

Отчёт по лабораторной работе 5

Архитектура компьютеров

Мухамметмырадов Рахым

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	15

Список иллюстраций

2.1	Создание каталога	6
2.2	Программа lab05-1.asm	7
2.3	Просмотр файла lab05-1.asm	8
2.4	Запуск программы lab05-1.asm	8
2.5	Копирование файла	9
2.6	Программа lab05-2.asm	10
2.7	Запуск программы lab05-2.asm	10
2.8	Программа lab05-2.asm	11
2.9	Запуск программы lab05-2.asm	11
2.10	Программа lab05-3.asm	12
2.11	Запуск программы lab05-3.asm	13
2.12	Программа lab05-4.asm	13
2.13	Запуск программы lab05-4.asm	14

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

1. Открыл Midnight Commander. Перешел в каталог ~/work/arch-рс. Создал каталог lab05.

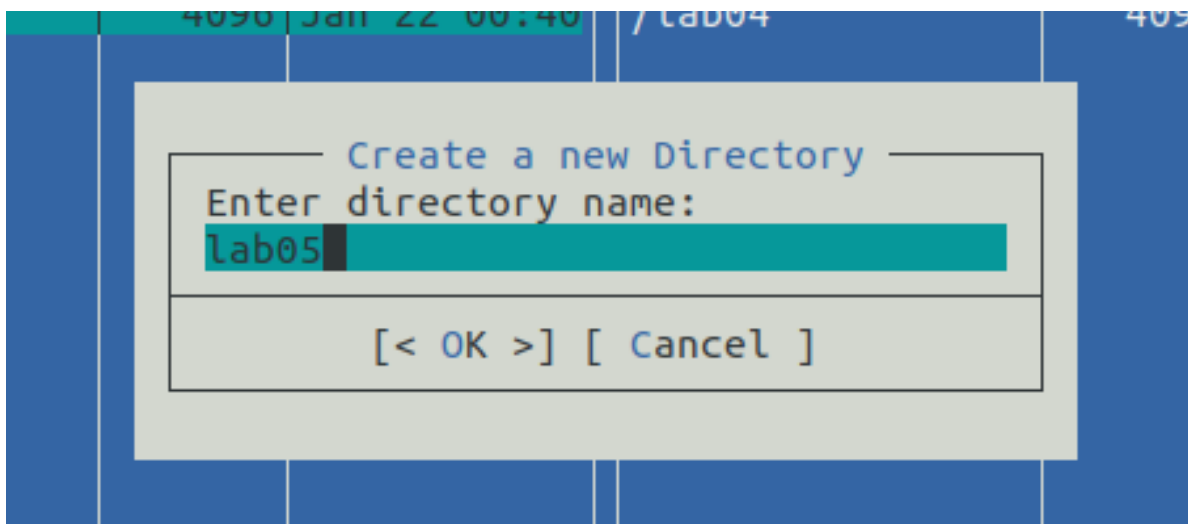
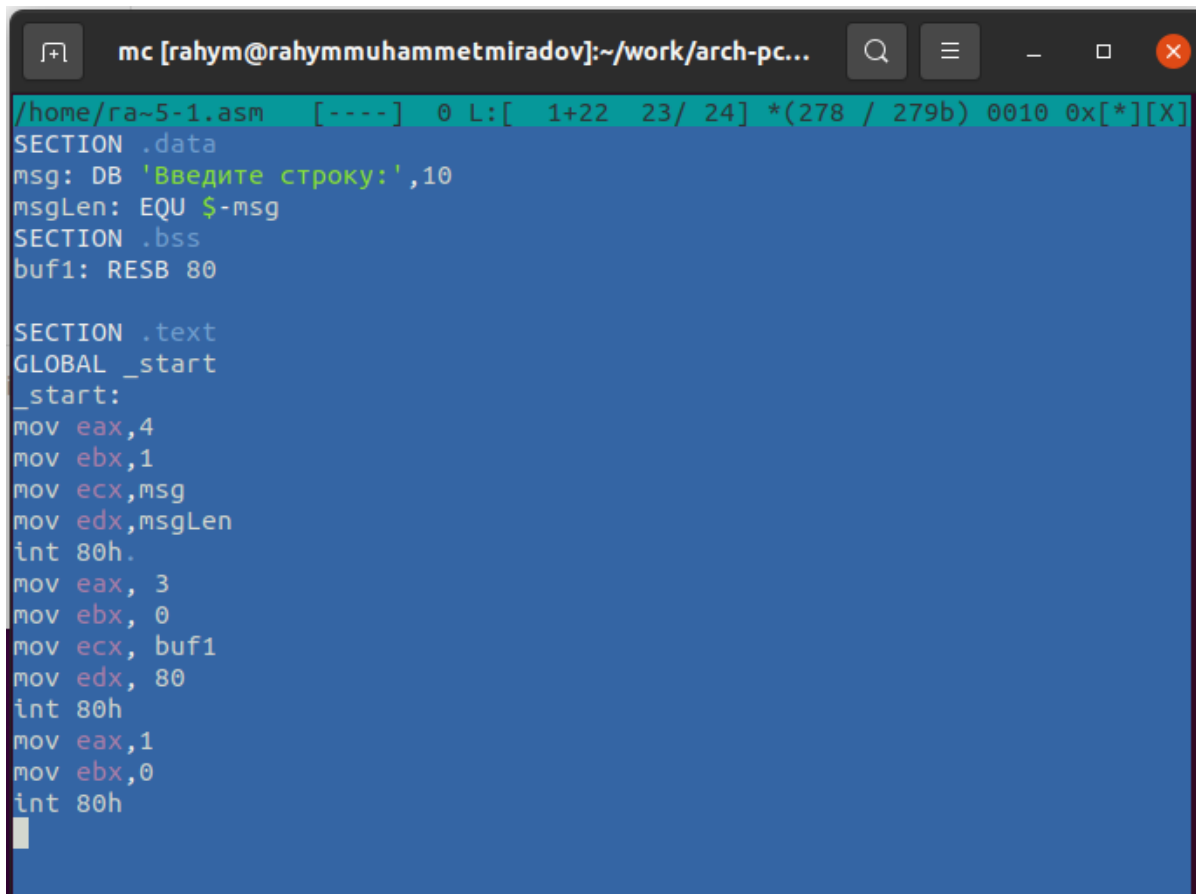


Рис. 2.1: Создание каталога

2. Создал файл lab05-1.asm. Открыл файл на редактирование и написал код.

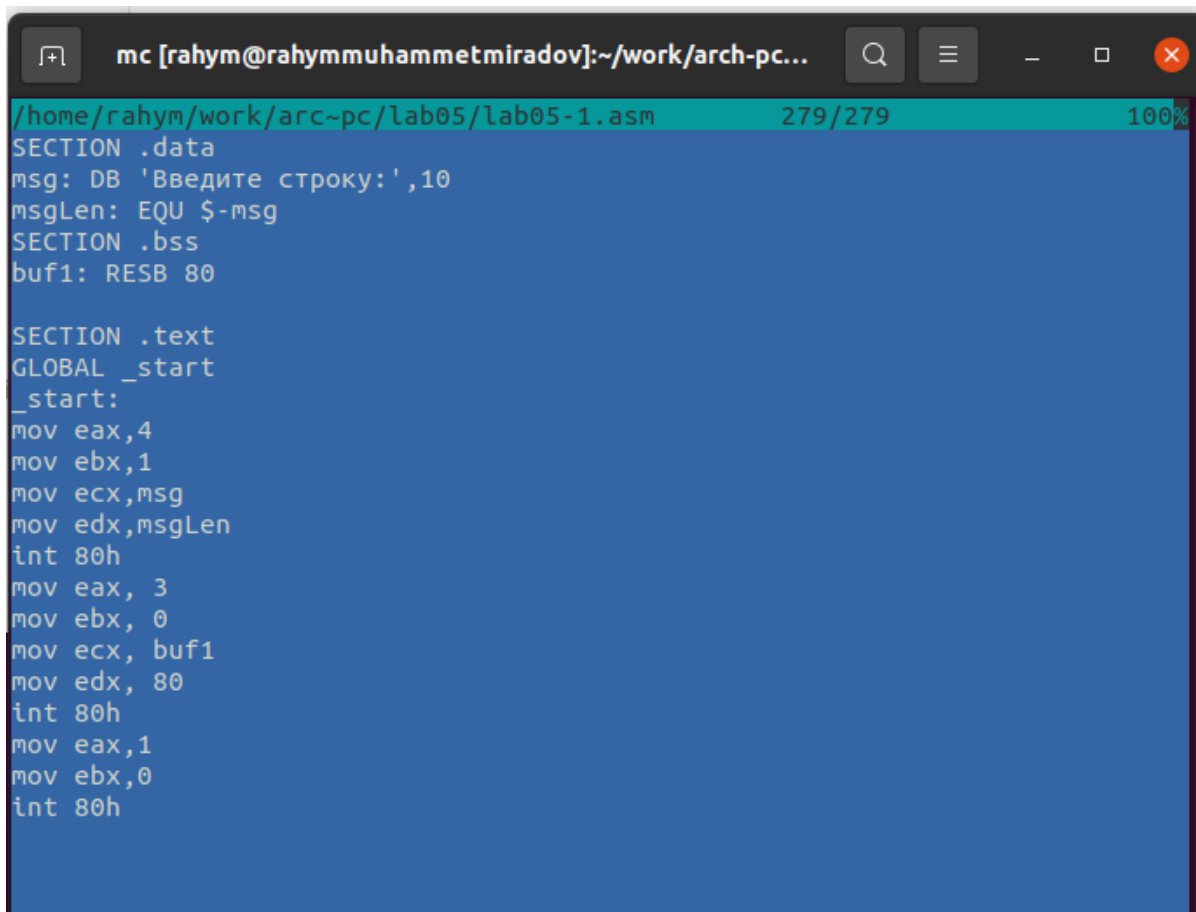


```
mc [rahym@rahymmuhammetmiradov]:~/work/arch-pc...
/home/ra~5-1.asm [----] 0 L:[ 1+22 23/ 24] *(278 / 279b) 0010 0x[*][X]
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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. 2.2: Программа lab05-1.asm

3. Открыл файл на просмотр и проверил набранный код.

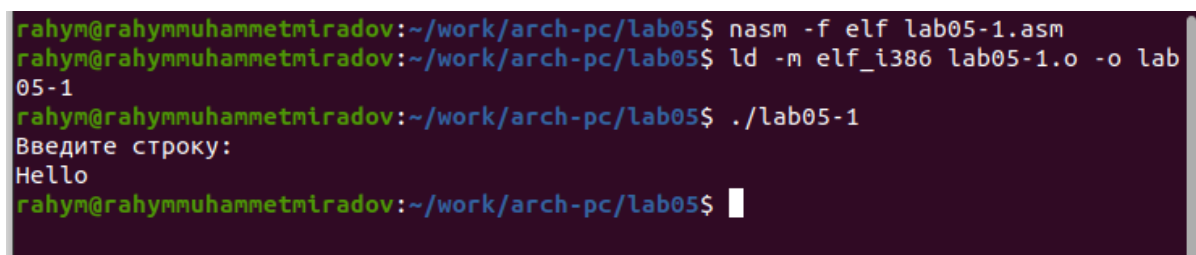


```
mc [rahym@rahymmuhammetmiradov]:~/work/arch-pc...
/home/rahym/work/arch-pc/lab05/lab05-1.asm 279/279 100%
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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: Просмотр файла lab05-1.asm

4. Получил исполняемый файл и проверил как он работает.



```
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ nasm -f elf lab05-1.asm
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-1.o -o lab05-1
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ./lab05-1
Введите строку:
Hello
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$
```

Рис. 2.4: Запуск программы lab05-1.asm

5. Скачал файл in_out.asm. Добавил файл in_out.asm в рабочий каталог. Скопировал lab05-1.asm в lab05-2.asm.

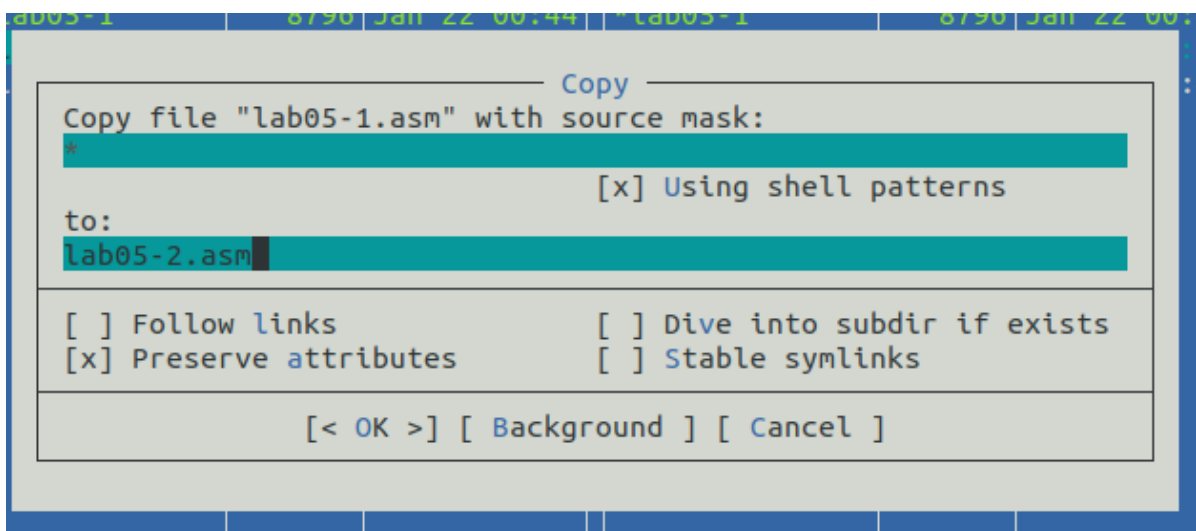
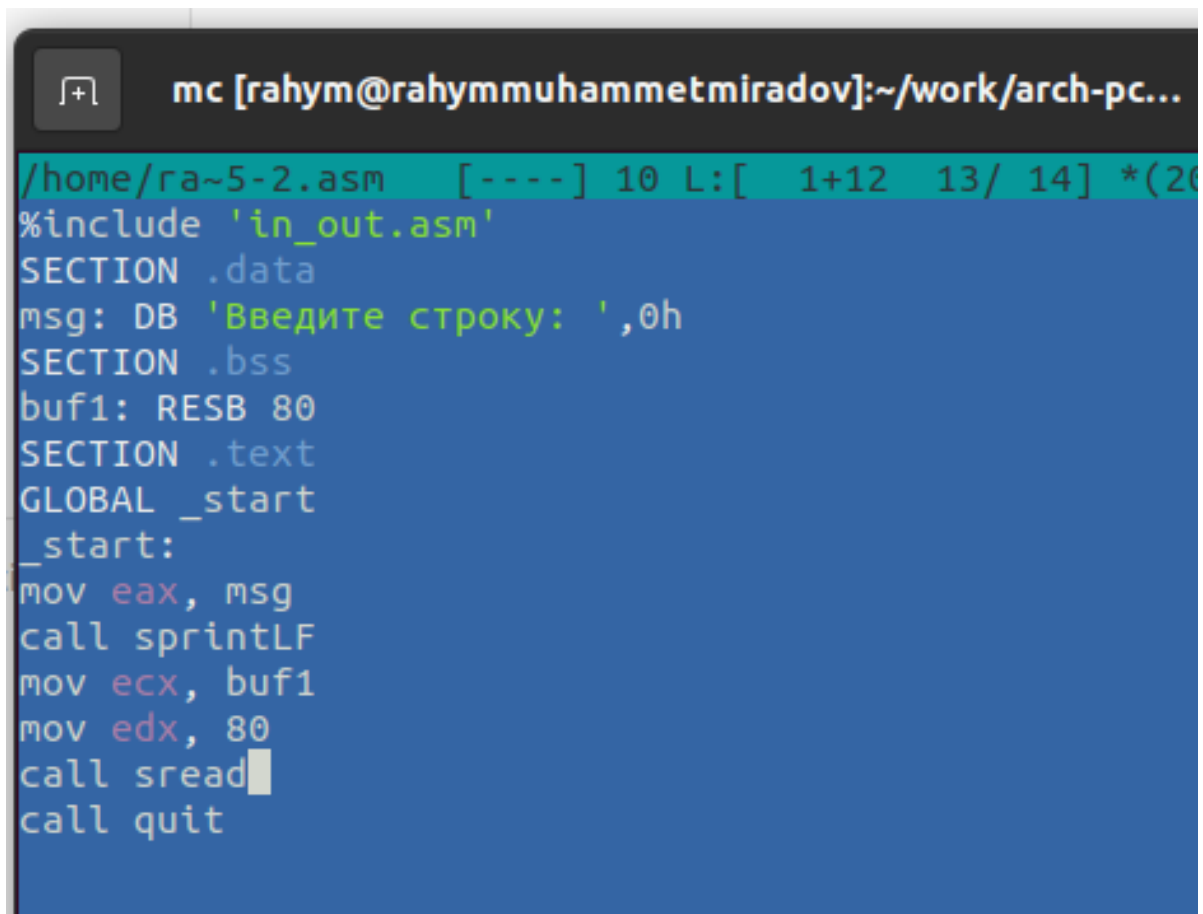


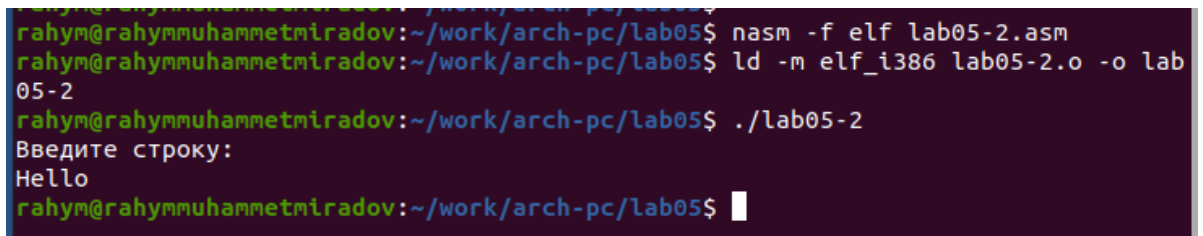
Рис. 2.5: Копирование файла

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



```
mc [rahym@rahymmuhammetmiradov]:~/work/arch-pc...  
/home/ra~5-2.asm [----] 10 L:[ 1+12 13/ 14] *(20  
%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
```

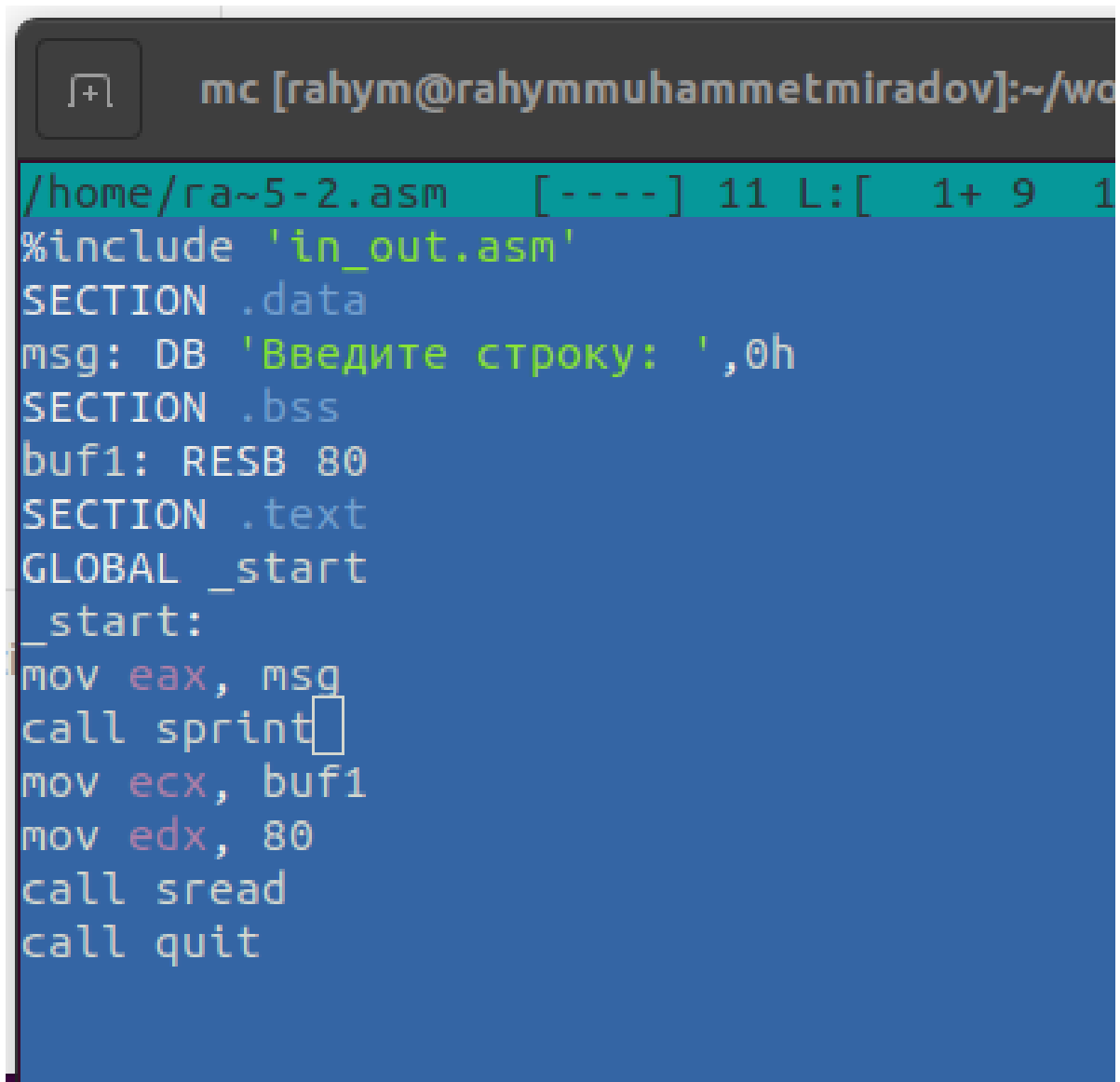
Рис. 2.6: Программа lab05-2.asm



```
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ./lab05-2  
Введите строку:  
Hello  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$
```

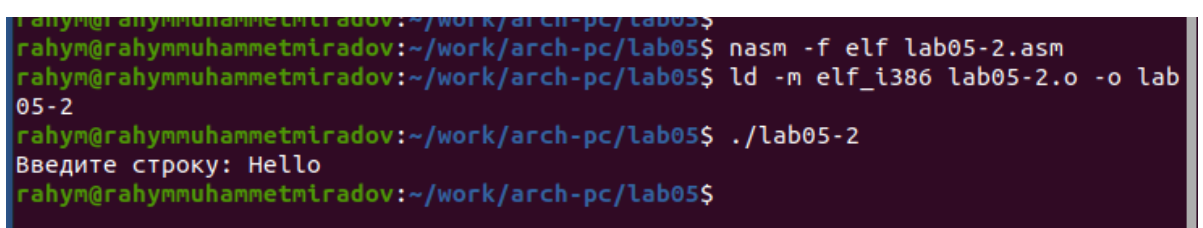
Рис. 2.7: Запуск программы lab05-2.asm

7. В файле lab5-2.asm заменил подпрограмму sprintLF на sprint. Заново собрал исполняемый файл. Теперь вывод строки происходит без перехода на следующую строку.



```
mc [rahym@rahymmuhammetmiradov]:~/wo
/home/ra~5-2.asm [ - - - - ] 11 L: [ 1+ 9 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 sprint
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 2.8: Программа lab05-2.asm



```
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab
05-2
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ./lab05-2
Введите строку: Hello
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$
```

Рис. 2.9: Запуск программы lab05-2.asm

8. Скопировал программу lab05-1.asm и изменил код, чтобы вывести

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



```
mc [rahym@rahymmuhammetmiradov]:~/wo
/home/ra~5-3.asm [----] 0 L:[ 3+ 6 9
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. 2.10: Программа lab05-3.asm

```

rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ nasm -f elf lab05-3.asm
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ./lab05-3
Введите строку:
Hello
Hello
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$

```

Рис. 2.11: Запуск программы lab05-3.asm

15. Скопировал программу lab05-2.asm и сделал аналогично заданию выше, но теперь используются возможности из файла in_out.asm.

```

mc [rahym@rahymmuhammetmiradov]:~/work/arch-
/home/ra~5-4.asm  [ - - - ]  0 L:[  1+12  13/ 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. 2.12: Программа lab05-4.asm

```
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ nasm -f elf lab05-4.asm  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-4.o -o lab  
05-4  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$ ./lab05-4  
Введите строку: Hello  
Hello  
rahym@rahymmuhammetmiradov:~/work/arch-pc/lab05$
```

Рис. 2.13: Запуск программы lab05-4.asm

3 Выводы

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