

Отчёт по лабораторной работе №5

Дисциплина: архитектура компьютера

Худдыева Дженнет

Содержание

| | |
|---|----|
| 1. Цель работы | 4 |
| 2. Задание | 5 |
| 3. Теоретическое введение | 6 |
| 4. Выполнение лабораторной работы | 8 |
| 4.1 Основы работы с mc | 8 |
| 4.2 Структура программы на языке ассемблера NASM | 10 |
| 4.3 Подключение внешнего файла | 11 |
| 4.4 Выполнение заданий для самостоятельной работы | 14 |
| 5. Выводы | 19 |
| 6. Список литературы | 20 |

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 .txt`) секция инициализированных (известных во время компиляции) данных (`SECTION .data`) и секция не инициализированных 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`, `DD`, `DQ`, `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 байта (четверное слово) `-DT` (define ten bytes) - определяет переменную размером в 10 байта. 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`

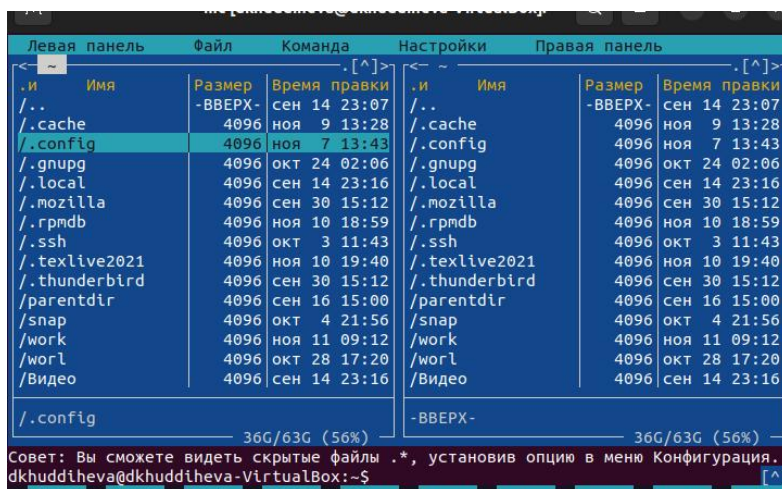
Здесь операнд `dst` - приёмник, а `src` - источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непосредственные значения (const). Инструкция языка ассемблера `int` предназначена для вызова прерывания с указанным номером.

`int n`

Здесь n-номер прерывания, принадлежащий диапазону 0-255. При программировании в Linux с использованием ядра sys_calls $n=80h$ (принято задавать в шестнадцатеричной системе счисления)

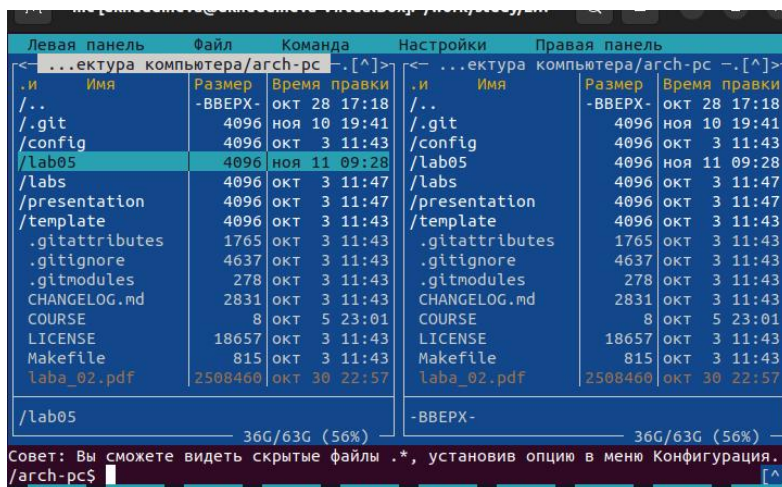
4 Выполнение лабораторной работы

Открываю Midnight Commander, введя в терминал `mc`



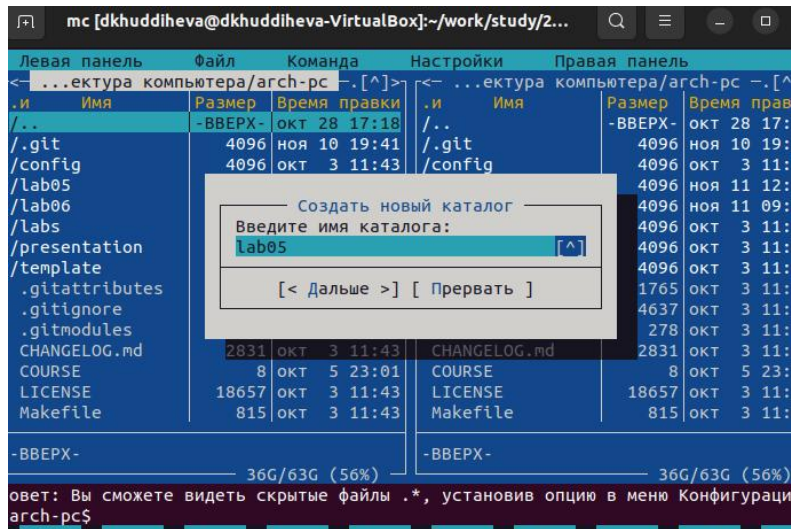
Открытый `mc`

Перехожу в каталог `~/work/study/2023-2024/Архитектура компьютера/arch-pc`, используя файловой менеджер `mc`



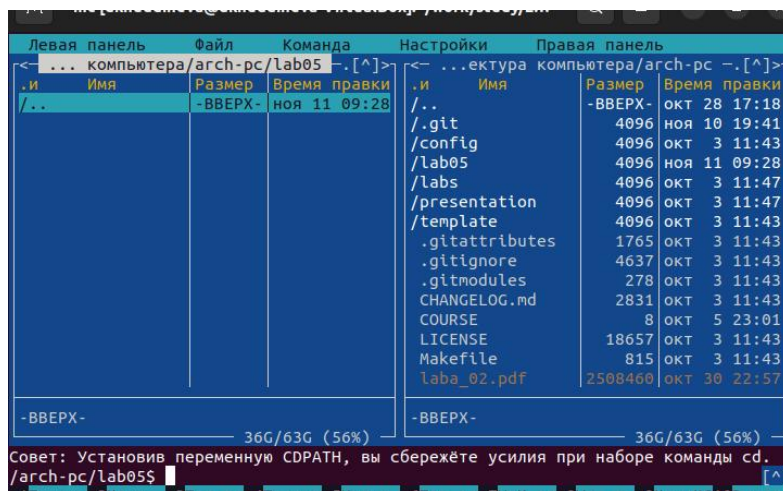
Перемещение между директориями

С помощью функциональной клавиши `F7` создаю каталог `lab05`



Создание каталога

Перехожу в созданный каталог



Перемещение между директориями

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

4.1 Структура программы на языке ассемблера NASM

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano. Ввожу в файл код программы для запроса строки у пользователя. Далее выхожу из файла (Ctrl+X), сохраняя изменения

```

GNU nano 6.2 /home/dkhuddheva/work/study/2023-2024/Архитектура компь
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 .txt ;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

```

Редактирование файла

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

```

/home/dkhuddheva/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05/lab5-1.asm 1614/1614
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 .txt ;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

```

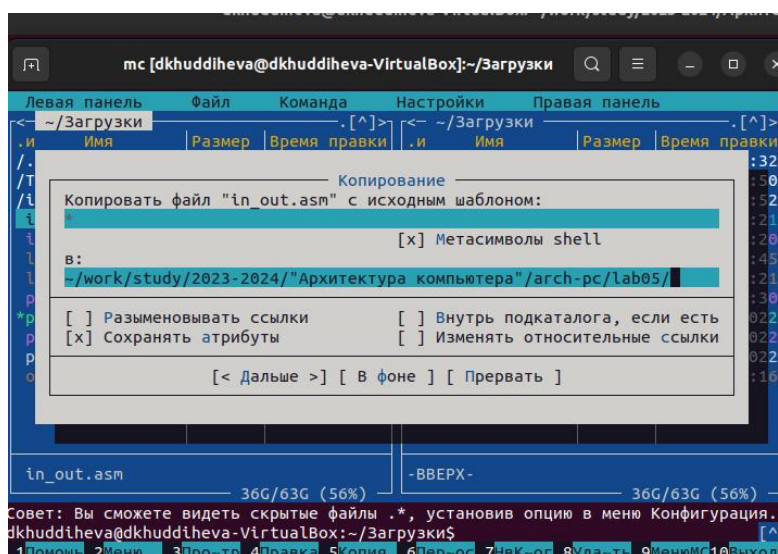
Открытие файла для просмотра

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`. Создался исполняемый файл `lab5-1`. Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждёт ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу


```
dkhuddiheva@dkhuddiheva-VirtualBox:~$ cd ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ mc
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab5-1.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ./lab5-1
Введите строку:
Худдыева Дженнет
```

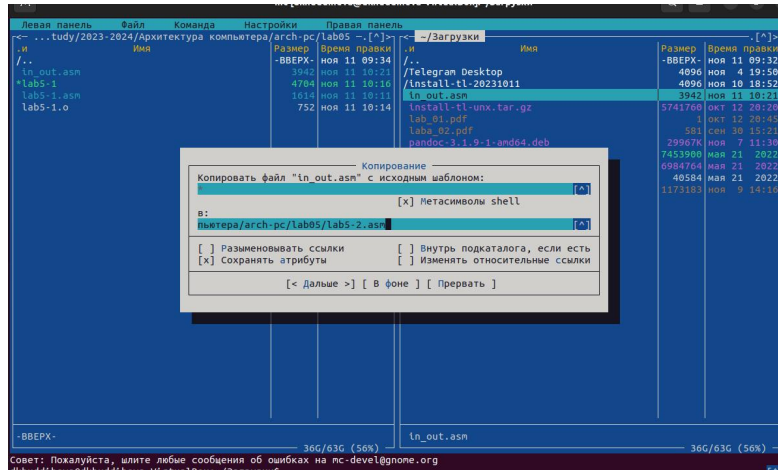
4.2 Подключение внешнего файла

Скачиваю файл in_out.asm со страницы в ТУИС. Он сохранился в каталоге “Загрузки” С помощью функциональной клавиши F5 копирую файл in_out.asm из каталога Загрузки в созданный каталог lab05



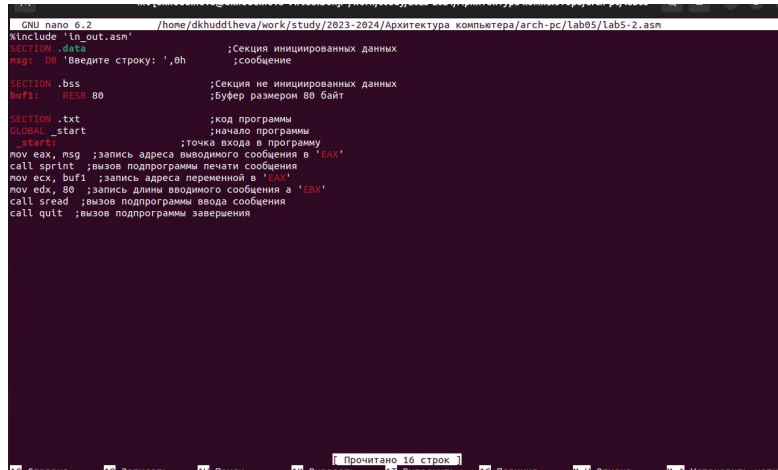
Копирование файла

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



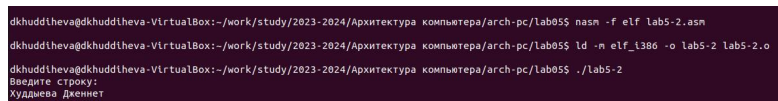
Копирование файла

Изменяю содержимого файла lab5-2.asm во встроенном редакторе nano



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`



Исполнение файла

Открываю файл `lab5-2.asm` для редактирования в nano функциональной клавиши F4. Изменяю в нём подпрограмму `sprintLF` на `sprint`. сохраняю файл для просмотра, чтобы проверить сохранение действий


```
mc [dkhuddiheva@dkhuddiheva-VirtualBox]:~/work/study/2...
/home/dkhuddiheva/work/~ch-pc/lab05/lab5-2.asm 1037/1037
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h ;Секция иницированных данных
;сообщение
SECTION .bss
buf1: RESB 80 ;Секция не иницированных данных
;Буфер размером 80 байт
SECTION .txt
GLOBAL _start ;код программы
_start: ;начало программы
;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
```

Отредактированный файл

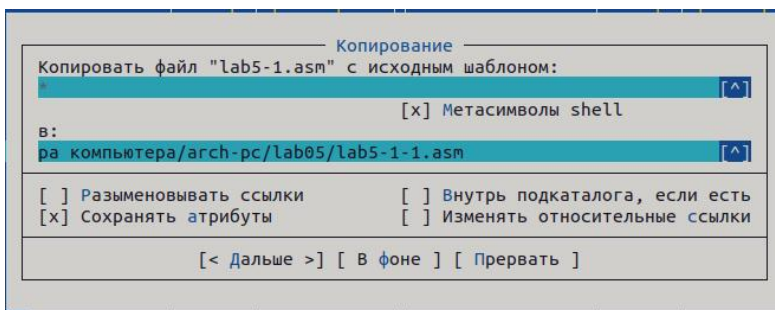
Снова транслирую файл,выполняю компоновку созданного объектного файла,запускаю новый исполняемый файл

```
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab5-2.asm
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ld -n elf_386 -o lab5-2-2 lab5-2.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ./lab5-2-2
Введите строку: Худдиева Дженет
```

Исполнение файла

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



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

```

GNU nano 6.2 /home/dkhuddiheva/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$
SECTION .data ;Секция инициализированных данных
msg: DB 'Введите строку:',10 ;сообщение плюс

msgLen: EQU $-msg ;длина переменной 'msg'

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,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

```

Редактирование файла

2.Создаю объектный файл lab5-1-1.o,отдаю его на обработку компоновщику,получаю исполняемый файл.Программа запрашивает ввод, ввожу свои ФИО,далее программа выводит введенные мною данные

```

dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
lab5-1-1.asm:23: error: parser: instruction expected
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ld -n elf_386 -o lab5-1-1 lab5-1.o
dkhuddiheva@dkhuddiheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Худячева Дхенне

```

Исполнение файла

Код программы из пункта 1:

```

SECTION .data ;Секция инициализированных данных

msg: DB 'Введите строку:',10

msglen: EQU $-msg ;Длина переменной 'msg'
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,4
mov ebx,1
mov ecx,msg
mov edx,msgLen

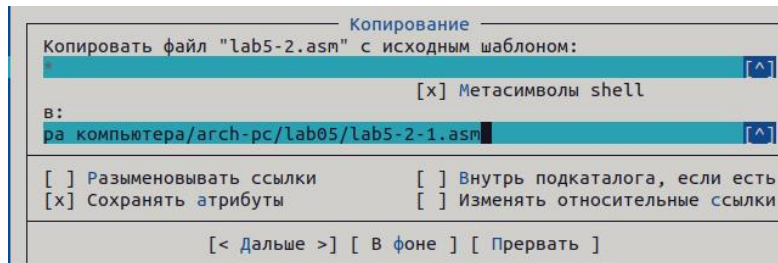
```

```

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

```

3.Создаю копии файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5



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

```

GNU nano 6.2 /home/dkhuddiheva/work/study/2023-2024/Архитект
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h ;Секция инициированных данных ;сообщение

SECTION .bss
buf1: RESB 80 ;Секция не инициированных данных ;Буфер размером 80 байт

SECTION .txt
GLOBAL _start
_start: ;код программы ;начало программы ;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
mov ebx, 1
mov ecx, buf1
int 80h
call quit ;вызов подпрограммы завершения

```

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

```
dkhuddheva@dkhuddheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ nasm -f elf lab5-2-1.asm
dkhuddheva@dkhuddheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
dkhuddheva@dkhuddheva-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab05$ ./lab5-2-1
Введите строку: Худяева Дженет
```

Исполнение файла

5 Выводы

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

6 Список литературы

Лабораторная работа №6