

Отчёт по лабораторной работе 5

Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM

Абдухалилов Мухаммадаюбхон Юсуфхонович НБИ-01-24

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Знакомство с Midnight Commander	6
2.2	Подключение внешнего файла in_out.asm	10
2.3	Задание для самостоятельной работы	15
3	Выводы	19

Список иллюстраций

2.1	Запуск Midnight Commander	6
2.2	Создание каталога	7
2.3	Создание файла lab05-1.asm	7
2.4	Программа в файле lab05-1.asm	8
2.5	Просмотр файла lab05-1.asm	9
2.6	Запуск программы lab05-1.asm	10
2.7	Копирование файла in_out.asm	11
2.8	Копирование файла lab05-1.asm	12
2.9	Программа в файле lab05-2.asm	13
2.10	Запуск программы lab05-2.asm	13
2.11	Программа в файле lab05-2.asm	14
2.12	Запуск программы lab05-2.asm	14
2.13	Копирование файла lab05-1.asm	15
2.14	Программа в файле lab05-3.asm	16
2.15	Запуск программы lab05-3.asm	16
2.16	Копирование файла lab05-2.asm	17
2.17	Программа в файле lab05-4.asm	18
2.18	Запуск 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 Выполнение лабораторной работы

2.1 Знакомство с Midnight Commander

Открыл Midnight Commander, с помощью клавишь со стрелками и Enter перешел в каталог ~/work/arch-рс. Далее нажал F7 и создал каталог lab05

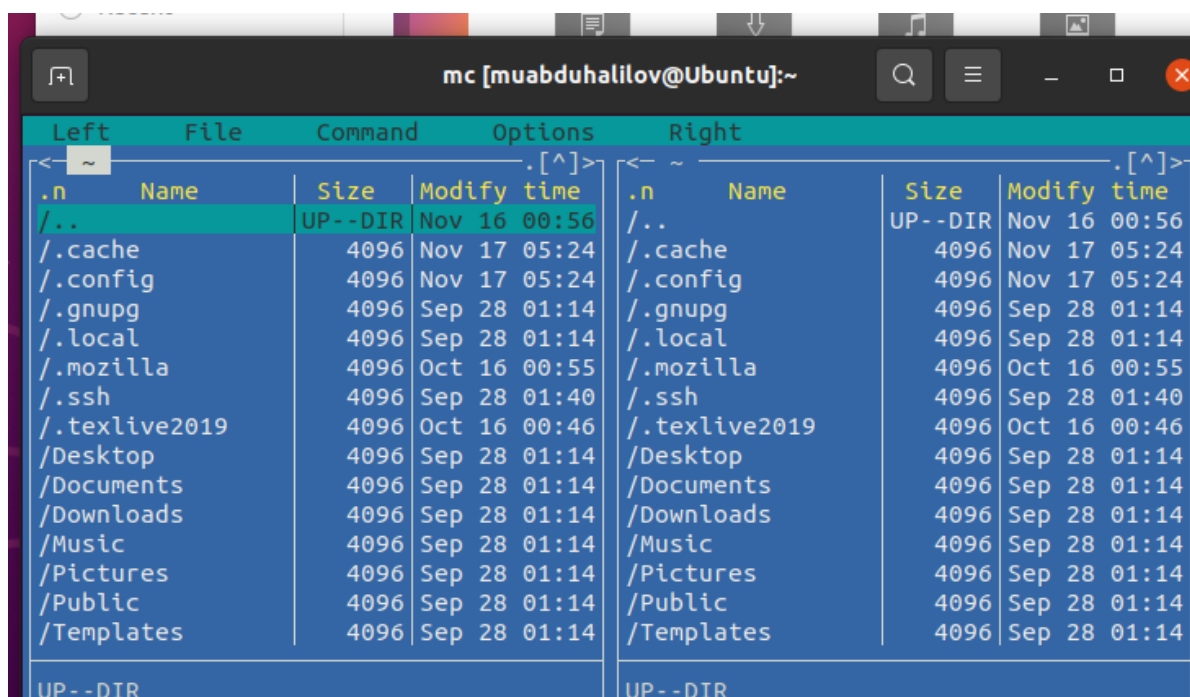


Рис. 2.1: Запуск Midnight Commander

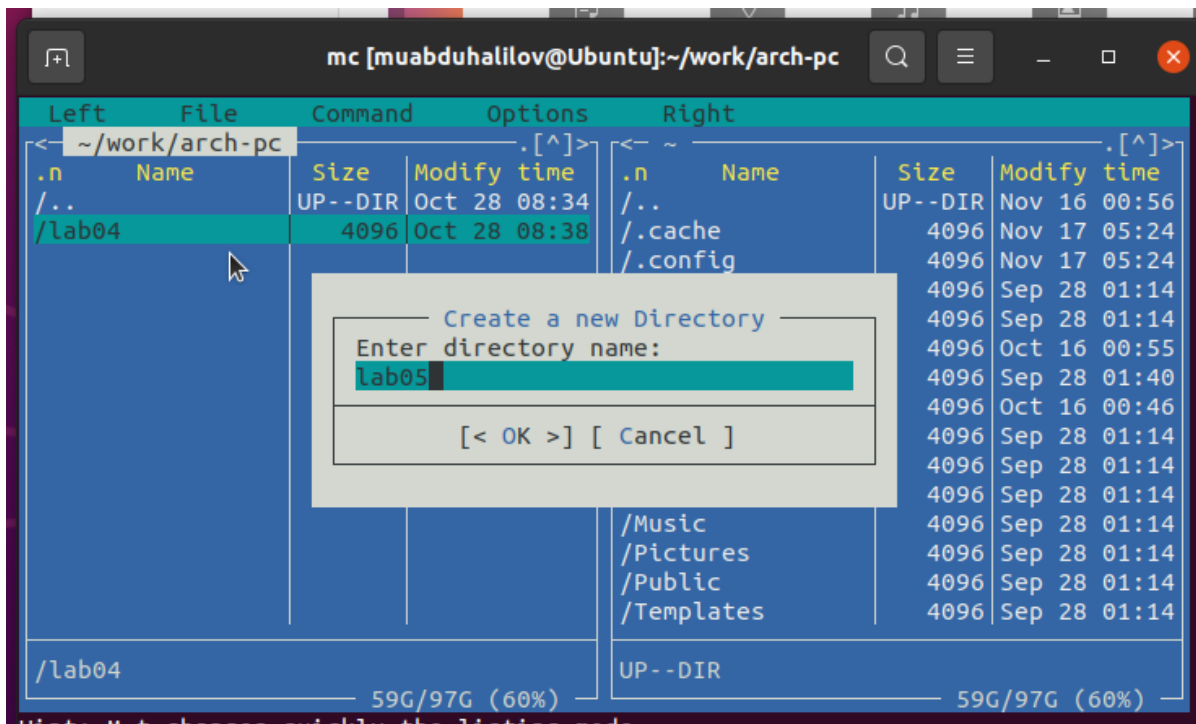


Рис. 2.2: Создание каталога

При помощи touch создал файл lab05-1.asm

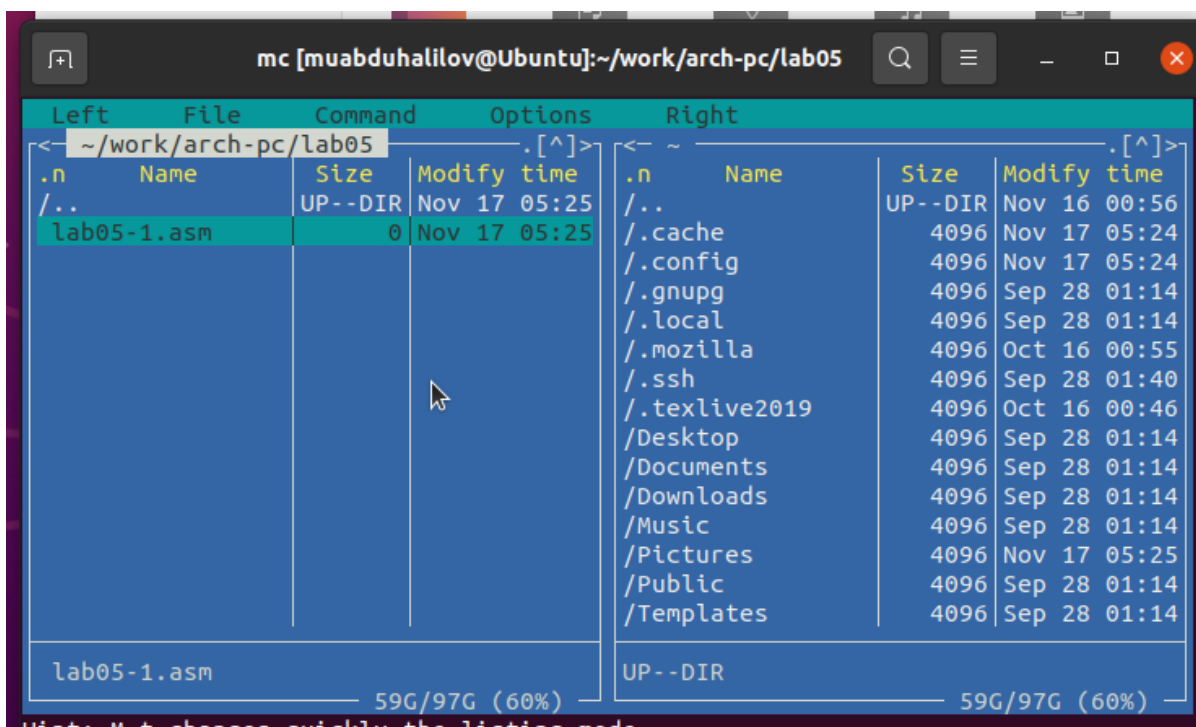
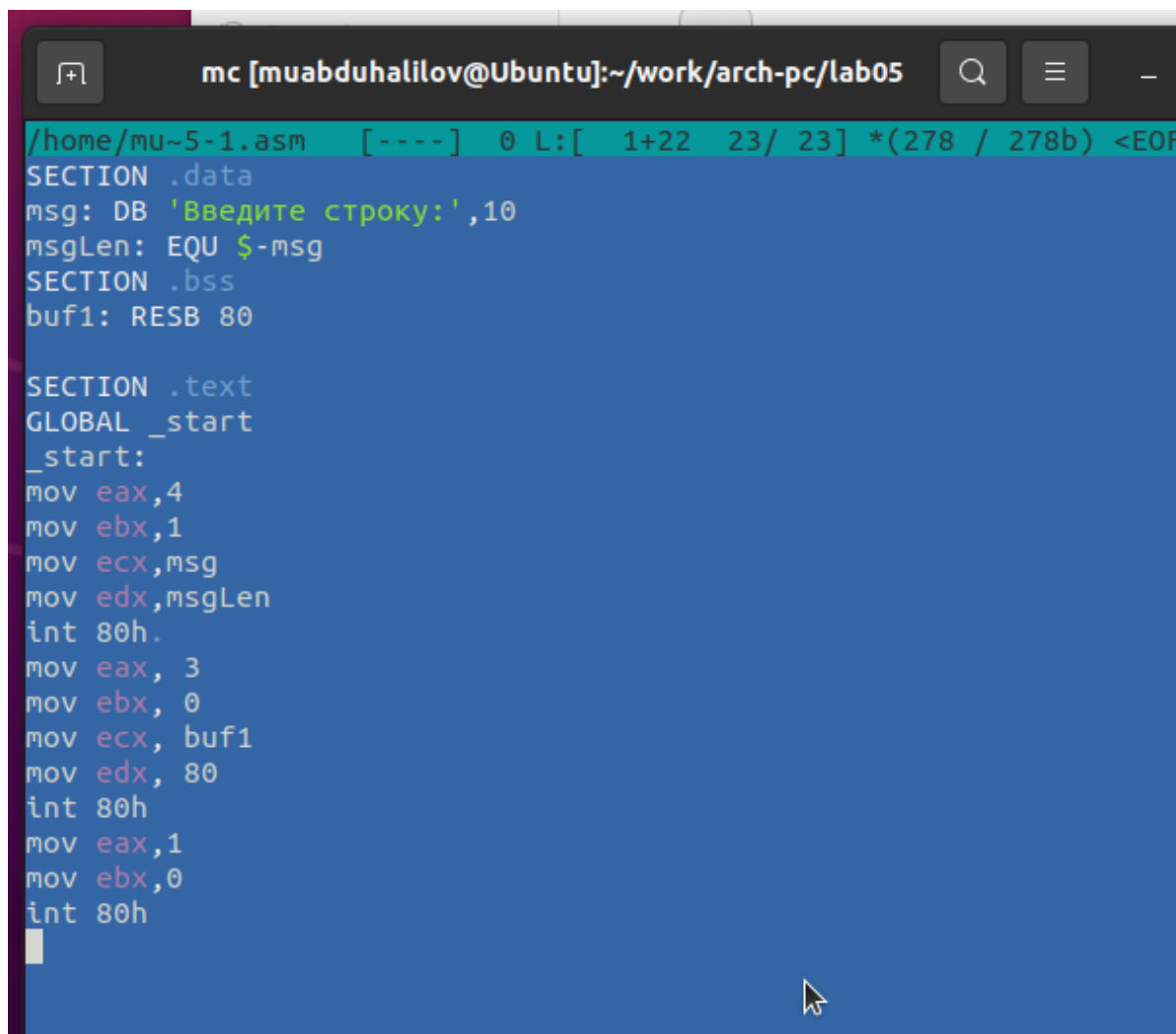


Рис. 2.3: Создание файла lab05-1.asm

Открыл файл на редактирование клавишей F4, выбрал редактор mceditor, написал код программы из задания.

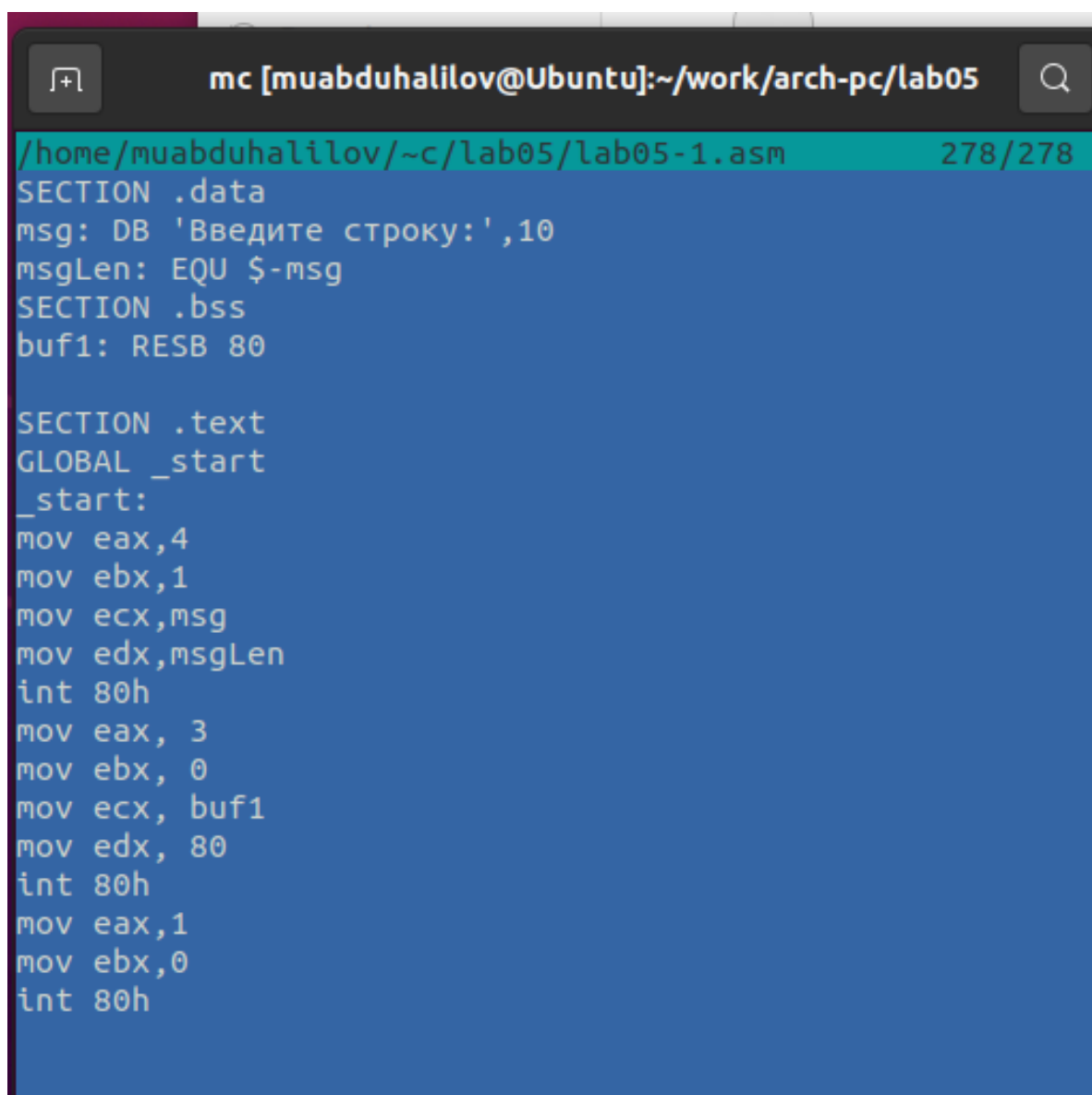


```
mc [muabduhalilov@Ubuntu]:~/work/arch-pc/lab05
/home/mu~5-1.asm [----] 0 L:[ 1+22 23/ 23] *(278 / 278b) <EOF
SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80

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: Программа в файле lab05-1.asm

Открыл файл на просмотр клавишей F3 и убедился, что он содержит набранный код.

A screenshot of a code editor window. The title bar shows the user 'muabduhalilov@Ubuntu' and the file path '~/work/arch-pc/lab05'. The editor displays assembly code for 'lab05-1.asm'. The code is organized into 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, starting with a global _start function. The logic involves setting up registers (eax, ebx, ecx, edx) and using system calls (int 80h) to interact with the user and the buffer. 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. 2.5: Просмотр файла lab05-1.asm

Транслировал файл программы в объектный файл, выполнил компоновку объектного файла, получил исполняемый файл программы и проверил ее работу.

```
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-1.asm
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-1.o -o lab05-1
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ./lab05-1
Введите строку:
test
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
```

Рис. 2.6: Запуск программы lab05-1.asm

2.2 Подключение внешнего файла in_out.asm

Для упрощения написания программ часто встречающиеся одинаковые участки кода (такие как, например, вывод строки на экран или выход из программы) можно оформить в виде подпрограмм и сохранить в отдельные файлы, а во всех нужных местах поставить вызов нужной подпрограммы. Это позволяет сделать основную программу более удобной для написания и чтения.

Для выполнения лабораторных работ используется файл in_out.asm, который содержит следующие подпрограммы:

- `slen` – вычисление длины строки (используется в подпрограммах печати сообщения для определения количества выводимых байтов);
- `sprint` – вывод сообщения на экран, перед вызовом `sprint` в регистр `eax` необходимо записать выводимое сообщение (`mov eax,;`);
- `sprintLF` – работает аналогично `sprint`, но при выводе на экран добавляет к сообщению символ перевода строки;
- `sread` – ввод сообщения с клавиатуры, перед вызовом `sread` в регистр `eax` необходимо записать адрес переменной в которую введенное сообщение буд записано (`mov eax,;`), в регистр `ebx` – длину вводимой строки (`mov ebx,;`);
- `iprint` – вывод на экран чисел в формате ASCII, перед вызовом `iprint` в регистр `eax` необходимо записать выводимое число (`mov eax,;`);

- `iprintLF` – работает аналогично `iprint`, но при выводе на экран после числа добавляет к символ перевода строки;
- `atoi` – функция преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`, перед вызовом `atoi` в регистр `eax` необходимо записать число (`mov eax,;`);
- `quit` – завершение программы.

Скачал файл `in_out.asm` и разместил его в рабочем каталоге. Для копирования используется клавиша F5. Для перемещения используется клавиша F6.

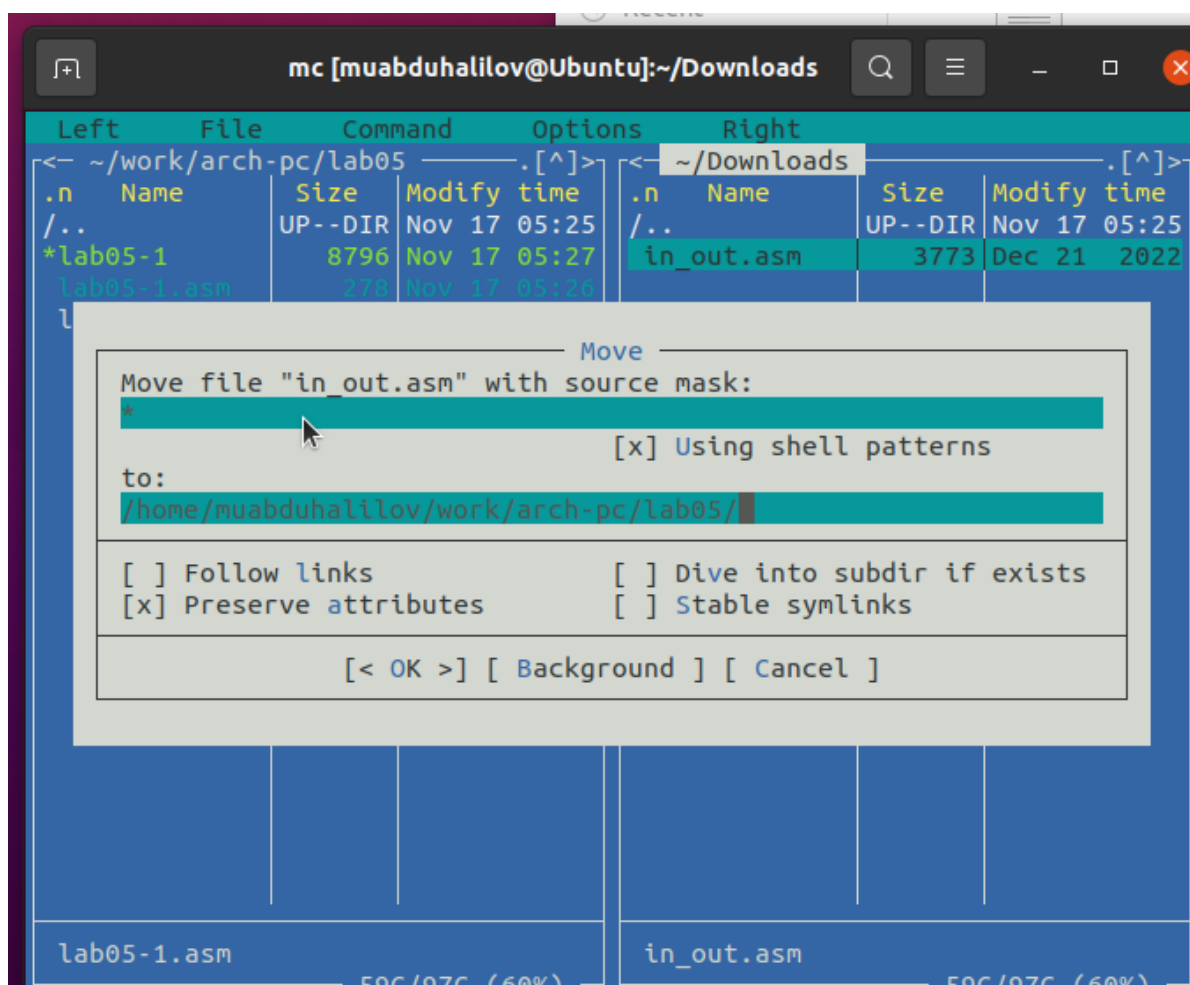


Рис. 2.7: Копирование файла `in_out.asm`

Скопировал `lab05-1.asm` в `lab05-2.asm`.

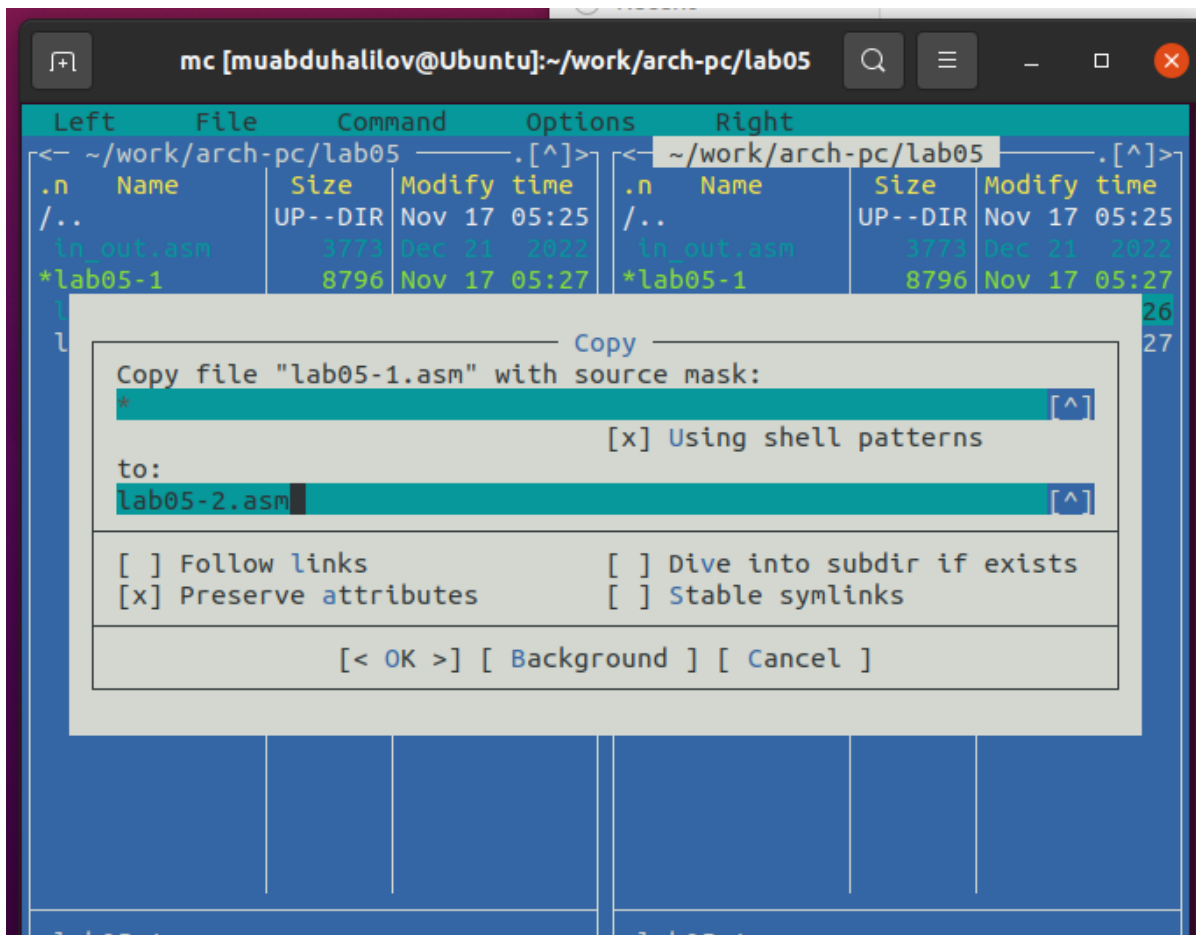
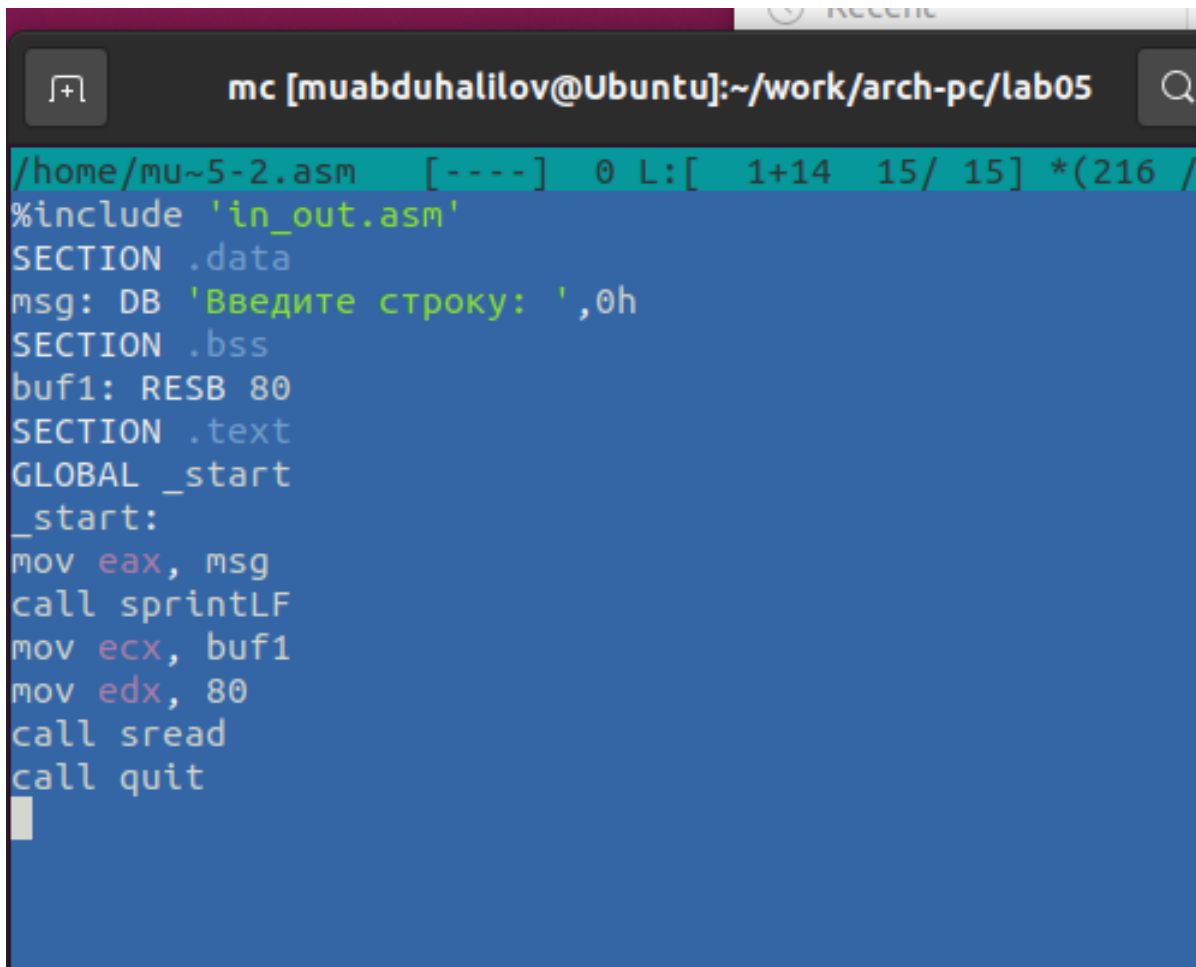


Рис. 2.8: Копирование файла lab05-1.asm

Написал код программы lab05-2.asm с использованием подпрограмм из внешнего файла in_out.asm . Скомпилировал программу и проверил запуск.

A screenshot of a code editor window titled 'mc [muabduhalilov@Ubuntu]:~/work/arch-pc/lab05'. The editor shows the contents of a file named '/home/mu~5-2.asm'. The code is written in assembly language and includes comments in Russian. The code defines a data section with a message, a bss section with a buffer, and a text section with a global _start label. The _start label contains instructions to move the message to the eax register, call sprintf, move the buffer to the ecx register, move 80 to the edx register, call sread, and call quit.

```
/home/mu~5-2.asm [----] 0 L:[ 1+14 15/ 15] *(216 /
#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 sprintf
mov ecx, buf1
mov edx, 80
call sread
call quit
```

Рис. 2.9: Программа в файле lab05-2.asm

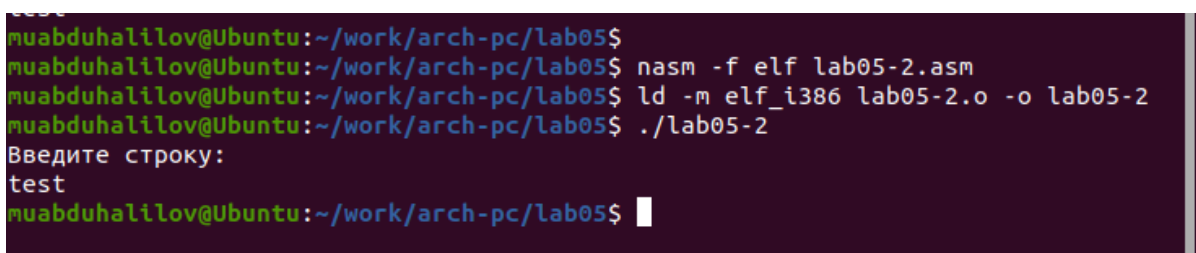
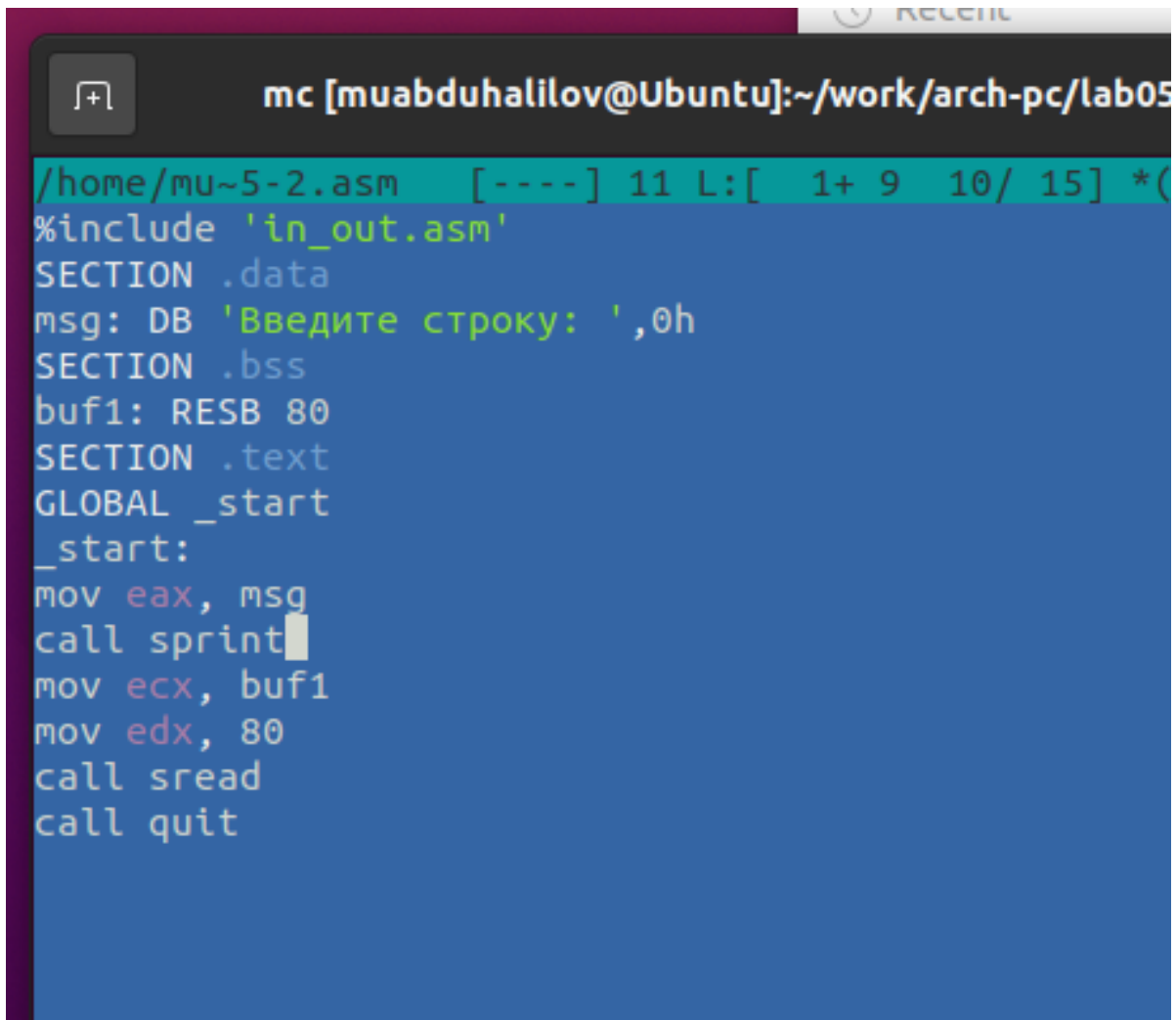
A screenshot of a terminal window showing the compilation and execution of the assembly program. The user runs 'nasm -f elf lab05-2.asm' to compile the assembly code into an object file 'lab05-2.o'. Then, they run 'ld -m elf_i386 lab05-2.o -o lab05-2' to link the object file into an executable file 'lab05-2'. Finally, they run './lab05-2' to execute the program. The program prompts the user to enter a string, and the user enters 'test'.

Рис. 2.10: Запуск программы lab05-2.asm

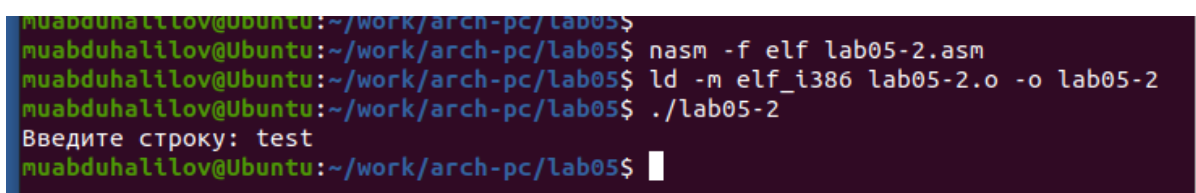
В файле lab5-2.asm заменил подпрограмму sprintf на printf. Заново собрал исполняемый файл. Теперь после вывода строки она не завершается символом перехода на новую строку.



The screenshot shows a code editor window titled 'mc [muabduhalilov@Ubuntu]:~/work/arch-pc/lab05'. The code is in assembly format and includes a header line with file statistics. The code defines a data section with a message, a bss section with a buffer, and a text section with the main logic. The logic involves printing the message and reading input into the buffer.

```
/home/mu~5-2.asm [ - - - ] 11 L:[ 1+ 9 10/ 15] *(
#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. 2.11: Программа в файле lab05-2.asm



The screenshot shows a terminal window with the following commands and output:

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

Рис. 2.12: Запуск программы lab05-2.asm

2.3 Задание для самостоятельной работы

Скопировал программу lab05-1.asm и изменил код, так чтобы она работала по следующему алгоритму:

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

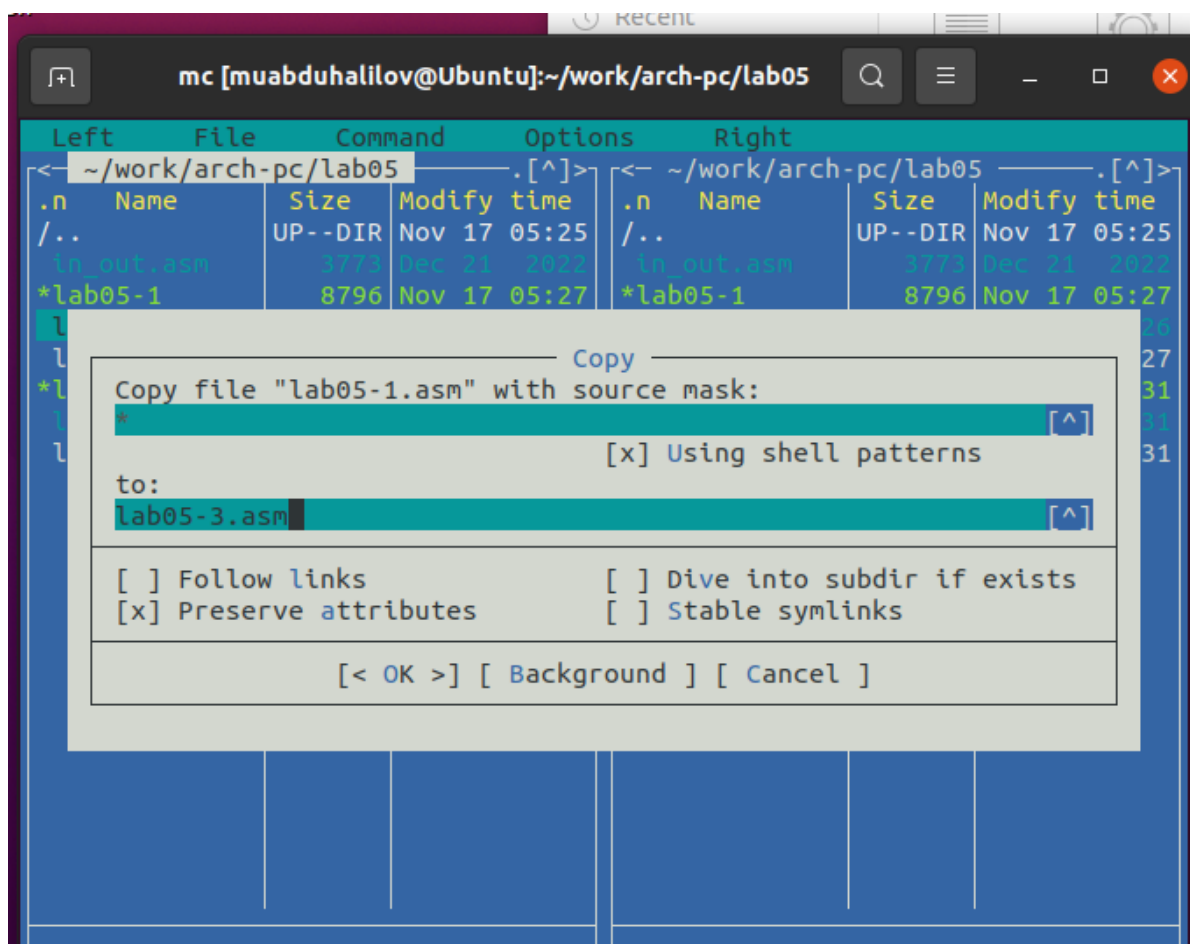
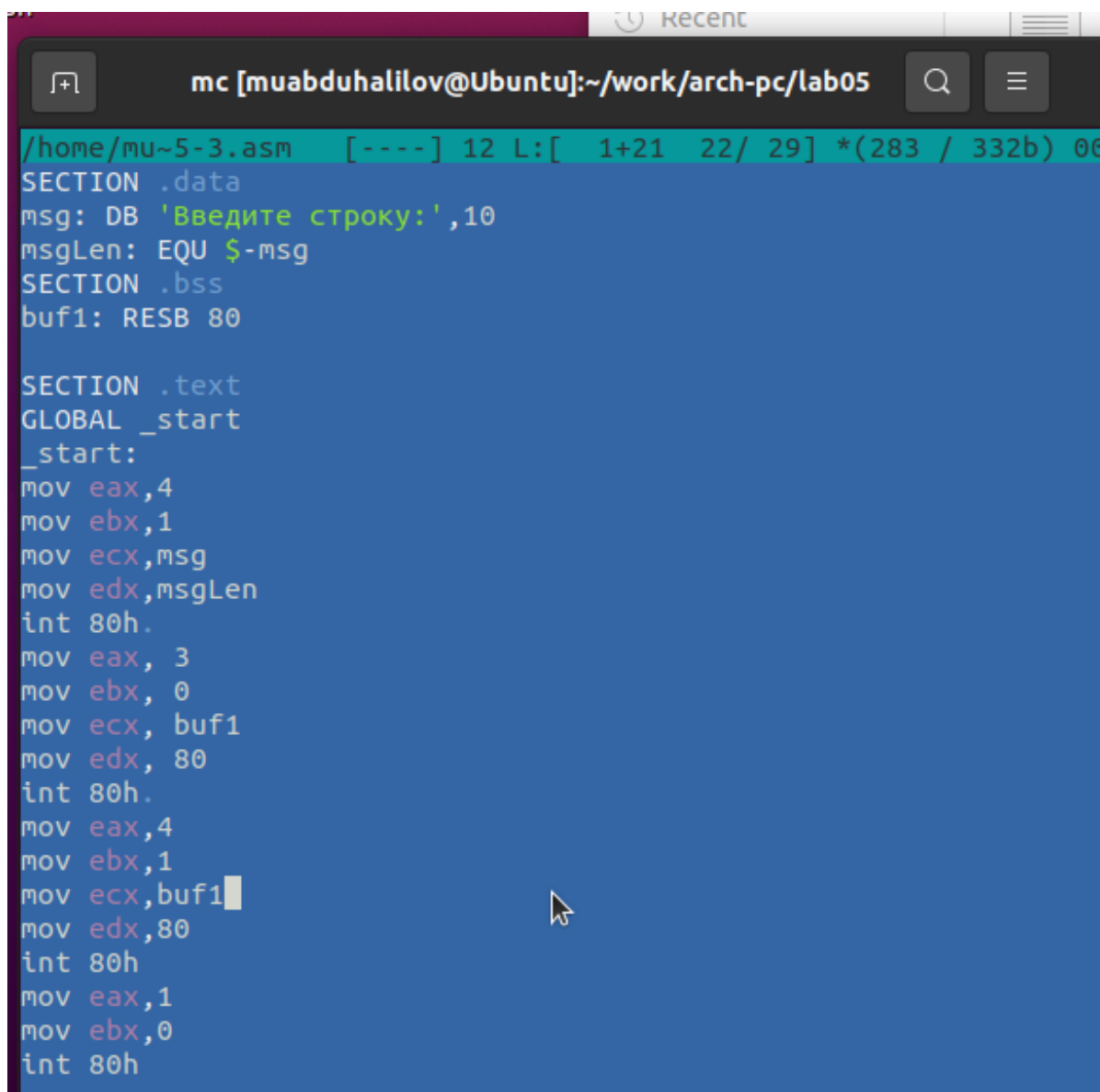


Рис. 2.13: Копирование файла lab05-1.asm

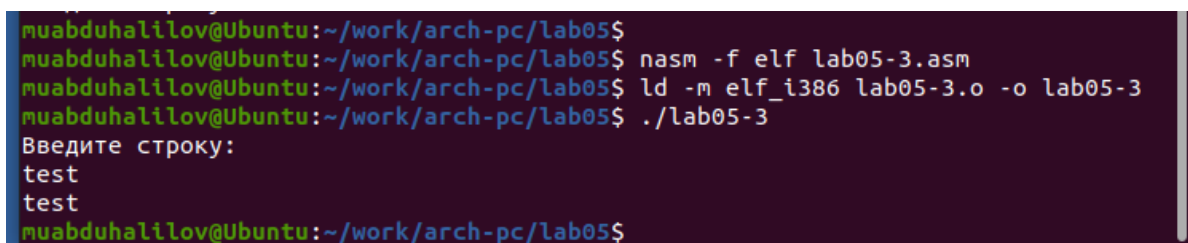


The screenshot shows a code editor window titled "mc [muabduhalilov@Ubuntu]:~/work/arch-pc/lab05". The code is for an assembly program named "lab05-3.asm". It defines a data section with a message "Введите строку:" and a length. It also defines a .bss section with a buffer "buf1" of size 80. The .text section contains the main logic, starting with a global "_start" symbol. The code uses various x86 instructions like "mov", "int", and "push" to set up registers and call system services (int 80h) to print the message and read input from the user.

```
/home/mu~5-3.asm [ - - - ] 12 L:[ 1+21 22/ 29] *(283 / 332b) 00
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. 2.14: Программа в файле lab05-3.asm



The screenshot shows a terminal window with the following commands and output: The user runs "nasm -f elf lab05-3.asm" to assemble the code. Then, they run "ld -m elf_i386 lab05-3.o -o lab05-3" to link the object file. Finally, they run "./lab05-3" to execute the program. The output shows the prompt "Введите строку:" followed by "test" being entered twice.

```
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-3.asm
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ./lab05-3
Введите строку:
test
test
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
```

Рис. 2.15: Запуск программы lab05-3.asm

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

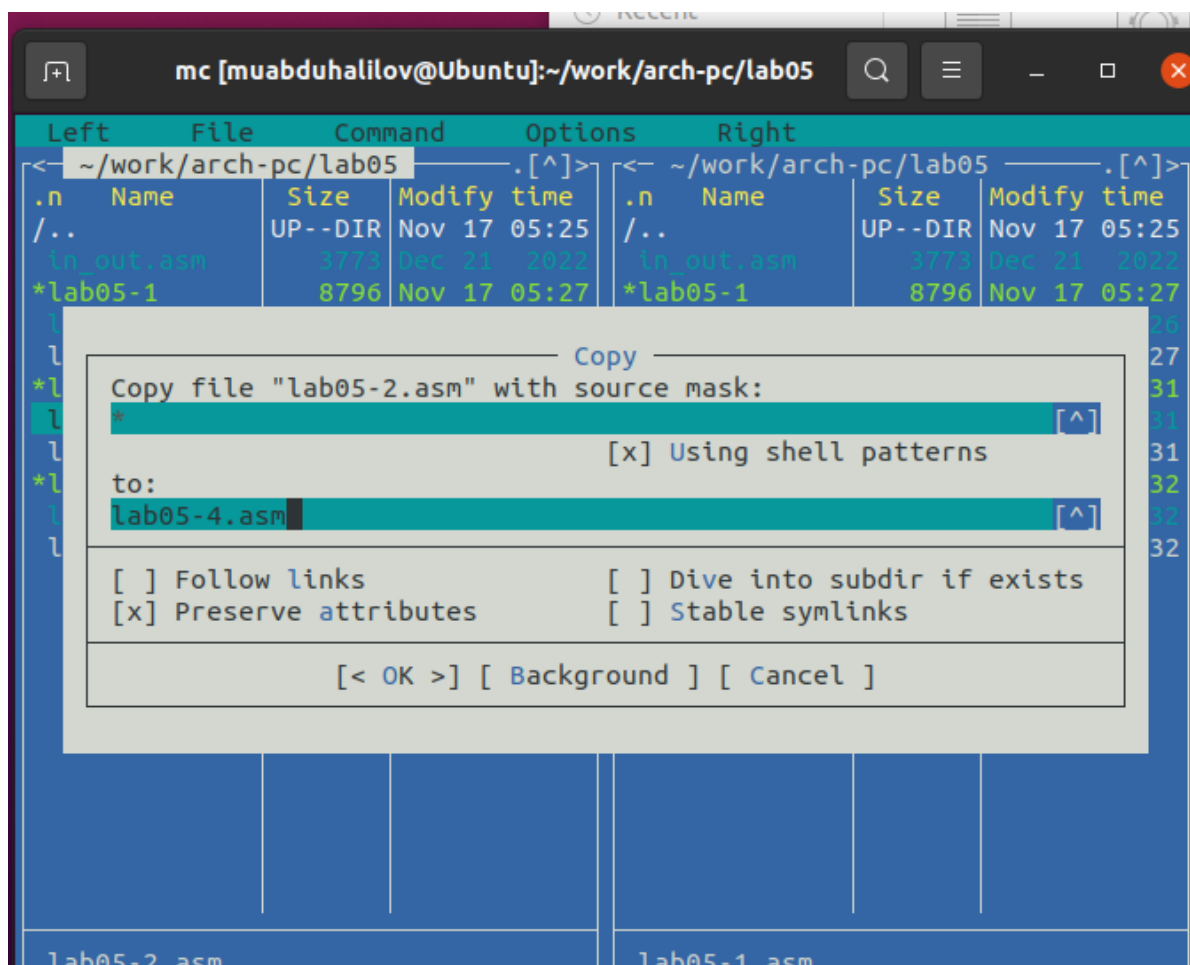
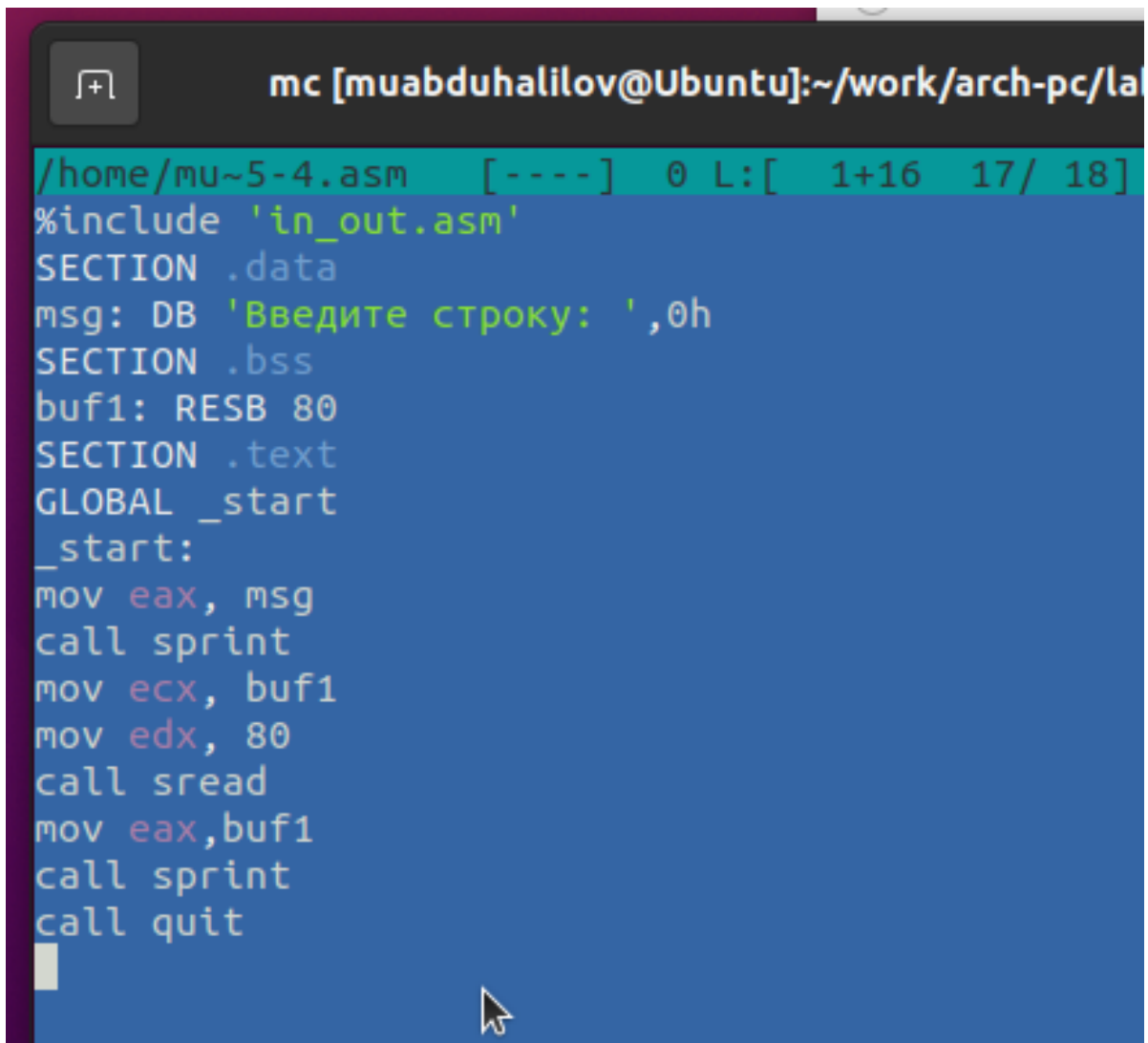
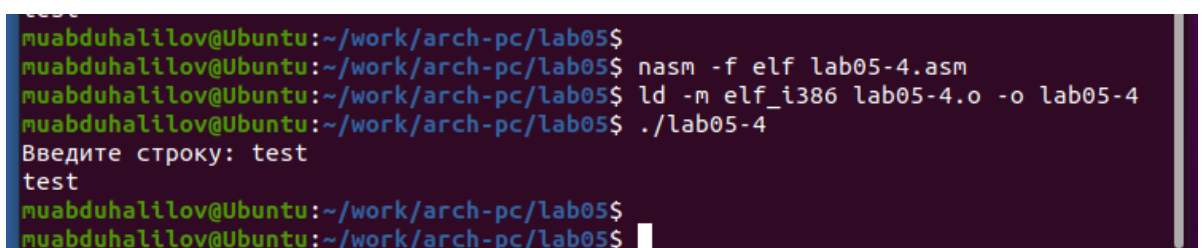


Рис. 2.16: Копирование файла lab05-2.asm



```
mc [muabduhalilov@Ubuntu]:~/work/arch-pc/lab05-4/
/home/mu~5-4.asm [----] 0 L:[ 1+16 17/ 18]
%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. 2.17: Программа в файле lab05-4.asm



```
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab05-4.asm
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-4.o -o lab05-4
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$ ./lab05-4
Введите строку: test
test
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
muabduhalilov@Ubuntu:~/work/arch-pc/lab05$
```

Рис. 2.18: Запуск программы lab05-4.asm

3 Выводы

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