

Отчёт по лабораторной работе 5

Архитектура компьютера

Малютина Софья Александровна

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, а также освоение инструкций языка ассемблера `mov` и `int`.

2 Выполнение лабораторной работы

2.1 Знакомство с Midnight Commander

Я открыла Midnight Commander и с помощью клавиш со стрелками и Enter перешла в каталог `~/work/arch-pc`. Затем нажала F7 и создала каталог `lab05`.

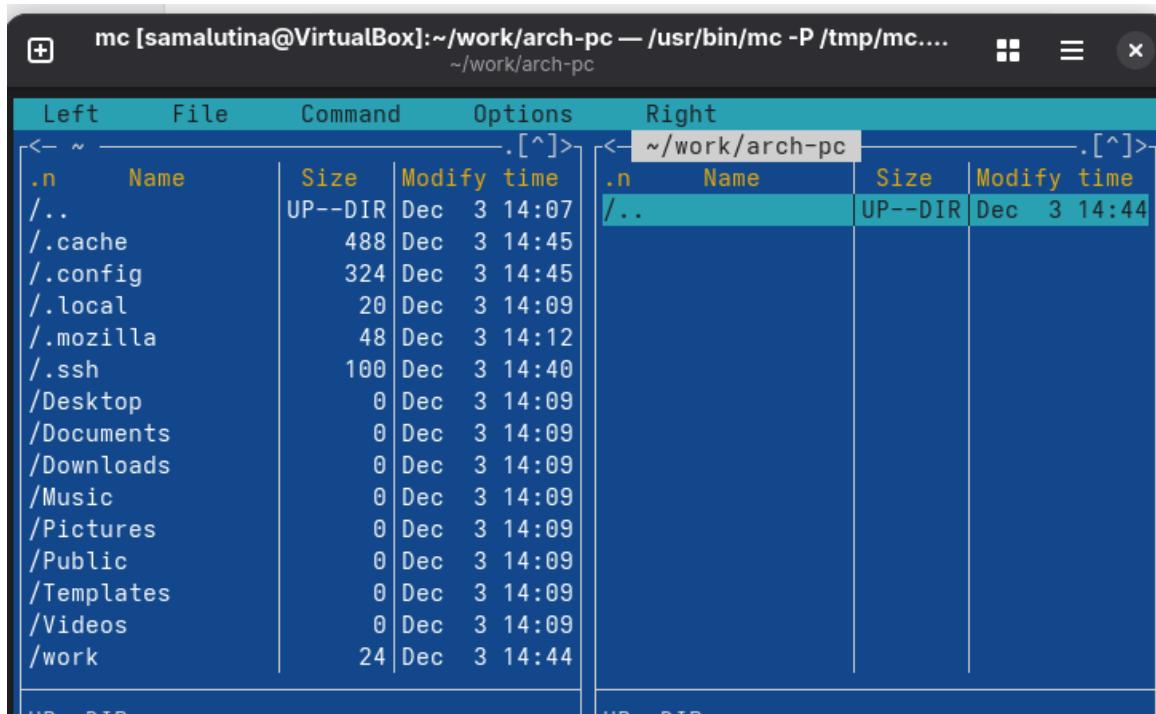


Рисунок 2.1: Запуск Midnight Commander

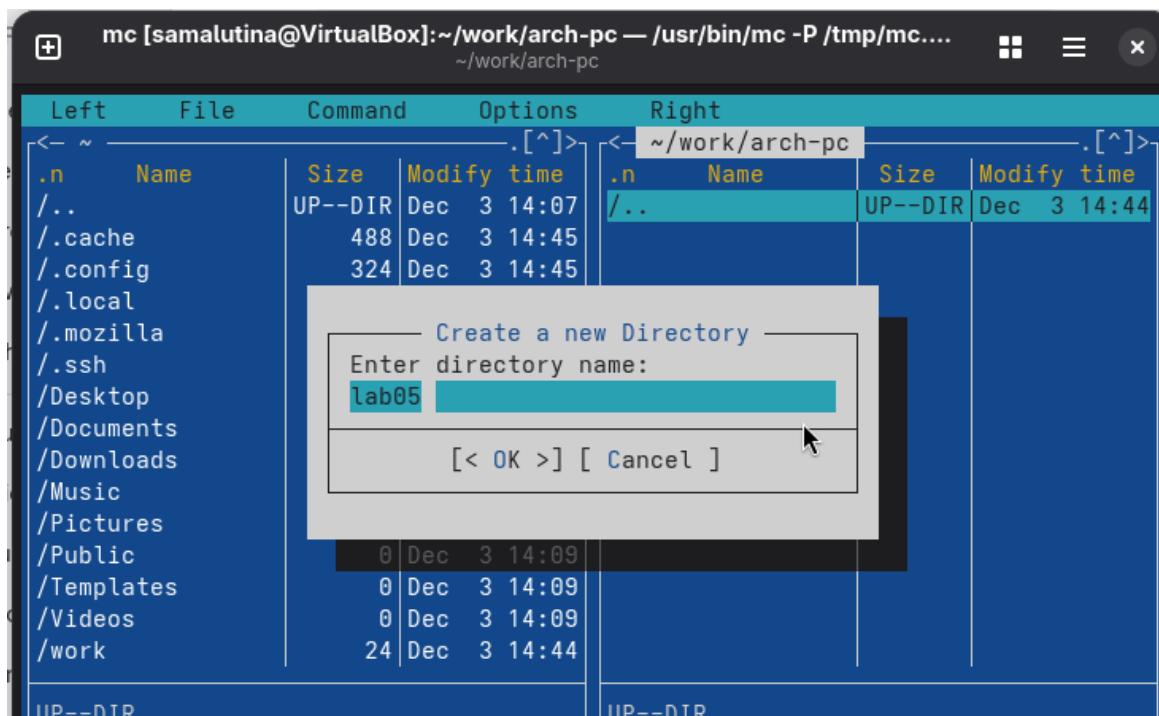


Рисунок 2.2: Создание каталога

При помощи команды touch я создала файл lab05-1.asm.

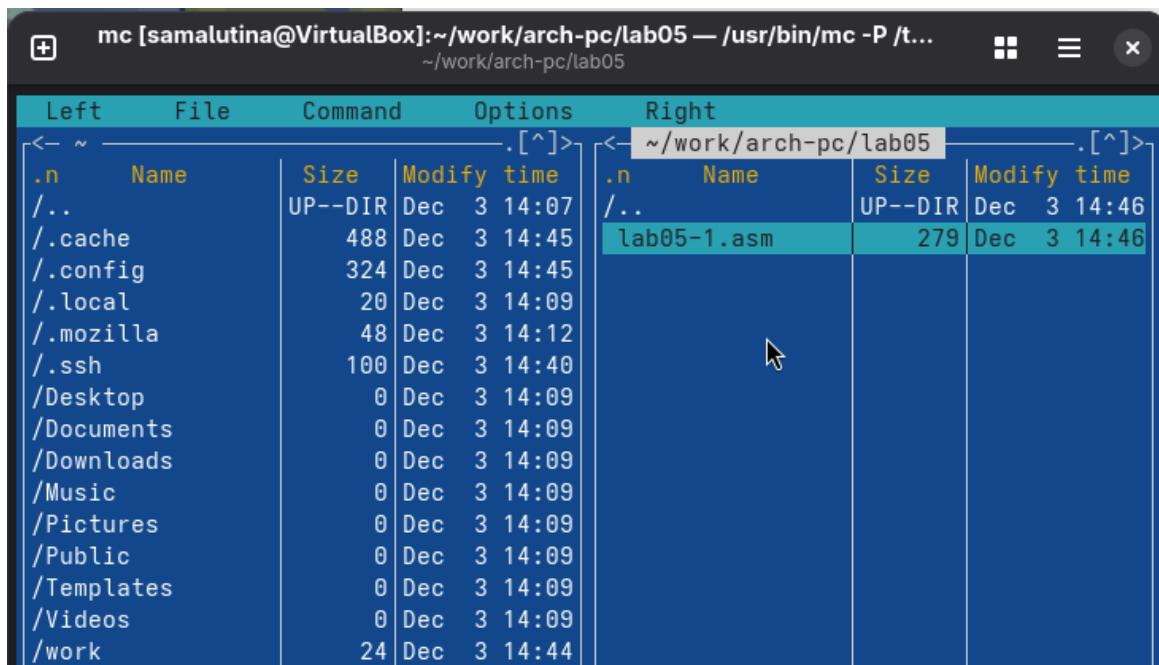


Рисунок 2.3: Создание файла lab05-1.asm

Открыла файл на редактирование, нажав F4, выбрала редактор mceditor и написала код программы из задания.



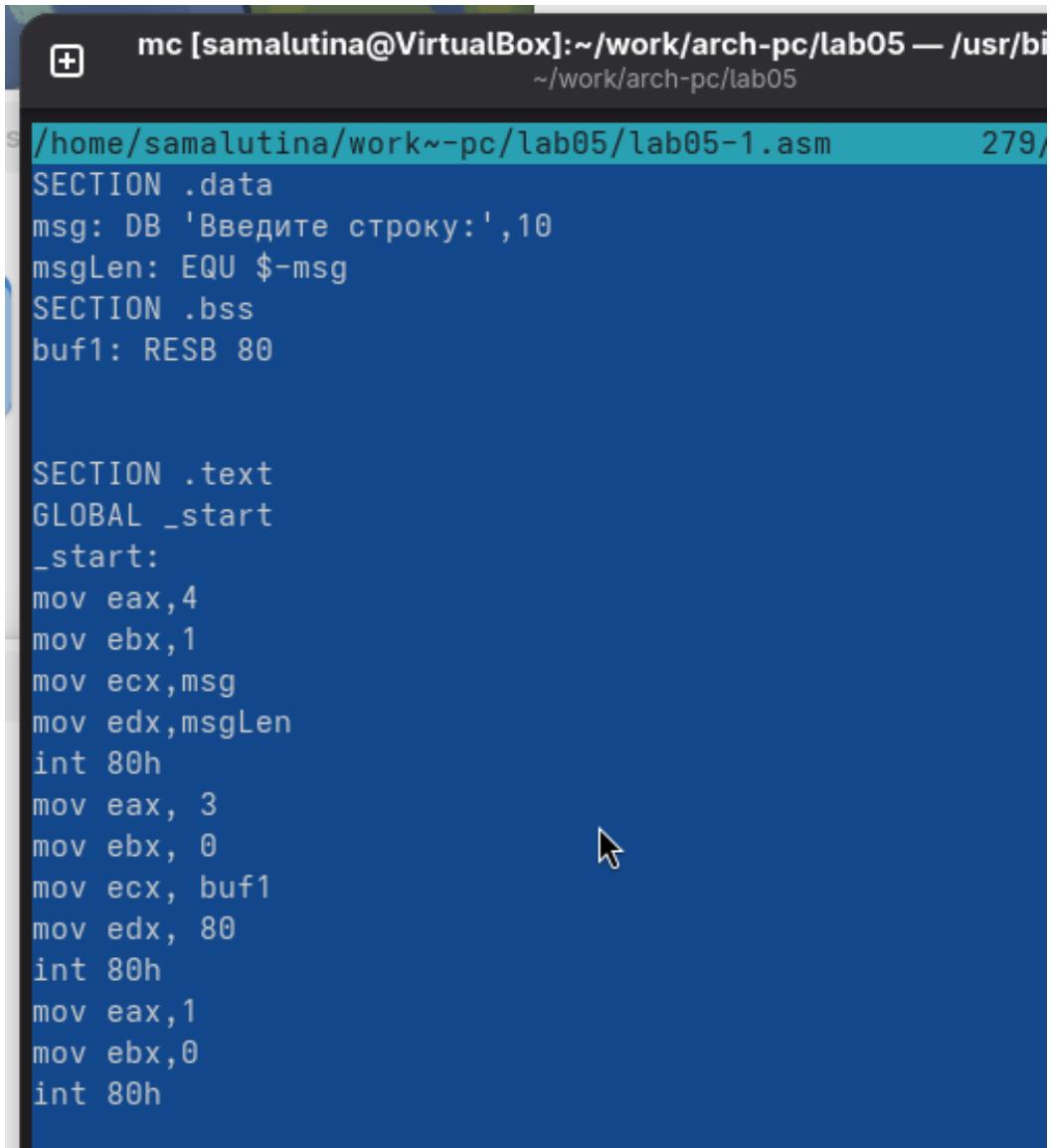
The screenshot shows the mc editor window with the following assembly code:

```
lab05-1.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,1
    mov ebx,0
    int 80h
```

Рисунок 2.4: Программа в файле lab05-1.asm

Открыла файл для просмотра, нажав F3, и убедилась, что он содержит написанный код.



The screenshot shows a terminal window titled 'mc [samalutina@VirtualBox]:~/work/arch-pc/lab05 — /usr/bi'. The current directory is ~/work/arch-pc/lab05. The file being viewed is lab05-1.asm. The assembly 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
```

Рисунок 2.5: Просмотр файла lab05-1.asm

Я транслировала файл программы в объектный файл, выполнила компоновку объектного файла и получила исполняемый файл программы, прове- рив ее работу.

```
samalutina@VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab05-1.asm
samalutina@VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-1.o -o lab05-1
samalutina@VirtualBox:~/work/arch-pc/lab05$ ./lab05-1
Введите строку:
privet
samalutina@VirtualBox:~/work/arch-pc/lab05$ █ █
```

Рисунок 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 вывод строки на экран или выход из программы, можно оформить в виде подпрограмм и сохранить в отдельные файлы. 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, <message>`);
- `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 будет записано (`mov eax, <buffer>`), в регистр `ebx` – длину вводимой строки (`mov ebx, <N>`);
- `iprint` – вывод на экран чисел в формате ASCII. Перед вызовом `iprint` в регистр `eax` необходимо записать выводимое число (`mov eax, <int>`);

- `iprintLF` – работает аналогично `iprint`, но при выводе на экран после числа добавляет 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, <int>`);
- `quit` – завершение программы.

Я скачала файл `in_out.asm` и разместила его в рабочем каталоге. Для копирования использовала клавишу F5, а для перемещения – клавишу F6.

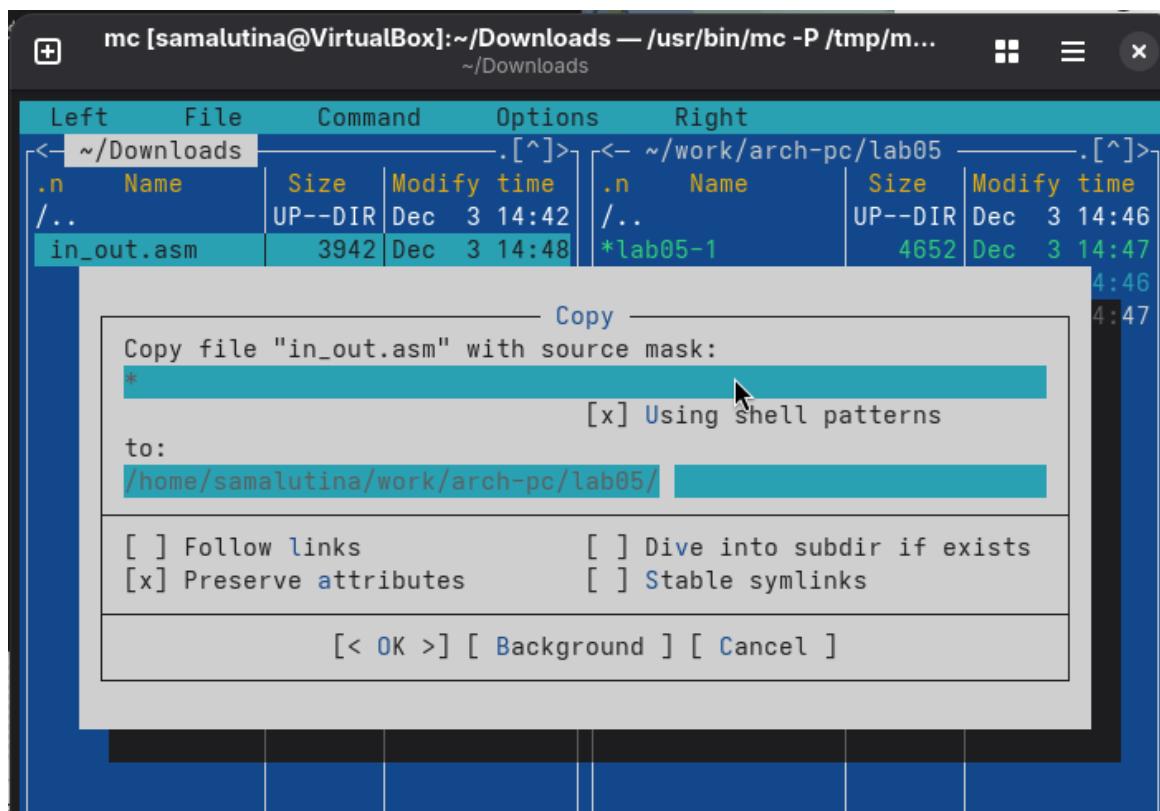


Рисунок 2.7: Копирование файла `in_out.asm`

Я скопировала `lab05-1.asm` в `lab05-2.asm`.

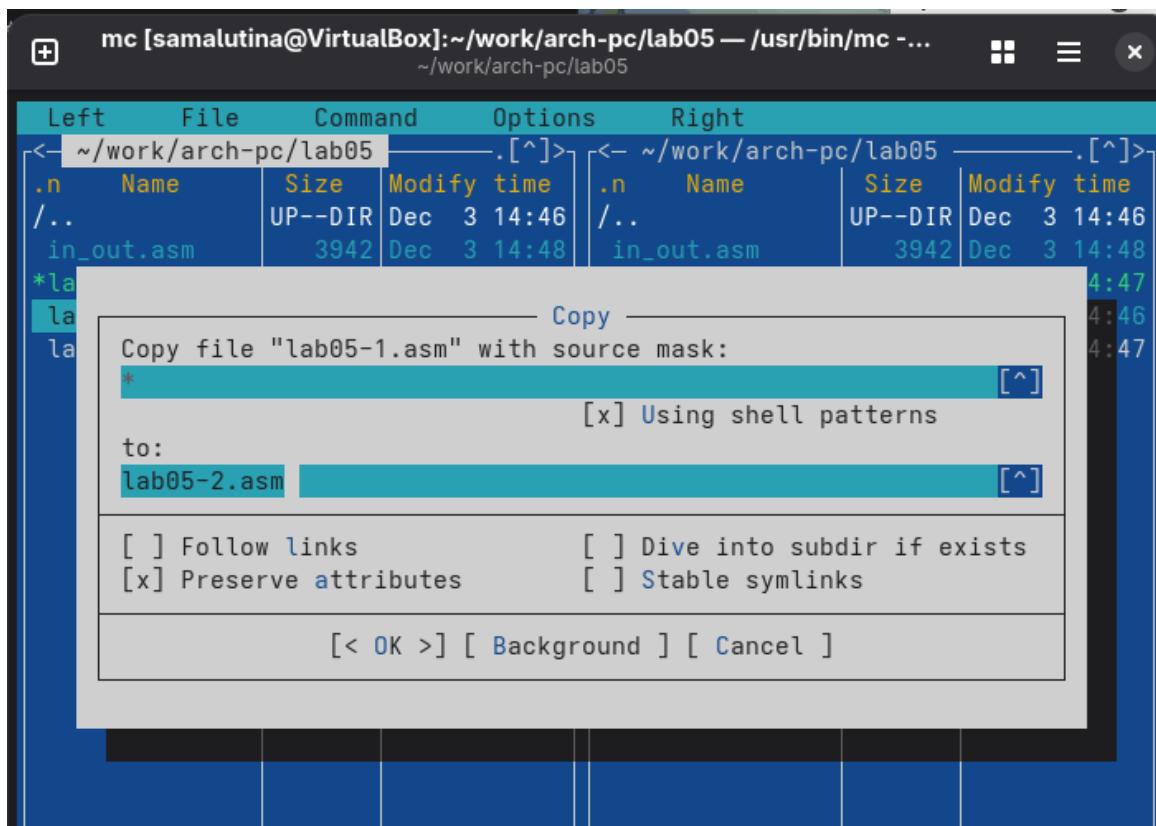
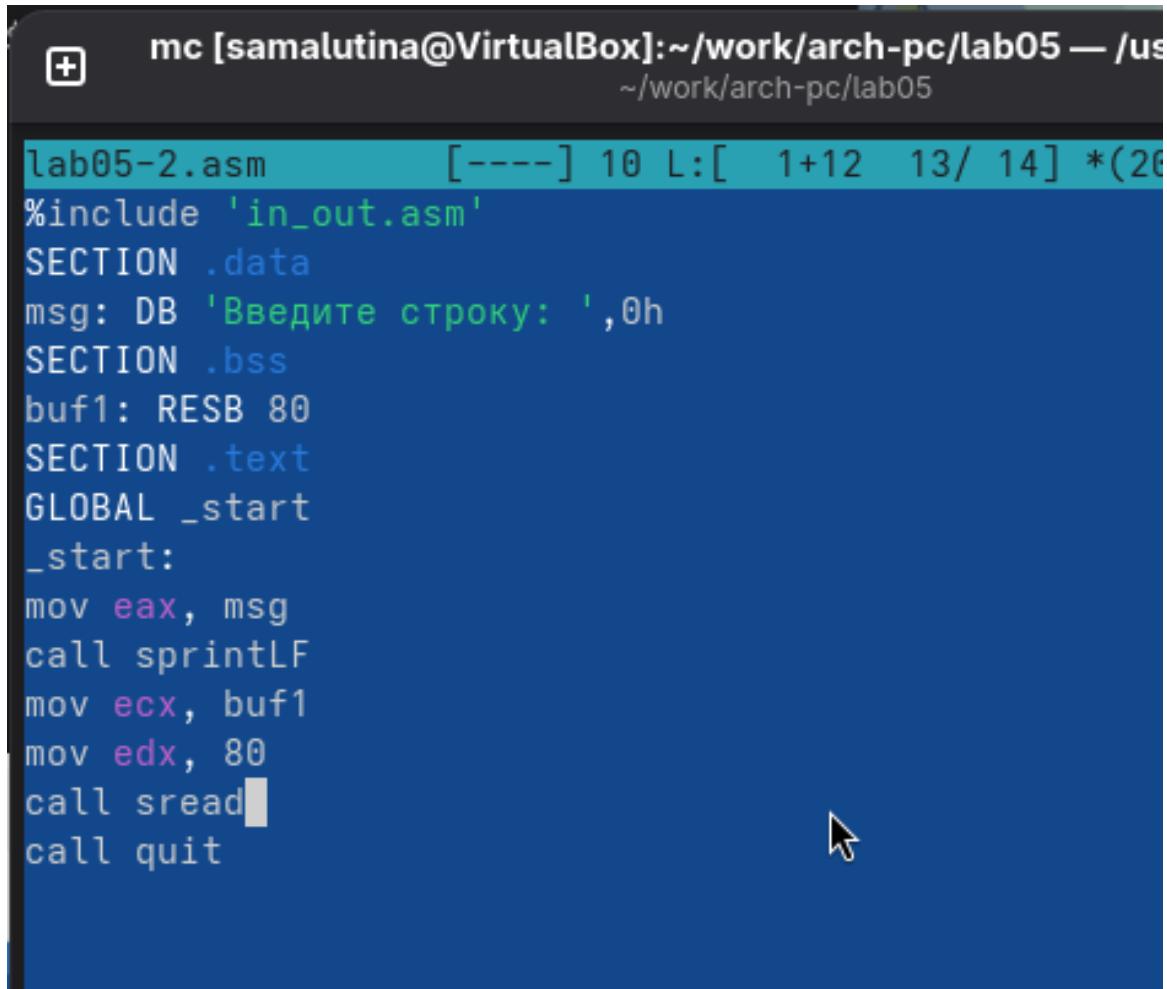


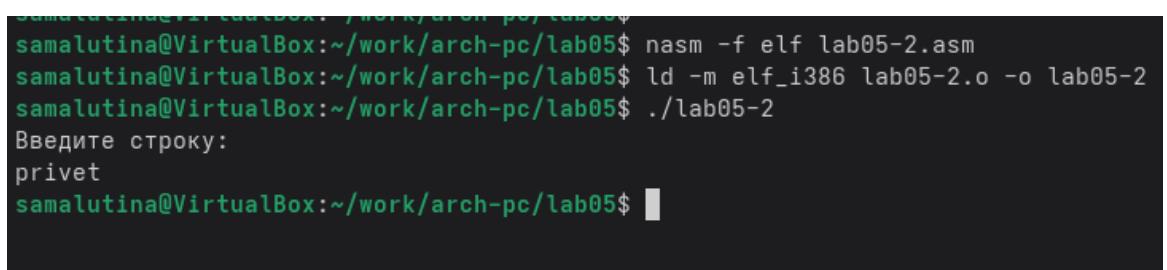
Рисунок 2.8: Копирование файла lab05-1.asm

Написала код программы lab05-2.asm, используя подпрограммы из внешнего файла in_out.asm. Скомпилировала программу и проверила запуск.



```
mc [samalutina@VirtualBox:~/work/arch-pc/lab05 — /us
~] 10 L:[ 1+12 13/ 14] *(26
lab05-2.asm      [----] 10 L:[ 1+12 13/ 14] *(26
%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
```

Рисунок 2.9: Программа в файле lab05-2.asm



```
samalutina@VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm
samalutina@VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2
samalutina@VirtualBox:~/work/arch-pc/lab05$ ./lab05-2
Введите строку:
privet
samalutina@VirtualBox:~/work/arch-pc/lab05$
```

Рисунок 2.10: Запуск программы lab05-2.asm

В файле lab05-2.asm я заменила подпрограмму `sprintLF` на `sprint`. Заново собрала исполняемый файл. Теперь после вывода строки она не завершается символом перехода на новую строку.

```
[+] mc [samalutina@VirtualBox]:~/work/arch-pc/lab05 — /u  
~/work/arch-pc/lab05  
  
lab05-2.asm      [----] 11 L:[ 1+ 9 10/ 14] *(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
```

Рисунок 2.11: Программа в файле lab05-2.asm

```
samatutina@VirtualBox:~/work/arch-pc/lab05$  
samatutina@VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab05-2.asm  
samatutina@VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-2.o -o lab05-2  
samatutina@VirtualBox:~/work/arch-pc/lab05$ ./lab05-2  
Введите строку: привет  
samatutina@VirtualBox:~/work/arch-pc/lab05$
```

Рисунок 2.12: Запуск программы lab05-2.asm

2.3 Задание для самостоятельной работы

Я скопировала программу lab05-1.asm и изменила код, так чтобы она работала по следующему алгоритму:

- вывести приглашение типа «Ведите строку:»;
- ввести строку с клавиатуры;
- вывести введённую строку на экран.

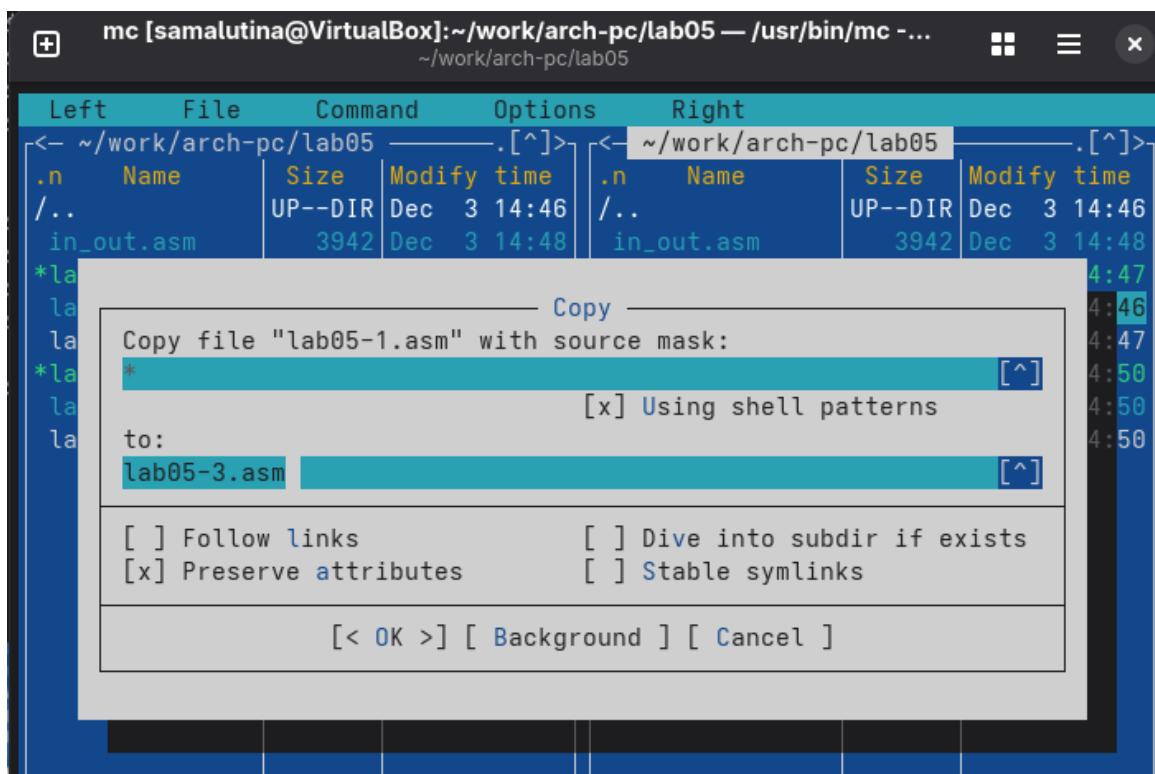


Рисунок 2.13: Копирование файла lab05-1.asm



The screenshot shows a terminal window titled 'mc [samalutina@VirtualBox]:' with the path '~ / work / arch - pc / lab05'. The file 'lab05-3.asm' is open. The code is as follows:

```
lab05-3.asm      [---]  0 L:[ 2+ 7  9/
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
```

Рисунок 2.14: Программа в файле lab05-3.asm



```
samalutina@VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab05-3.asm
samalutina@VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-3.o -o lab05-3
samalutina@VirtualBox:~/work/arch-pc/lab05$ ./lab05-3
Введите строку:
privet
privet
samalutina@VirtualBox:~/work/arch-pc/lab05$
```

Рисунок 2.15: Запуск программы lab05-3.asm

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

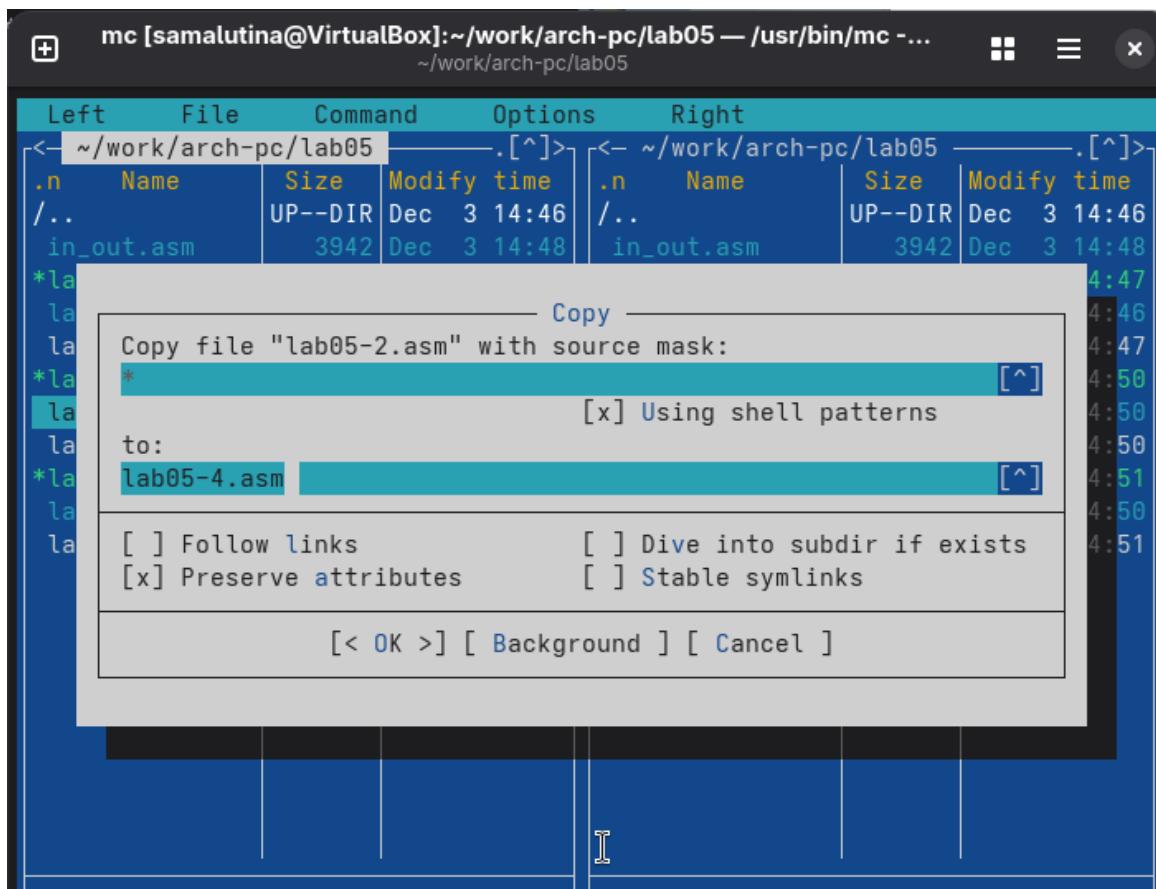


Рисунок 2.16: Копирование файла lab05-2.asm

```
+ mc [samalutina@VirtualBox]:~/work/arch-pc/lab05 — /usr/bin/n
~ /work/arch-pc/lab05

lab05-4.asm      [----]  9 L:[ 1+15 16/ 16] *(238 / 2
%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
```

Рисунок 2.17: Программа в файле lab05-4.asm

```
samalutina@VirtualBox:~/work/arch-pc/lab05$ 
samalutina@VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab05-4.asm
samalutina@VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 lab05-4.o -o lab05-4
samalutina@VirtualBox:~/work/arch-pc/lab05$ ./lab05-4
Введите строку: привет
привет
samalutina@VirtualBox:~/work/arch-pc/lab05$
```

Рисунок 2.18: Запуск программы lab05-4.asm

3 Выводы

Я научилась писать базовые ассемблерные программы и освоила ассемблерные инструкции `mov` и `int`.