

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Краснопер Данила Олегович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Структура программы на языке ассемблера NASM	11
4.2	Подключение внешнего файла	14
4.3	Выполнение заданий для самостоятельной работы	17
5	Выводы	21

Список иллюстраций

4.1	Открытый тс	9
4.2	Перемещение между директориями	10
4.3	Создание каталога	10
4.4	Перемещение между директориями	11
4.5	Создание файла	11
4.6	Открытие файла для редактирования	12
4.7	Редактирование файла	12
4.8	Открытие файла для просмотра	13
4.9	Компиляция файла и передача на обработку компоновщику	13
4.10	Исполнение файла	13
4.11	Скачанный файл	14
4.12	Копирование файла	14
4.13	Копирование файла	15
4.14	Редактирование файла	15
4.15	Исполнение файла	16
4.16	Отредактированный файл	16
4.17	Исполнение файла	17
4.18	Копирование файла	17
4.19	Редактирование файла	18
4.20	Исполнение файла	18
4.21	Копирование файла	19
4.22	Редактирование файла	19
4.23	Исполнение файла	20

Список таблиц

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 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

Открываю Midnight Commander, введя в терминал mc (рис. 4.1).

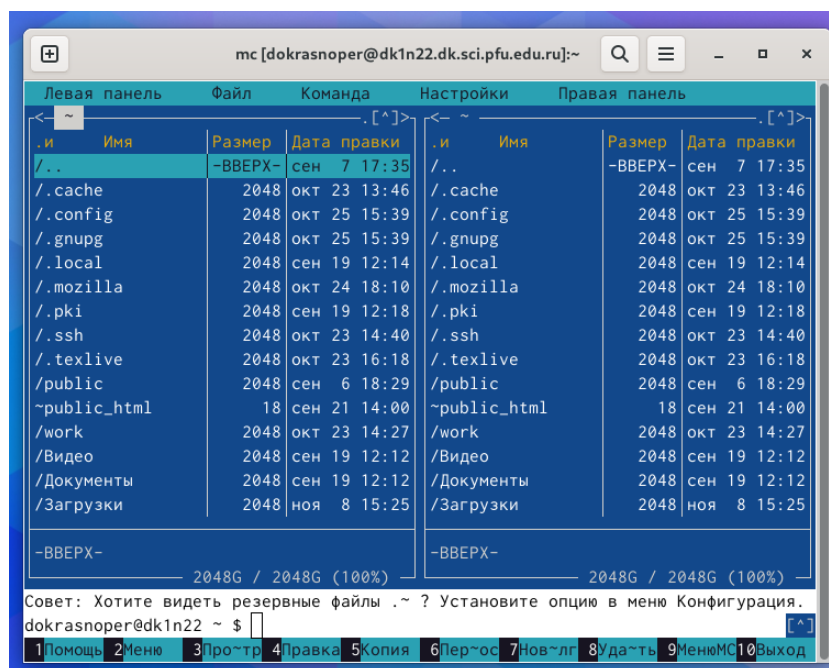


Рис. 4.1: Открытый mc

Перехожу в каталог ~/work/study/2024-2025/Архитектура Компьютера/arch-
rs, используя файловый менеджер mc (рис. 4.2)

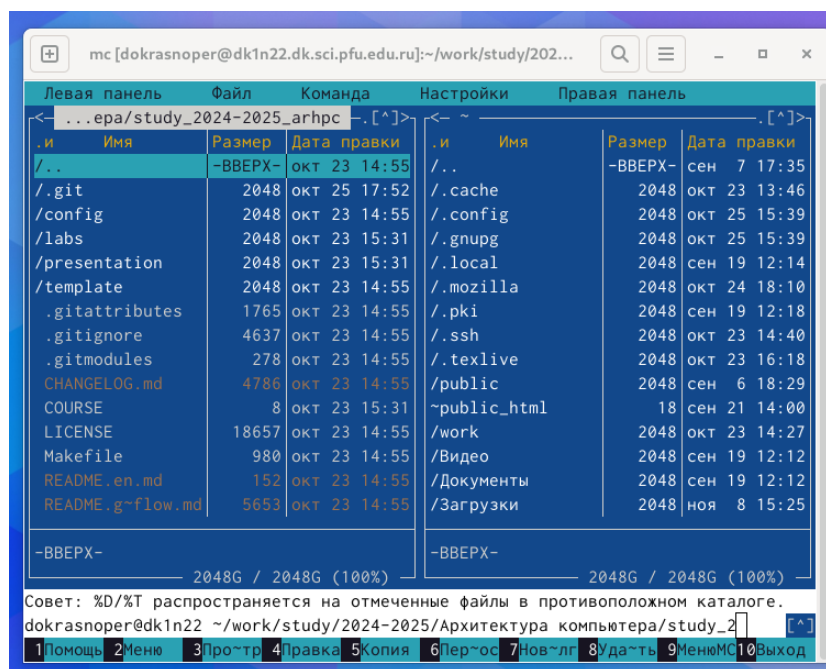


Рис. 4.2: Перемещение между директориями

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

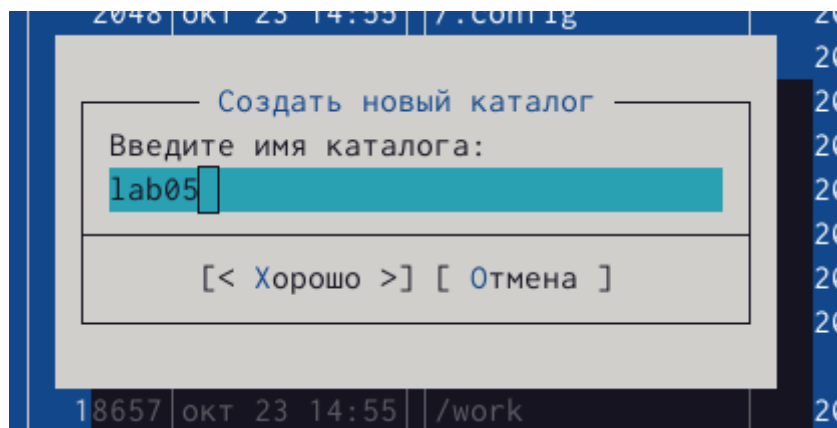


Рис. 4.3: Создание каталога

Переходу в созданный каталог (рис. 4.4).

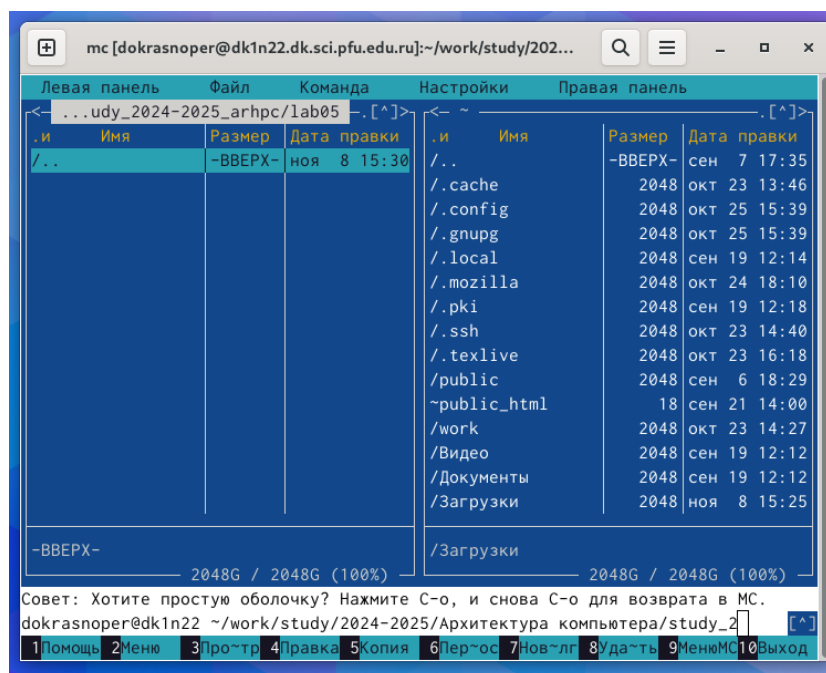


Рис. 4.4: Перемещение между директориями

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

.и	Имя	Размер	Дата правки
/..	-ВВЕРХ-	ноя 8 15:30	
lab5-1.asm	0	ноя 8 15:34	

Рис. 4.5: Создание файла

4.1 Структура программы на языке ассемблера NASM

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

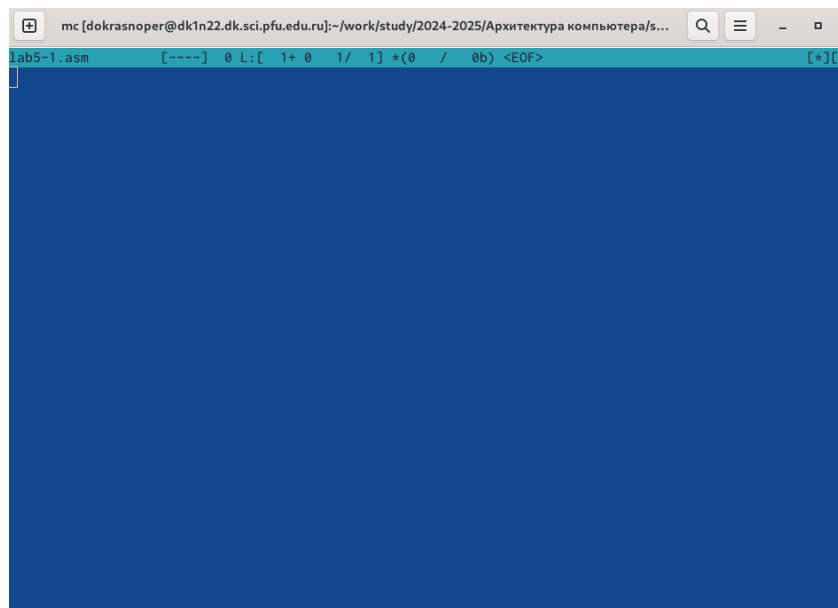


Рис. 4.6: Открытие файла для редактирования

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

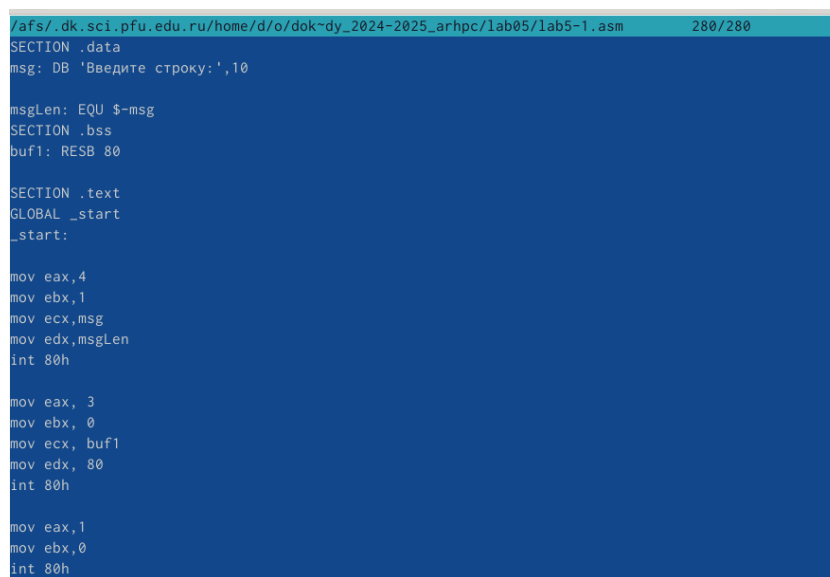


Рис. 4.7: Редактирование файла

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

```
..study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/lab05/lab5-1.asm
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

[ Прочитано 26 строк ]
^G Справка ^O Записать ^F Поиск ^K Вырезать ^T Выполнить M-U Отмена
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^C Позиция M-E Повтор
```

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`.

```
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/lab05 $ nasm -f elf lab5-1.asm
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/lab05 $
```

Рис. 4.9: Компиляция файла и передача на обработку компоновщику

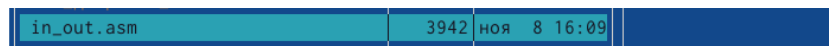
Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. 4.10).

```
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/lab05 $ ./lab5-1
Введите строку:
Краснопер Данила Олегович
```

Рис. 4.10: Исполнение файла

4.2 Подключение внешнего файла

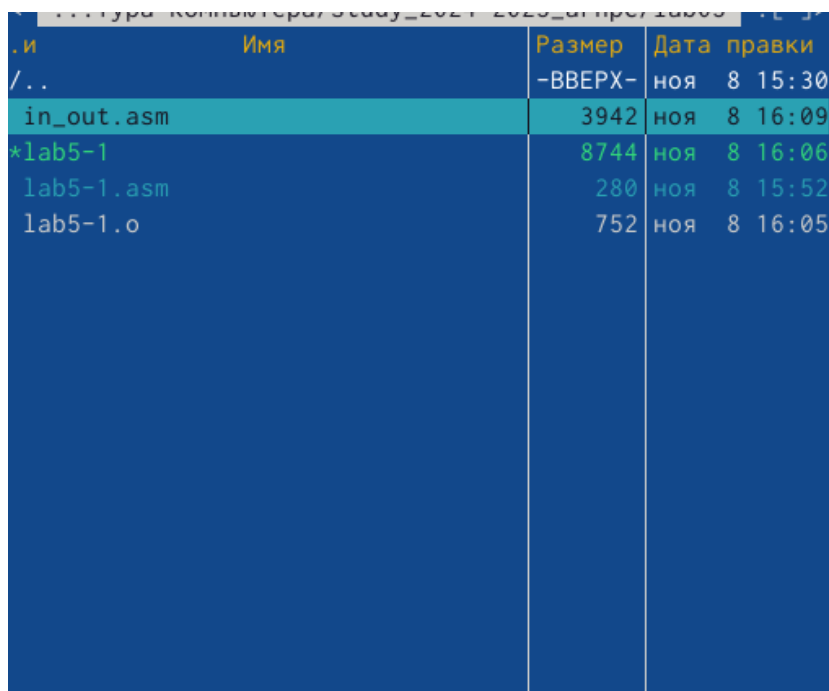
Скачиваю файл in_out.asm со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. 4.11).



in_out.asm	3942	ноя 8 16:09
------------	------	-------------

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).



Имя	Размер	Дата правки
in_out.asm	3942	ноя 8 16:09
*lab5-1	8744	ноя 8 16:06
lab5-1.asm	280	ноя 8 15:52
lab5-1.o	752	ноя 8 16:05

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).

Имя	Размер	Дата	Правки
in_out.asm	3942	ноя 8 15:30	
*lab5-1	8744	ноя 8 16:06	
lab5-1.asm	280	ноя 8 15:52	
lab5-1.o	752	ноя 8 16:05	
lab5-2.asm	280	ноя 8 15:52	

Рис. 4.13: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе mscedit (рис. 4.14), чтобы в программе использовались подпрограммы из внешнего файла in_out.asm.

```
; 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 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
```

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 объектно-

го 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).

```
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ nasm -f elf lab5-2.asm
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ./lab5-2
Введите строку:
Краснопер Данила Олегович
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $
```

Рис. 4.15: Исполнение файла

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

```
;------;
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;------;
%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 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
call quit ; вызов подпрограммы завершения
```

Рис. 4.16: Отредактированный файл

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


```
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ nasm -f elf lab5-2.asm
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ld -m elf_i386 -o lab5-2-2 lab5-2.o
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ./lab5-2-2
Введите строку: Краснопер Данила Олегович
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $
```

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`.

4.3 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).

.и	Имя	Размер	Дата правки	.и	Имя	Размер	Дата правки
./..		-ВВЕРХ-	ноя 8 15:30	./..		-ВВЕРХ-	ноя 8 15:30
	in_out.asm	3942	ноя 8 16:09		in_out.asm	3942	ноя 8 16:09
	*lab5-1	8744	ноя 8 16:06		*lab5-1	8744	ноя 8 16:06
	lab5-1-1.asm	280	ноя 8 15:52		lab5-1-1.asm	280	ноя 8 15:52
	lab5-1.asm	280	ноя 8 15:52		lab5-1.asm	280	ноя 8 15:52
	lab5-1.o	752	ноя 8 16:05		lab5-1.o	752	ноя 8 16:05
	*lab5-2	9092	ноя 8 16:31		*lab5-2	9092	ноя 8 16:31
	*lab5-2-2	9092	ноя 8 16:35		*lab5-2-2	9092	ноя 8 16:35
	lab5-2.asm	1223	ноя 8 16:33		lab5-2.asm	1223	ноя 8 16:33

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).

```

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).

```

dokrasnoper@edk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ nasm -f elf lab5-1-1.asm
dokrasnoper@edk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
dokrasnoper@edk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ./lab5-1-1
Введите строку:
Краснопер Данила Олегович
Краснопер Данила Олегович
dokrasnoper@edk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $

```

Рис. 4.20: Исполнение файла

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

lab5-1-1.o	784	ноя 8 16:47	lab5-1-1.o	784	ноя 8 16:47
lab5-1.asm	280	ноя 8 15:52	lab5-1.asm	280	ноя 8 15:52
lab5-1.o	752	ноя 8 16:05	lab5-1.o	752	ноя 8 16:05
*lab5-2	9092	ноя 8 16:31	*lab5-2	9092	ноя 8 16:31
lab5-2-1.asm	1223	ноя 8 16:33	lab5-2-1.asm	1223	ноя 8 16:33
*lab5-2-2	9092	ноя 8 16:35	*lab5-2-2	9092	ноя 8 16:35
lab5-2.asm	1223	ноя 8 16:33	lab5-2.asm	1223	ноя 8 16:33
lab5-2.asm.save	281	ноя 8 16:19	lab5-2.asm.save	281	ноя 8 16:19
lab5-2.o	1312	ноя 8 16:34	lab5-2.o	1312	ноя 8 16:34
lab5-2.asm			-BPEPX-		

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).

```

;-----
%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 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

```

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).

```
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ nasm -f elf lab5-2-1.asm
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $ ./lab5-2-1
Введите строку: Краснопер Данила Олегович
Краснопер Данила Олегович
dokrasnoper@dk1n22 ~/work/study/2024-2025/Архитектура компьютера/study_2024-2025
_arhpc/lab05 $
```

Рис. 4.23: Исполнение файла

5 Выводы

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