

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

Студент: Белакова Полина Вячеславовна

Ст.билет: 1032252589

Группа: НКАбд-01-25

МОСКВА

2025 г

Цель работы

Приобретение практических навыков работы в Midnight Commander.
Освоение инструкций языка ассемблера mov и int.

Выполнение работы

В командной строке ввожу команду mc, чтобы открыть Midnight Commander (рисунок 1).

```
pvbelakova2@localhost-live:~$ mc
pvbelakova2@localhost-live:~/arch-pc/labs$ touch lab5-1.asm
pvbelakova2@localhost-live:~/arch-pc/labs$
```

Рисунок 1 - команда mc.

Используя клавиши \uparrow , \downarrow и Enter перехожу в каталог $\sim/\text{arch-pc}$ созданный при выполнении лабораторной работы №4.

С помощью клавиши F7 создаю папку lab05 и перехожу в созданный каталог (рисунок 2).

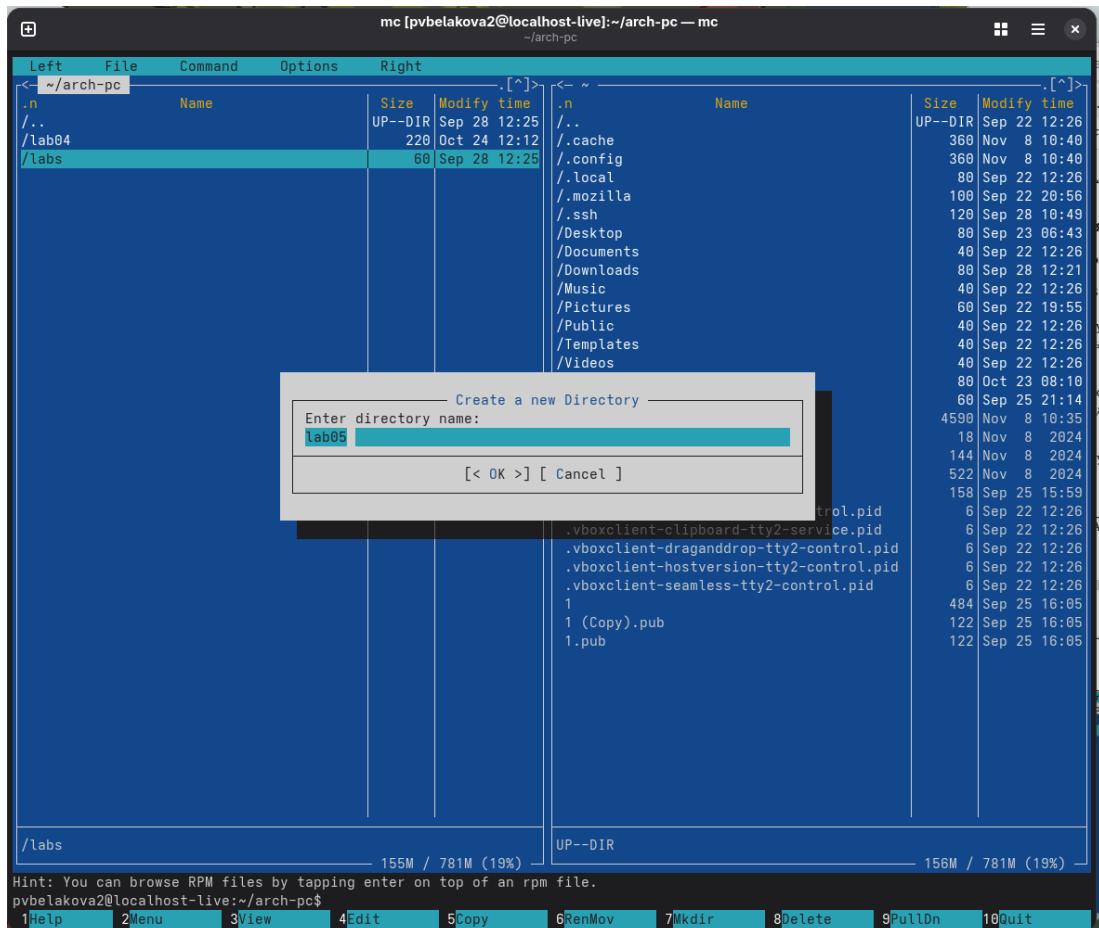


Рисунок 2 - Создание каталога lab7.

Создаю файл lab5-1.asm с помощью команды touch (рисунок 3).

The screenshot shows the terminal window of the Midnight Commander (mc) file manager. The left pane displays the directory structure under `~/arch-pc/lab05`, and the right pane shows the contents of the current directory. A file named `lab5-1.asm` is highlighted in the left pane. In the bottom command line, the command `touch lab5-1.asm` is being typed, with the cursor positioned at the end of the command. The status bar at the bottom indicates the memory usage: `155M / 781M (19%)`.

Рисунок 3 - создание файла lab5-1.asm.

С помощью клавиши F4 открываю файл lab5-1.asm для редактирования во встроенным редакторе. Ввожу код программы (рисунок 4), сохраняю изменения и закрываю файл (рисунок5).

The screenshot shows a terminal window titled 'mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — mc'. The window displays assembly code for a program named 'lab5-1.asm'. The code is written in AT&M assembly language and includes comments explaining the purpose of each section and instruction. The assembly code is as follows:

```
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
52 Демидова А. В.
Архитектура ЭВМ
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write'
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов 'read' -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
mov eax,3 ; Системный вызов для чтения (sys_read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov ecx,buf1 ; Адрес буфера под вводимую строку
mov edx,80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

The terminal window has a menu bar at the top with items like 'File', 'Edit', 'View', 'Search', 'Run', 'Help', and 'Exit'. At the bottom, there is a toolbar with numbered buttons from 1 to 10 corresponding to the menu items.

Рисунок 4 - ввод кода программы.

С помощью клавиши F3 открываю файл lab5-1.asm для просмотра, чтобы убедиться, что изменения сохранились (рисунок 5).

The screenshot shows the Mars Cross Assembler (mc) interface. The title bar reads "mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — mc". The main window displays the assembly code for "lab5-1.asm". The code includes sections for data and bss, defines a message string, and implements a simple I/O loop using system calls for write and read. The assembly code is annotated with comments explaining its purpose. The bottom of the window shows a toolbar with numbered buttons from 1 to 10, corresponding to various functions like Help, Unwrap, Quit, Hex, Goto, Search, Raw, Format, and Exit.

```
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
52 Демидова А. В.
Архитектура ЭВМ
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов `write` -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов `read` -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
mov eax,3 ; Системный вызов для чтения (sys_read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov ecx,buf1 ; Адрес буфера под вводимую строку
mov edx, 80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов `exit` -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рисунок 5 - содержимое файла lab5-1.asm.

Создаю объектный файл для lab5-1.asm. Затем выполняю компоновку объектного файла и запускаю получившийся исполняемый файл (рисунок 6).

```
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-1.asm
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
pvbelakova2@localhost-live:~/arch-pc/lab05$ ./lab5-1
Введите строку:
Belakova Polina
pvbelakova2@localhost-live:~/arch-pc/lab05$
```

Рисунок 6 - создание объектного файла, исполняемого файла, запуск программы и выполнение программы (вывод текста и ввод данных).

Подключение внешнего файла in_out.asm

Скачиваю файл in_out.asm со страницы курса в ТУИС. В одной из панелей mc открываю каталог с файлом lab5-1.asm. В другой панели каталог со скаченным файлом in_out.asm (рисунок 7).

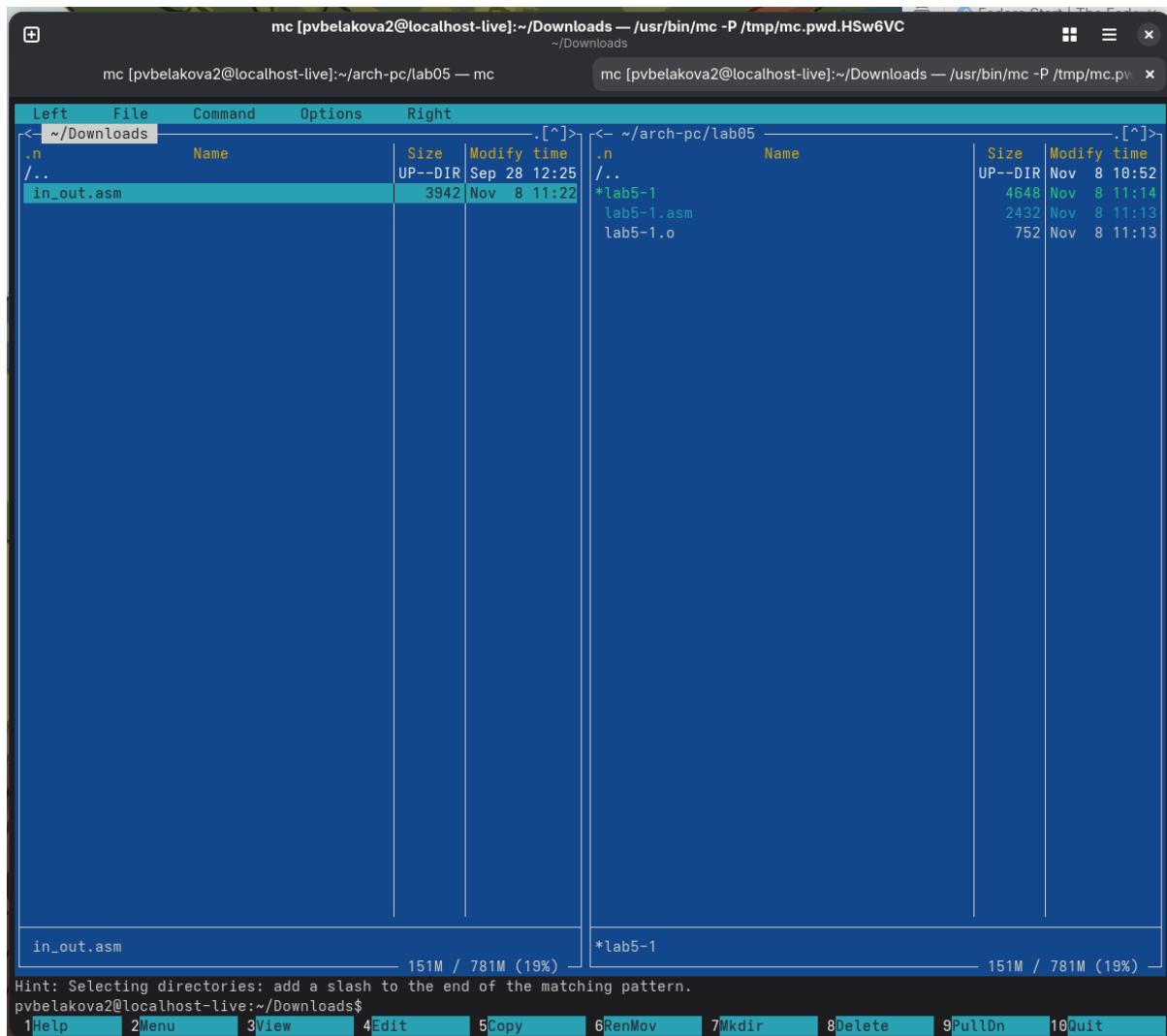


Рисунок 7 - окно midnight commander с двумя открытыми панелями.

Копирую файл in_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5 (рисунок 8).

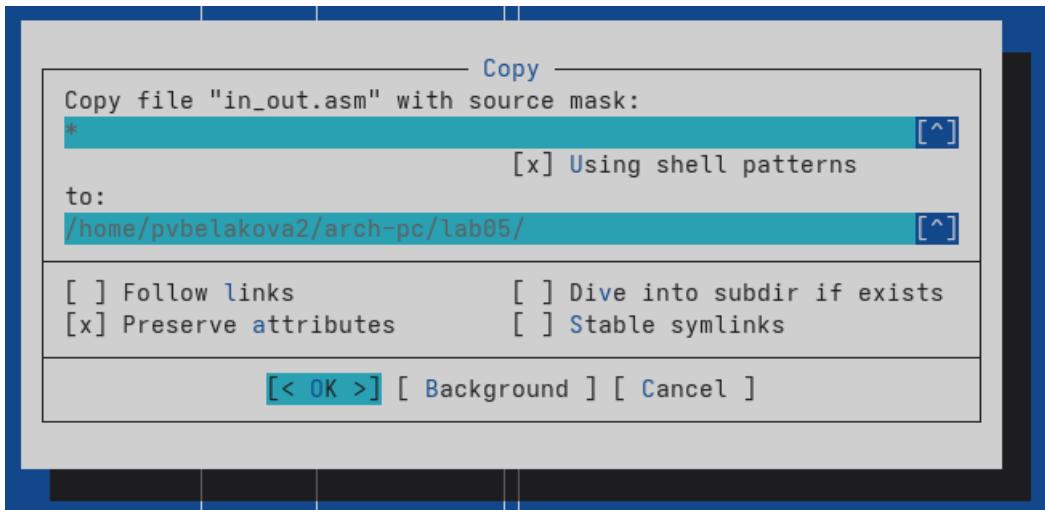


Рисунок 8 - копирование файла

С помощью клавиши F6 создаю копию файла lab5-1.asm с именем lab5-2.asm (рисунок 9).

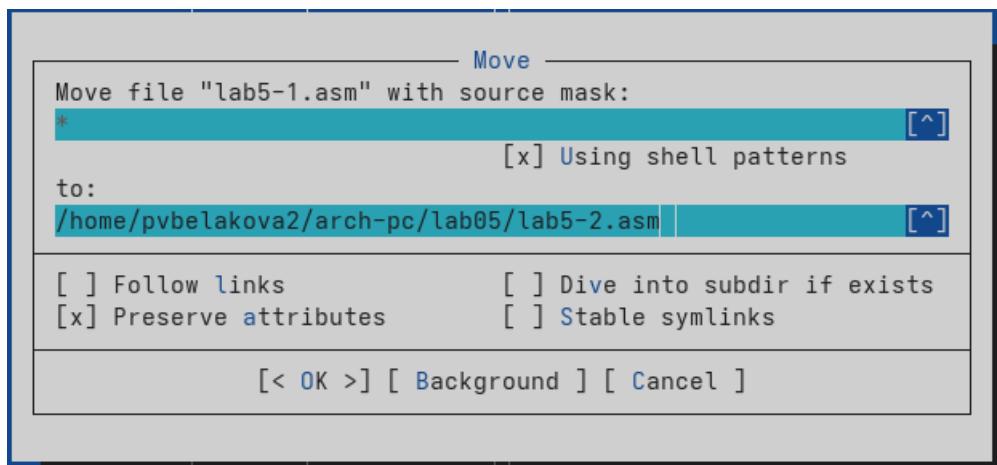


Рисунок 9 - Создание копии файла.

Изменяю текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm (рисунок 10).

```
[+] mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — /usr/bin/mc -P /tmp/mc.pwd.HSw6VC  
mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — mc  
lab5-2.asm      [-M--] 54 L:[ 1+14 15/ 17] *(1076/1224b) 0010 0x00A  
----  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
----  
%include 'in_out.asm' ; подключение внешнего файла  
SECTION .data ; Секция инициализированных данных  
msg: DB 'Введите строку: ',0h ; сообщение  
SECTION .bss ; Секция не инициализированных данных  
buf1: RESB 80 ; Буфер размером 80 байт  
SECTION .text ; Код программы  
GLOBAL _start ; Начало программы  
_start: ; Точка входа в программу  
    mov eax, msg ; запись адреса выводимого сообщения в `EAX`  
    call sprintLF ; вызов подпрограммы печати сообщения  
    mov ecx, buf1 ; запись адреса переменной в `EAX`  
    mov edx, 80 ; запись длины вводимого сообщения в `EBX`  
    call sread ; вызов подпрограммы ввода сообщения  
    call quit ; вызов подпрограммы завершения
```

Рисунок 10 - текст программы lab5-2.asm.

Создаю объектный и исполняемый файл и проверяю его работу (рисунок 11).

```
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-1.asm  
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-2.asm  
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o~  
ld: cannot find lab5-2.o~: No such file or directory  
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o  
pvbelakova2@localhost-live:~/arch-pc/lab05$ ./lab5-2  
Введите строку:  
Belakova Polina  
pvbelakova2@localhost-live:~/arch-pc/lab05$
```

Рисунок 11 - создание объектного и исполняемого файла, запуск программы.

В файле lab5-2.asm заменяю подпрограмму sprintLF на sprint (рисунок 12), создаю объектный и исполняемый файлы и проверяю его работу (рисунок 13).

```
mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — mc
mc [pvbelakova2@localhost-live]:~/arch-pc/lab05 — mc
lab5-2.asm      [-M--] 41 L:[ 1+16 17/ 17 ] *(1222/1222b) <EOF>
-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
    mov eax, msg ; запись адреса выводимого сообщения в `EAX`
    call sprint ; вызов подпрограммы печати сообщения
    mov ecx, buf1 ; запись адреса переменной в `ECX`
    mov edx, 80 ; запись длины вводимого сообщения в `EBX`
    call sread ; вызов подпрограммы ввода сообщения
    call quit ; вызов подпрограммы завершения
```

Рисунок 12 - Замена подпрограммы `sprintLF` на `sprint` в коде `lab5-2.asm`.

```
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-2.asm -o lab5-2.2.o
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-2_2 lab5-2.2.o
ld: cannot find lab5-2_2.o: No such file or directory
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-2_2 lab5-2.2.o
pvbelakova2@localhost-live:~/arch-pc/lab05$ ./lab5-2_2
Введите строку: Belakova Polina
pvbelakova2@localhost-live:~/arch-pc/lab05$
```

Рисунок 13 - Создание объектного и исполняемого файла и запуск программы.

При использовании подпрограммы `sprintLF` новые данные вводятся с новой строки, при использовании подпрограммы `sprint` данные вводятся с той же строки.

Задание для самостоятельной работы

- Создаю копию файла `lab5-1.asm` с названием `lab5-1_task.asm` (рисунок 14). Редактирую программу, так чтобы после ввода данных осуществлялся их вывод (рисунок 15).

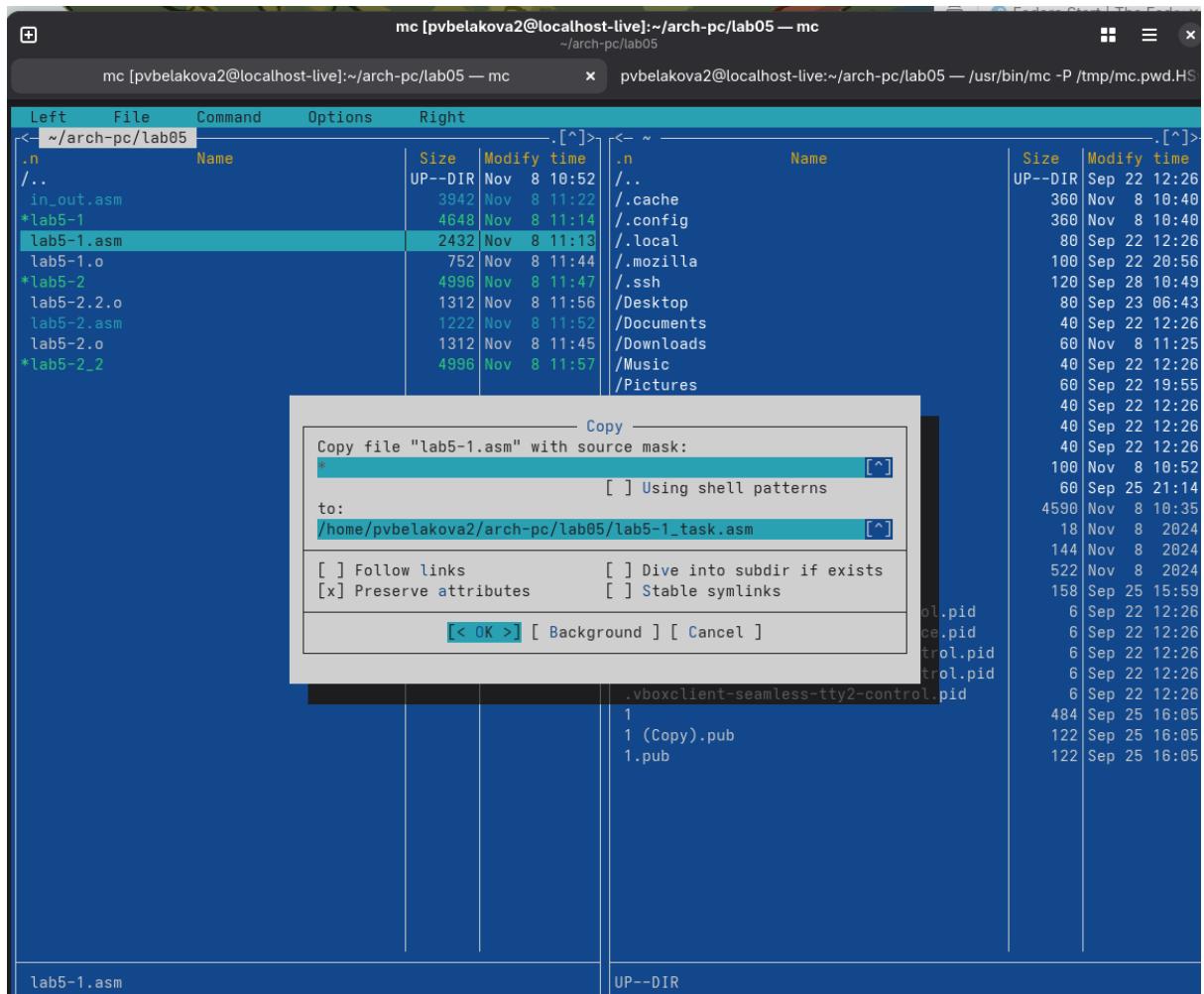


Рисунок 14 - копирование файла.

```
mc [pbelakova2@localhost-live]:~/arch-pc/lab05 — mc x pbelakova2@localhost-live: ~/arch-pc/lab05 — mc

lab5_1_task.asm [-M--] 12 L:[ 1+35 36/ 43] *(2464/2897b) 0032 0x020
;-----;
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----;
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов `write` -
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов `read` -
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
mov eax,3 ; Системный вызов для чтения (sys_read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov ecx,buf1 ; Адрес буфера под вводимую строку
mov edx,80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- Системный вызов `write` -
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,buf1 ; Адрес строки 'msg' в 'ecx'
int 80h ; Вызов ядра
;----- Системный вызов `exit` -
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рисунок 15 - Код программы.

2. Создаю объектный и исполняемый файл и проверяю его работу.

Программа выводит данные, введенные ранее (рисунок 16).

```

Belakova Polina
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-1_task.asm
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-1_task lab5-1_task.o
pvbelakova2@localhost-live:~/arch-pc/lab05$ ./lab5-1_task
Введите строку:
Belakova Polina
Belakova Polina
pvbelakova2@localhost-live:~/arch-pc/lab05$

```

Рисунок 16 - создание объектного и исполняемого файлов, запуск программы.

3. Создаю копию файла lab5-2.asm и называю ее lab5-2_task.asm (рисунок 17).

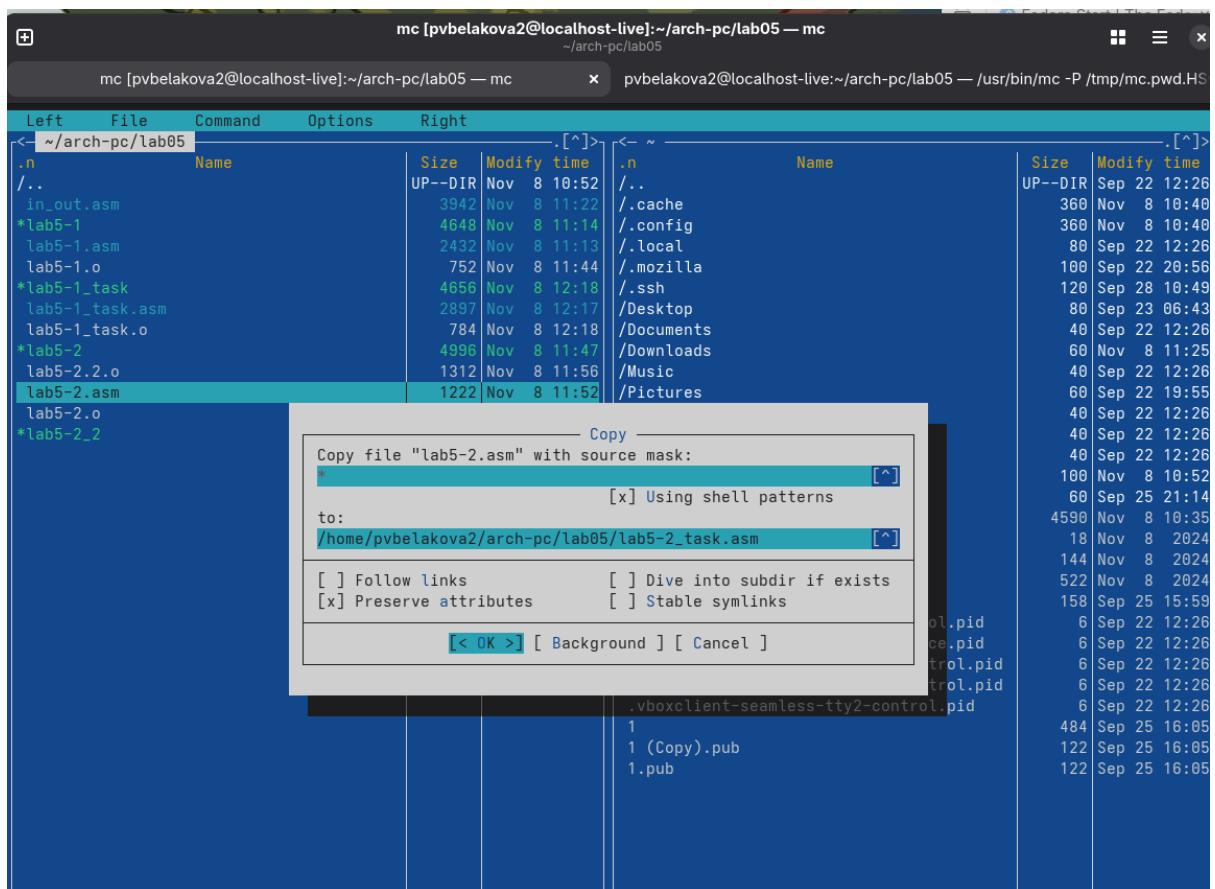
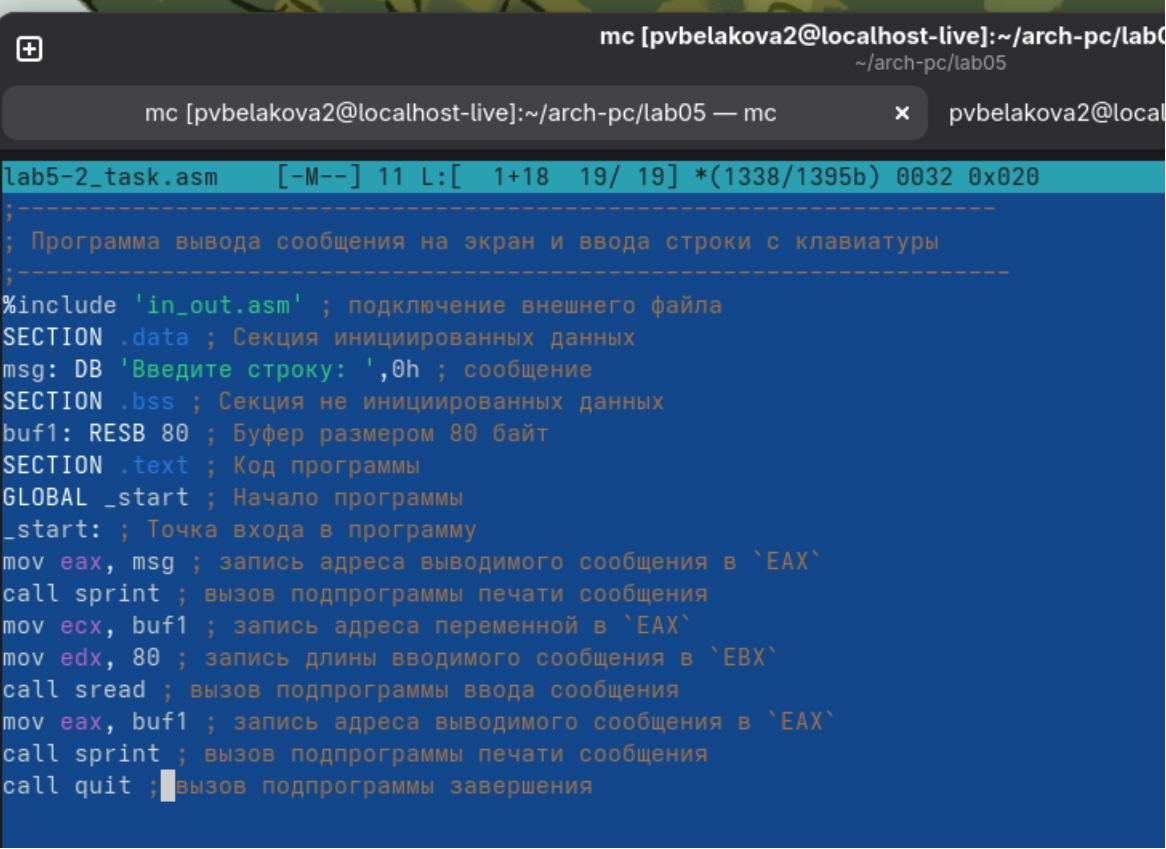


Рисунок 17 - копирование файла.

Исправляю текст программы с использование подпрограмм из внешнего файла `in_out.asm`, так чтобы она в конце выводила данные, которые были введены ранее (рисунок 18).

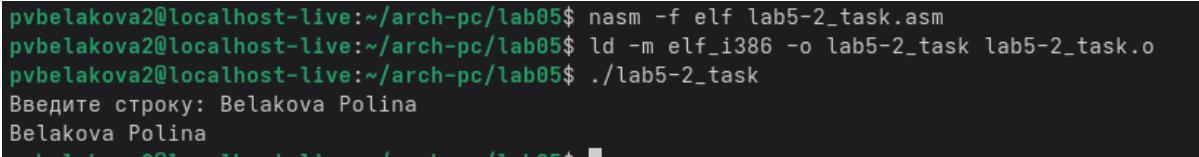


The screenshot shows a terminal window titled 'mc [pvbelakova2@localhost-live]:~/arch-pc/lab05'. The command 'mc' was run to open the file 'lab5-2_task.asm'. The code in the editor is as follows:

```
lab5-2_task.asm [-M--] 11 L:[ 1+18 19/ 19] *(1338/1395b) 0032 0x020
-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
    mov eax, msg ; запись адреса выводимого сообщения в `EAX`
    call sprint ; вызов подпрограммы печати сообщения
    mov ecx, buf1 ; запись адреса переменной в `EAX`
    mov edx, 80 ; запись длины вводимого сообщения в `EBX`
    call sread ; вызов подпрограммы ввода сообщения
    mov eax, buf1 ; запись адреса выводимого сообщения в `EAX`
    call sprint ; вызов подпрограммы печати сообщения
    call quit ; вызов подпрограммы завершения
```

Рисунок 19 -измененный код программы в файле lab5-2_task.asm.

4. Создаю объектный и исполняемый файл и проверяю его работу (рисунок 20). Программа выводит введенные ранее данные.



```
pvbelakova2@localhost-live:~/arch-pc/lab05$ nasm -f elf lab5-2_task.asm
pvbelakova2@localhost-live:~/arch-pc/lab05$ ld -m elf_i386 -o lab5-2_task lab5-2_task.o
pvbelakova2@localhost-live:~/arch-pc/lab05$ ./lab5-2_task
Введите строку: Belakova Polina
Belakova Polina
pvbelakova2@localhost-live:~/arch-pc/lab05$
```

Рисунок 20 - создание объектного и исполняемого файлов, запуск и программы

Выводы

В ходе лабораторной работы были приобретены практические навыки работы в Midnight Commander. Освоены инструкции mov и int языка ассемблер.