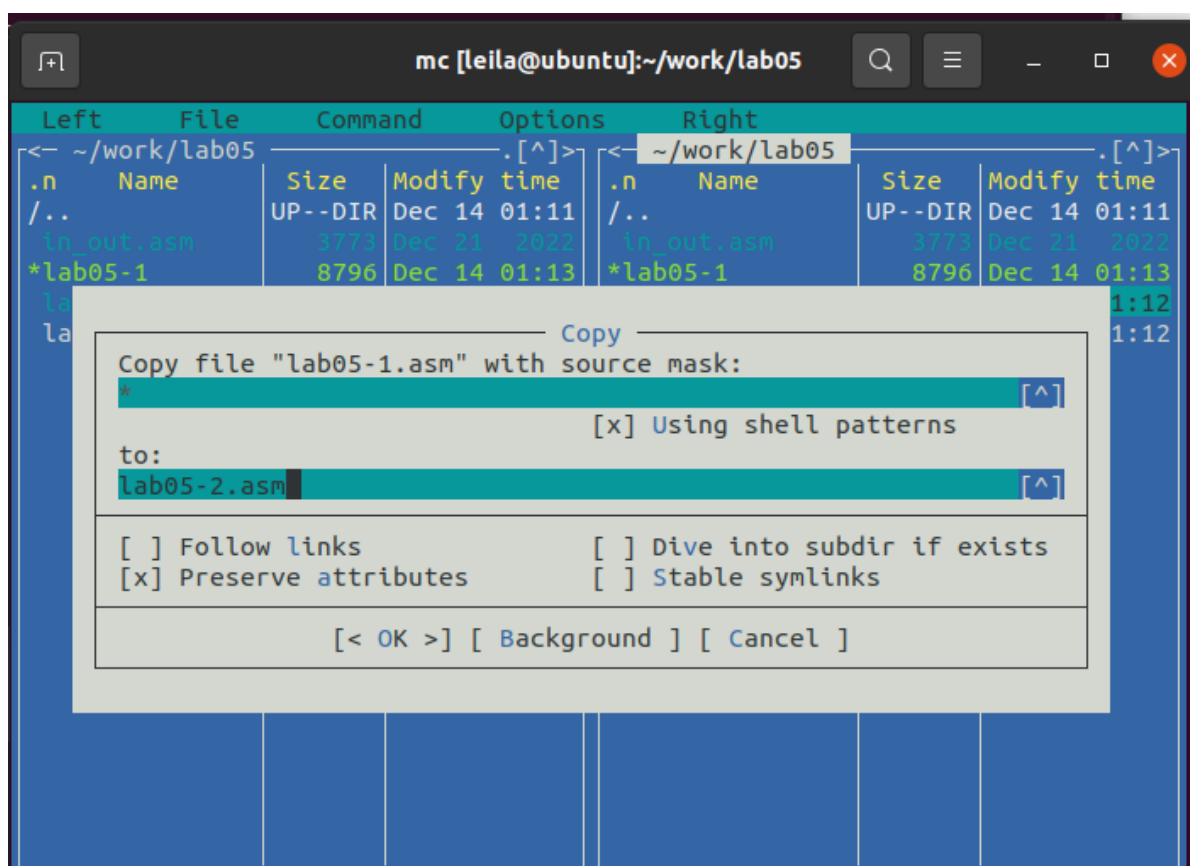
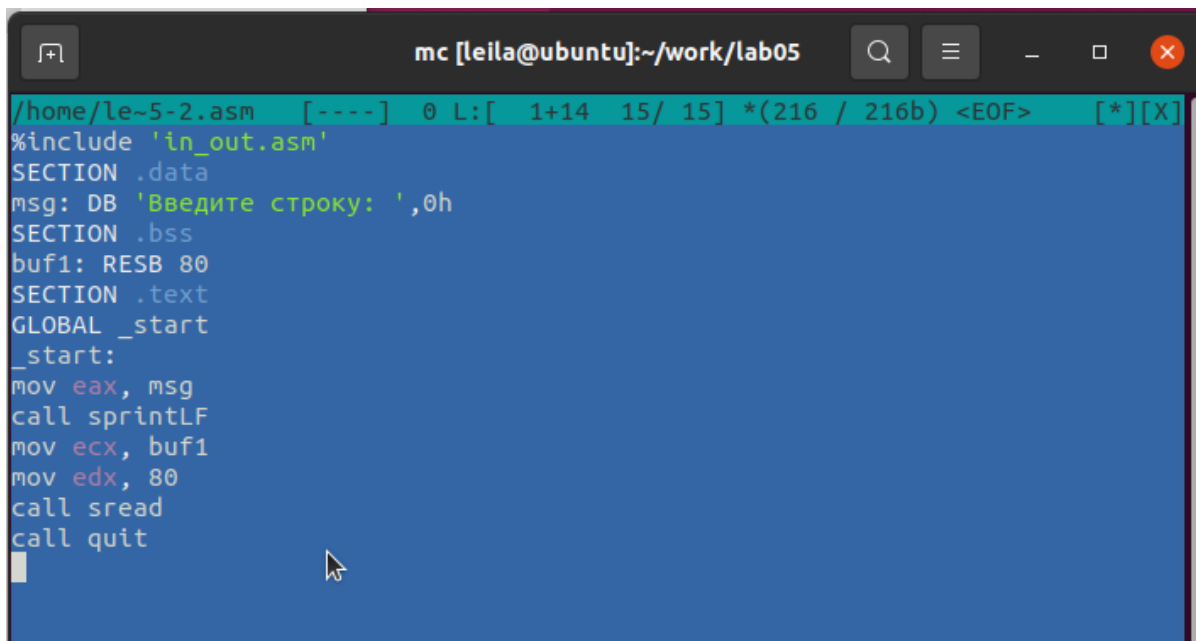


Рис. 4.8: Копирование файла lab05-1.asm

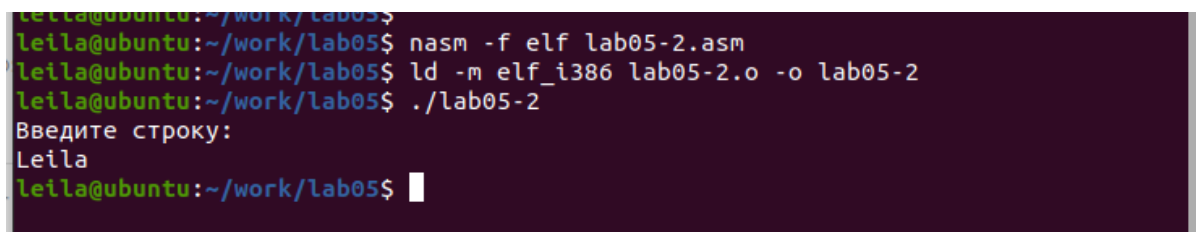
Затем написала код программы lab05-2.asm, используя подпрограммы из внешнего файла in_out.asm.



```
/home/le~5-2.asm [----] 0 L:[ 1+14 15/ 15] *(216 / 216b) <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 sprintLF
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 4.9: Редактирование файла lab05-2.asm

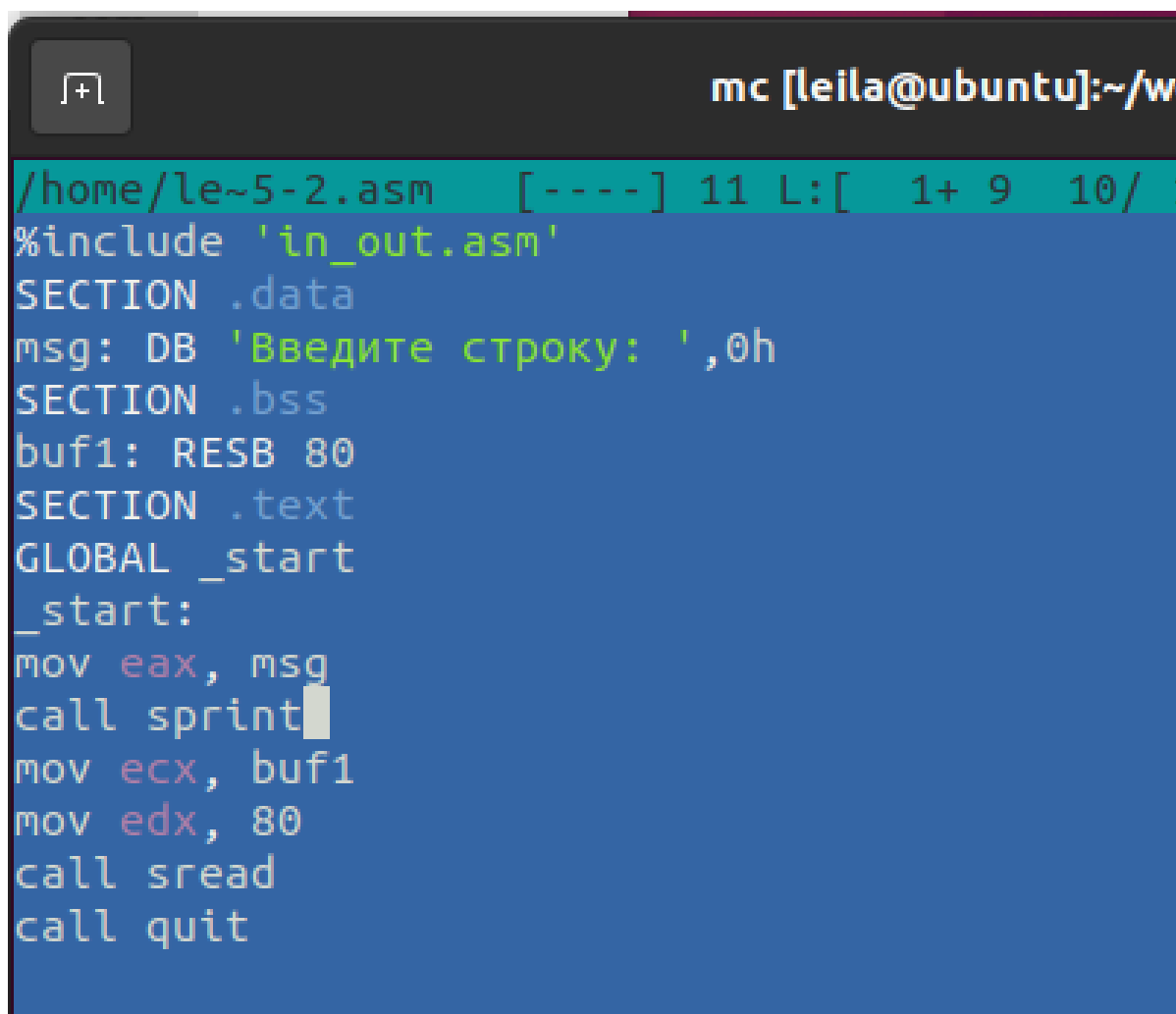
Скомпилировала программу и проверила ее запуск.



```
leila@ubuntu:~/work/lab05$
leila@ubuntu:~/work/lab05$ nasm -f elf lab05-2.asm
leila@ubuntu:~/work/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
leila@ubuntu:~/work/lab05$ ./lab05-2
Введите строку:
Leila
leila@ubuntu:~/work/lab05$
```

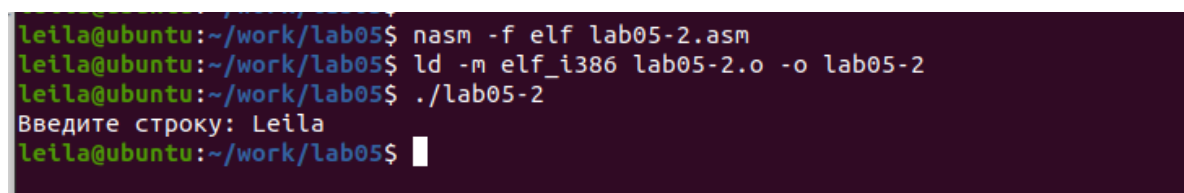
Рис. 4.10: Тестирование программы lab05-2.asm

В файле lab05-2.asm заменила вызов подпрограммы sprintLF на sprint. Пересобрала исполняемый файл. Теперь после вывода строки символ перехода на новую строку отсутствует.



```
mc [leila@ubuntu]:~/w
/home/le~5-2.asm [----] 11 L:[ 1+ 9 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 sprint
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 4.11: Редактирование файла lab05-2.asm

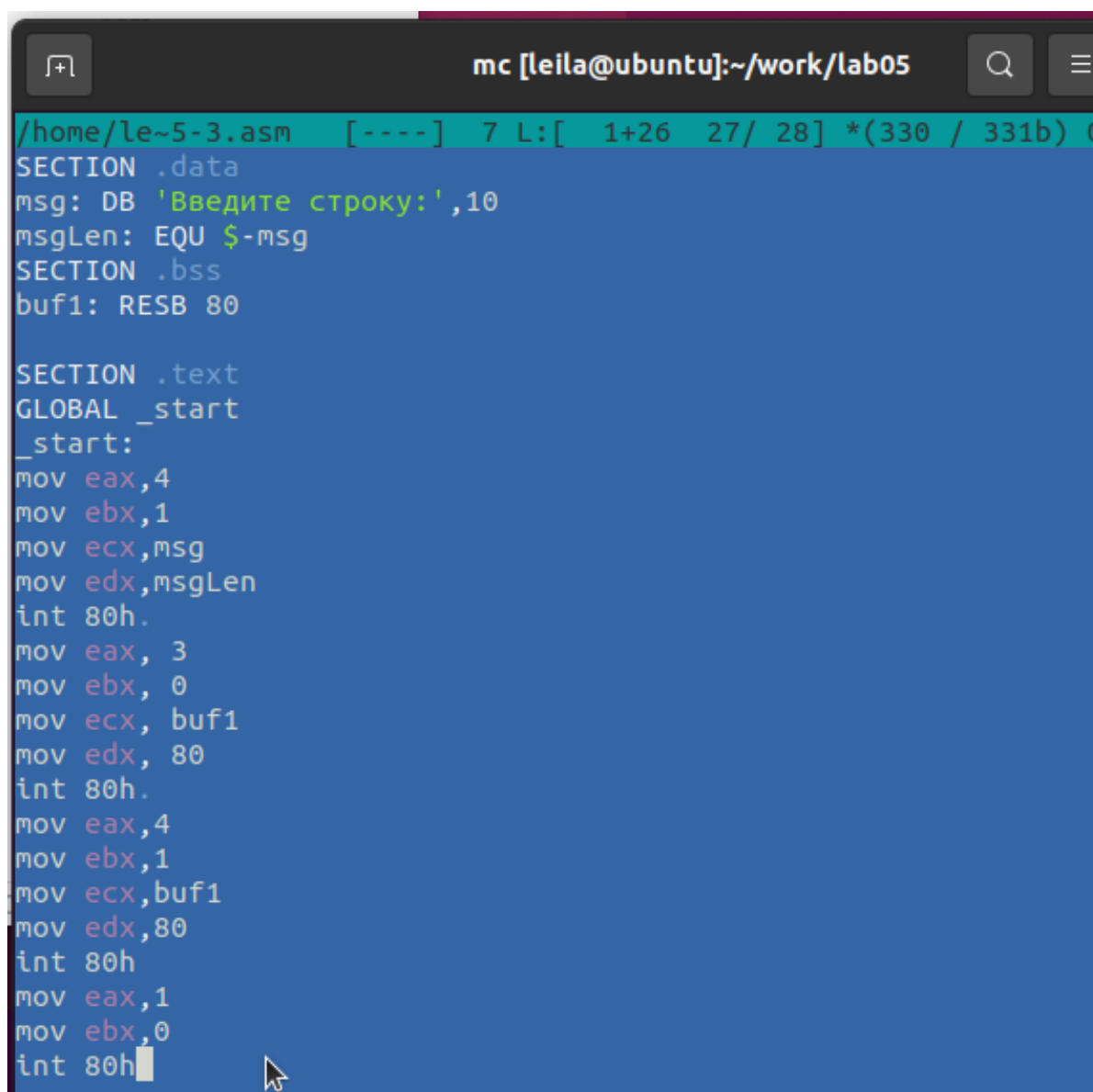


```
leila@ubuntu:~/work/lab05$ nasm -f elf lab05-2.asm
leila@ubuntu:~/work/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
leila@ubuntu:~/work/lab05$ ./lab05-2
Введите строку: Leila
leila@ubuntu:~/work/lab05$
```

Рис. 4.12: Тестирование программы lab05-2.asm

4.3 Выполнение заданий для самостоятельной работы

Скопировала программу lab05-1.asm и внесла изменения в код, чтобы программа работала по следующему алгоритму: она выводит приглашение вида “Введите строку:”, считывает строку с клавиатуры и выводит введенную строку на экран.



```
mc [leila@ubuntu]:~/work/lab05
/home/le~5-3.asm [----] 7 L:[ 1+26 27/ 28] *(330 / 331b) 0
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: Редактирование файла lab05-3.asm

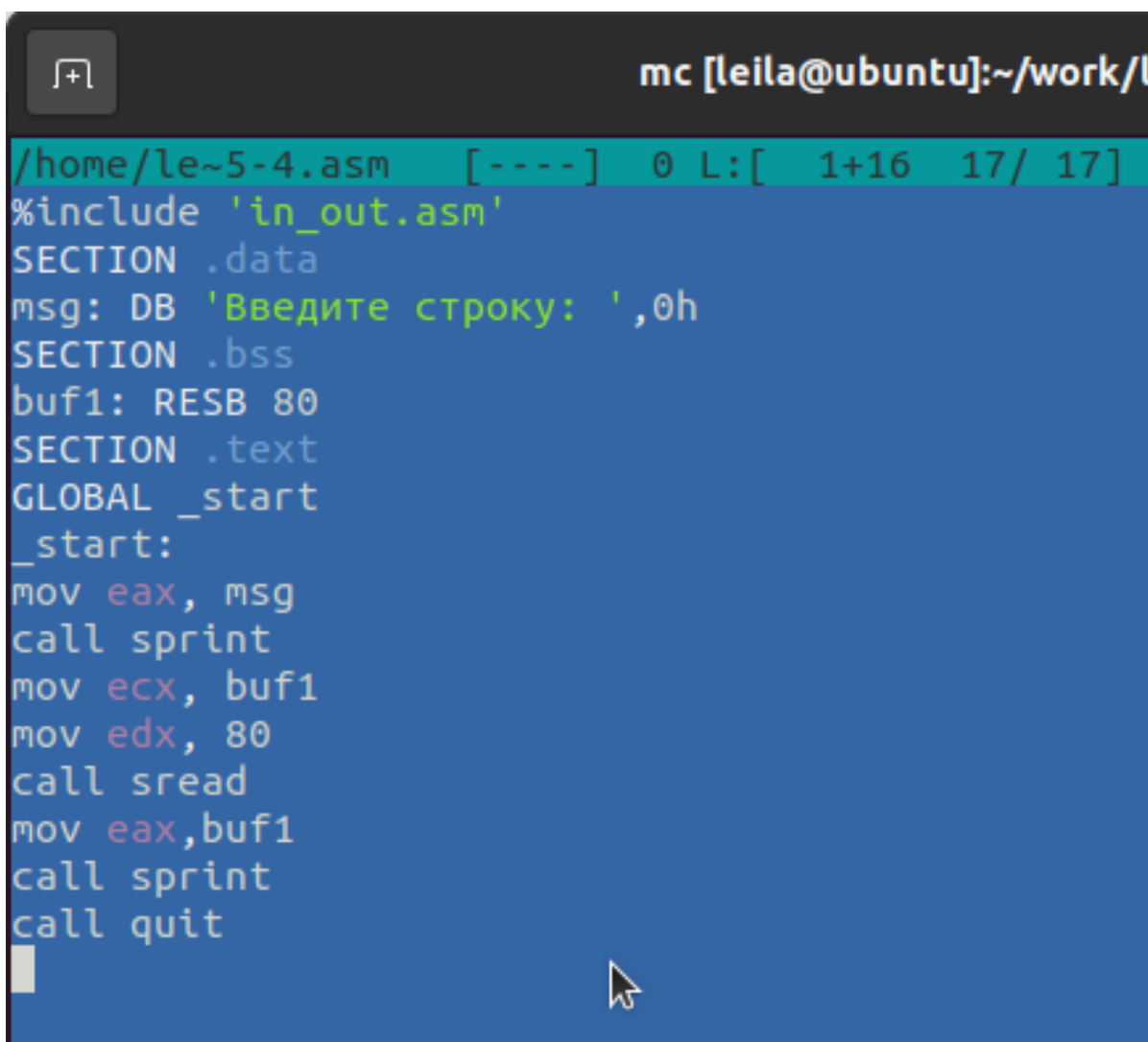
```

leila@ubuntu:~/work/lab05$ nasm -f elf lab05-3.asm
leila@ubuntu:~/work/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
leila@ubuntu:~/work/lab05$ ./lab05-3
Введите строку:
Leila
Leila
leila@ubuntu:~/work/lab05$

```

Рис. 4.14: Тестирование программы lab05-3.asm

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



```

mc [leila@ubuntu]:~/work/
/home/le~5-4.asm [----] 0 L:[ 1+16 17/ 17]
%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: Редактирование файла lab05-4.asm

```
leila@ubuntu:~/work/lab05$ nasm -f elf lab05-4.asm
leila@ubuntu:~/work/lab05$ ld -m elf_i386 lab05-4.o -o lab05-4
leila@ubuntu:~/work/lab05$ ./lab05-4
Введите строку: Leila
Leila
leila@ubuntu:~/work/lab05$
```

Рис. 4.16: Тестирование программы lab05-4.asm

5 Выводы

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