

# **Отчет по лабораторной работе №5**

**Дисциплина: архитектура компьютера**

**Чувакина Мария Владимировна**

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
4.1	Основы работы с тс	8
4.2	Структура программы на языке ассемблера NASM	9
4.3	Подключение внешнего файла	11
4.4	Выполнение заданий для самостоятельной работы	14
<b>5</b>	<b>Выводы</b>	<b>20</b>
<b>6</b>	<b>Список литературы</b>	<b>21</b>

# Список иллюстраций

4.1 Открытый mc .....	8
4.2 Перемещение между директориями .....	8
4.3 Создание каталога.....	9
4.4 Перемещение между директориями .....	9
4.5 Создание файла .....	9
4.6 Открытие файла для редактирования.....	10
4.7 Редактирование файла.....	10
4.8 Открытие файла для просмотра.....	10
4.9 Компиляция файла и передача на обработку компоновщику . . . . .	11
4.10 Исполнение файла.....	11
4.11 Скачанный файл.....	11
4.12 Копирование файла .....	12
4.13 Копирование файла .....	12
4.14 Редактирование файла.....	13
4.15 Исполнение файла.....	13
4.16 Отредактированный файл.....	14
4.17 Исполнение файла.....	14
4.18 Копирование файла .....	15
4.19 Редактирование файла.....	15
4.20 Исполнение файла.....	16
4.21 Копирование файла .....	17
4.22 Редактирование файла.....	17
4.23 Исполнение файла.....	18

# 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 mc
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 иницированных (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 иницированных 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-байтное 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).





С помощью функциональной клавиши F7 создаю каталог lab05 (рис. [4.3]).

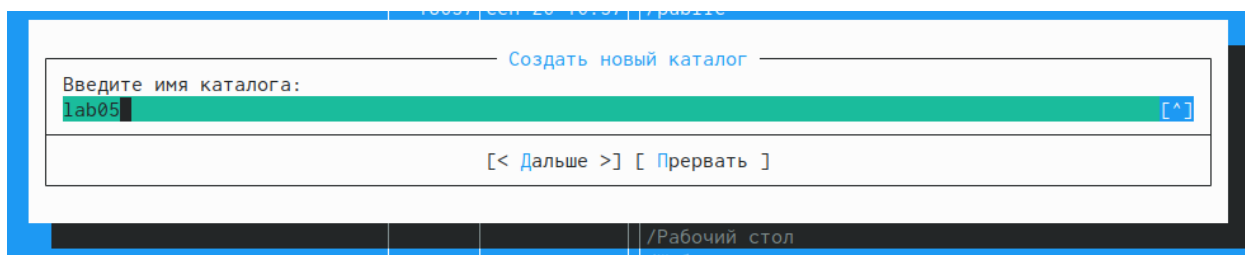


Рис. 4.3: Создание каталога

Переходу в созданный каталог (рис. [4.4]).

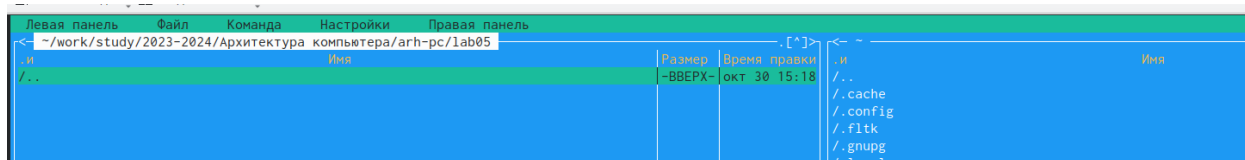


Рис. 4.4: Перемещение между директориями

В строке ввода прописываю команду touch lab5-1.asm, чтобы создать файл, в котором буду работать (рис. [4.5]).

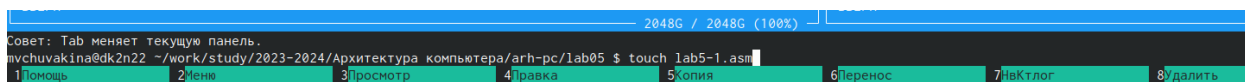


Рис. 4.5: Создание файла

## 4.2 Структура программы на языке ассемблера NASM

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano (рис. [4.6]).

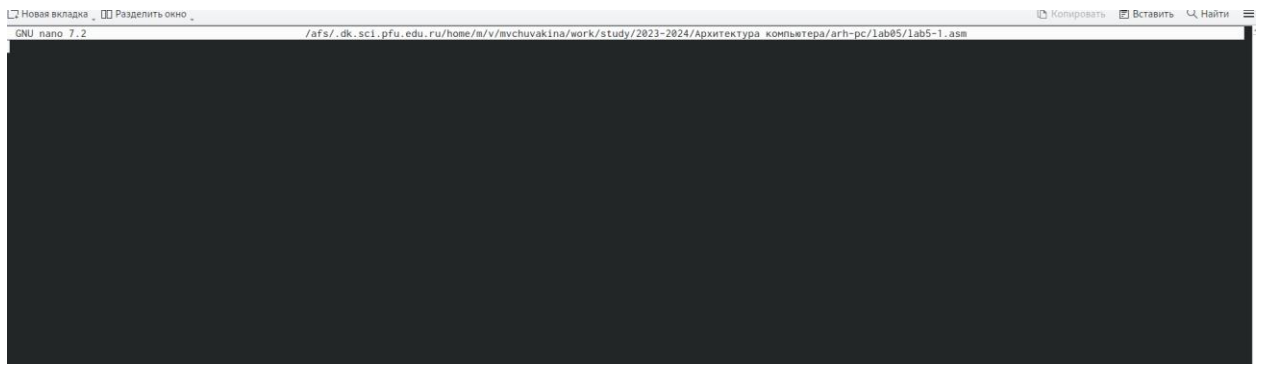


Рис. 4.6: Открытие файла для редактирования

Ввожу в файл код программы для запроса строки у пользователя (рис. [4.7]). Далее выхожу из файла (Ctrl+X), сохраняя изменения (Y, Enter).

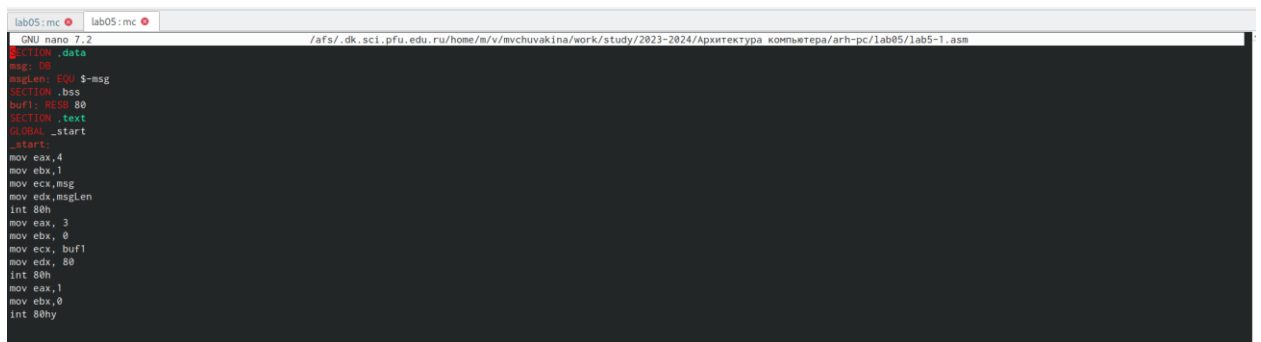


Рис. 4.7: Редактирование файла

С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы (рис. [4.8]).

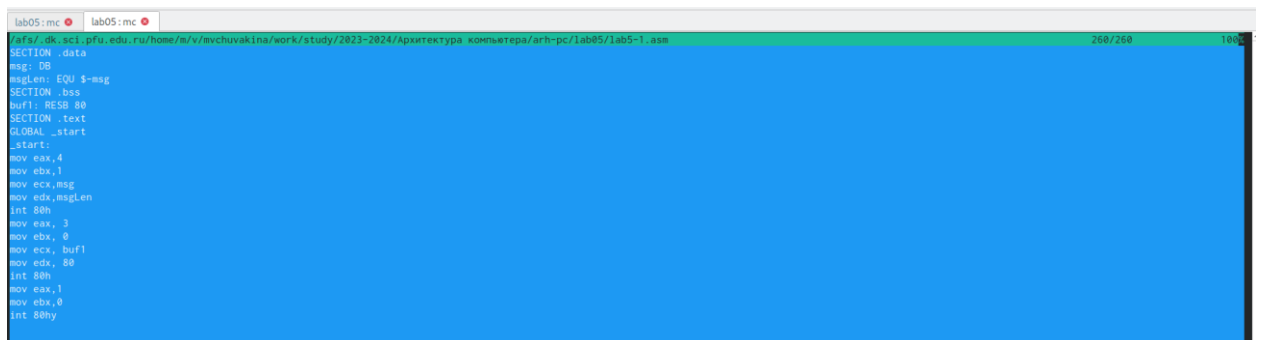


Рис. 4.8: Открытие файла для просмотра

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-1.asm`. Создался объектный файл `lab5-1.o`. Выполняю компоновку объектного

файла с помощью команды `ld -m elf_i386 -o lab5-1 lab5-1.o` (рис. [4.9]). Создался исполняемый файл `lab5-1`.

```
nvchuvakina@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/argh-pc/lab05 $ nasm -f elf lab5-1.asm
nvchuvakina@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/argh-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
nvchuvakina@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/argh-pc/lab05 $
```

Рис. 4.9: Компиляция файла и передача на обработку компоновщику

Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. [4.10]).

```
nvchuvakina@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/argh-pc/lab05 $ ./lab5-1
Введите строку:
Чувакина Мария Владимировна
```

Рис. 4.10: Исполнение файла

4.3 Подключение внешнего файла

Скачиваю файл `in_out.asm` со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. [4.11]).

Левая панель	Файл	Команда	Настройки	Правая панель			
~/work/study/2023-2024/Архитектура компьютера/arghpc				~/Загрузки			
o	Имя	Размер	Время ["]	o	Имя	Размер	Время ["]
./		-BBERX-	окт 10 11:23	./		-BBERX-	ноя 7 11:38
./git		2048	окт 17 11:36	1.2	1.2. Подключение комментариев к условию задачи: файл.docx	38022	ноя 11 11:38
./config		2048	сен 26 10:57	1.3	1.3. Подключение комментариев к условию задачи(1).docx	38170	ноя 11 11:38
./lab04		2048	окт 17 11:08	1.4	1.4. Подключение комментариев к условию задачи(1).docx	38118	ноя 11 11:38
./lab05		2048	ноя 7 11:38	1.5	1.5. Подключение комментариев к условию задачи: файл.docx	38200	ноя 11 11:38
./lab06		2048	окт 30 15:19	lab01.asm.txt		850	ноя 11 11:37
./labs		2048	окт 10 11:24	lab02.asm		3944	ноя 7 11:37
./presentation		2048	сен 26 11:00	lab03.asm.txt		850	ноя 11 11:37
./template		2048	сен 26 10:57	lab04.asm.txt		101100	ноя 11 11:37
./gitattributes		1768	сен 26 10:57	lab05.asm.txt		101100	ноя 11 11:37
./gitignore		4632	сен 26 10:57	lab06.asm.txt		101100	ноя 11 11:37
./gitmodules		279	сен 26 10:57	lab07.asm.txt		101100	ноя 11 11:37
./README.md		1012	сен 26 10:57	lab08.asm.txt		101100	ноя 11 11:37
COURSE		0	сен 26 11:00	lab09.asm.txt		101100	ноя 11 11:37
LICENSE		18657	сен 26 10:57	lab10.asm.txt		101100	ноя 11 11:37
Makefile		819	сен 26 10:57	lab11.asm.txt		101100	ноя 11 11:37
./lab01		100	сен 26 10:57	lab12.asm.txt		101100	ноя 11 11:37
./lab02		1012	сен 26 10:57	lab13.asm.txt		101100	ноя 11 11:37
./lab03		1012	сен 26 10:57	lab14.asm.txt		101100	ноя 11 11:37
./prepare		0	сен 26 11:00	без имени.o		4984	окт 12 11:57
./lab02_Чувакина.отчет		0	сен 26 11:05	lab15.asm.txt		101100	ноя 11 11:37

Рис. 4.11: Скачанный файл

С помощью функциональной клавиши F5 копирую файл in\_out.asm из каталога Загрузки в созданный каталог lab05 (рис. [4.12]).

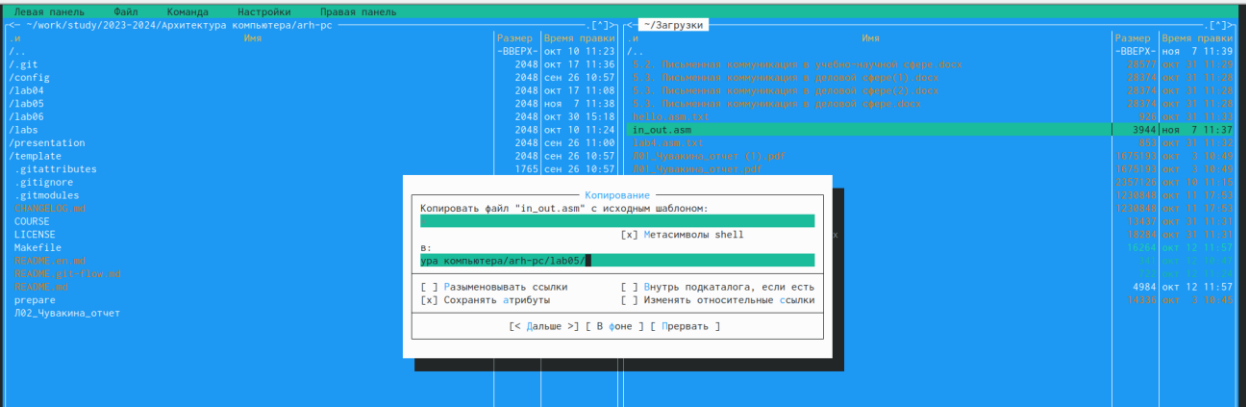


Рис. 4.12: Копирование файла

С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя для копии файла (рис. [4.13]).

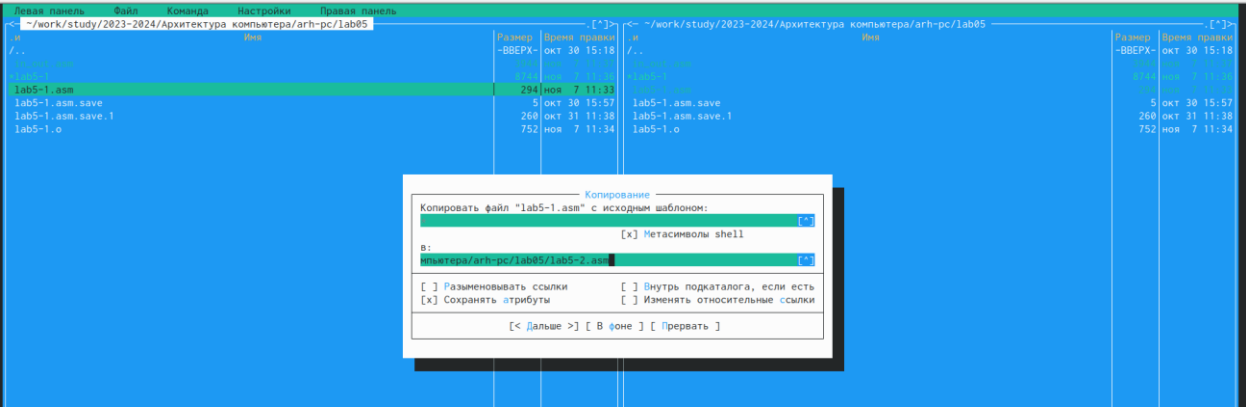


Рис. 4.13: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano (рис. [4.14]), чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.

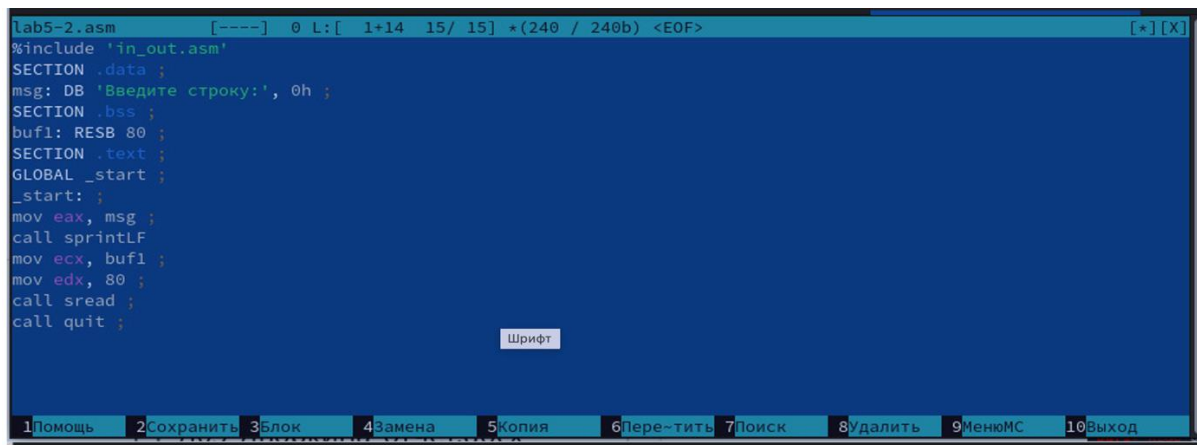


Рис. 4.14: Редактирование файла

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-2.asm`. Создался объектный файл `lab5-2.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-2 lab5-2.o`. Создался исполняемый файл `lab5-2`. Запускаю исполняемый файл (рис. [4.15]).

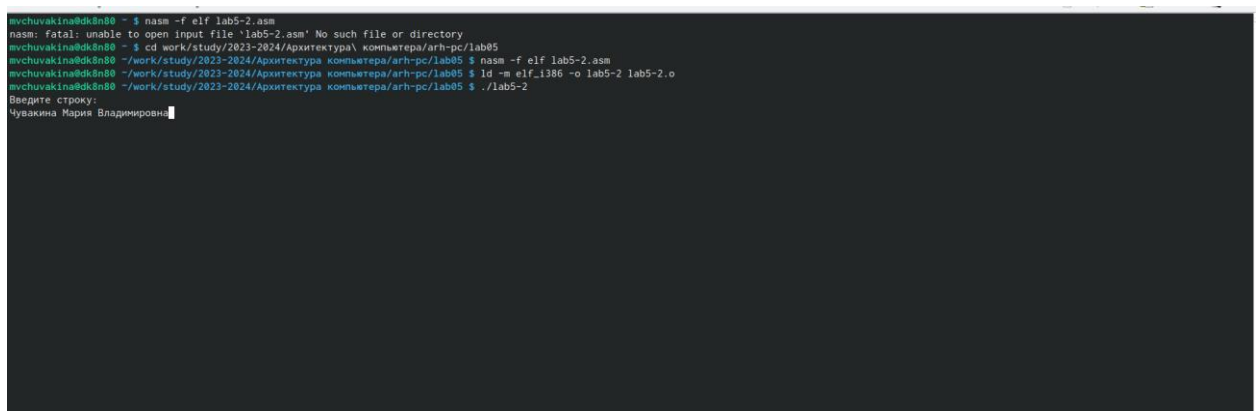


Рис. 4.15: Исполнение файла

Открываю файл `lab5-2.asm` для редактирования в папо функциональной клавишей F4. Изменяю в нем подпрограмму `sprintLF` на `sprint`. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий (рис. [4.16]).

```
%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 ;
```

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

Рис. 4.16: Отредактированный файл

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. [4.17]).

```
nvchuvakina@dkn88 ~ $ nasm -f elf lab5-2.asm
nasm: fatal: unable to open input file 'lab5-2.asm' No such file or directory
nvchuvakina@dkn88 ~ $ cd /work/study/2023-2024/Архитектура\ компьютера/arh-pc/lab05
nvchuvakina@dkn88 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ nasm -f elf lab5-2.asm
nvchuvakina@dkn88 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
nvchuvakina@dkn88 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ./lab5-2
Введите строку:
Чувачкина Мария Владимировна
```

Рис. 4.17: Исполнение файла

Разница между первым исполняемым файлом lab5-2 и вторым lab5-2-2 в том, что запуск первого запрашивает ввод с новой строки, а программа, которая исполняется при запуске второго, запрашивает ввод без переноса на новую строку, потому что в этом заключается различие между подпрограммами sprintLF и sprint.

## 4.4 Выполнение заданий для самостоятельной работы

1. Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5 (рис. [4.18]).

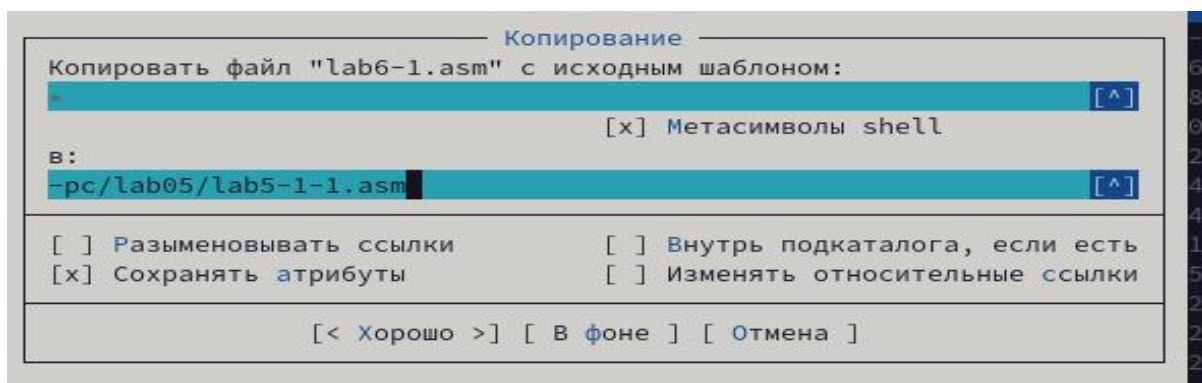


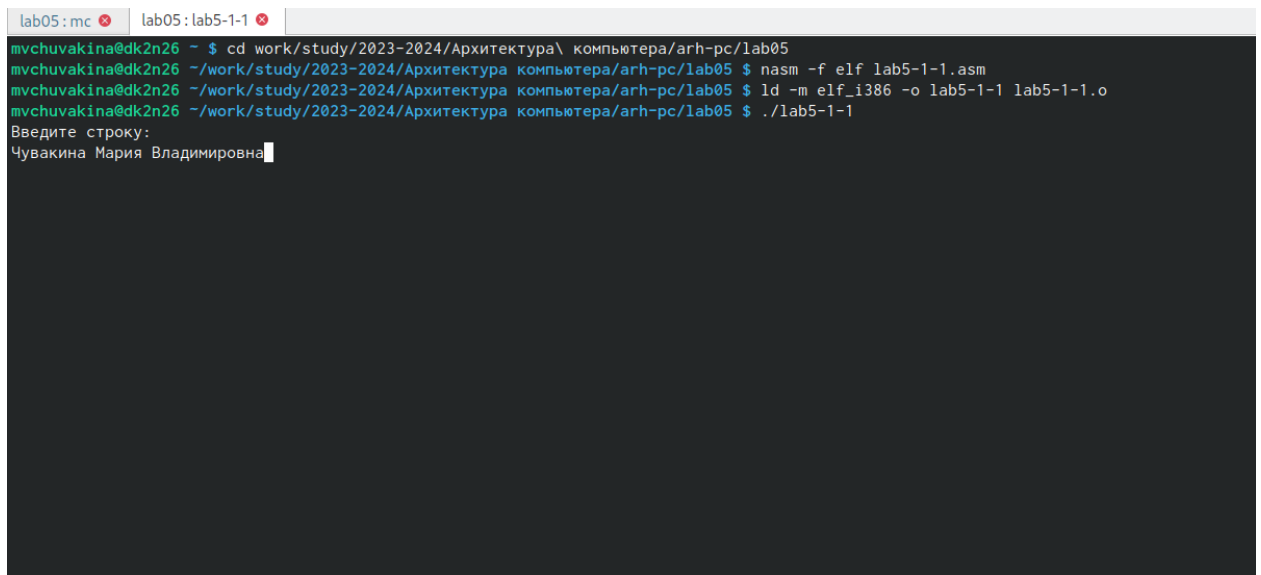
Рис. 4.18: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [4.19]).

```
GNU nano 7.2 /afs/.dk.sci.pfu.edu.ru/home/m/v/mvchuvakina/work/study/2023-2024/Архитектура компь
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. 4.19: Редактирование файла

2. Создаю объектный файл lab5-1-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-1-1, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. [4.20]).



```
lab05: mc x lab05: lab5-1-1 x
mvchuvakina@dk2n26 ~ $ cd work/study/2023-2024/Архитектура\ компьютера/arh-pc/lab05
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ nasm -f elf lab5-1-1.asm
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ./lab5-1-1
Введите строку:
Чувакина Мария Владимировна
```

Рис. 4.20: Исполнение файла

Код программы из пункта 1:

```
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициированных 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,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,buf1 ; Адрес строки buf1 в ecx
mov edx,buf1 ; Размер строки buf1
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** ; Вызов ядра

3. Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5 (рис. [4.21]).

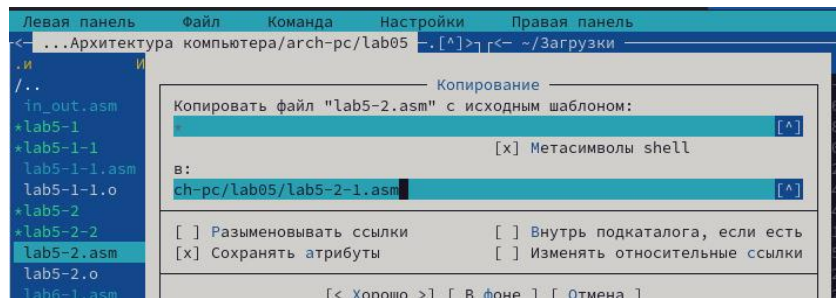


Рис. 4.21: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. [4.22]).

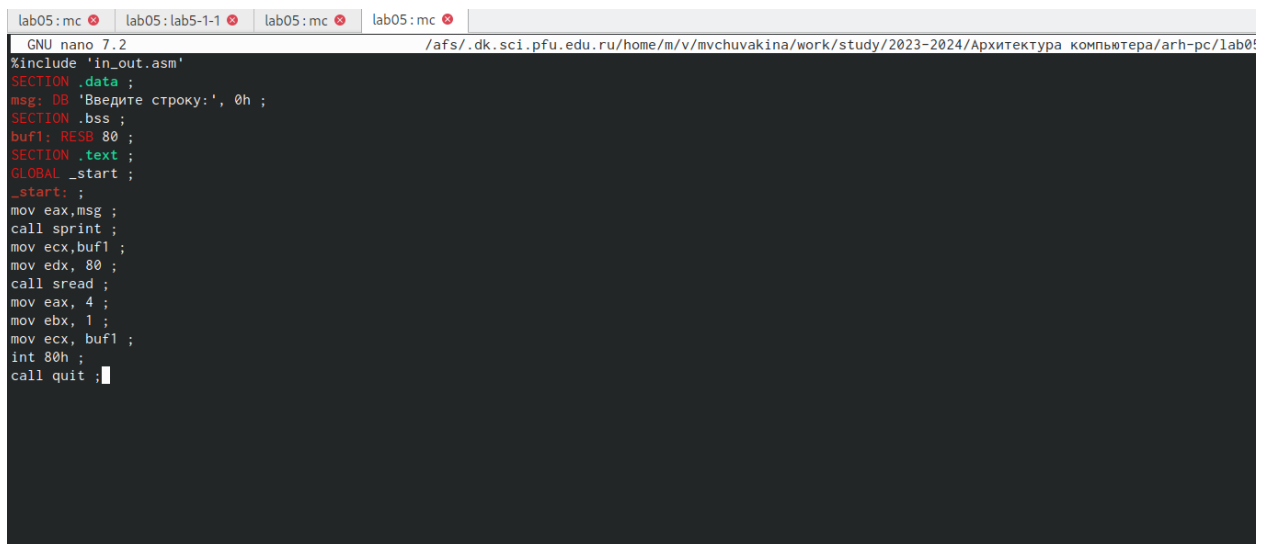
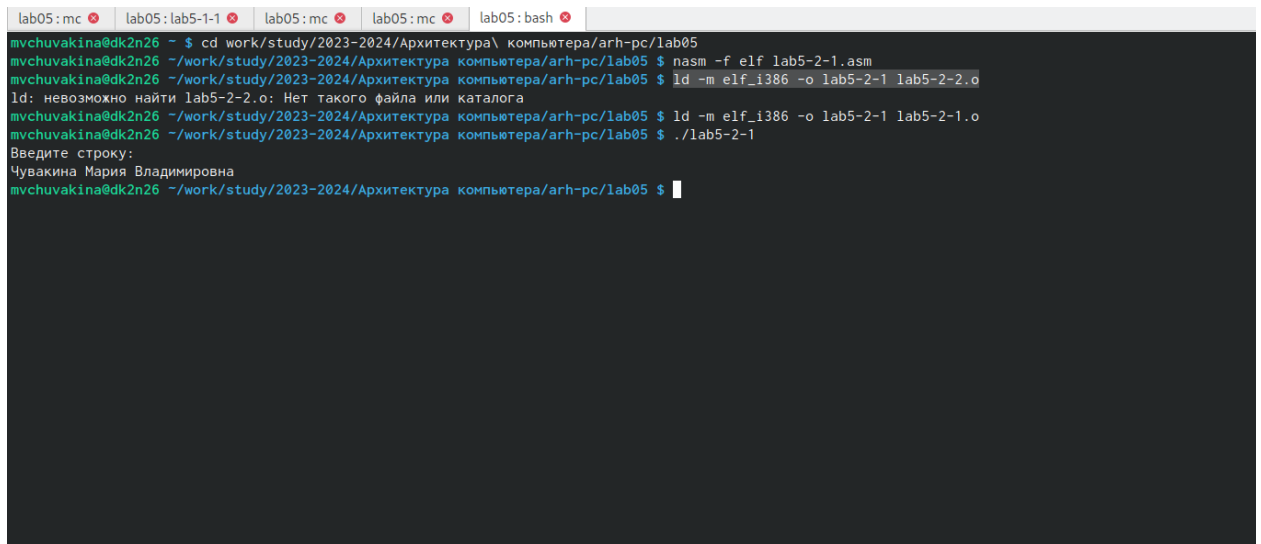


Рис. 4.22: Редактирование файла

4. Создаю объектный файл lab5-2-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2-1, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. [4.23]).



```
lab05:mc lab05:lab5-1-1 lab05:mc lab05:mc lab05:mc lab05:bash
mvchuvakina@dk2n26 ~ $ cd work/study/2023-2024/Архитектура\ компьютера/arh-pc/lab05
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ nasm -f elf lab5-2-1.asm
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-2.o
ld: невозможно найти lab5-2-2.o: Нет такого файла или каталога
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $ ./lab5-2-1
Введите строку:
Чувакина Мария Владимировна
mvchuvakina@dk2n26 ~/work/study/2023-2024/Архитектура компьютера/arh-pc/lab05 $
```

Рис. 4.23: Исполнение файла

Код программы из пункта 3:

```
%include 'in_out.asm'
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 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,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,buf1 ; Адрес строки buf1 в ecx
```

**int** 80h ; Вызов ядра

**call** quit ; вызов подпрограммы завершения

## 5 Выводы

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

## **6 Список литературы**

1. Лабораторная работа №5