

Лабораторная работа номер 5

Архитектура компьютера

Титков Ярослав Максимович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Выводы:	20

Список иллюстраций

4.1	Пользуясь командой <code>mc</code> открыл Midnight Commander	9
4.2	Создал папку <code>lab2</code>	10
4.3	С помощью команды <code>touch</code> создал <code>asm</code> файл и проверил в Midnight Commander	11
4.4	С помощью клавиши <code>F4</code> открыл файл <code>lab5-1.asm</code> и ввёл предложенный текст	12
4.5	Прописываю команды <code>NASM</code> и запускаю файл	13
4.6	С помощью клавиши <code>F6</code> создал копию файла <code>lab5-1.asm</code> с именем <code>lab5-2.asm</code>	14
4.7	С помощью команды <code>папо</code> исправил текст в новом файле	15
4.8	Отредактировал текст так, чтобы строки были на одном уровне и запустил файл	16
4.9	Скопировал файл <code>lab5-1.asm</code> и задал ему имя <code>lab5-1-1.asm</code> , а затем с помощью <code>папо</code> изменил алгоритм на нужный для выполнения самостоятельной работы	17
4.10	Проделал тоже самое с файлом <code>lab5-2</code>	18
4.11	запустил файл <code>lab5-2-2</code>	19

Список таблиц

1 Цель работы

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

2 Задание

1. Основы работы с тс
2. Структура программ
3. Подключение файлов
4. Фиксация всей работы
5. Самостоятельная работа

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 (четырёх- рённое 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

1. Основы работы с Midnight Commander

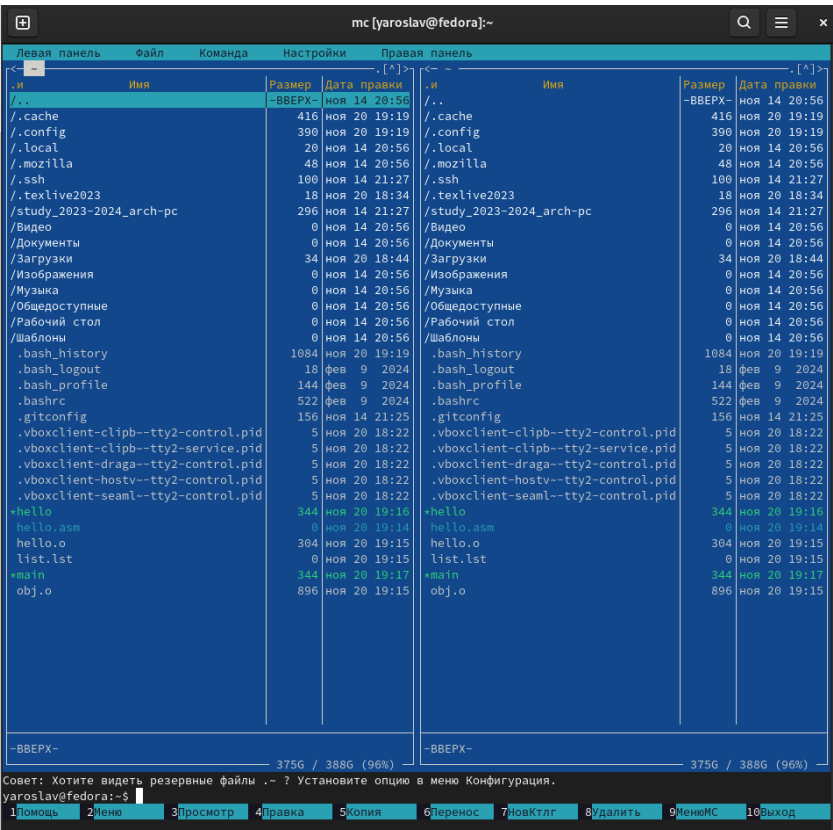


Рис. 4.1: Пользуясь командой mc открыл Midnight Commander

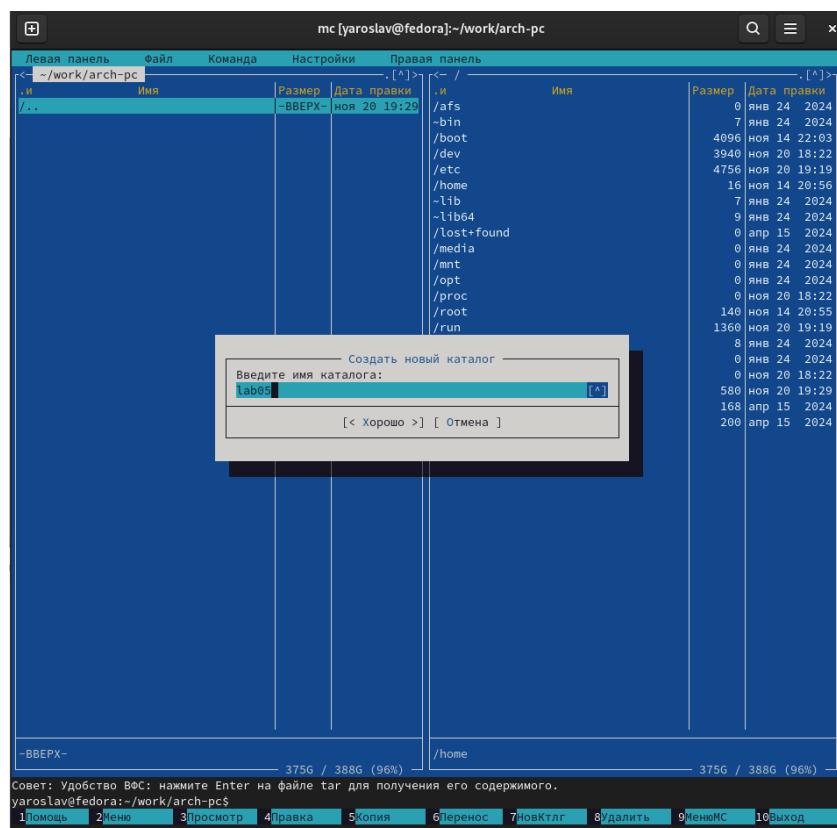


Рис. 4.2: Создал папку lab2

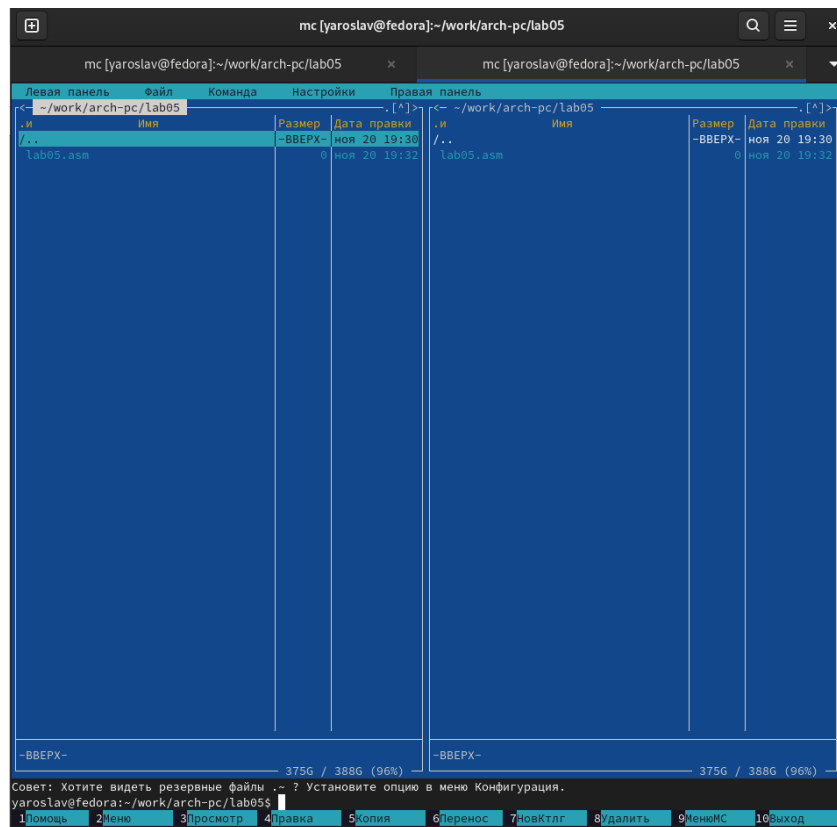
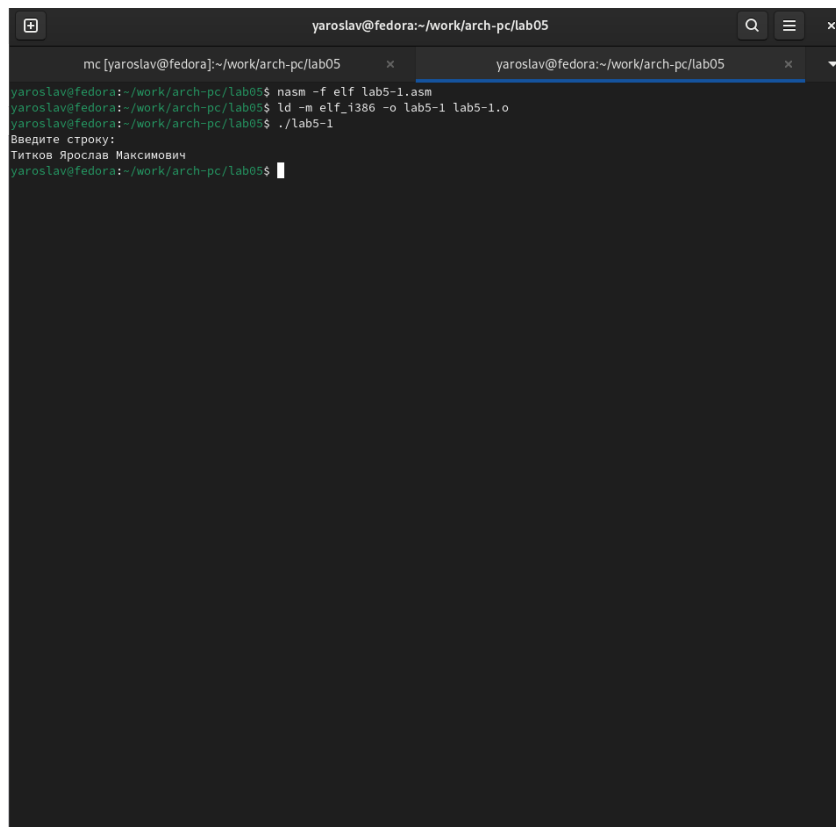


Рис. 4.3: С помощью команды `touch` создал `asm` файл и проверил в Midnight Commander

```
mc [yaroslav@fedora]:~/work/arch-pc/lab05
/home/yaroslav/work/arch-pc/lab05/lab5-1.asm 1296/1296 100%
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
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
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
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h

1Помощь 2Разверн 3Выход 4Нех 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход
```

Рис. 4.4: С помощью клавиши F4 открыл файл lab5-1.asm и ввёл предложенный текст



```
yaroslav@fedora:~/work/arch-pc/lab05
mc [yaroslav@fedora]:~/work/arch-pc/lab05
yaroslav@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
yaroslav@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
yaroslav@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Титков Ярослав Максимович
yaroslav@fedora:~/work/arch-pc/lab05$
```

Рис. 4.5: Прописываю команды NASM и запускаю файл

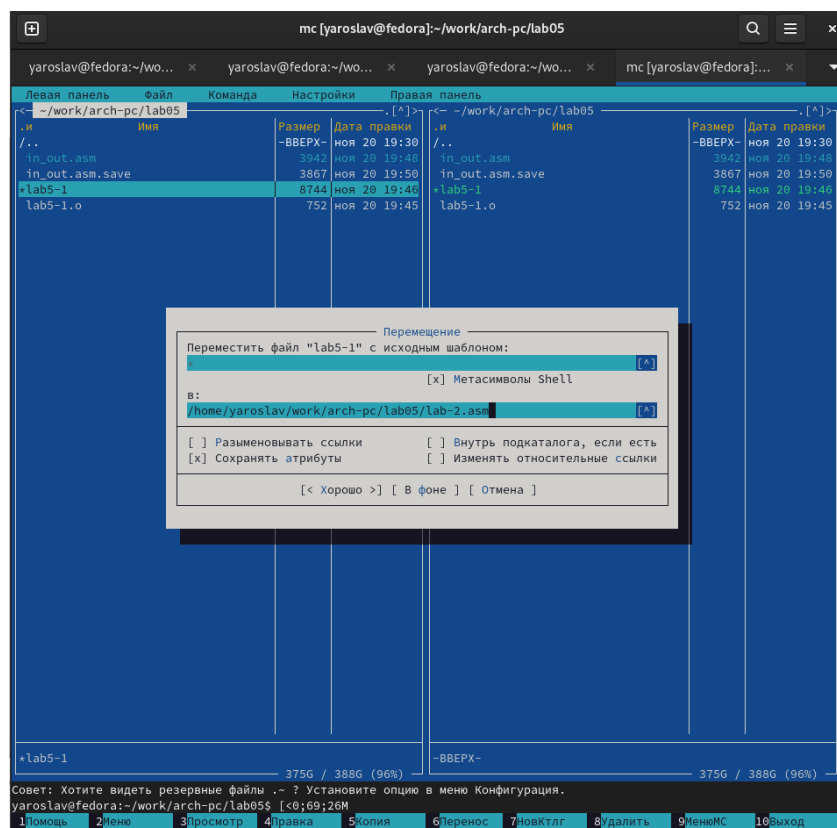
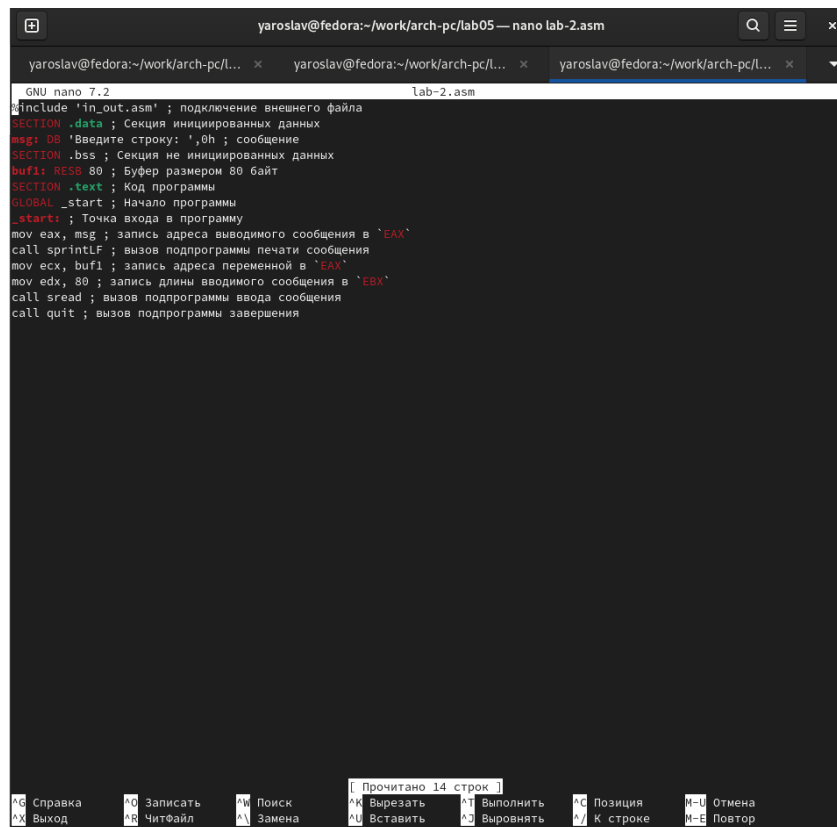


Рис. 4.6: С помощью клавиши F6 создал копию файла lab5-1.asm с именем lab5-2.asm



```
GNU nano 7.2 lab-2.asm
#include "in_out.asm" ; подключение внешнего файла
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не иницированных 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 sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Прочитано 14 строк

Ctrl-G Справка	Ctrl-O Записать	Ctrl-W Поиск	Ctrl-I Прочитано 14 строк	Ctrl-T Выполнить	Ctrl-Q Позиция	Ctrl-U Отмена
Ctrl-X Выход	Ctrl-R ЧитФайл	Ctrl-A Замена	Ctrl-K Вырезать	Ctrl-V Вставить	Ctrl-J Вывод	Ctrl-E Повтор

Рис. 4.7: С помощью команды nano исправил текст в новом файле

```
yaroslav@fedora:~/work/arch-pc/lab05$ nano lab5-2.asm
yaroslav@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
yaroslav@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
yaroslav@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Титков Ярослав
yaroslav@fedora:~/work/arch-pc/lab05$
```

Рис. 4.8: Отредактировал текст так, чтобы строки были на одном уровне и запустил файл

2. Задания для самостоятельной работы:

3. Создание копии файла lab5-1.asm и изменение программы

Я создал копию файла lab5-1.asm и назвал её lab5-1-modified.asm. Затем я внес изменения в программу, чтобы она работала по следующему алгоритму:

Вывести приглашение типа "Введите строку:".

Ввести строку с клавиатуры.

Вывести введенную строку на экран.

2. Получение исполняемого файла и проверка работы

Я получил исполняемый файл и проверил его работу. На приглашение ввести строку я ввел свою фамилию. 3. Создание копии файла lab5-2.asm и исправление программы

Я создал копию файла lab5-2.asm и назвал её lab5-2-modified.asm. Затем я исправил текст программы с использованием подпрограмм из внешнего файла in_out.asm, чтобы она работала по следующему алгоритму:

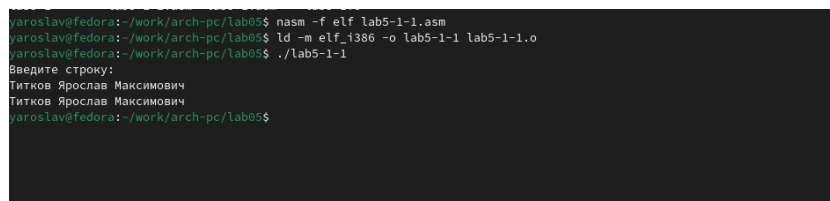
Вывести приглашение типа "Введите строку:".

Ввести строку с клавиатуры.

Вывести введенную строку на экран.

4. Создание исполняемого файла и проверка работы

Я создал исполняемый файл и проверил его работу.



```
yaroslav@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
yaroslav@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
yaroslav@fedora:~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Титков Ярослав Максимович
Титков Ярослав Максимович
yaroslav@fedora:~/work/arch-pc/lab05$
```

Рис. 4.9: Скопировал файл lab5-1.asm и задал ему имя lab5-1-1.asm, а затем с помощью nano изменил алгоритм на нужный для выполнения самостоятельной работы

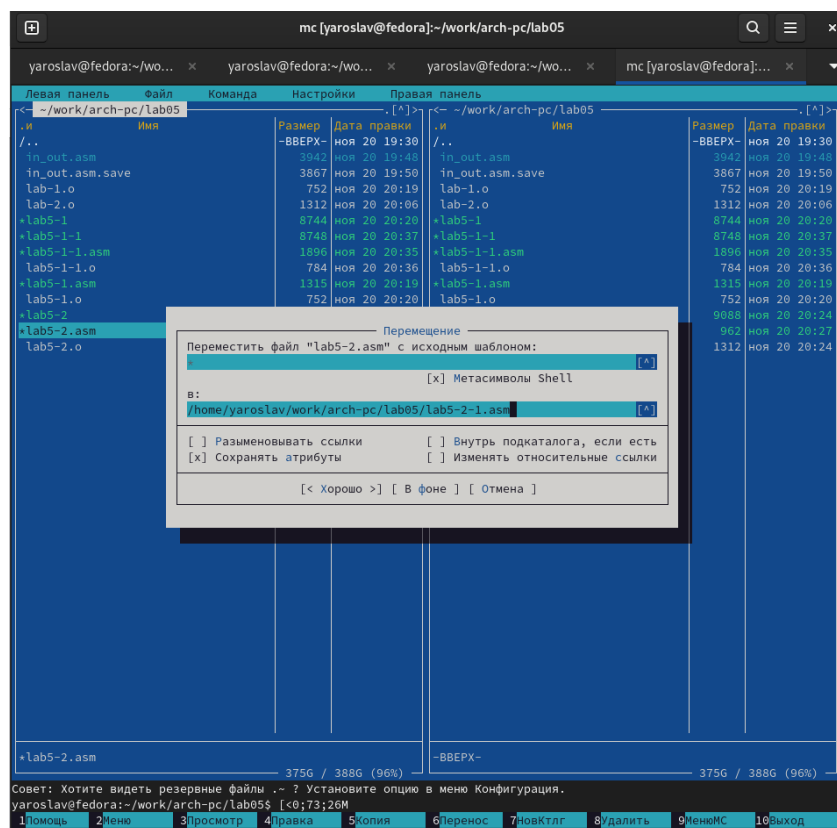
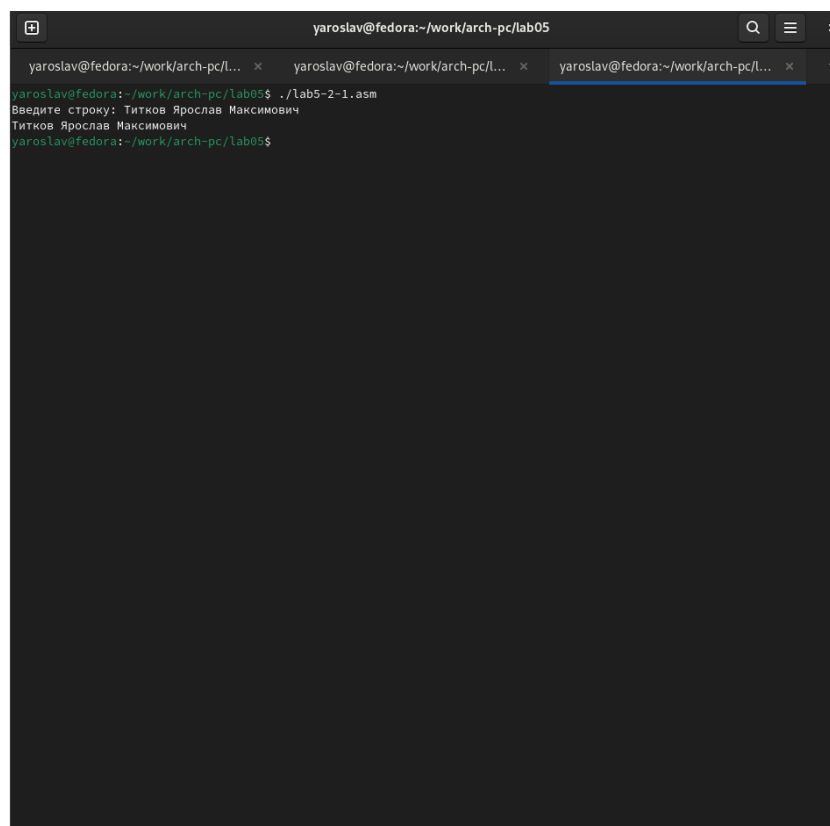


Рис. 4.10: Проделал тоже самое с файлом lab5-2



The image shows a terminal window with a dark background. The title bar at the top reads "yaroslav@fedora:~/work/arch-pc/lab05". There are three tabs open, all with the same title "yaroslav@fedora:~/work/arch-pc/L...". The terminal content shows the user running the command `./lab5-2-1.asm`. The output of the program is "Введите строку: Титков Ярослав Максимович" followed by "Титков Ярослав Максимович". The prompt `yaroslav@fedora:~/work/arch-pc/lab05$` is visible at the bottom.

```
yaroslav@fedora:~/work/arch-pc/lab05$ ./lab5-2-1.asm
Введите строку: Титков Ярослав Максимович
Титков Ярослав Максимович
yaroslav@fedora:~/work/arch-pc/lab05$
```

Рис. 4.11: запустил файл lab5-2-2

5 Выводы:

При выполнении данной лабораторной работы я приобрел практические навыки работы в Midnight Commander, а также освоил инструкции языка ассемблера `mov` и `int`.