

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Исупова Кристина Павловна

Оглавление

1 Цель работы	3
2 Задание	4
3 Теоретическое введение	5
4 Выполнение лабораторной работы	6
4.1 Основы работы с Midnight Commander	6
4.2 Работа в NASM	9
4.3 Подключение внешнего файла	10
4.4 Задание для самостоятельной работы	13
5 Выводы	20
Список литературы	21

1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в Midnight Commander, освоение инструкций языка ассемблера `mov` и `int`.

2 Задание

1. Основы работы с `ms`
2. Структура программы на языке ассемблера `NASM`
3. Подключение внешнего файла
4. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Midnight Commander (или просто `mc`) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. `mc` является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (`SECTION .text`), секция инициированных (известных во время компиляции) данных (`SECTION .data`) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (`SECTION .bss`). Для объявления инициированных данных в секции `.data` используются директивы `DB`, `DW`, `DD`, `DQ` и `DT`, которые резервируют память и указывают, какие значения должны храниться в этой памяти: - `DB` (`define byte`) — определяет переменную размером в 1 байт; - `DW` (`define word`) — определяет переменную размером в 2 байта (слово); - `DD` (`define double word`) — определяет переменную размером в 4 байта (двойное слово); - `DQ` (`define quad word`) — определяет переменную размером в 8 байт (учетверённое слово); - `DT` (`define ten bytes`) — определяет переменную размером в 10 байт. Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву `DB` в связи с особенностями хранения данных в оперативной памяти. Инструкция языка ассемблера `mov` предназначена для дублирования данных источника в приёмнике.

`mov dst,src`

Здесь операнд `dst` — приёмник, а `src` — источник. В качестве операнда могут выступать регистры (`register`), ячейки памяти (`memory`) и непосредственные значения (`const`). Инструкция языка ассемблера `int` предназначена для вызова прерывания с указанным номером.

`int n`

Здесь `n` — номер прерывания, принадлежащий диапазону 0–255. При программировании в Linux с использованием вызовов ядра `sys_calls` `n=80h` (принято задавать в шестнадцатеричной системе счисления).

4 Выполнение лабораторной работы

4.1 Основы работы с Midnight Commander

Введя соответствующую команду в терминале (Рис. **Error! Reference source not found.**) я открываю Midnight Commander (Рис. 2)

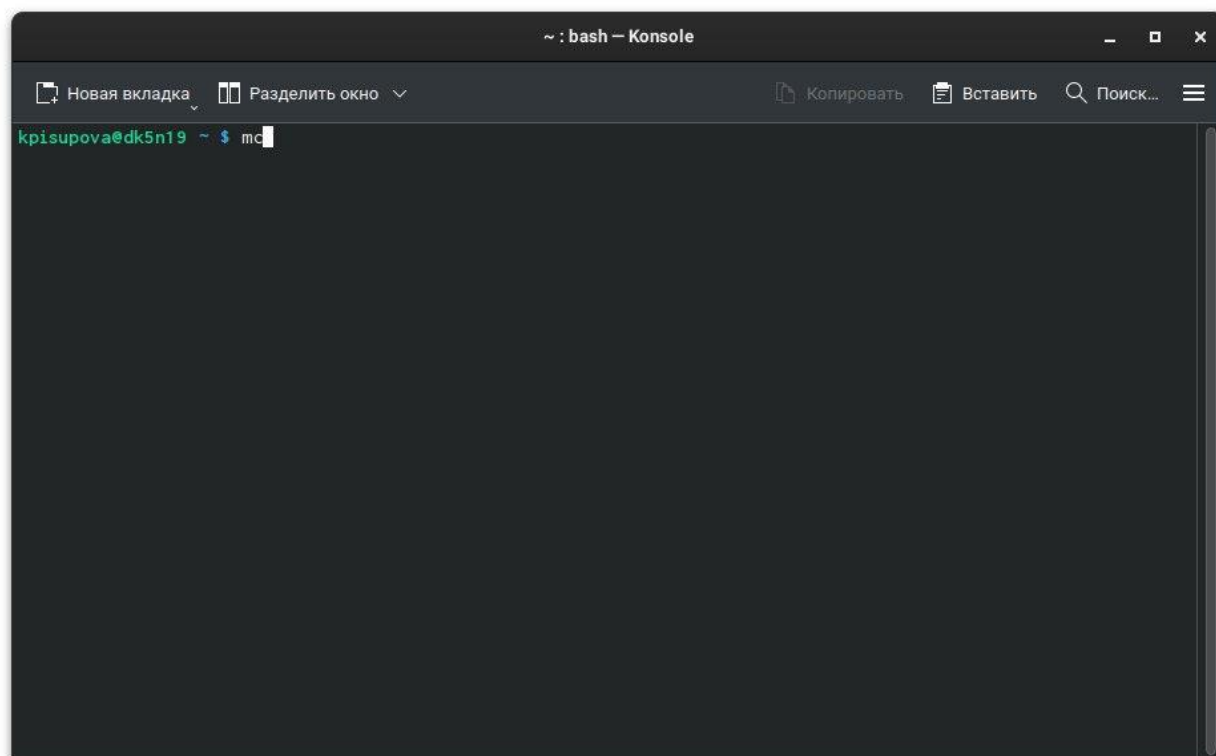


Рис. 1 Открытие Midnight Commander.

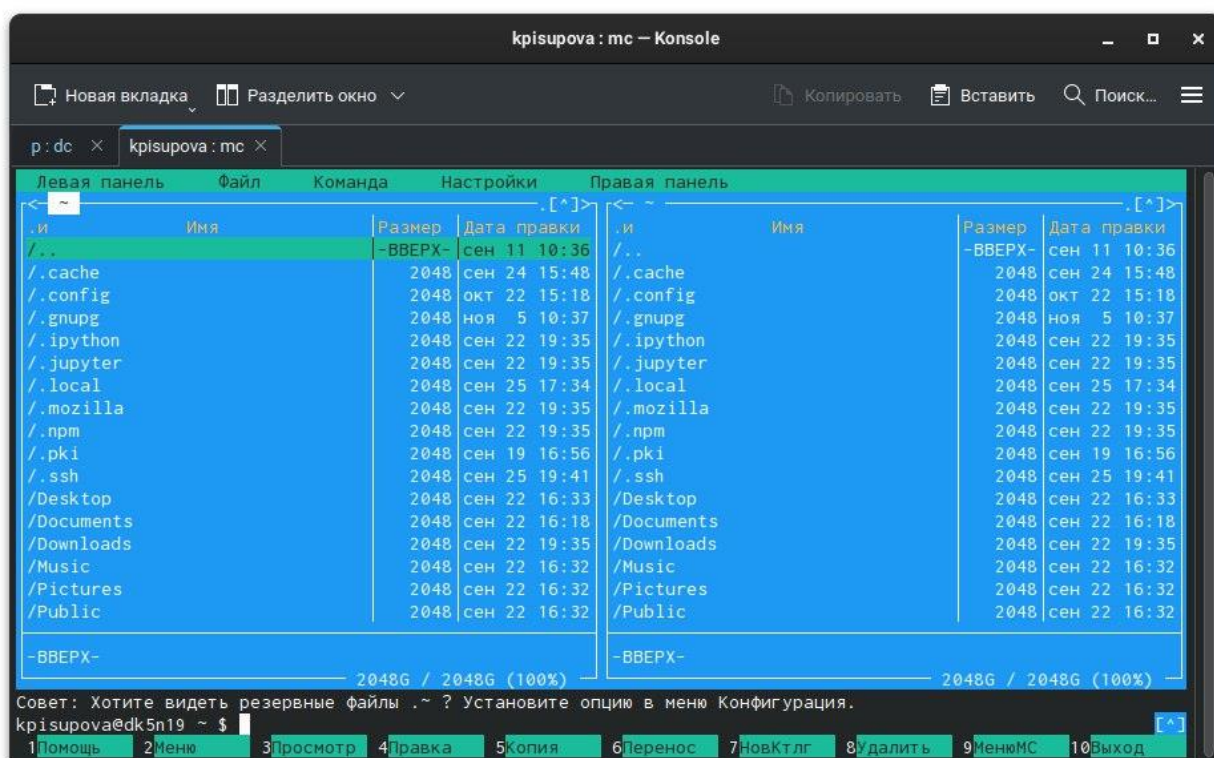


Рис. 2 Интерфейс Midnight Commander

Перехожу в созданный каталог в предыдущей лабораторной работе (Рис. 3).

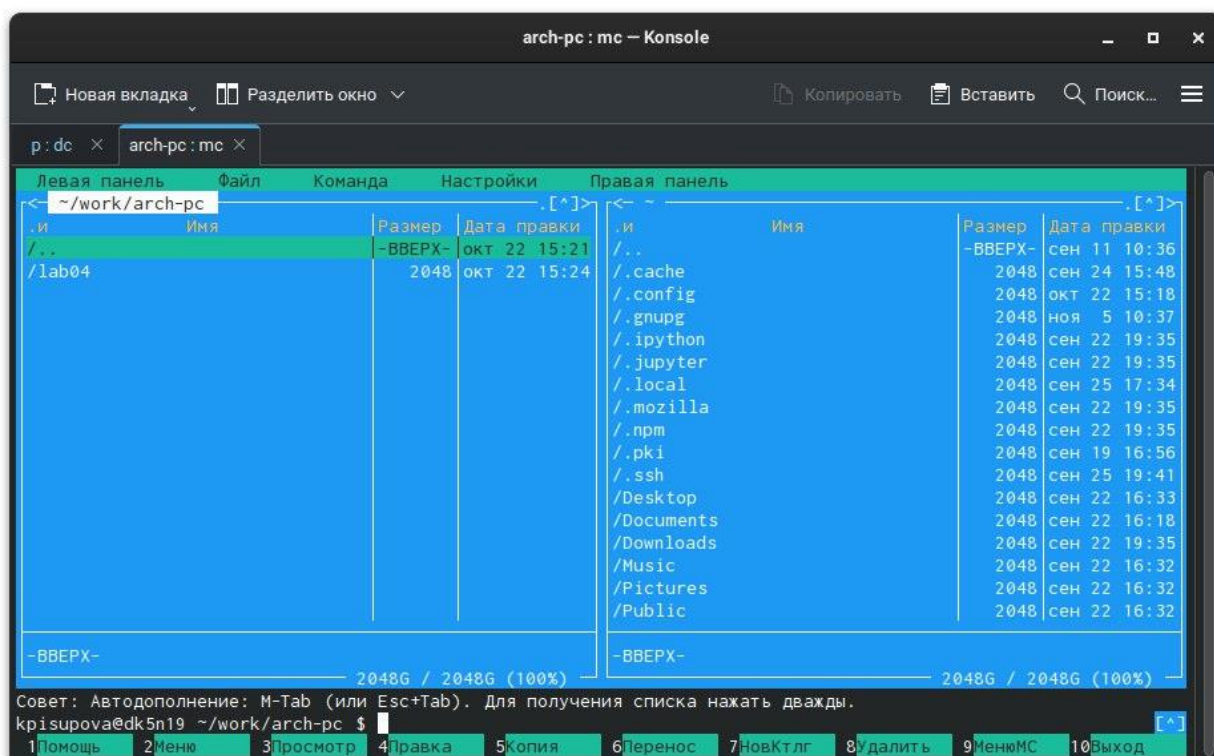


Рис. 3 Открытый каталог arch-pc.

С помощью функциональной клавиши, я создаю подкаталог lab05, в котором буду работать (Рис. 4).

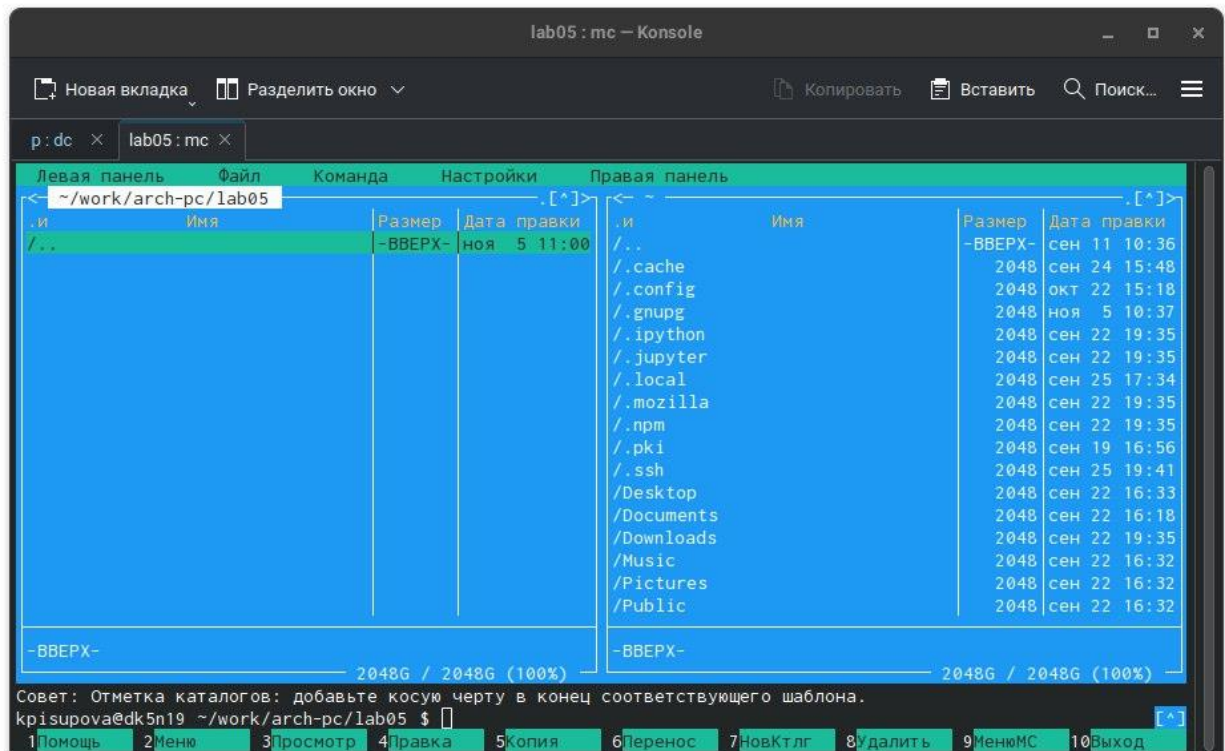


Рис. 4 Создание рабочего подкаталога

В строке ввода вводжу команду touch и создаю файл (Рис. 5).

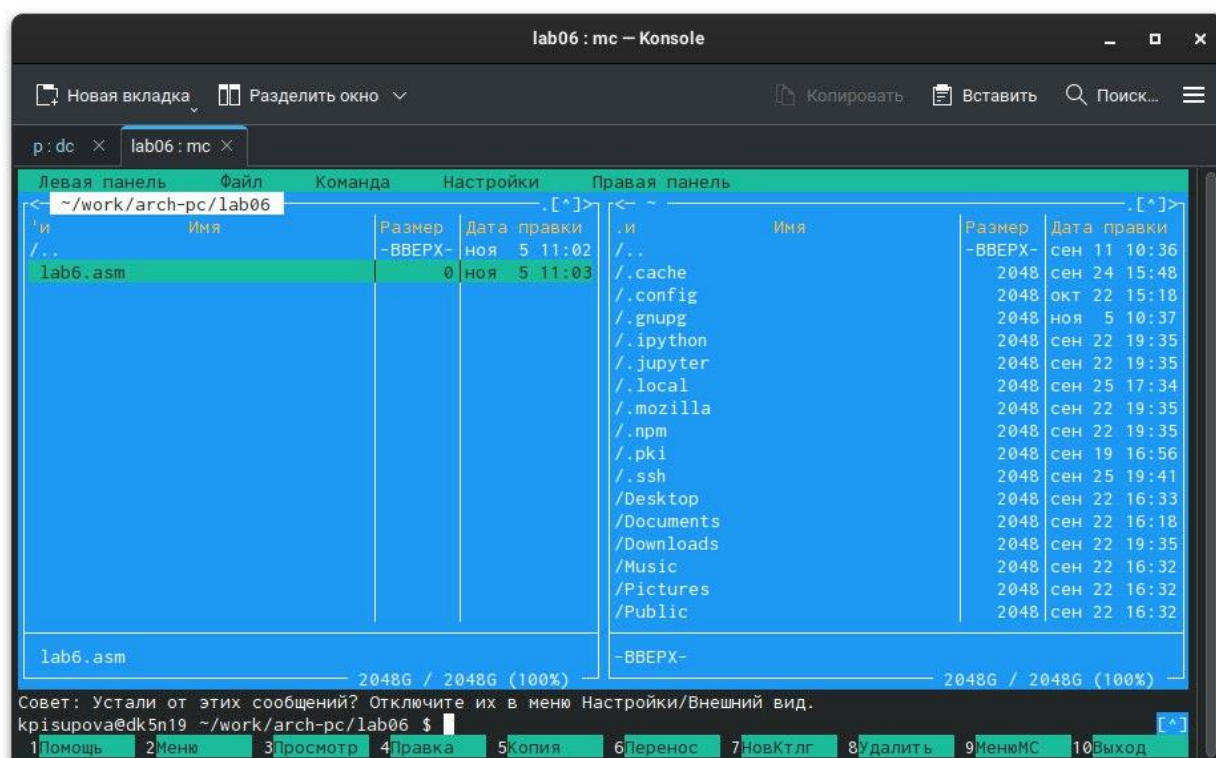


Рис. 5 Создание файла в Midnight Commander

4.2 Работа в NASM

С помощью F4 открываю только что созданный файл и вношу код с листинга (Рис. 6).

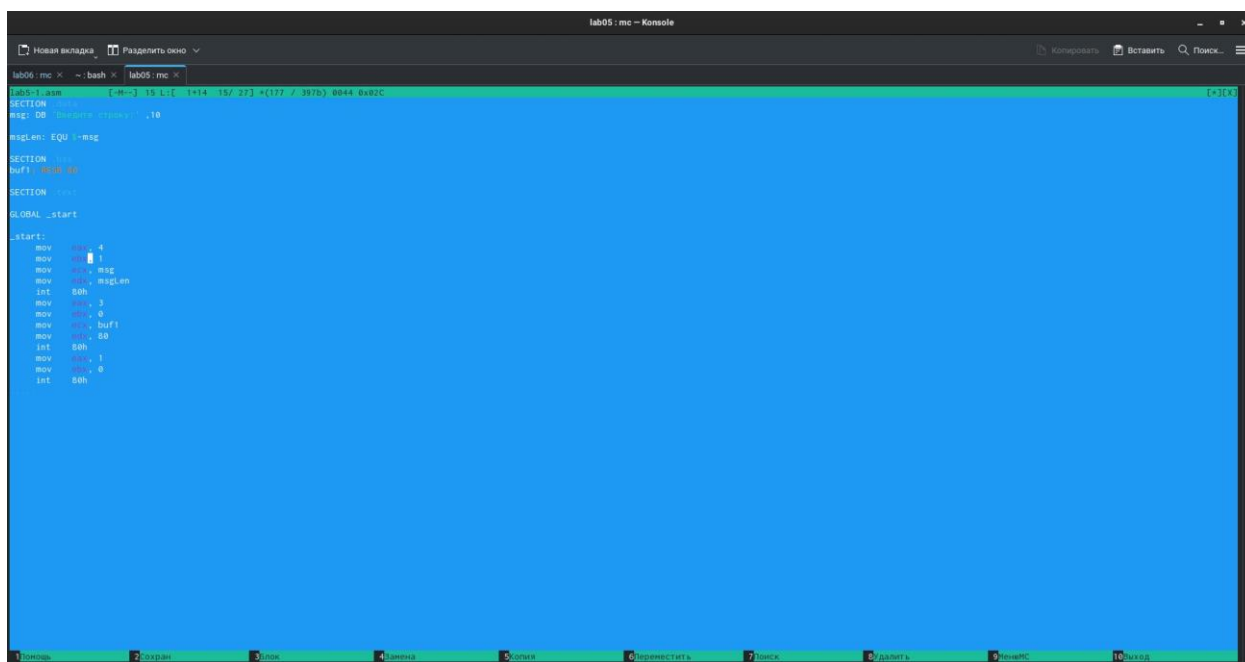


Рис. 6 Редактирование файла в Midnight Commander

Проверяю сохраненные изменения с помощью клавиши F3 (Рис. 7).

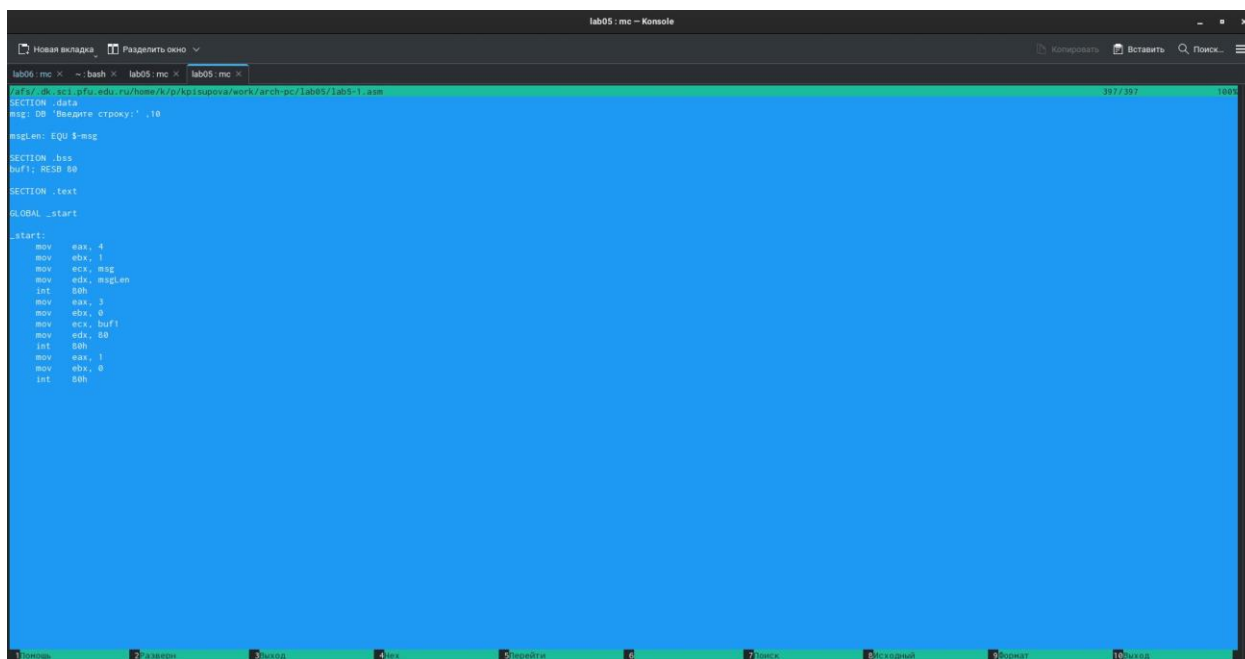


Рис. 7 Проверка сохранения сделанных изменений

Транслирую и компоную измененный файл, запускаю (Рис. 8).

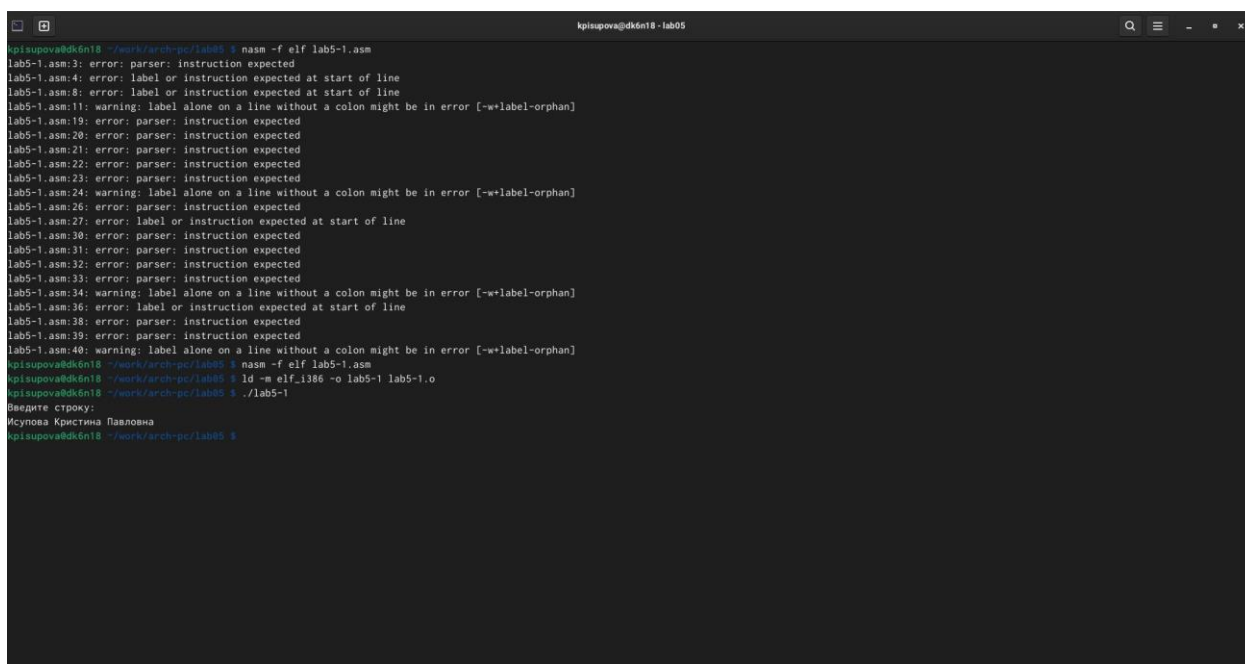


Рис. 8 Трансляция, компоновка и последующий запуск программы

4.3 Подключение внешнего файла

Скачанный с ТУИС файл сохраняю в общую папку на своем компьютере, на виртуальной машине в интерфейсе Midnight Commander перехожу в директорию общей папки, копирую файл в рабочий подкаталог. (Рис. 9).

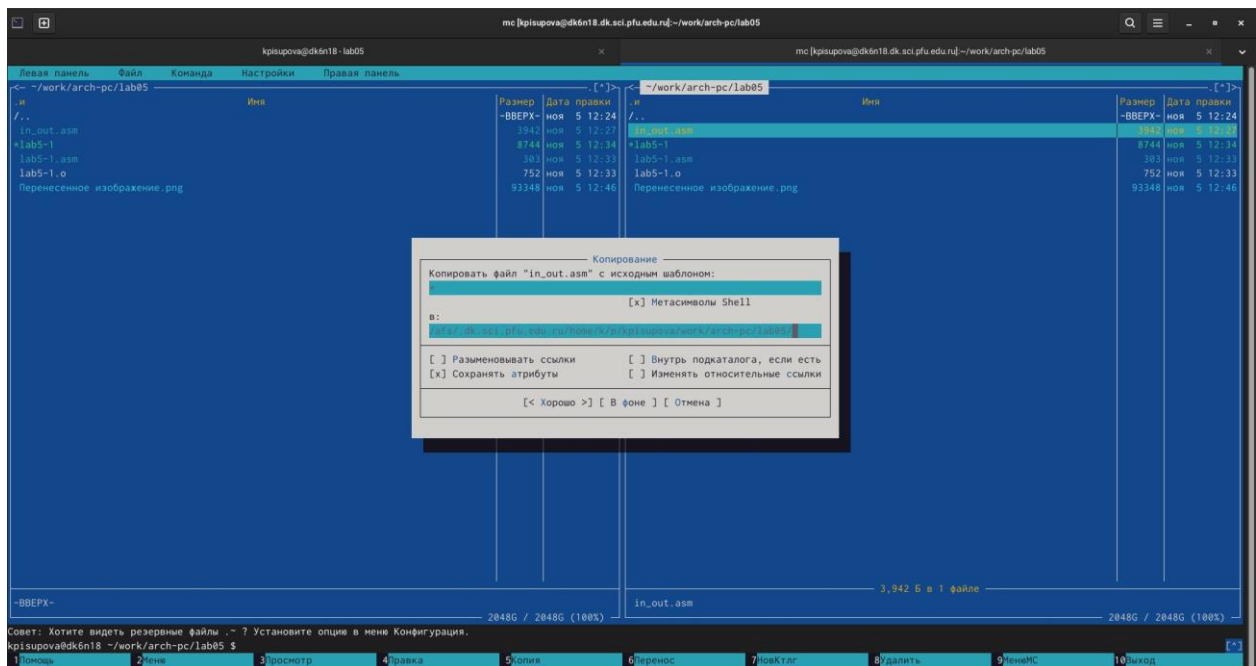


Рис. 9 Копирование файла в рабочий каталог

Создаю копию файла для последующей работы с ним (Рис. 10).

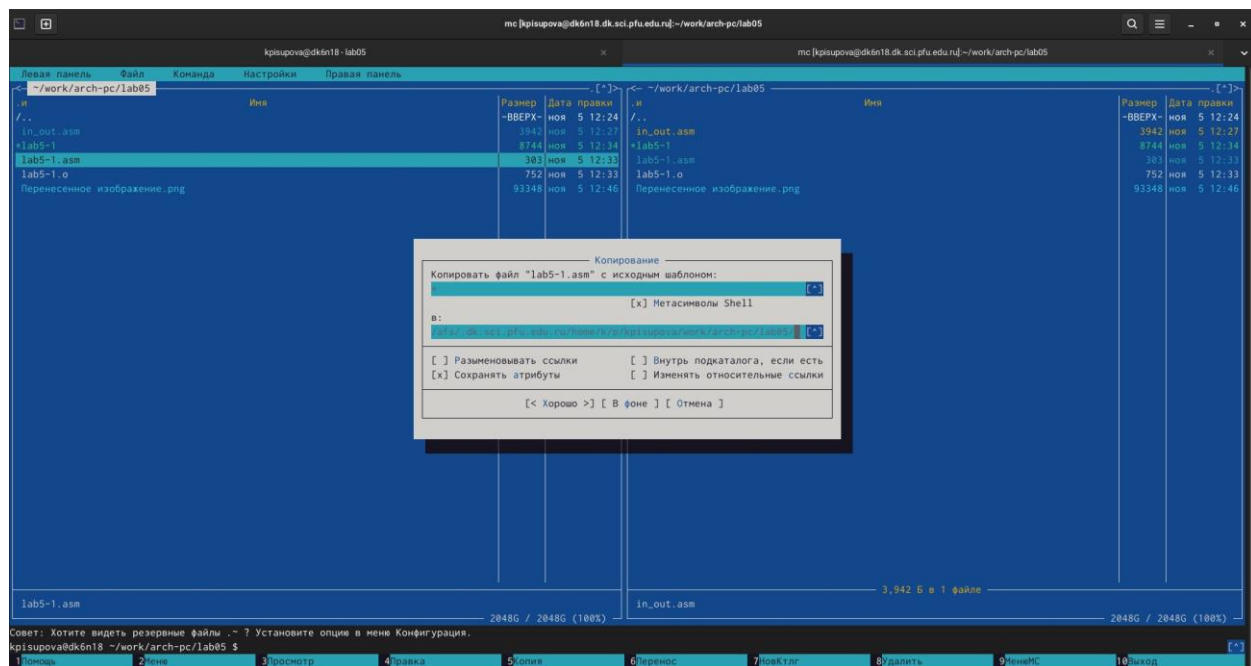


Рис. 10 Создание копии файла в Midnight Commander

В копии файла подключаю подпрограмм из подключенного файла (Рис. 11).



Рис. 11 Изменение программы

Транслирую, компоную и запускаю программу с подключенным файлом (Рис. 12).

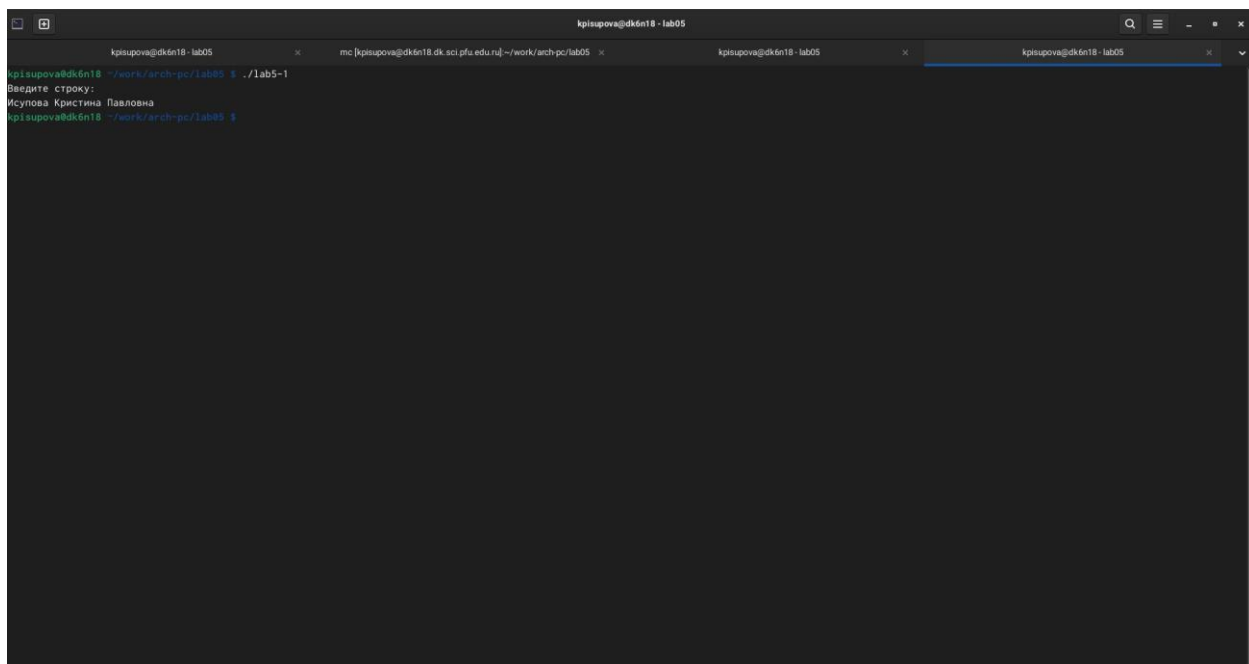
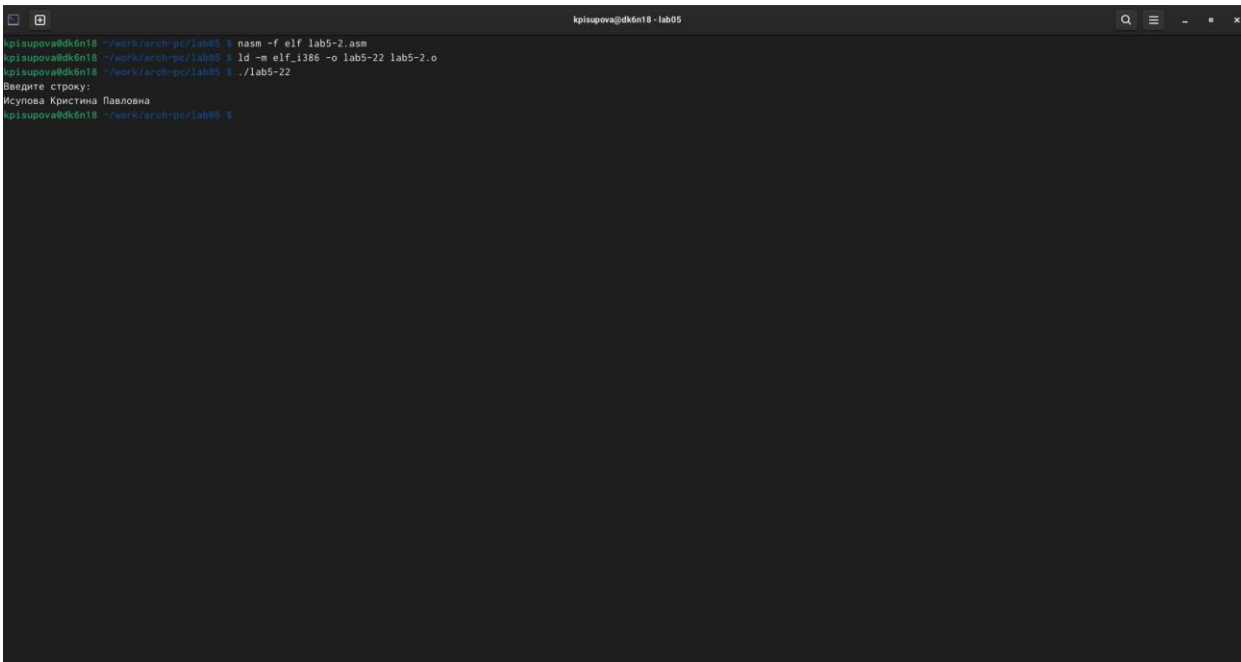


Рис. 12 Запуск измененной программы

Редактирую файл и заменяю в нем подпрограмму sprintLF на sprint. Разница подпрограмм в том, что вторая вызывает ввод на той же строке (Рис. 13).

4.4 Задание для самостоятельной работы

A screenshot of a terminal window with a dark background. The window title is 'kpisupova@dkn18 - lab05'. The terminal shows the following commands and output:

```
kpisupova@dkn18 ~/work/arch-ipc/lab05 $ nasm -f elf lab5-2.asm
kpisupova@dkn18 ~/work/arch-ipc/lab05 $ ld -m elf_i386 -o lab5-22 lab5-2.o
kpisupova@dkn18 ~/work/arch-ipc/lab05 $ ./lab5-22
Введите строку:
Исупова Кристина Павловна
kpisupova@dkn18 ~/work/arch-ipc/lab05 $
```

Рис. 13 Запуск изменённой программы с другой подпрограммой

Создаю копию lab5-1.asm, редактирую так, чтобы в конце выводилась введённая мною строка с клавиатуры (Рис. 14).

```

SECTION .data

msg: DB 'Введите строку:',10
msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text

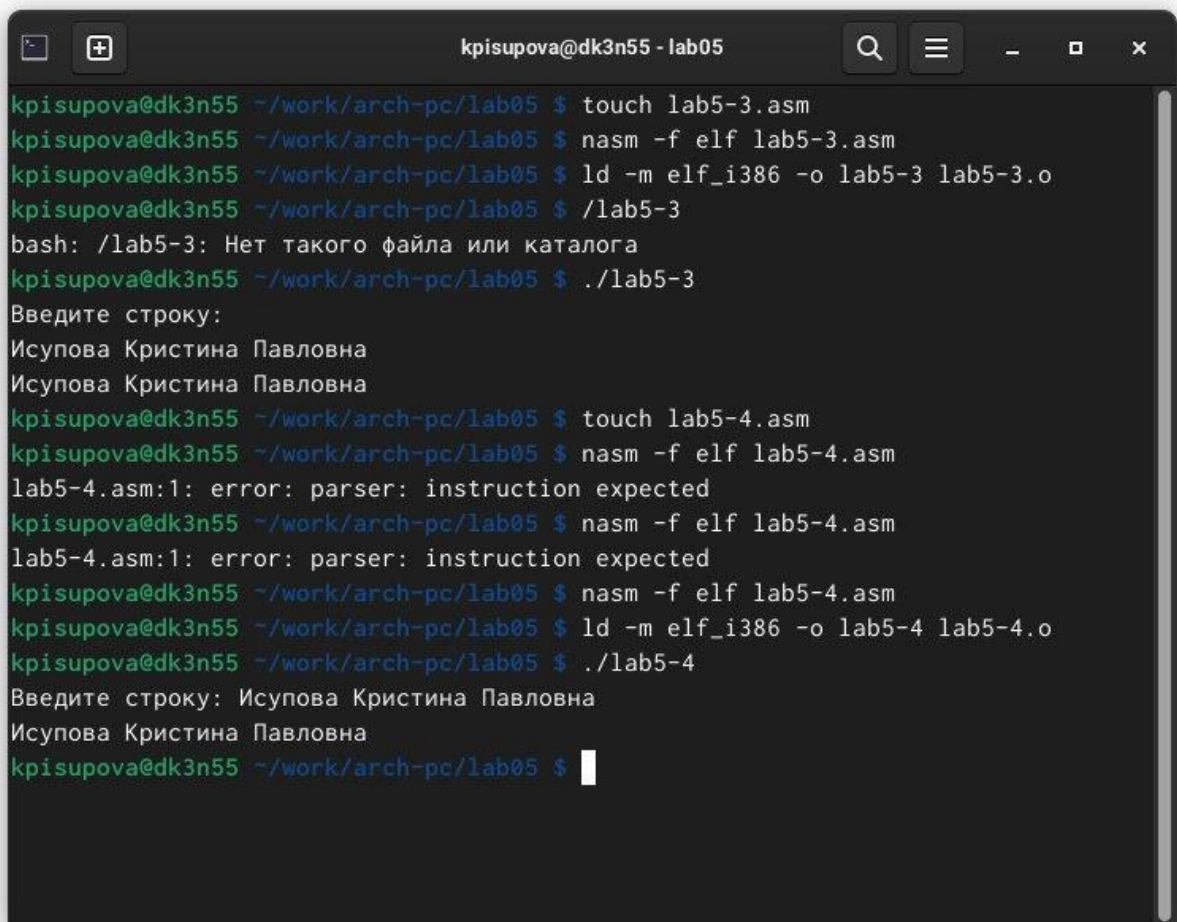
GLOBAL _start

_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, buf1
    int     80h
    mov     eax, 1
    mov     ebx, 0
    int     80h

```

Рис. 14 Редактирование копии

Транслирую, компоную и запускаю свою программу (Рис. 15).



The screenshot shows a terminal window with the title 'kpisupova@dk3n55 - lab05'. The user is in the directory ~/work/arch-pc/lab05. They create a file lab5-3.asm, compile it with nasm -f elf, and then link it with ld -m elf_i386 -o lab5-3 lab5-3.o. They attempt to run ./lab5-3, but get an error: 'Нет такого файла или каталога'. They then create lab5-4.asm, compile it with nasm -f elf, and link it with ld -m elf_i386 -o lab5-4 lab5-4.o. They attempt to run ./lab5-4, but get an error: 'Введите строку: Исупова Кристина Павловна'. The terminal shows the input 'Исупова Кристина Павловна' and the prompt 'kpisupova@dk3n55 ~/work/arch-pc/lab05 \$'.

```
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ touch lab5-3.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-3.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-3 lab5-3.o
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-3
bash: ./lab5-3: Нет такого файла или каталога
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-3
Введите строку:
Исупова Кристина Павловна
Исупова Кристина Павловна
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ touch lab5-4.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
lab5-4.asm:1: error: parser: instruction expected
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
lab5-4.asm:1: error: parser: instruction expected
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-4 lab5-4.o
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-4
Введите строку: Исупова Кристина Павловна
Исупова Кристина Павловна
kpisupova@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 15 Запуск своей программы.

Код прикладываю

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, buf1
int 80h
mov eax, 1
mov ebx, 0
int 80h
```

Создаю копию lab5-2.asm, редактирую так, чтобы в конце выводилась введенная мною строка с клавиатуры (Рис. 16).


```
%include 'in_out.asm'

SECTION .data

msg: DB 'Введите строку: ', 0h
msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

    mov eax, msg
    call sprint

    mov ecx, buf1
    mov edx, 80

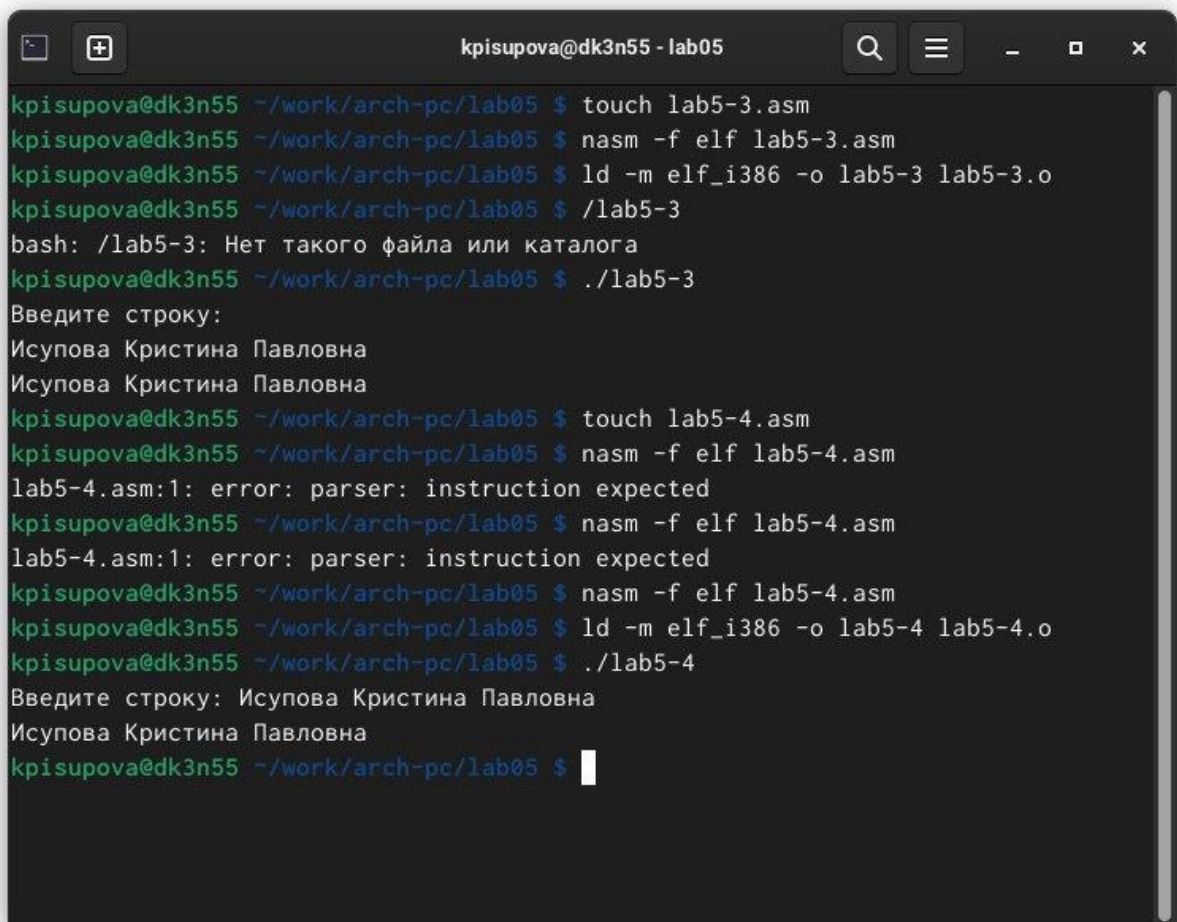
    call sread

    mov eax, 4
    mov ebx, 1
    mov ecx, buf1
    int 80h

    call quit
```

Рис. 16 Редактирование копии

Транслирую, компоную и запускаю свою программу (Рис. 17).

A terminal window titled 'kpisupova@dk3n55 - lab05' with standard window controls. The terminal shows the following commands and output:

```
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ touch lab5-3.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-3.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-3 lab5-3.o
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-3
bash: ./lab5-3: Нет такого файла или каталога
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-3
Введите строку:
Исупова Кристина Павловна
Исупова Кристина Павловна
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ touch lab5-4.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
lab5-4.asm:1: error: parser: instruction expected
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
lab5-4.asm:1: error: parser: instruction expected
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-4.asm
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-4 lab5-4.o
kpisupova@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-4
Введите строку: Исупова Кристина Павловна
Исупова Кристина Павловна
kpisupova@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 17 Запуск своей программы

Код прикладываю

```
%include 'in_out.asm'
```

```
SECTION .data
```

```
msg: DB 'Введите строку: ', 0h
```

```
msgLen: EQU $-msg
```

```
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, 4  
mov ebx, 1  
mov ecx, buf1  
int 80h  
call quit
```

5 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы в Midnight Commander, а также освоила

Список литературы

1. [Пример выполнения лабораторной работы](#)
2. [Курс на ТУИС](#)
3. [Лабораторная работа №5](#)
4. [Программирование на языке ассемблера NASM Столяров А. В.](#)