

# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

## ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

дисциплина: Архитектура компьютера

Студент: Парчиев Султан Багаудинович

Группа: НММ бд-03-24

МОСКВА

2024 г.

# Содержание

<b>1 Цель работы</b>	<b>4</b>
<b>2 Задание</b>	<b>5</b>
<b>3 Теоретическое введение</b>	<b>6</b>
<b>4 Выполнение лабораторной работы</b>	<b>7</b>
<b>5 Выводы</b>	<b>16</b>
<b>Список литературы</b>	<b>17</b>

## Список иллюстраций

<b>4.1 Открытый тс</b>	<b>7</b>
<b>4.2 Перемещение между директориями</b>	<b>8</b>
<b>4.3 Создание каталога</b>	<b>8</b>
<b>4.4 Перемещение между директориями</b>	<b>9</b>
<b>4.5 Созданный файл</b>	<b>9</b>
<b>4.6 Открытие файла для редактирования</b>	<b>9</b>
<b>4.7 Редактирование файла</b>	<b>10</b>
<b>4.8 Открытие файла для просмотра</b>	<b>10</b>
<b>4.9 Копирование файла</b>	<b>11</b>
<b>4.10 Редактирование файла</b>	<b>12</b>
<b>4.11 Исполнение файла</b>	<b>12</b>
<b>4.12 Исполнение файла</b>	<b>13</b>
<b>4.13 Копирование файла</b>	<b>13</b>
<b>4.14 Отредактированный файл</b>	<b>14</b>
<b>4.15 Исполнение файла</b>	<b>14</b>
<b>4.16 Отредактированный файл</b>	<b>15</b>
<b>4.17 Исполнение файла</b>	<b>15</b>

# 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-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).

# 4 Выполнение лабораторной работы

## 4.1 Основы работы с mc

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

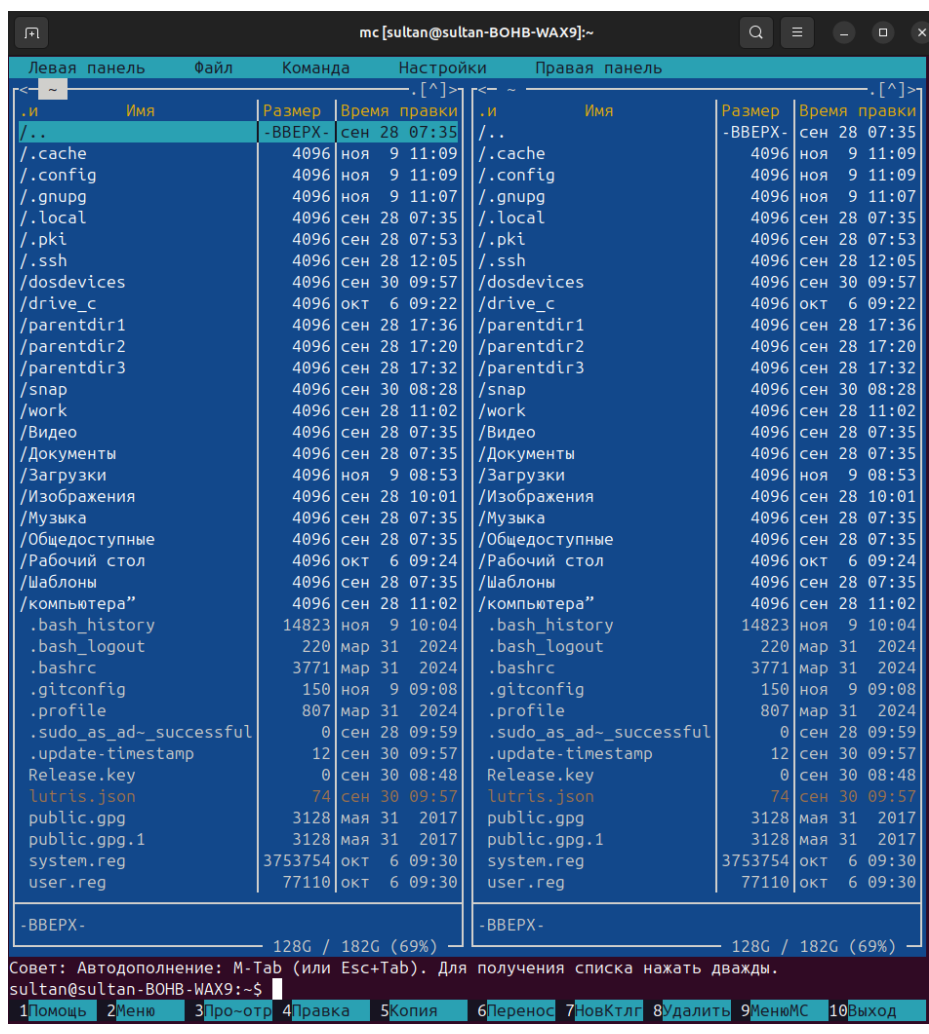


Рис. 4.1: Открытый mc

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

...5/Архитектура компьютера/arch-рс			
Имя	Размер	Время правки	
/..	-ВВЕРХ-	ноя 9	09:54
/.git	4096	ноя 9	10:03
/config	4096	ноя 1	11:32
/labs	4096	ноя 9	09:54
/presentation	4096	ноя 9	09:54
/template	4096	ноя 1	11:32
.gitattributes	1765	ноя 1	11:32
.gitignore	4637	ноя 1	11:32
.gitmodules	278	ноя 1	11:32
CHANGELOG.md	4786	ноя 1	11:32
COURSE	8	ноя 9	09:55
LICENSE	18657	ноя 1	11:32
Makefile	980	ноя 1	11:32
README.en.md	152	ноя 1	11:32
README.git-flow.md	5653	ноя 1	11:32
README.md	4304	ноя 1	11:32
prepare	0	ноя 9	09:54

Рис. 4.2: Перемещение между директориями

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

Создать новый каталог

Введите имя каталога:

lab05

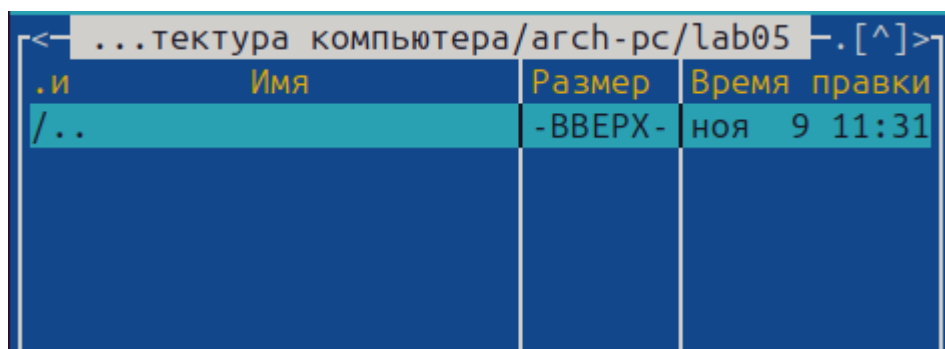
[< Хорошо >]

[Отмена]



Рис. 4.3: Создание каталога

Перехожу в созданный каталог (рис. 4.4).

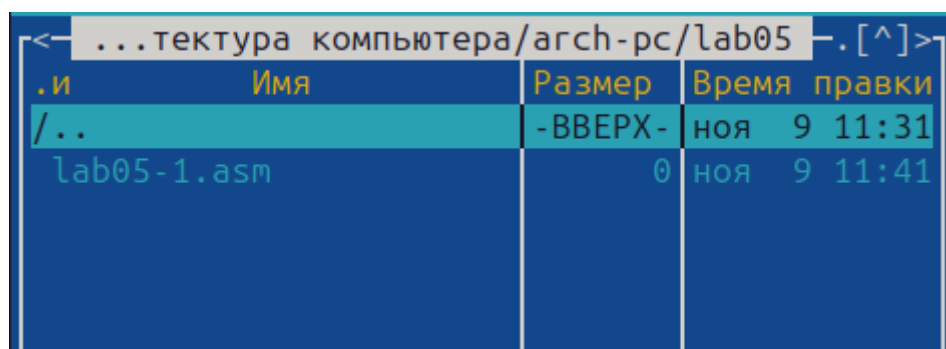


A terminal window showing a directory listing. The prompt is `<- ...текстура компьютера/arch-pc/lab05 -.[^]>`. The listing shows a directory `/..` with size `-ВВЕРХ-` and modification time `ноя 9 11:31`.

.и	Имя	Размер	Время правки
/..		-ВВЕРХ-	ноя 9 11:31

Рис. 4.4: Перемещение между директориями

В строке ввода прописываю команду `touch lab05-1.asm`, чтобы создать файл, в котором буду работать (рис. 4.5).



A terminal window showing a directory listing after creating a file. The prompt is `<- ...текстура компьютера/arch-pc/lab05 -.[^]>`. The listing shows the directory `/..` and a new file `lab05-1.asm` with size `0` and modification time `ноя 9 11:41`.

.и	Имя	Размер	Время правки
/..		-ВВЕРХ-	ноя 9 11:31
lab05-1.asm		0	ноя 9 11:41

Рис. 4.5: Созданный файл

## 4.2 Структура программы на языке ассемблера NASM

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

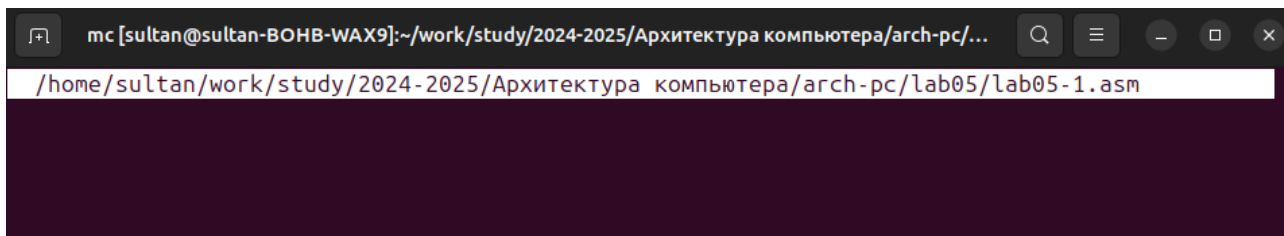


Рис. 4.6: Открытие файла для редактирования

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

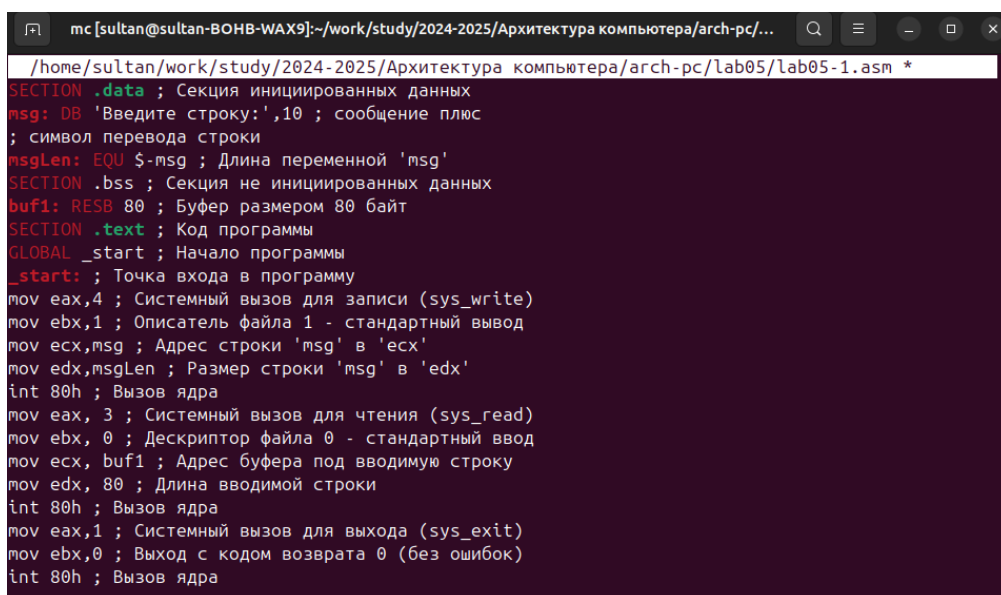


Рис. 4.7: Редактирование файла

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

```
mc [sultan@sultan-BOHB-WAX9]:~/work/study/2024-2025/Архитектура компьютера/arch-pc/...
/home/sultan/work/study/2024-2-epa/arch-pc/lab05/lab05-1.asm 1315/1315 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 ; Вызов ядра
```

Рис. 4.8: Открытие файла для просмотра

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab05-1.asm`. Создался объектный файл `lab05-1.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab05-1 lab05-1.o`. Создался исполняемый файл `lab05-1`.

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

### 4.3 Подключение внешнего файла

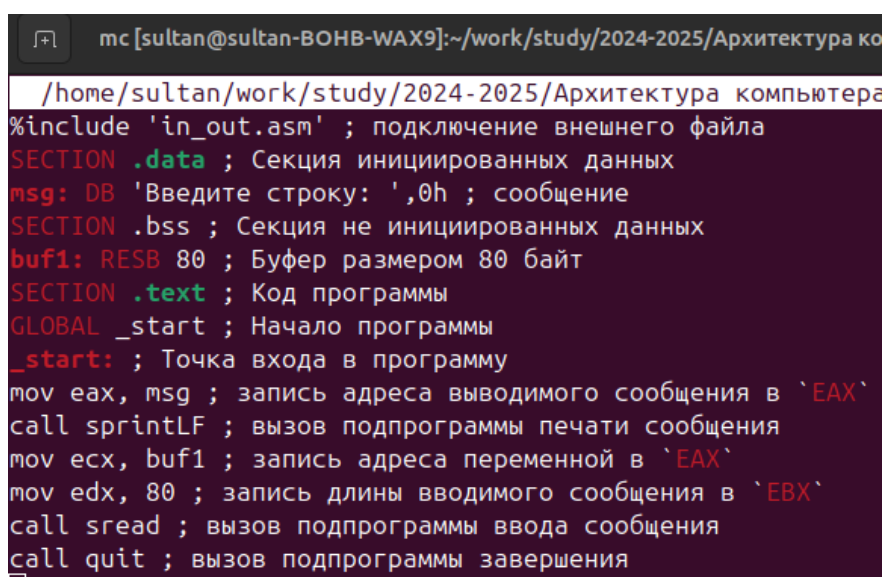
Скачиваю файл `in_out.asm` со страницы курса в ТУИС. С помощью функциональной клавиши F5 копирую файл `in_out.asm` из каталога Загрузки в созданный каталог `lab05` (рис. 4.9).

Левая панель	Файл	Команда	Настройка
...тектура компьютера/arch-pc/lab05			
.и	Имя	Размер	Время правки
/..		-ВВЕРХ-	ноя 9 11:31
	in_out.asm	3942	ноя 9 08:53
	*lab05-1	8748	ноя 9 12:08
	lab05-1.asm	1315	ноя 9 11:55
	lab05-1.o	752	ноя 9 12:05

Рис. 4.9: Копирование файла

С помощью функциональной клавиши F5 копирую файл lab6-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя для копии файла.

Изменяю содержимое файла lab6-2.asm во встроенном редакторе nano (рис. 4.10), чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.



```
mc [sultan@sultan-BOHB-WAX9]:~/work/study/2024-2025/Архитектура ко
/home/sultan/work/study/2024-2025/Архитектура компьютера
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку: ',0h ; сообщение
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, 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 sread ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 4.10: Редактирование файла

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab05-2.asm`. Создался объектный файл lab05-2.o. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab05-2 lab05-2.o` Создался исполняемый файл lab05-2. Запускаю исполняемый файл (рис.4.11).

```
sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab05-2.asm

sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab05-2 lab05-2.o

sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ./lab05-2
Введите строку:
Парчиев Султан Багаудинович
```

Рис. 4.11: Исполнение файла

Открываю файл lab6-2.asm для редактирования в nano функциональной клавишей F4. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения. Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. 4.12).

```
sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab05-2-2 lab05-2.o

sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ./lab05-2-2
Введите строку:
Парчиев Султан Багаудинович
```

Рис. 4.12: Исполнение файла

Разница между первым исполняемым файлом lab05-2 и вторым lab05-2-2 в том, что запуск первого запрашивает ввод с новой строки, а программа, которая выполняется 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 lab05-1.asm с именем lab05-1-1.asm с помощью функциональной клавиши F5 (рис. 4.13).

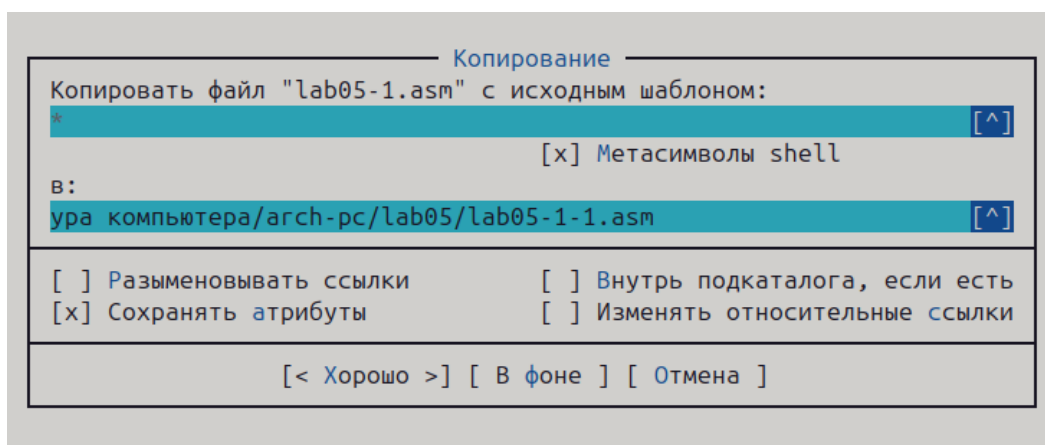
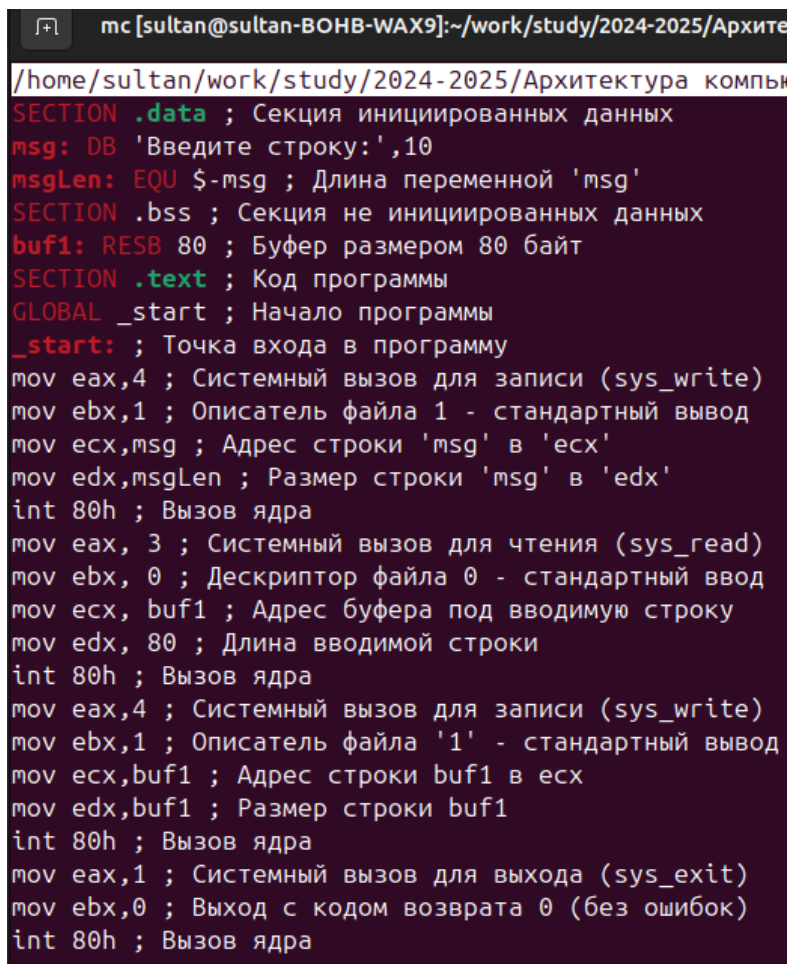


Рис. 4.13: Копирование файла

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. 4.14).



```
mc [sultan@sultan-BOHB-WAX9]:~/work/study/2024-2025/Архитектура компь
/home/sultan/work/study/2024-2025/Архитектура компь
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
```

Рис. 4.14: Отредактированный файл

2. Создаю объектный файл lab05-1-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab05-1-1, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. 4.15).

```
sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab05-1-1.asm

sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab05-1-1 lab05-1-1.o

sultan@sultan-BOHB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ./lab05-1-1
Введите строку:
Парчиев Султан Багаудинович
Парчиев Султан Багаудинович
```

Рис. 4.15: Исполнение файла

3. Создаю копию файла lab05-2.asm с именем lab05-2-1.asm с помощью функциональной клавиши F5. 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. 4.16).

```
/home/sultan/work/study/2024-2025/Архитектура компьютера/a
%include 'in_out.asm'
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку: ',0h ; сообщение
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, 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. 4.16: Отредактированный файл

4. Создаю объектный файл lab05-2-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab05-2-1, запускаю полученный исполняемый файл.

Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее

прог  
рамм  
а  
ВЫВО  
ДИТ  
ВВЕД

```
sultan@sultan-B0NB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab05-2-1.asm  
sultan@sultan-B0NB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab05-2-1 lab05-2-1.o  
sultan@sultan-B0NB-WAX9:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab05$ ./lab05-2-1  
Введите строку: Парчиев Султан Багаудинович  
Парчиев Султан Багаудинович
```

енные мною данные (рис. 4.17).

Рис. 4.17: Исполнение файла

## 5 Выводы

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



## **Список литературы**

### **1. Архитектура ЭВМ**