

Отчёт по лабораторной работе 5

Дисциплина: архитектура компьютера

Видмаер Егор

Содержание

1	Цель работы	5
2	Задания	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Работа в Midnight Commander	8
4.2	Подключение внешнего файла in_out.asm	12
4.3	Задание для самостоятельной работы	16
5	Выводы	20

Список иллюстраций

4.1	Создание каталога	8
4.2	Создание файла lab05-1.asm	9
4.3	Программа lab05-1.asm	10
4.4	Просмотр файла lab05-1.asm	11
4.5	Проверка программы lab05-1.asm	12
4.6	Копирование файла	12
4.7	Программа lab05-2.asm	13
4.8	Проверка программы lab05-2.asm	14
4.9	Программа lab05-2.asm	15
4.10	Проверка программы lab05-2.asm	15
4.11	Программа lab05-3.asm	16
4.12	Проверка программы lab05-3.asm	17
4.13	Программа lab05-4.asm	18
4.14	Проверка программы lab05-4.asm	18

Список таблиц

1 Цель работы

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

2 Задания

1. Изучить основы работы с Midnight Commander
2. Изучить инструкции ассемблера
3. Выполнить самостоятельное задание по изменению программы

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.

4 Выполнение лабораторной работы

4.1 Работа в Midnight Commander

Открыл Midnight Commander. Перешел в каталог ~/work/arch-rc. Создал каталог lab05 (рис. [4.1])

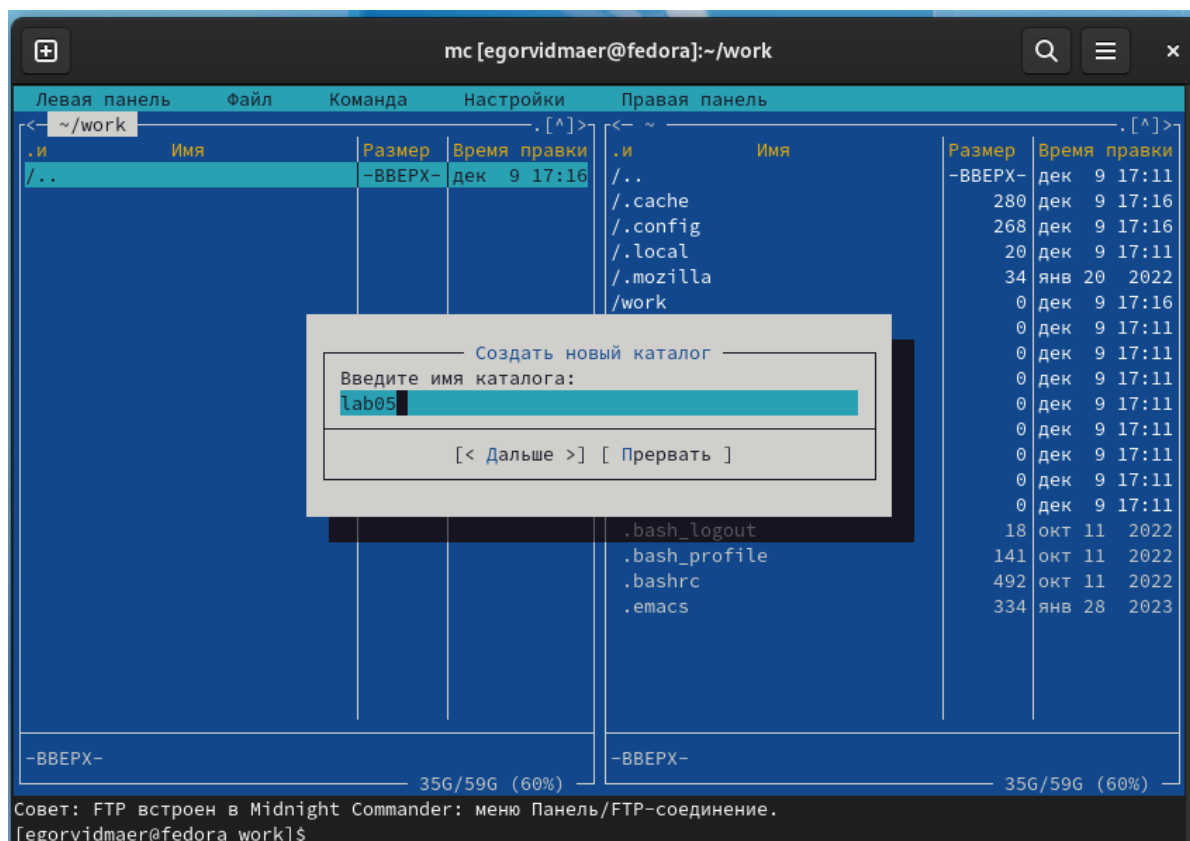


Рис. 4.1: Создание каталога

Создал файл lab05-1.asm (рис. [4.2])

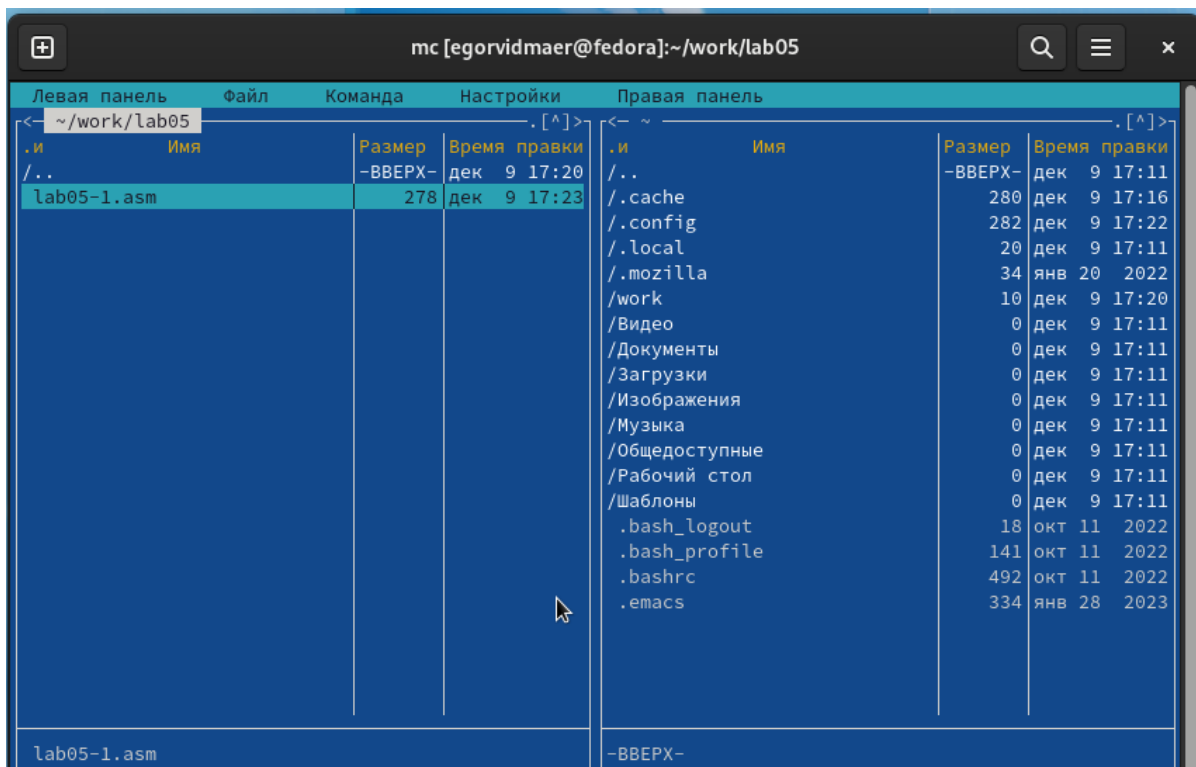
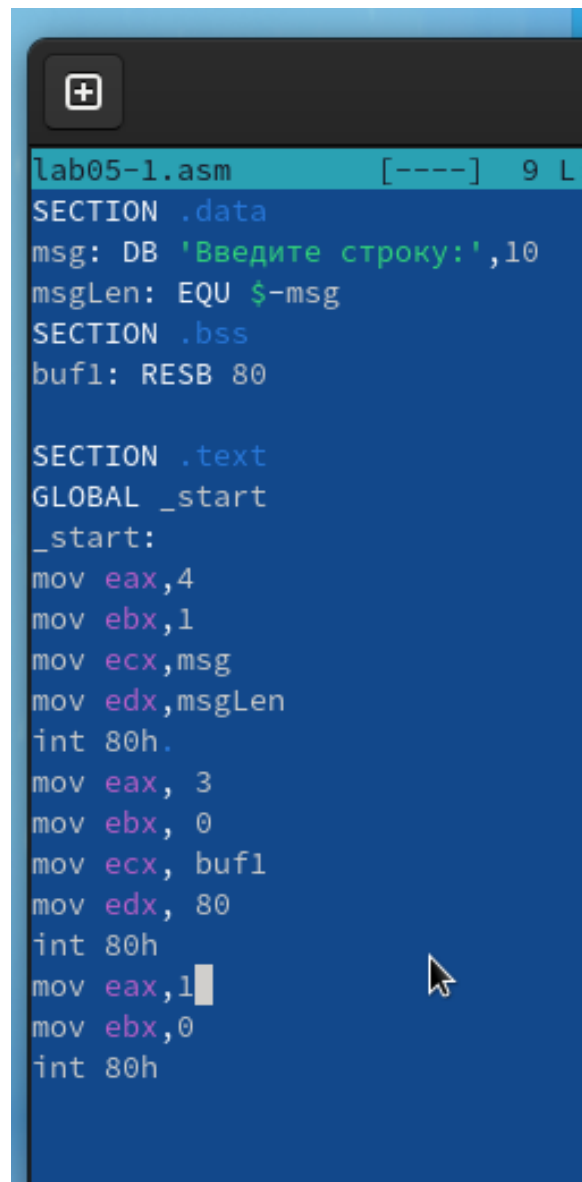


Рис. 4.2: Создание файла lab05-1.asm

Открыл файл на редактирование. Написал код. (рис. [4.3])

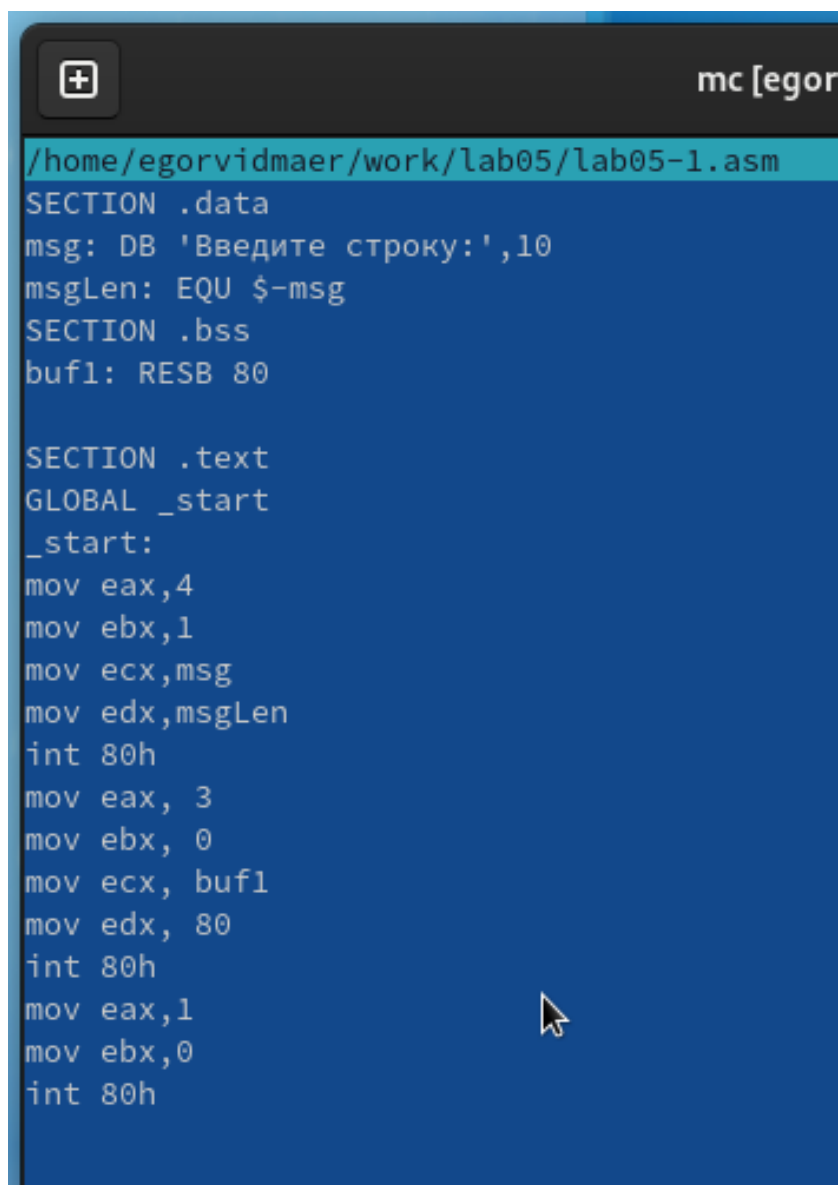


```
lab05-1.asm [----] 9 L:
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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. 4.3: Программа lab05-1.asm

Открыл файл для просмотра и убедился, что он содержит написанный код. (рис. [4.4])



```
mc [egor  
/home/egorvidmaer/work/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. 4.4: Просмотр файла lab05-1.asm

Получил исполняемый файл программы и проверил его работу.(рис. [4.5])

```

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

```

Рис. 4.5: Проверка программы lab05-1.asm

4.2 Подключение внешнего файла in_out.asm

Скачал файл in_out.asm. Добавил файл in_out.asm в рабочий каталог. Скопировал lab05-1.asm в lab05-2.asm. (рис. [4.6])

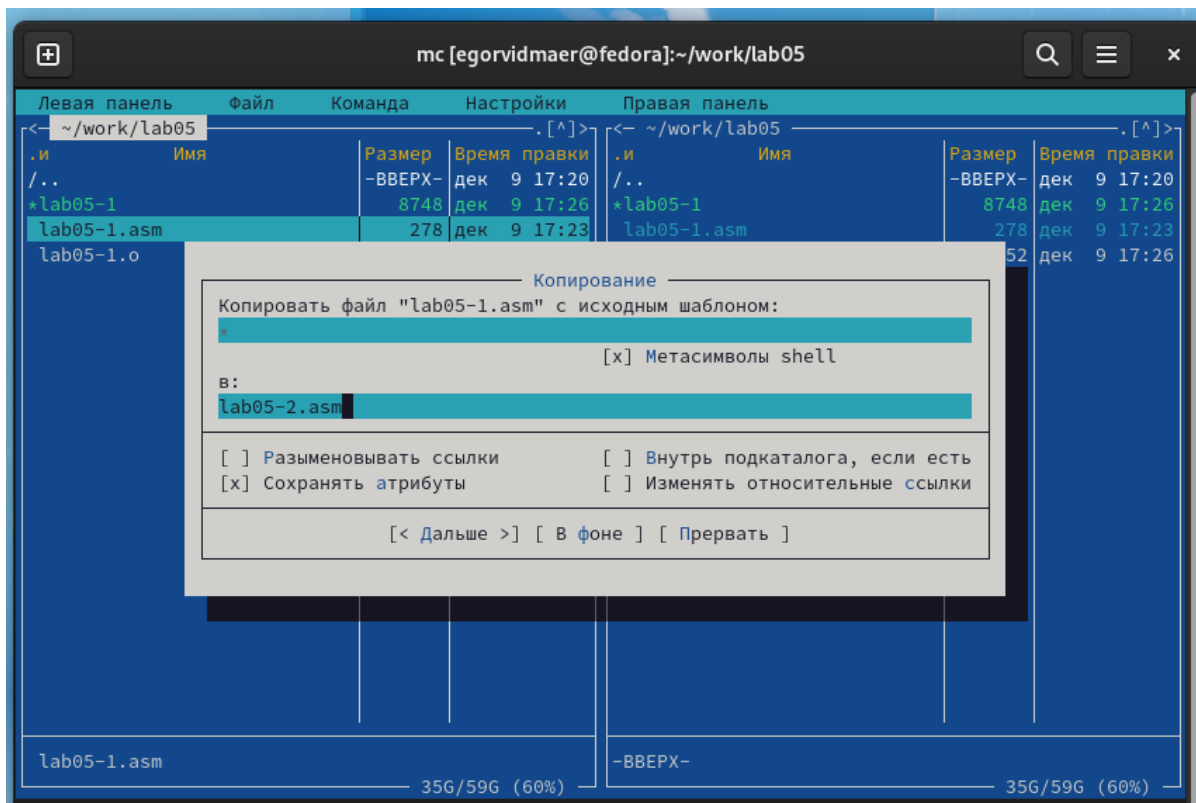
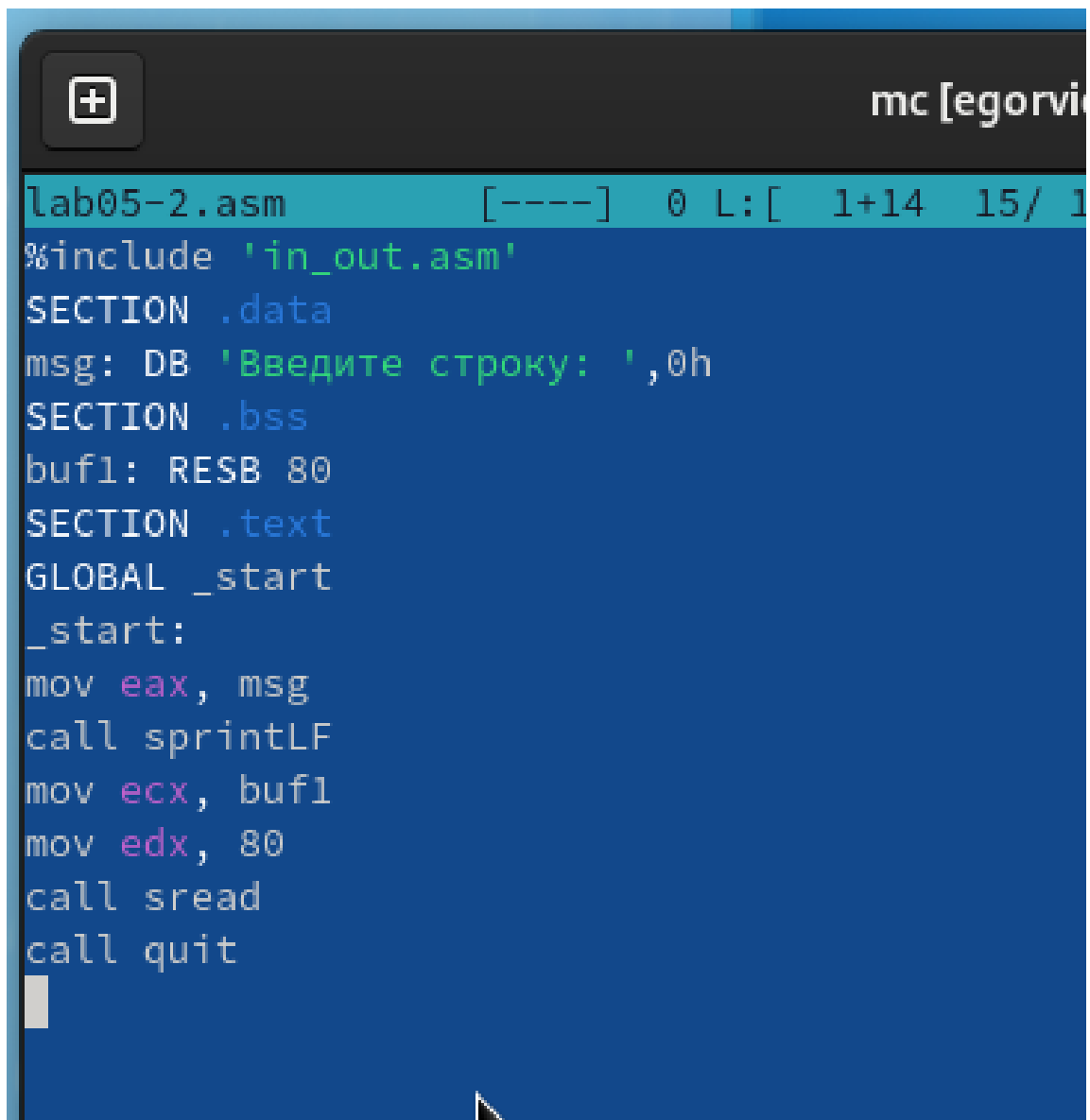


Рис. 4.6: Копирование файла

Написал код программы lab05-2.asm. (рис. [4.7]) Скомпилировал программу и

проверил запуск. (рис. [4.8])



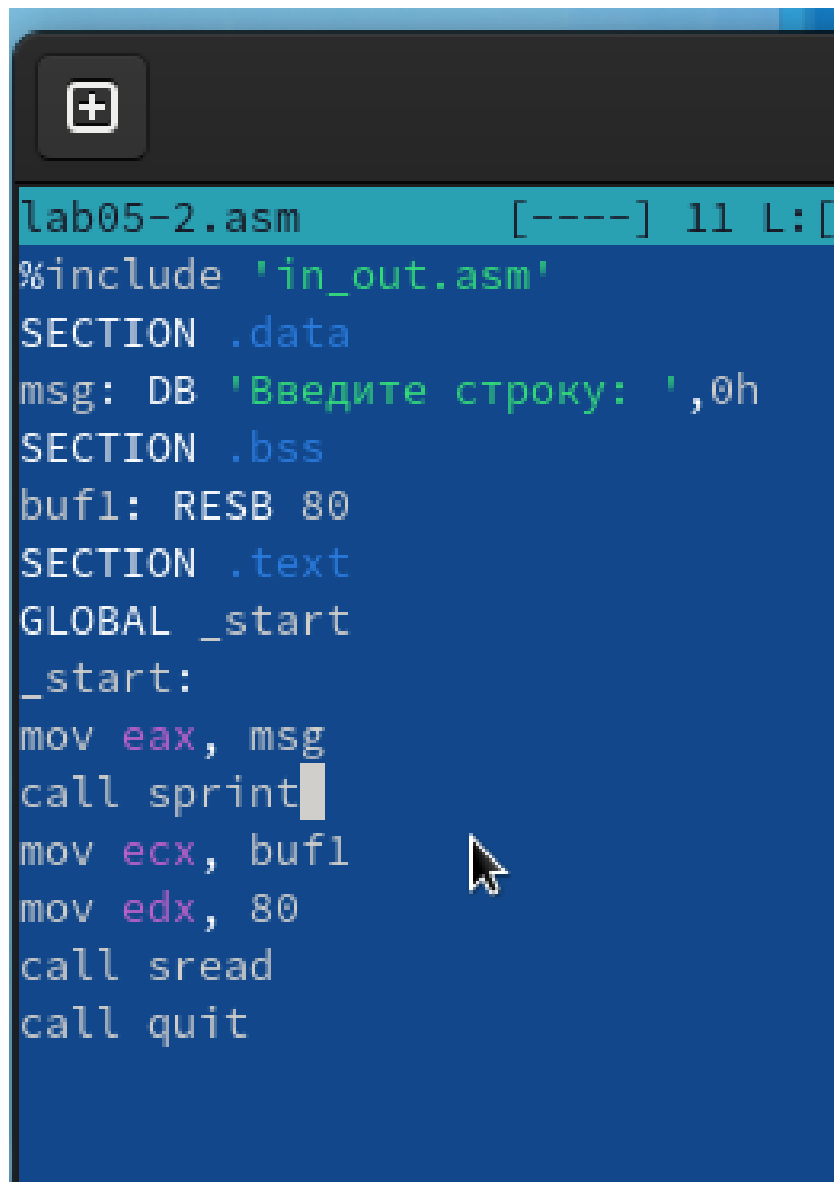
```
lab05-2.asm [-----] 0 L: [ 1+14 15/ 1
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 4.7: Программа lab05-2.asm

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

Рис. 4.8: Проверка программы lab05-2.asm

В файле lab5-2.asm я заменил подпрограмму sprintLF на sprint (рис. [4.9]). Затем я снова собрал исполняемый файл (рис. [4.10]). Теперь после вывода строки она не завершается символом перехода на новую строку.



```
lab05-2.asm [----] 11 L: [
#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
```

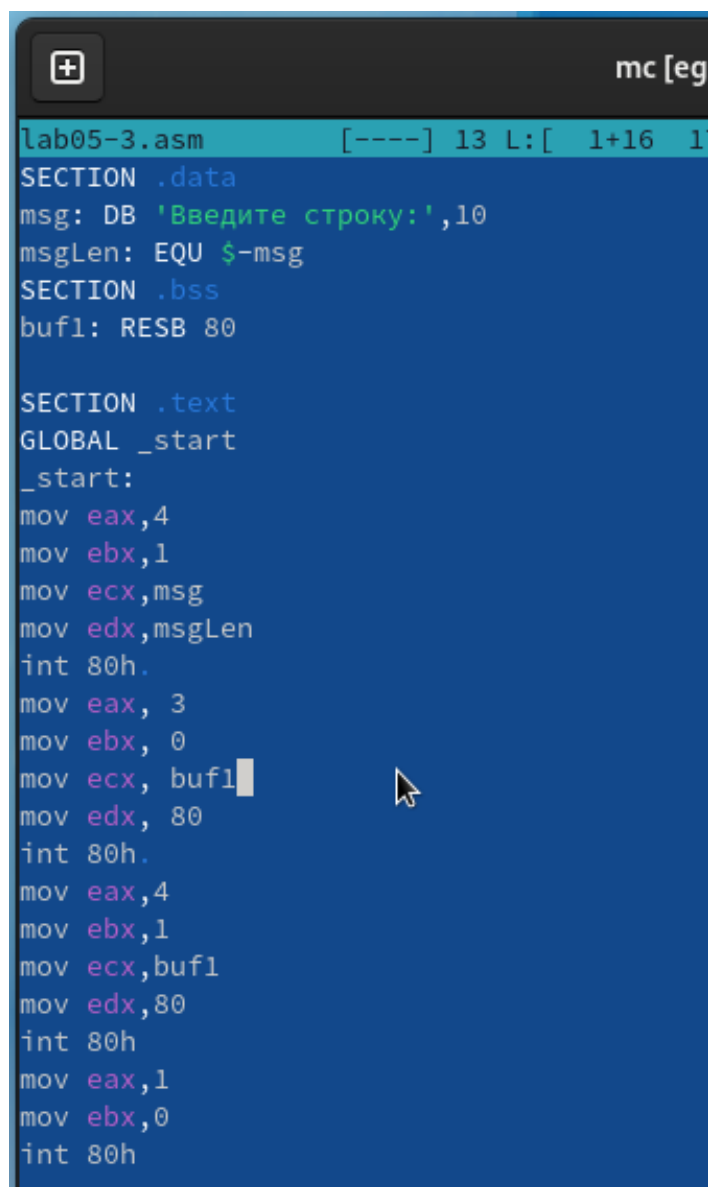
Рис. 4.9: Программа lab05-2.asm

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

Рис. 4.10: Проверка программы lab05-2.asm

4.3 Задание для самостоятельной работы

Скопировал программу lab05-1.asm и изменил код, чтобы программа выводила приглашение типа “Введите строку:”, затем считывала строку с клавиатуры и выводила введенную строку на экран. (рис. [4.11], рис. [4.12])



```
lab05-3.asm [----] 13 L: [ 1+16 17
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

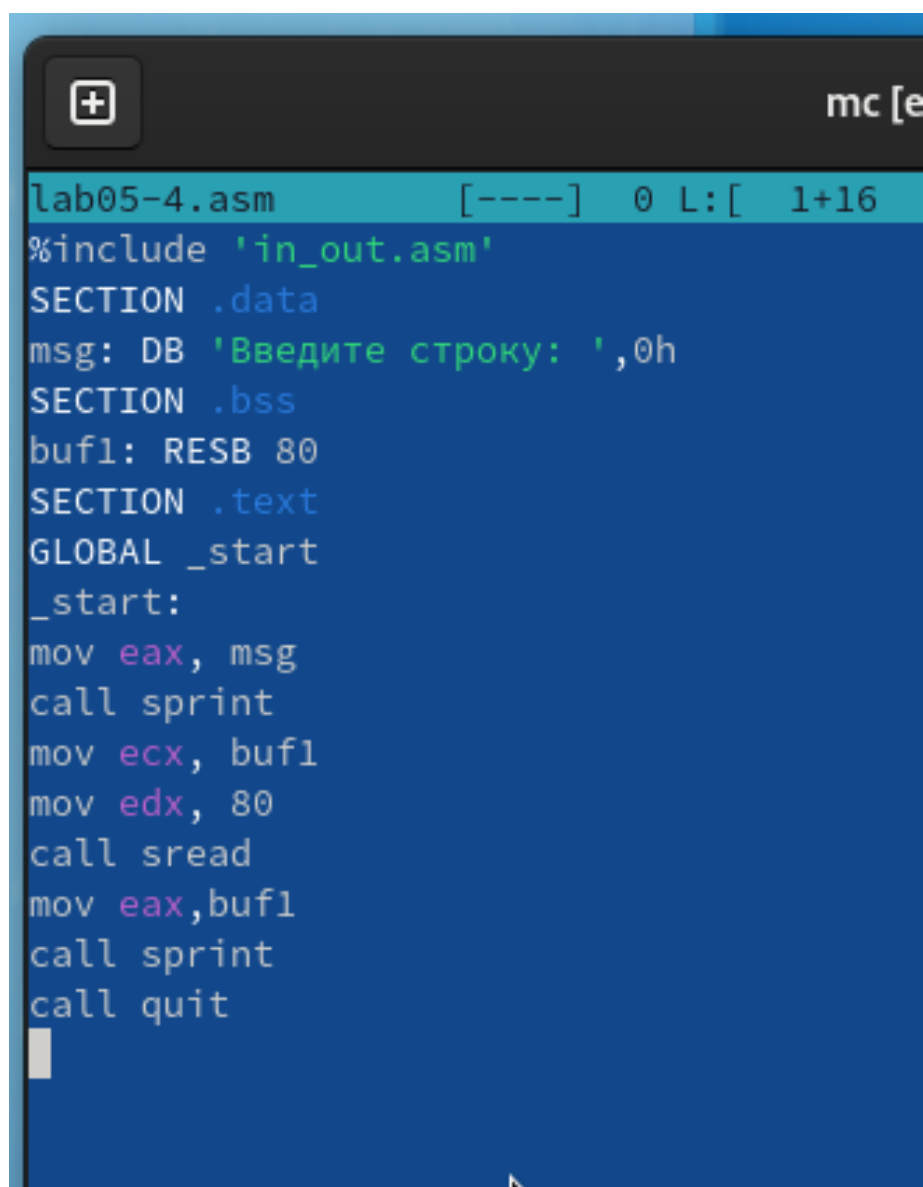
SECTION .text
GLOBAL _start
_start:
mov eax,4
mov ebx,1
mov ecx,msg
mov edx,msgLen
int 80h.
mov eax, 3
mov ebx, 0
mov ecx, buf1
mov edx, 80
int 80h.
mov eax,4
mov ebx,1
mov ecx,buf1
mov edx,80
int 80h
mov eax,1
mov ebx,0
int 80h
```

Рис. 4.11: Программа lab05-3.asm


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

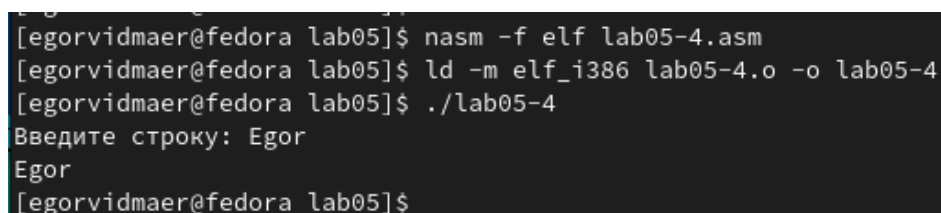
Рис. 4.12: Проверка программы lab05-3.asm

Также я скопировал программу lab05-2.asm и внес соответствующие изменения в код, чтобы программа выводила приглашение типа “Введите строку:”, затем считывала строку с клавиатуры и выводила введенную строку на экран.(рис. [4.13], рис. [4.14])



```
lab05-4.asm [----] 0 L:[ 1+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. 4.13: Программа lab05-4.asm



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

Рис. 4.14: Проверка программы lab05-4.asm

Отличие этих двух реализаций заключается в том, что файл `in_out.asm` содержит уже готовые подпрограммы для обеспечения ввода/вывода. Таким образом, нам остается только разместить данные в нужных регистрах и вызвать желаемую подпрограмму с помощью инструкции `call`.

5 Выводы

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