

# **Отчёт по лабораторной работе**

**Дисциплина: архитектура компьютера**

**Назармамадов Умед Джамshedович**

# Содержание

- 1 Цель работы
- 2 Задание
- 3 Теоретическое введение
- 4 Выполнение лабораторной работы
- 5 Выводы

Список литературы

## Список иллюстраций

4.1	Midnight commander . . . . .	9
4.2	Перемещение между директориями . . . . .	10
4.3	Перемещение между директориями . . . . .	10
4.4	Создание файла . . . . .	10
4.5	Открытие файла . . . . .	11
4.6	Редактирование файла . . . . .	11
4.7	Открытие файла . . . . .	12
4.8	Компиляция файла . . . . .	12
4.9	Запуск файла . . . . .	12
4.10	Скачивание файла . . . . .	13
4.11	Копирование файла . . . . .	13
4.12	Копирование файла . . . . .	13
4.13	Редактирование файла . . . . .	14
4.14	Исполнение файла . . . . .	14
4.15	Исполнение файла . . . . .	14
4.16	Отредактирование файла . . . . .	15
4.17	Запуск файла . . . . .	15

# 1 Цель работы

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

## 2 Задание

1. Основы работы с тс
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 инициализированных (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 инициализированных 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 (четверное слово);
- DT (define ten bytes) — определяет переменную размером в 10 байт.

Директивы используются для объявления простых переменных и для объявления массивов.

вов. Для определения строк принято использовать директиву DB в связи с особенностями хранения данных в оперативной памяти. Синтаксис директив определения данных следующий: DB [ , ] [ , ]

Для объявления неиницированных данных в секции .bss используются директивы resb,

resw, resd и другие, которые сообщают ассемблеру, что необходимо зарезервировать за- данное количество ячеек памяти.

## 4 Выполнение лабораторной работы

Открываю Midnight Commander, введя в терминал mc.

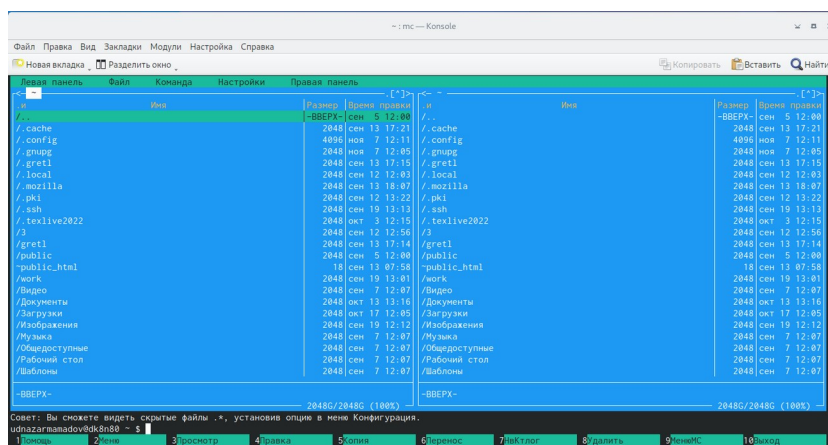


Рис. 4.1: Midnight commander

Перехожу в каталог `~/work/study/2022-2023/Архитектура Компьютера/arch-рс`, используя файловый менеджер mc.



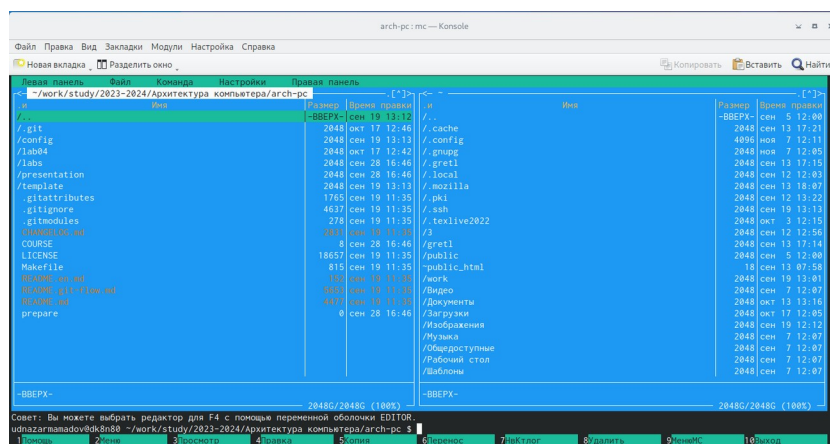


Рис. 4.2: Перемещение между директориями

С помощью функциональной клавиши F7 создаю каталог lab05 и перехожу в созданный каталог.

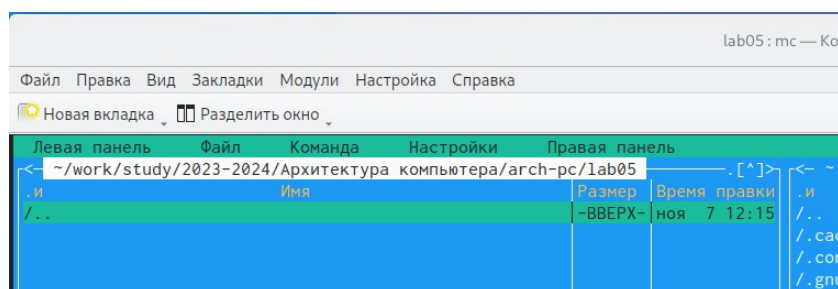


Рис. 4.3: Перемещение между директориями

Прописываю команду touch lab5.asm, чтобы создать файл, в котором буду работать.

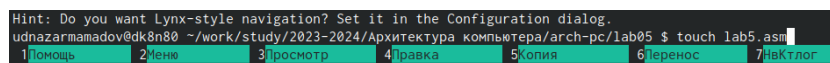


Рис. 4.4: Создание файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano.

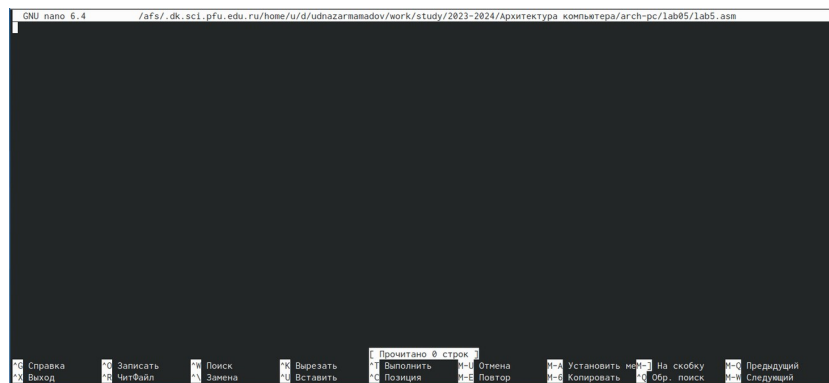


Рис. 4.5: Открытие файла

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

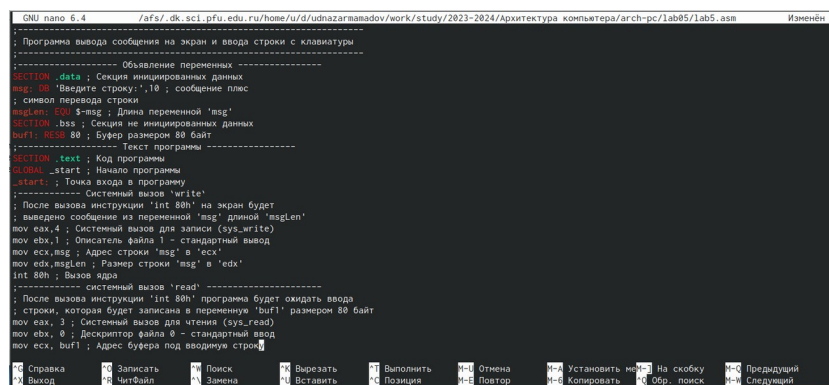


Рис. 4.6: Редактирование файла

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

```

~/fs/~/dk.sci.pfu.edu.ru/home/u/d/udnazaromadov/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05/lab5.asm 2071/2432
-----
Программа вывода сообщения на экран и ввода строки с клавиатуры
-----
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ', 10 ; сообщение плюс
; символ перевода строки
msglen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buff1: RESB 80 ; Буфер размером 80 байт
----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write' -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msglen'
mov eax, 4 ; Системный вызов для записи (sys_write)
mov ebx, 1 ; Файловый дескриптор stdout
mov ecx, msg ; Адрес строки 'msg' в 'ecx'
mov edx, msglen ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- Системный вызов 'read' -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buff1' размером 80 байт
mov eax, 3 ; Системный вызов для чтения (sys_read)
mov ebx, 0 ; Дескриптор файла stdin - стандартный ввод
mov ecx, buff1 ; Адрес буфера под вводную строку
mov edx, 80 ; Длина вводной строки
int 80h ; Вызов ядра
-----
1.Консоль 2.Загрузка 3.Выход 4.Помощь 5.Перейти 6. 7.Список 8.Скорректировать 9.Формат 10.Выход

```

Рис. 4.7: Открытие файла

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5.asm`. Создался объектный файл `lab5.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5 lab5.o`.

```

$ nasm -f elf lab5.asm
$ ld -m elf_i386 -o lab5 lab5.o

```

Рис. 4.8: Компиляция файла

Запускаю исполняемый файл.

```

udnazaromadov@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05 $ ./lab5
Введите строку:
Назармадов Умед Джамшедович

```

Рис. 4.9: Запуск 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`

Имя	Размер	Время правки
in_out.asm	3942	ноя 7 12:21

Рис. 4.10: Скачивание файла

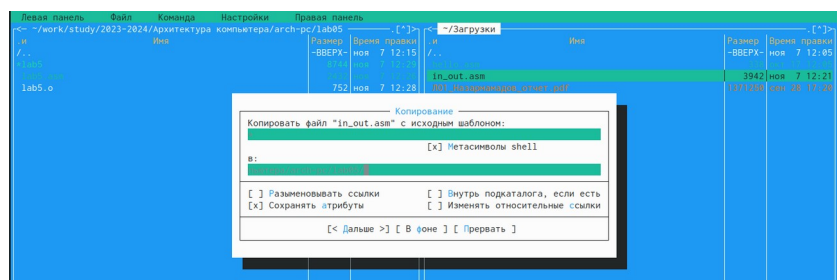


Рис. 4.11: Копирование файла

С помощью функциональной клавиши F5 копирую файл lab5 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя для копии файла

Имя	Размер	Время правки
in_out.asm	3942	ноя 7 12:21
lab5	8744	ноя 7 12:29
lab5-2.asm	2432	ноя 7 12:28
lab5.asm	2432	ноя 7 12:28
lab5.o	752	ноя 7 12:28

Рис. 4.12: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.

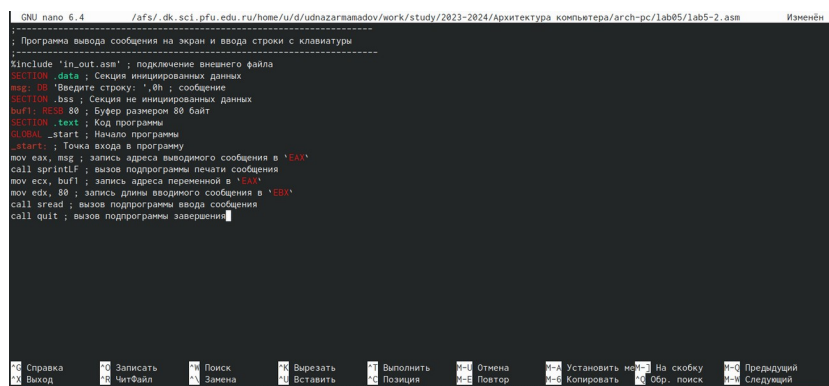


Рис. 4.13: Редактирование файла

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

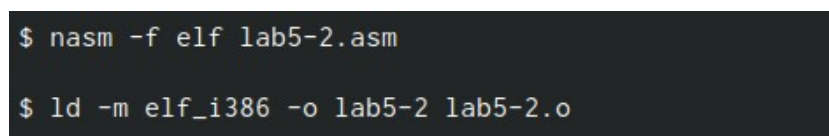


Рис. 4.14: Исполнение файла

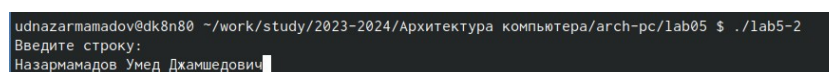


Рис. 4.15: Исполнение файла

Открываю файл `lab5-2.asm` для редактирования в `nano` функциональной клавишей F4. Изменяю в нем подпрограмму `sprintf` на `sprint`. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий

```

lab5-2.asm      [-M--] 41 L:[ 1+16 17/ 19] *(1222/1224b) 0010 0x00A
; ~~~~~
; 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 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: Отредактированное файла

Запускаю новый исполняемый файл.

```

udnazarmamadov@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05 $ nasm -f elf lab5-2.asm
udnazarmamadov@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-2 lab5-2.o
udnazarmamadov@dk8n80 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05 $ ./lab5-2-2
Введите строку: Назармадов Умед Джамшедов:

```

Рис. 4.17: Запуск файла

## 5 Выводы

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

# Список литературы

1. Лабораторная работа №5