

Отчёт по лабораторной работе 5

Архитектура компьютера

Ходжамедов Давуд НБИбд-02-23

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	20

Список иллюстраций

2.1	Создание каталога	6
2.2	Создание файла lab05-1.asm	7
2.3	Программа в файле lab05-1.asm	8
2.4	Просмотр файла lab05-1.asm	10
2.5	Сборка и проверка программы lab05-1.asm	11
2.6	Копирование файла	11
2.7	Программа в файле lab05-2.asm	12
2.8	Сборка и проверка программы lab05-2.asm	13
2.9	Программа в файле lab05-2.asm	14
2.10	Сборка и проверка программы lab05-2.asm	15
2.11	Программа в файле lab05-3.asm	16
2.12	Сборка и проверка программы lab05-3.asm	17
2.13	Программа в файле lab05-4.asm	18
2.14	Сборка и проверка 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 Выполнение лабораторной работы

Открыл Midnight Commander

Перешел в каталог ~/work/arch-pc

Создал каталог lab05

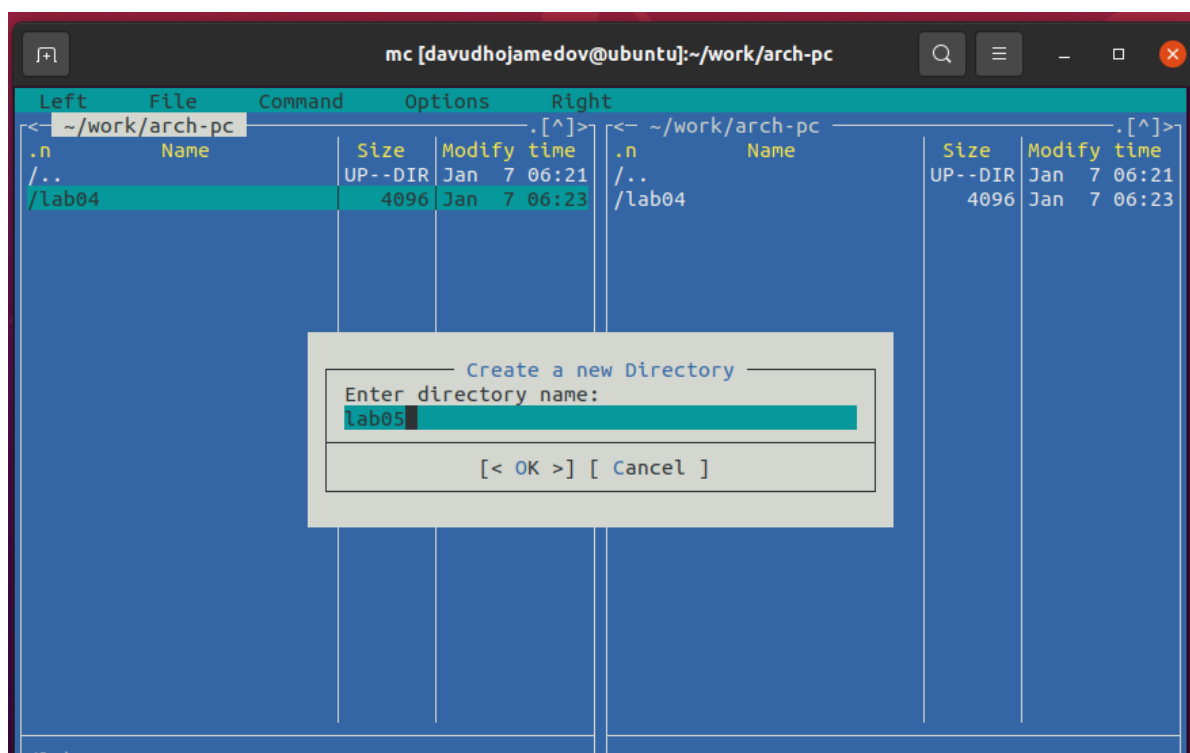


Рис. 2.1: Создание каталога

Создал файл lab05-1.asm

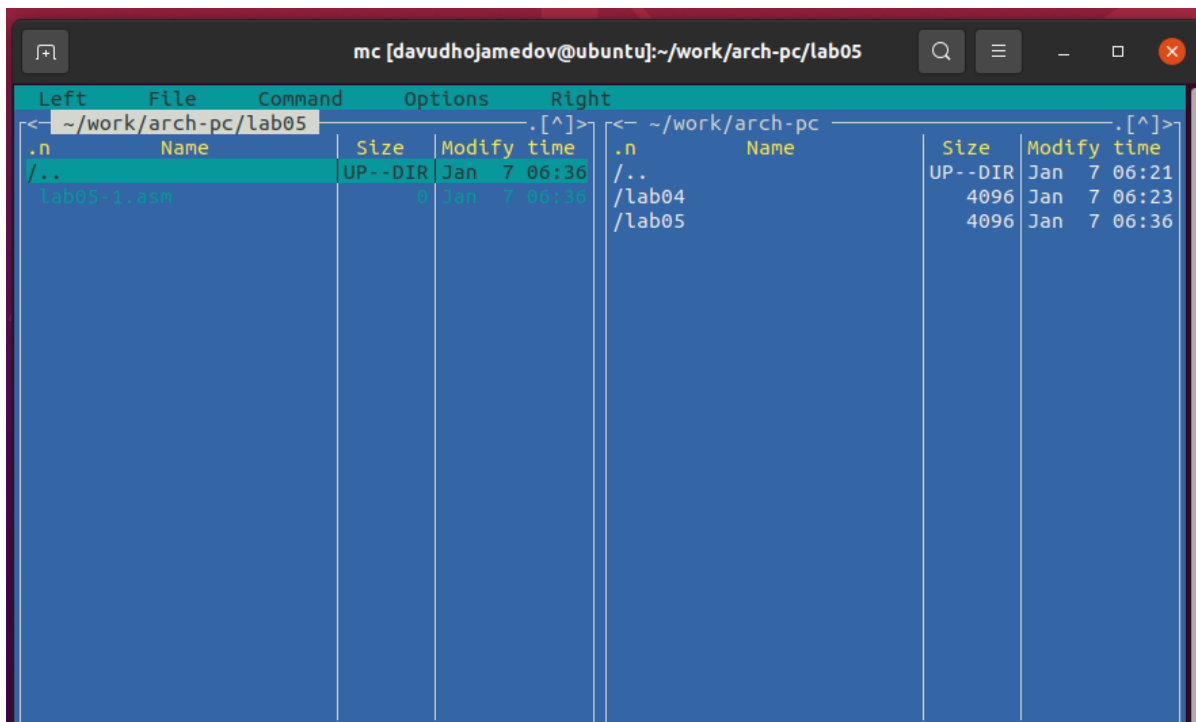
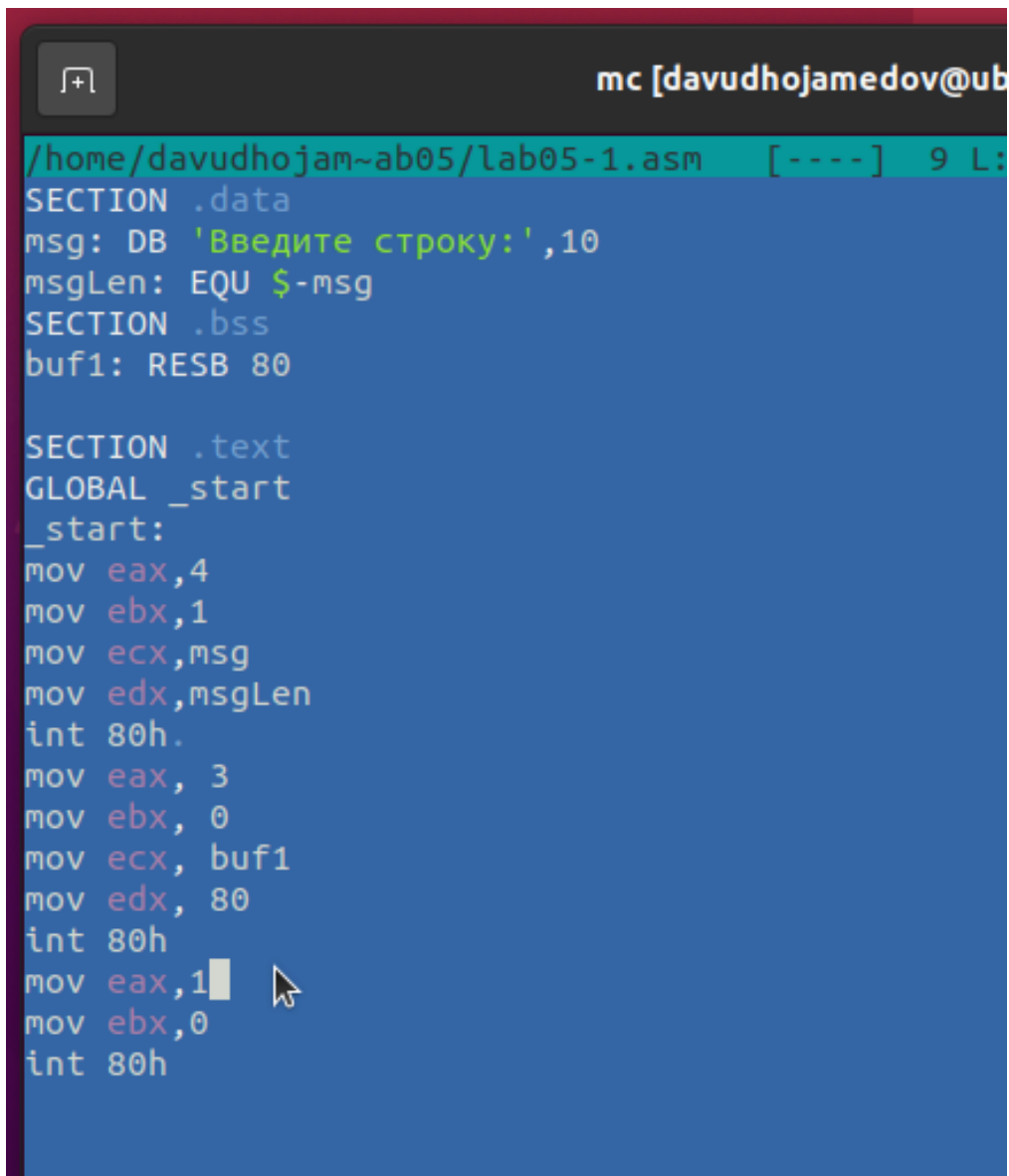


Рис. 2.2: Создание файла lab05-1.asm

Открыл файл на редактирование

Написал код



```
mc [davudhojamedov@ub
/home/davudhojam~ab05/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. 2.3: Программа в файле lab05-1.asm

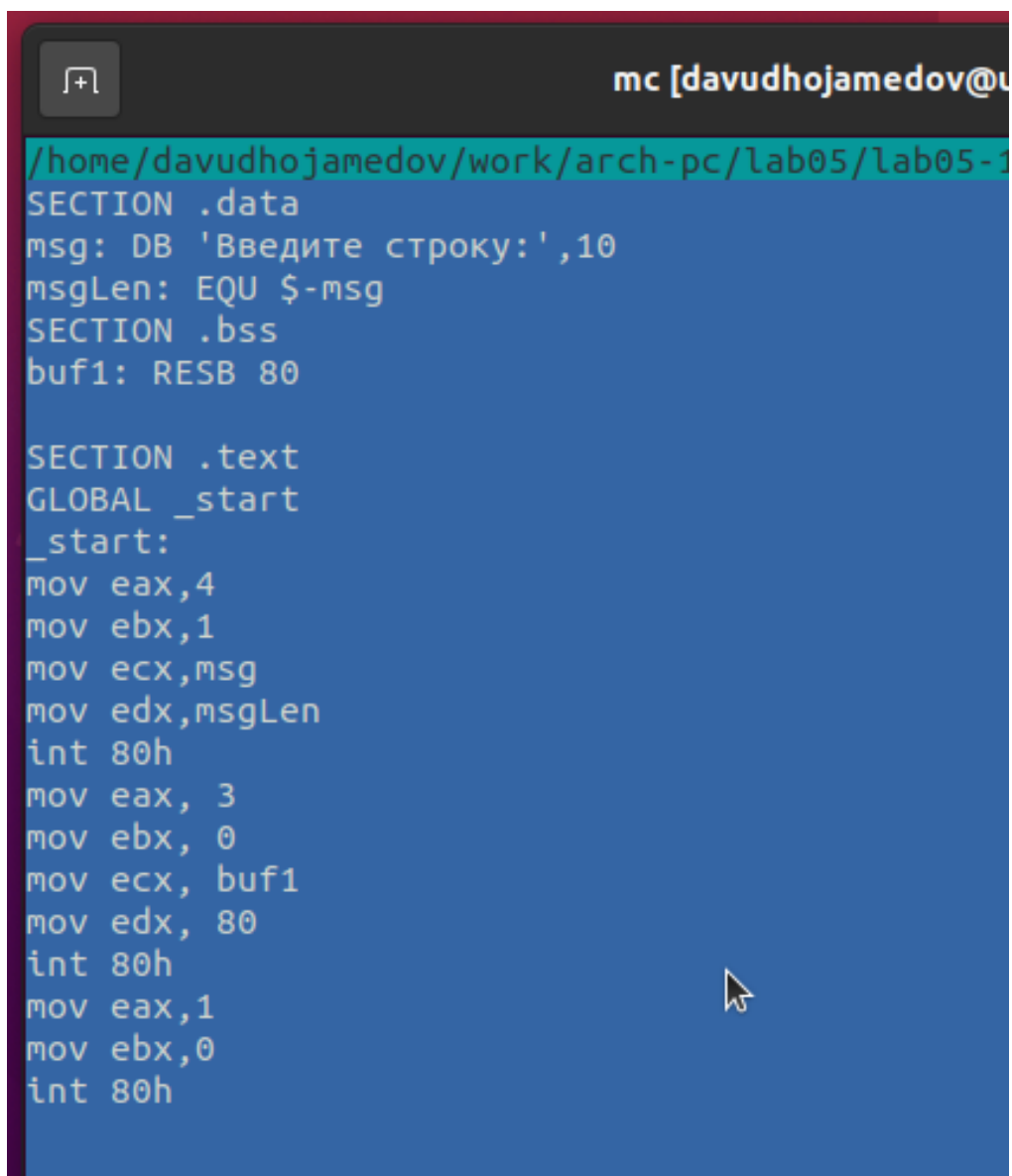
Также добавлю код программы в отчет.

```
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
```

Открыл файл на просмотр и убелился, что он содержит набранный код.



```
mc [davudhojamedov@u
/home/davudhojamedov/work/arch-pc/lab05/lab05-1
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. 2.4: Просмотр файла lab05-1.asm

Получил исполняемый файл программы и проверил ее работу.

```

davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-1.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-1.o -o lab05-1
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-1
Введите строку:
Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$

```

Рис. 2.5: Сборка и проверка программы lab05-1.asm

Скачал файл in_out.asm.

Добавил файл in_out.asm в рабочий каталог.

Скопировал lab05-1.asm в lab05-2.asm.

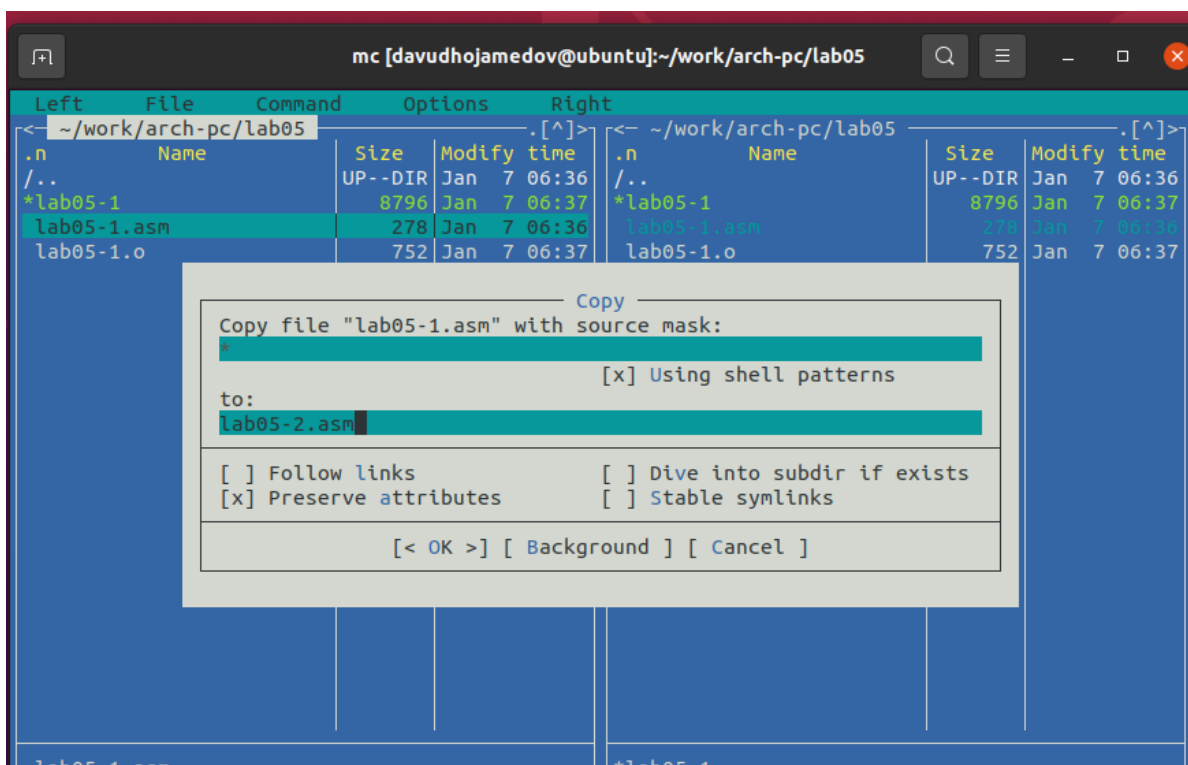
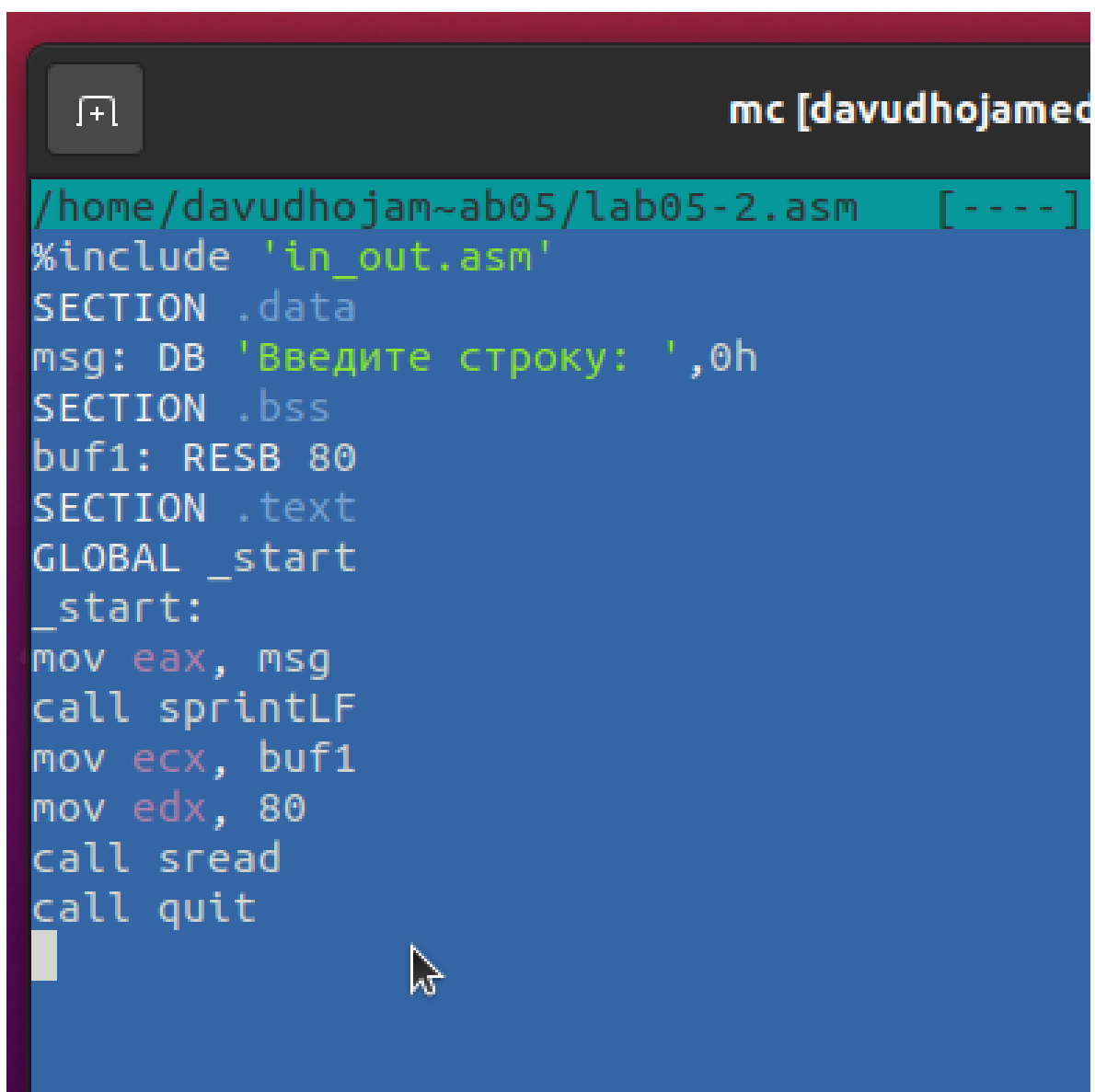


Рис. 2.6: Копирование файла

Написал код программы lab05-2.asm. Скомпилировал программу и проверили запуск.



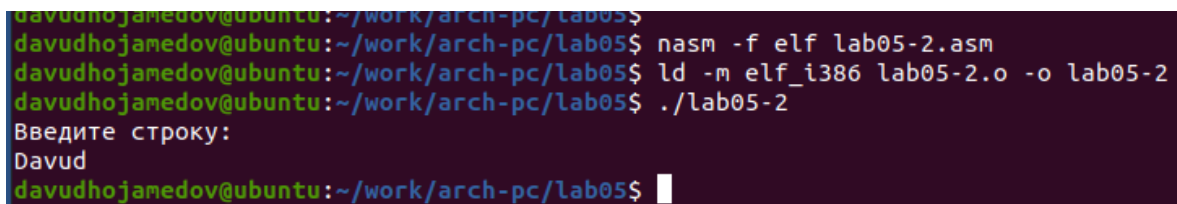
```
mc [davudhojamec
/home/davudhojam~ab05/lab05-2.asm [ - - - - ]
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 2.7: Программа в файле lab05-2.asm

Добавлю код программы в отчет.

```
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h
SECTION .bss
buf1: RESB 80
```

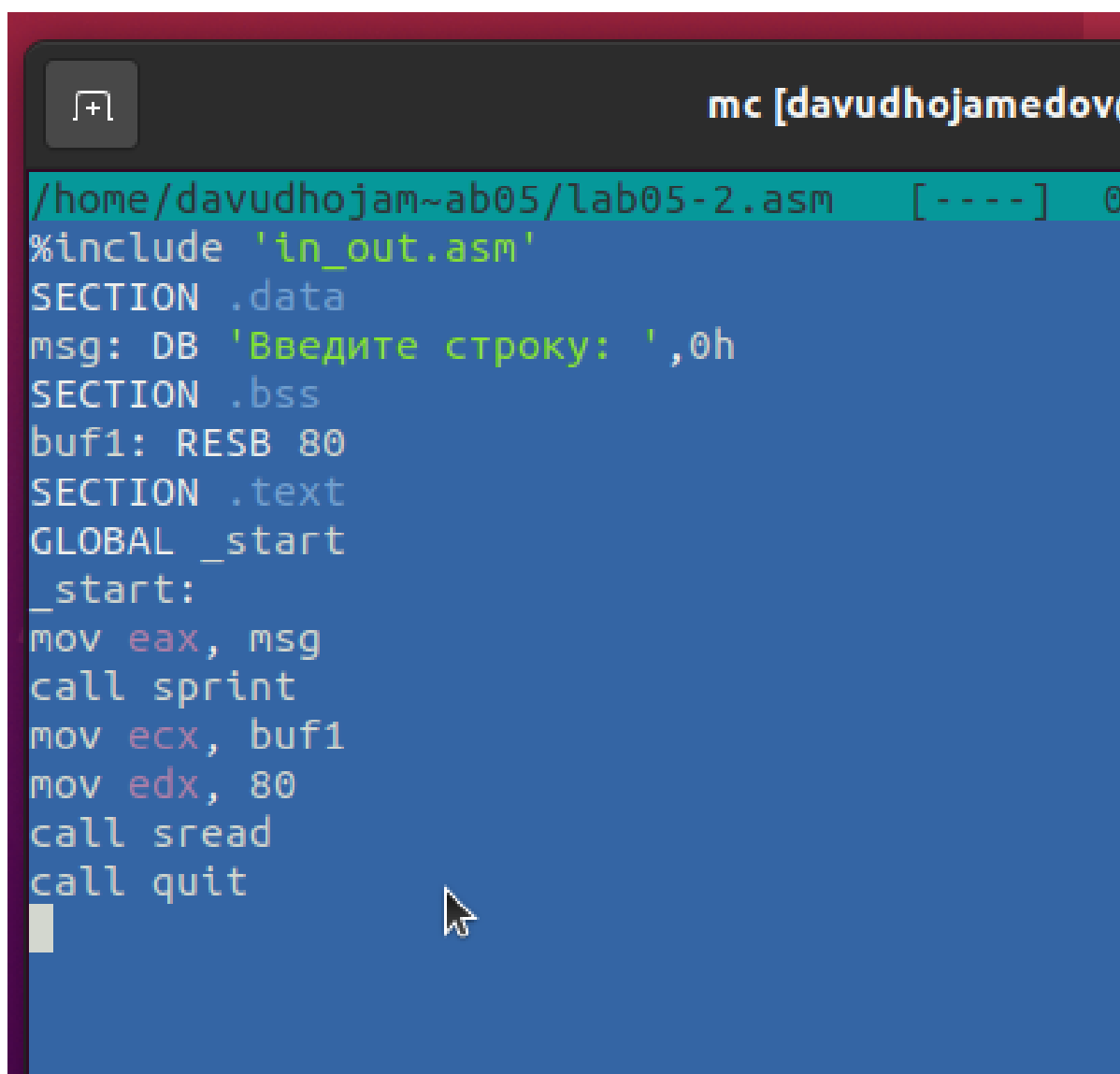
```
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, buf1
mov edx, 80
call sread
call quit
```



```
davudhojamedov@ubuntu:~/work/arch-pc/lab05$
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-2
Введите строку:
Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$
```

Рис. 2.8: Сборка и проверка программы lab05-2.asm

В файле lab5-2.asm заменил подпрограмму sprintLF на sprint. Заново собрал исполняемый файл. Теперь после вывода строки она не завершается символом перехода на новую строку.



```
mc [davudhojamedov  
/home/davudhojam~ab05/lab05-2.asm [ - - - - ]  
%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. 2.9: Программа в файле lab05-2.asm

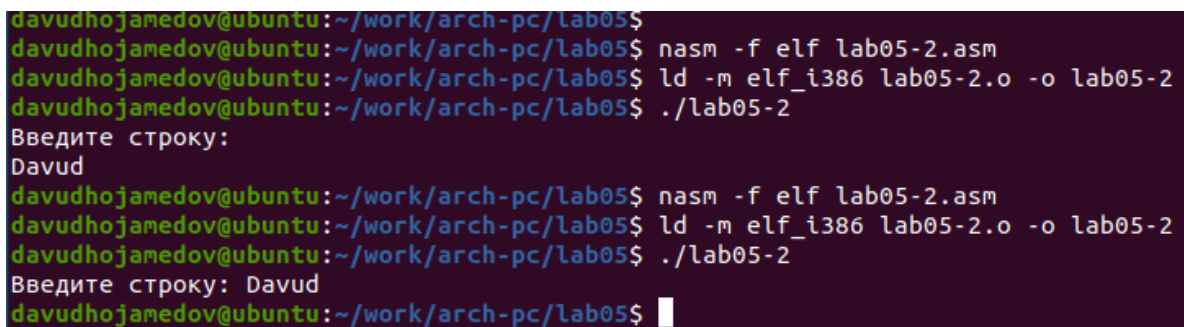
Покажу код программы в отчете.

```
%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

```



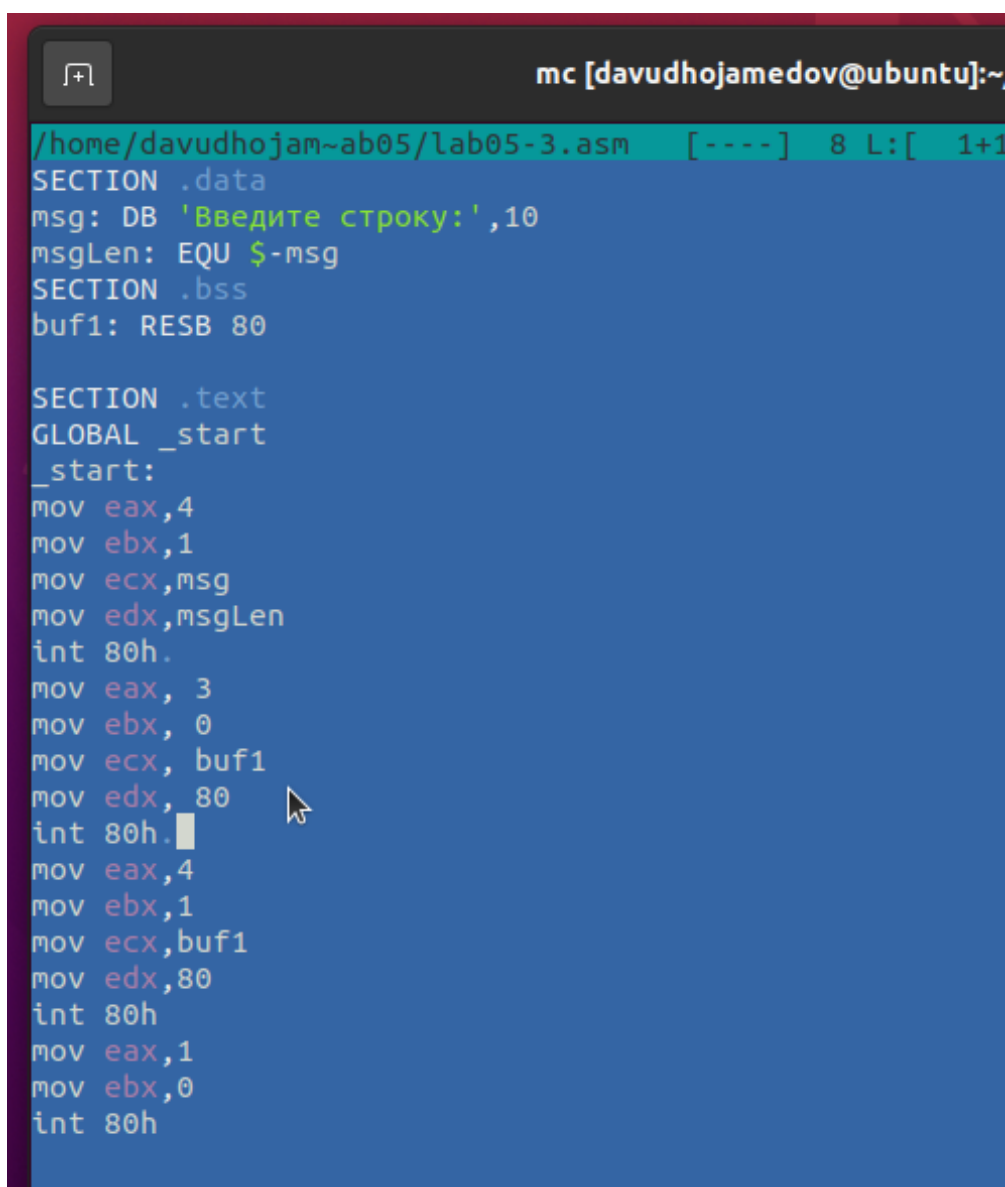
```

davudhojamedov@ubuntu:~/work/arch-pc/lab05$
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-2
Введите строку:
Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-2
Введите строку: Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ █

```

Рис. 2.10: Сборка и проверка программы lab05-2.asm

Скопировал программу lab05-1.asm и изменил код, чтобы вывести приглашение типа “Введите строку:”, ввести строку с клавиатуры, вывести введенную строку на экран.



```
mc [davudhojamedov@ubuntu]:~/
/home/davudhojam~ab05/lab05-3.asm [----] 8 L:[ 1+1
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. 2.11: Программа в файле lab05-3.asm

Покажу код программы в отчете.

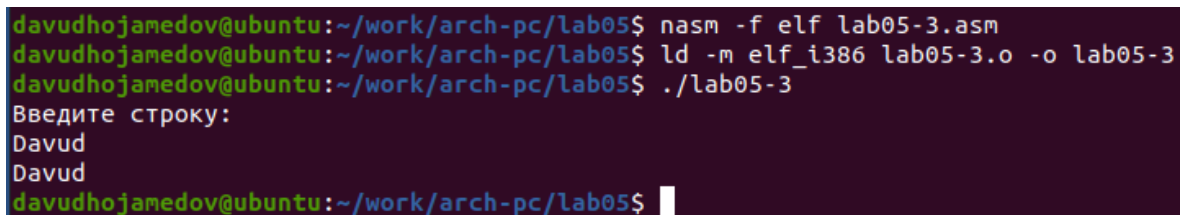
```
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80
```



```

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

```



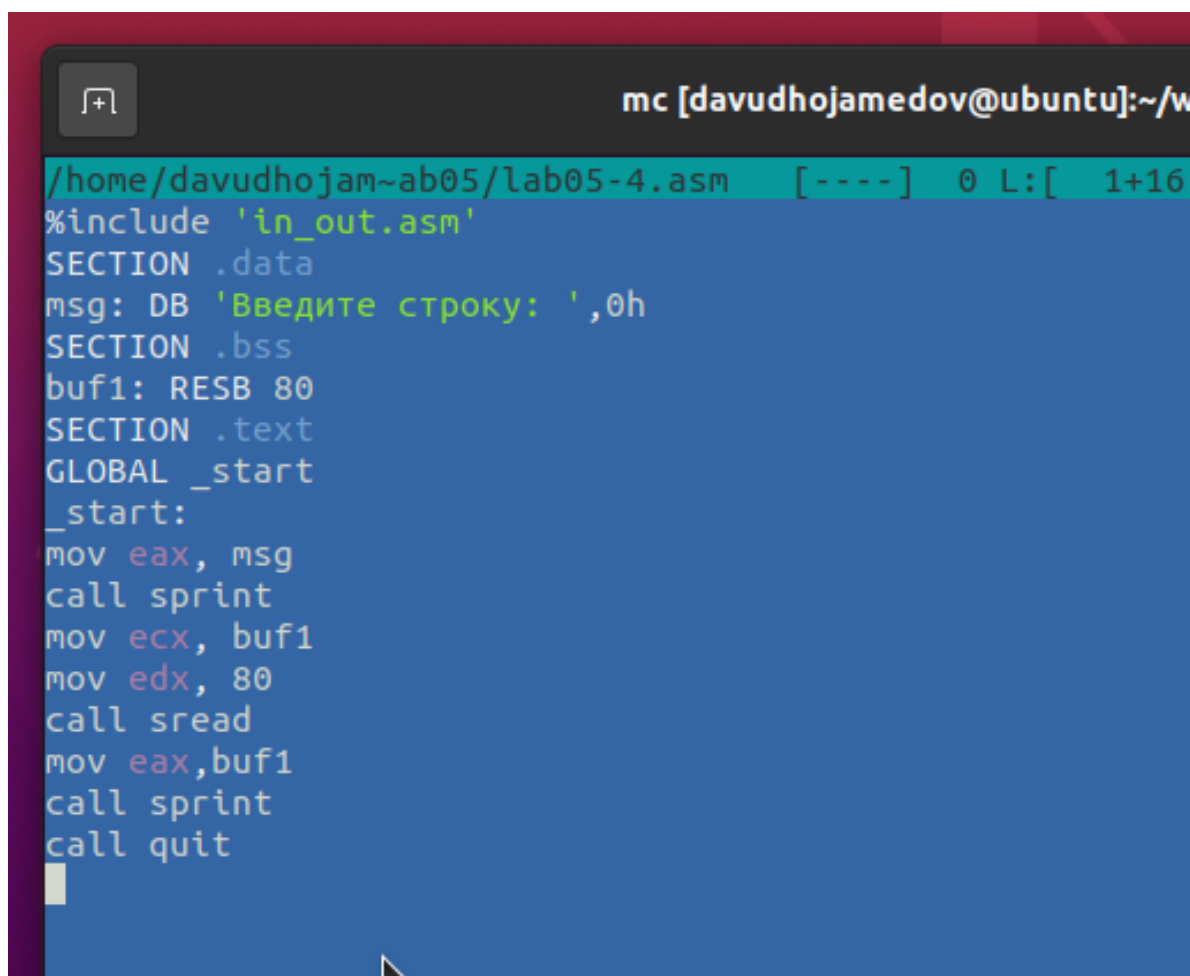
```

davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-3.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-3
Введите строку:
Davud
Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ █

```

Рис. 2.12: Сборка и проверка программы lab05-3.asm

Скопировал программу lab05-2.asm и изменил код, чтобы вывести приглашение типа “Введите строку:”, ввести строку с клавиатуры, вывести введённую строку на экран.



```
mc [davudhojamedov@ubuntu]:~/w
/home/davudhojam~ab05/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. 2.13: Программа в файле lab05-4.asm

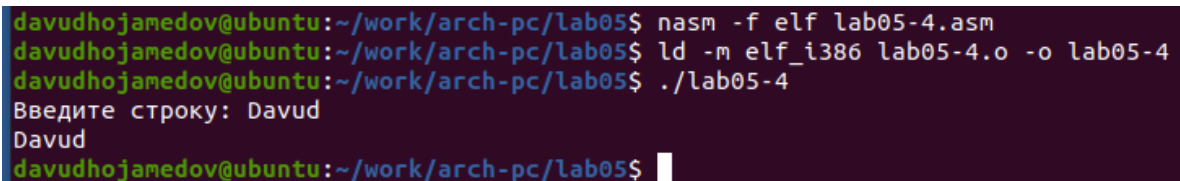
Покажу код программы в отчете.

```
%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

```



```

davudhojamedov@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-4.asm
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-4.o -o lab05-4
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ ./lab05-4
Введите строку: Davud
Davud
davudhojamedov@ubuntu:~/work/arch-pc/lab05$ █

```

Рис. 2.14: Сборка и проверка программы lab05-4.asm

Отличие этих двух реализаций 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 call.

3 Выводы

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