

# **Отчёт по лабораторной работе №5**

**Дисциплина: архитектура компьютера**

Черкашина Ангелина Максимовна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
4.1	Основы работы с тс . . . . .	9
4.2	Структура программы на языке ассемблера NASM . . . . .	12
4.3	Подключение внешнего файла . . . . .	14
4.4	Выполнение заданий для самостоятельной работ . . . . .	19
<b>5</b>	<b>Выводы</b>	<b>24</b>
<b>6</b>	<b>Список литературы</b>	<b>25</b>

## Список иллюстраций

4.1	Открытый Midnight Commander . . . . .	9
4.2	Перемещение между директориями . . . . .	10
4.3	Создание каталога . . . . .	11
4.4	Перемещение между директориями . . . . .	11
4.5	Создание файла . . . . .	12
4.6	Открытие файла lab5-1.asm для редактирования . . . . .	12
4.7	Редактирование файла lab5-1.asm . . . . .	13
4.8	Открытие файла lab5-1.asm для просмотра . . . . .	13
4.9	Компиляция файла и передача на обработку компоновщику . . . .	14
4.10	Исполнение файла . . . . .	14
4.11	Скачанный файл in_out.asm . . . . .	14
4.12	Копирование файла в нужный каталог . . . . .	15
4.13	Копирование файла с другим именем . . . . .	16
4.14	Редактирование файла lab5-2.asm . . . . .	17
4.15	Компиляция, компоновка и исполнение файла . . . . .	17
4.16	Изменение подпрограммы программы файла lab5-2.asm . . . . .	18
4.17	Просмотр отредактированного файла lab5-2.asm . . . . .	18
4.18	Исполнение отредактированного файла . . . . .	19
4.19	Копирование файла с новым именем . . . . .	19
4.20	Редактирование файла lab5-1-1.asm . . . . .	20
4.21	Просмотр отредактированного файла lab5-1-1.asm . . . . .	21
4.22	Исполнение файла lab5-1-1 . . . . .	21
4.23	Копирование файла с новым именем . . . . .	22
4.24	Редактирование файла lab5-2-1.asm . . . . .	22
4.25	Просмотр отредактированного файла lab5-2-1.asm . . . . .	23
4.26	Исполнение файла lab5-2-1 . . . . .	23

## 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 тс
2. Структура программы на языке ассемблера NASM
3. Подключение внешнего файла
4. Выполнение заданий для самостоятельной работ

### 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).





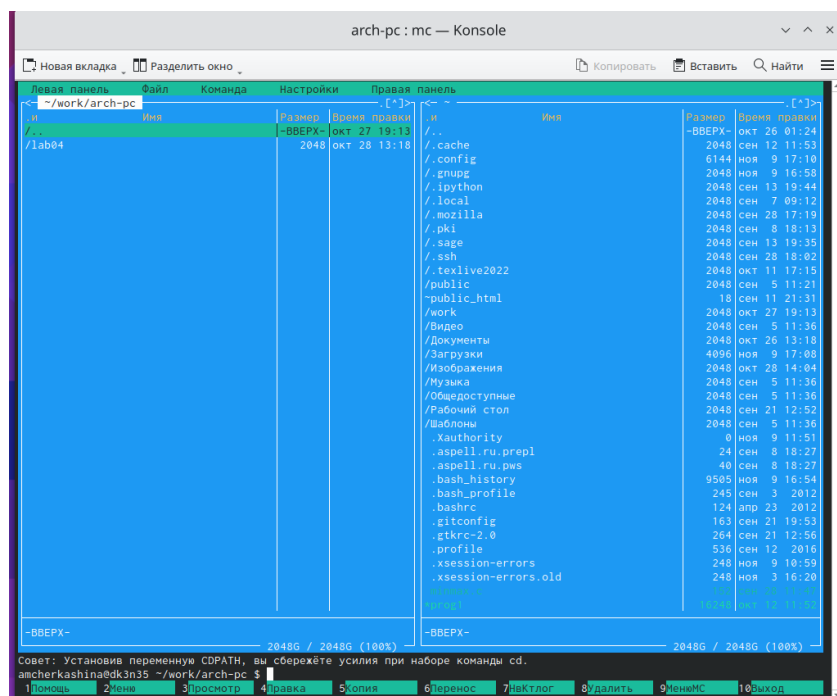


Рис. 4.2: Перемещение между директориями

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

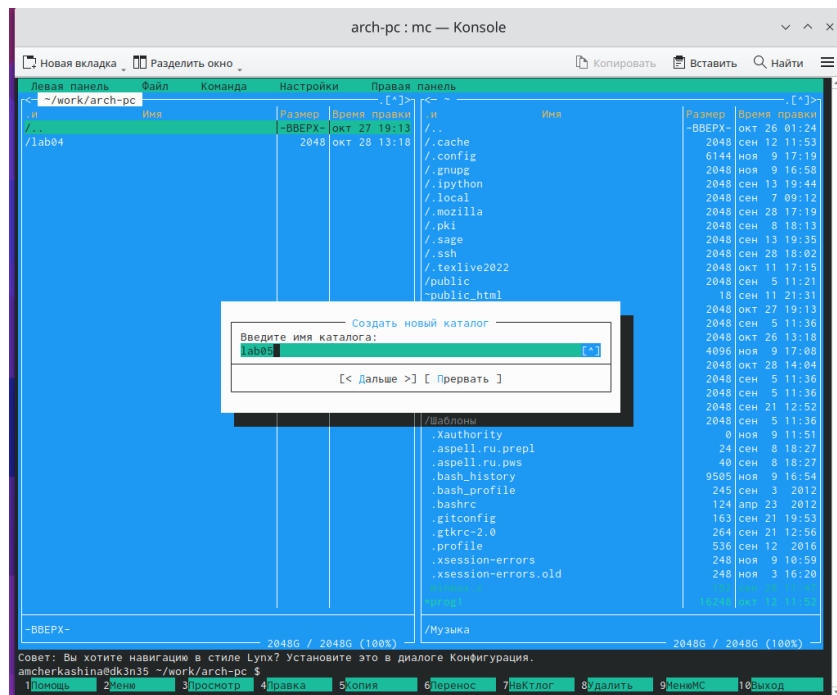


Рис. 4.3: Создание каталога

Перехожу в созданный каталог(рис. 4.4).

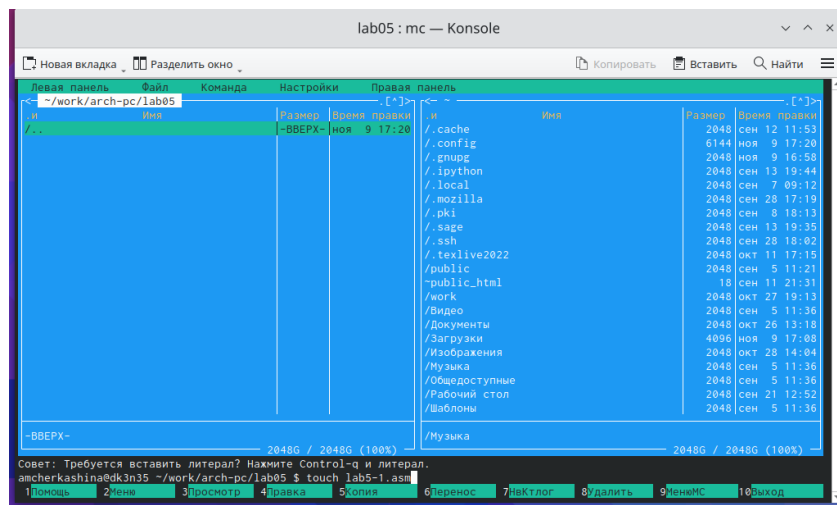


Рис. 4.4: Перемещение между директориями

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

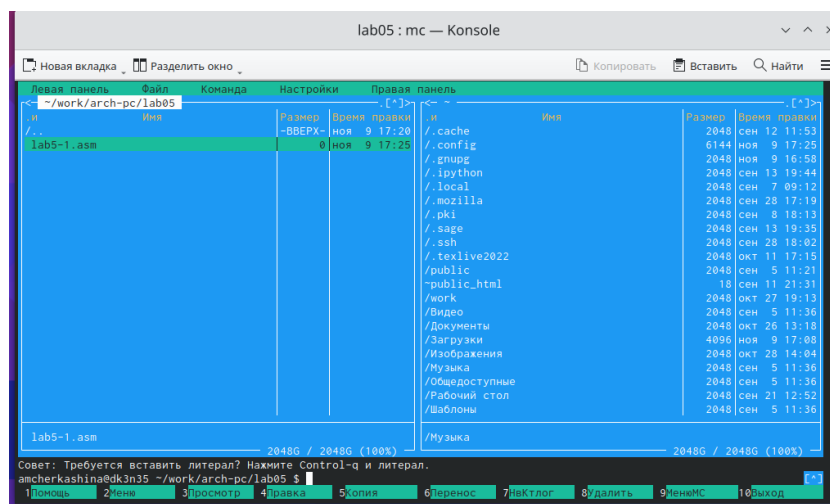


Рис. 4.5: Создание файла

## 4.2 Структура программы на языке ассемблера NASM

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

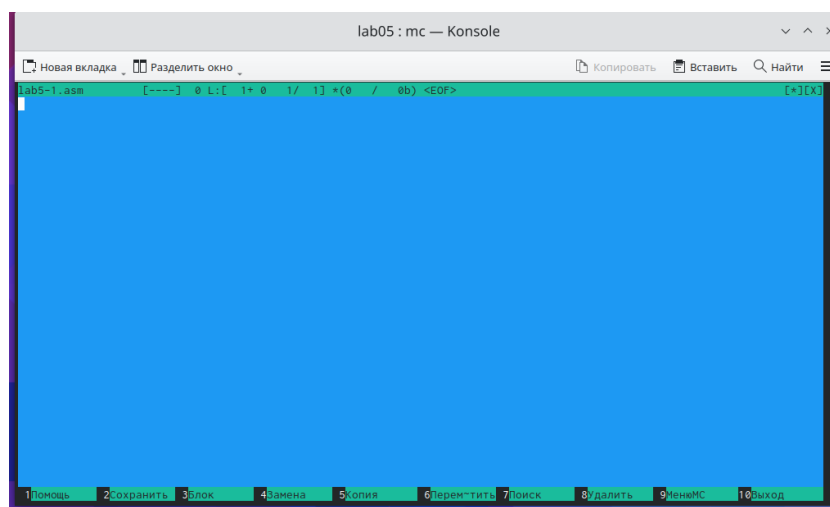


Рис. 4.6: Открытие файла lab5-1.asm для редактирования

Ввожу в файл код программы вывода сообщения на экран и ввода строки с клавиатуры (рис. 4.7). Сохраняю изменения и закрываю файл.

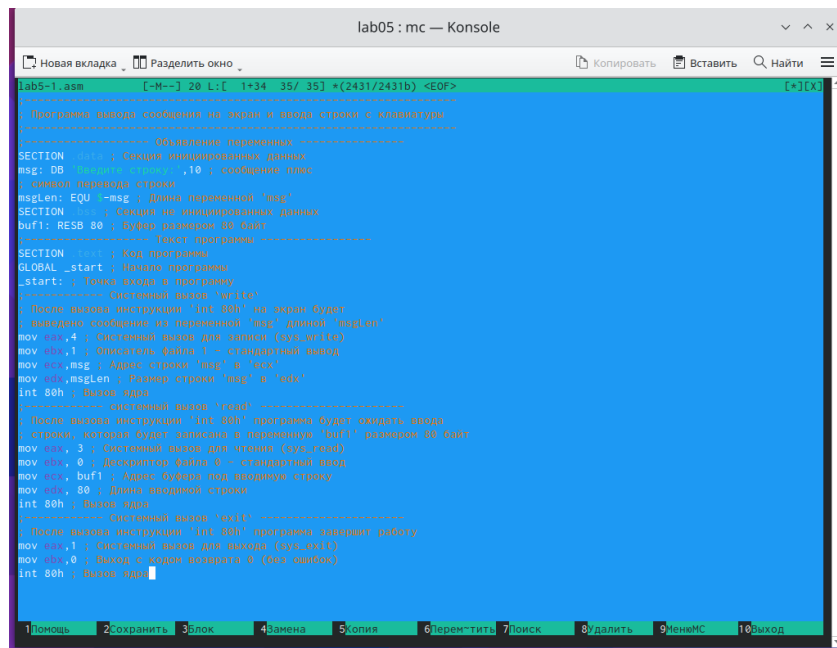


Рис. 4.7: Редактирование файла lab5-1.asm

С помощью функциональной клавиши F3 открываю файл lab5-1.asm для просмотра, чтобы убедиться, что файл содержит текст программы (рис. 4.8).

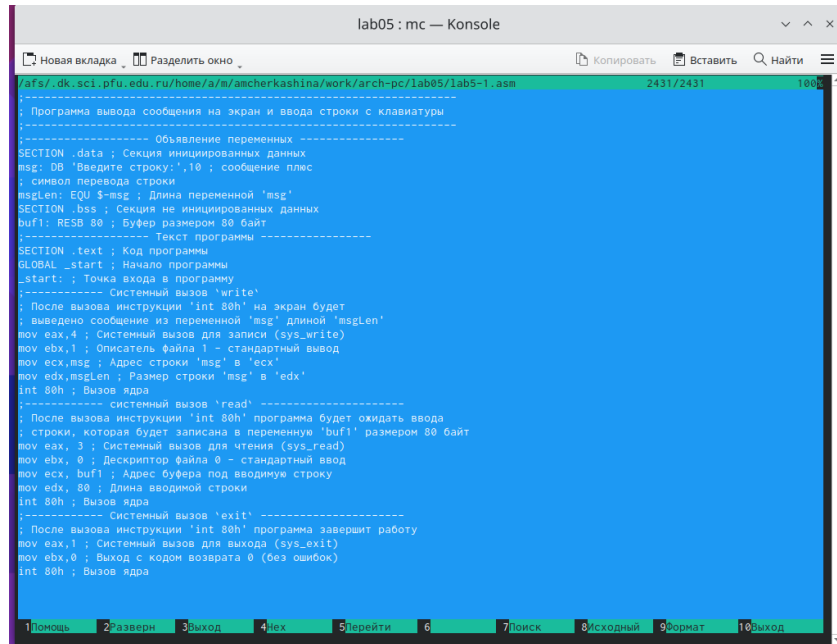


Рис. 4.8: Открытие файла lab5-1.asm для просмотра

Транслирую текст программы файла lab5-1.asm 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 (рис. 4.9).

```
amcherkashina@dk3n35 ~ $ cd work/arch-pc/lab05
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
```

Рис. 4.9: Компиляция файла и передача на обработку компоновщику

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

```
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Черкашина Ангелина Максимовна
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $
```

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).

~/Загрузки		Имя	Размер	Время	Правки
..		Telegram Desktop	2048	ноя 9 17:51	
in_out.asm			3942	сен 14 09:32	
lab1_Черкашина_ответ.docx			1004560	сен 30 14:00	
lab1_Черкашина_ответ.pdf			1751521	сен 30 14:02	
lab2_Черкашина_ответ.pdf			950350	сен 30 16:36	
лабораторный №1.docx			1830927	сен 30 13:07	
лабораторная работа 2.docx			15703	сен 14 10:10	
Редактировать: Простой оператор SELECT.docx			18522	сен 14 10:08	

Рис. 4.11: Скачанный файл in\_out.asm

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

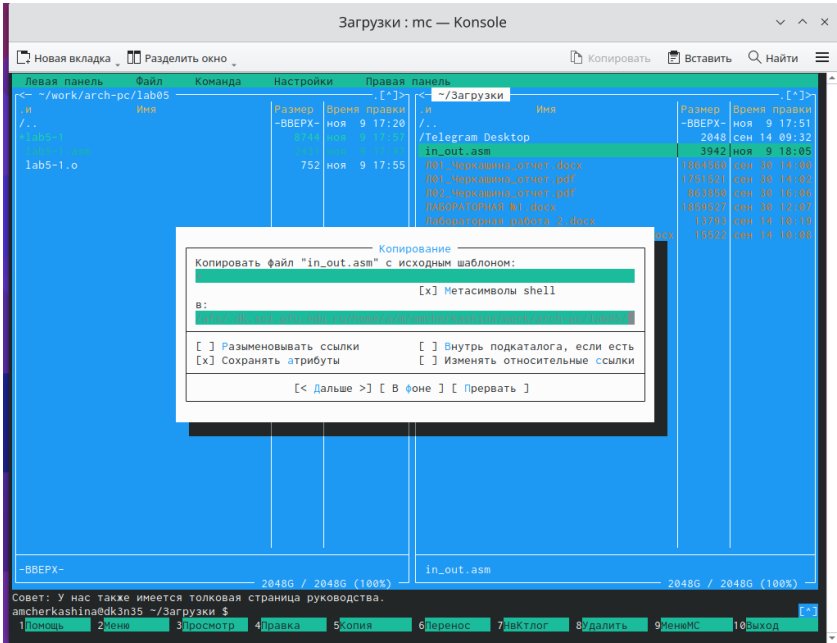


Рис. 4.12: Копирование файла в нужный каталог

С помощью функциональной клавиши F5 создаю копию файла lab5-1 в том же каталоге, но с другим именем (lab5-2.asm). Для этого в появившемся окне mc прописываю путь к копии файла с новым именем (рис. 4.13).

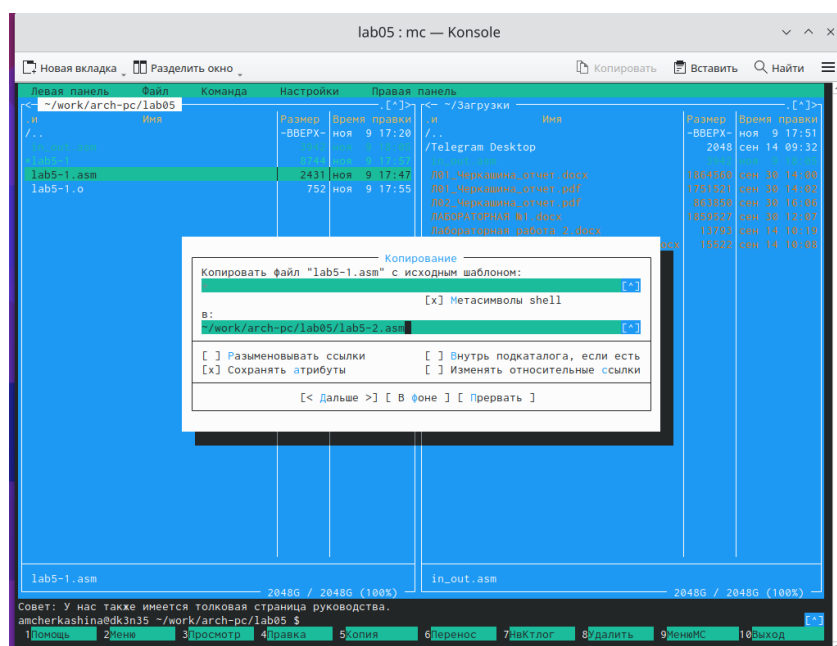


Рис. 4.13: Копирование файла с другим именем

Изменяю содержимое файла lab5-2.asm во встроенном редакторе mcedit так, чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm (рис. ??).



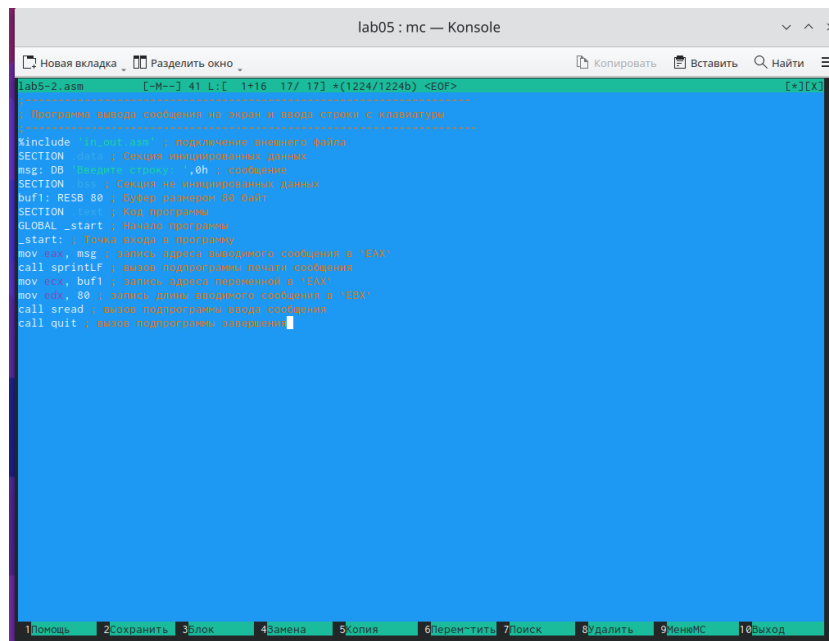


Рис. 4.14: Редактирование файла lab5-2.asm

Транслирую текст программы файла lab5-2.asm 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 получившийся исполняемый файл (рис. 4.15).

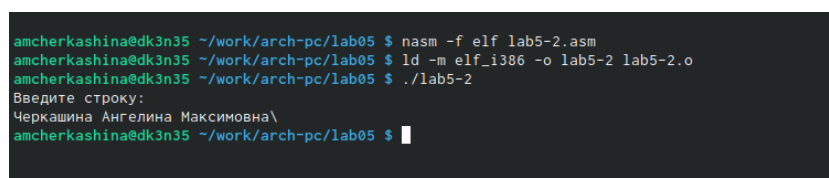
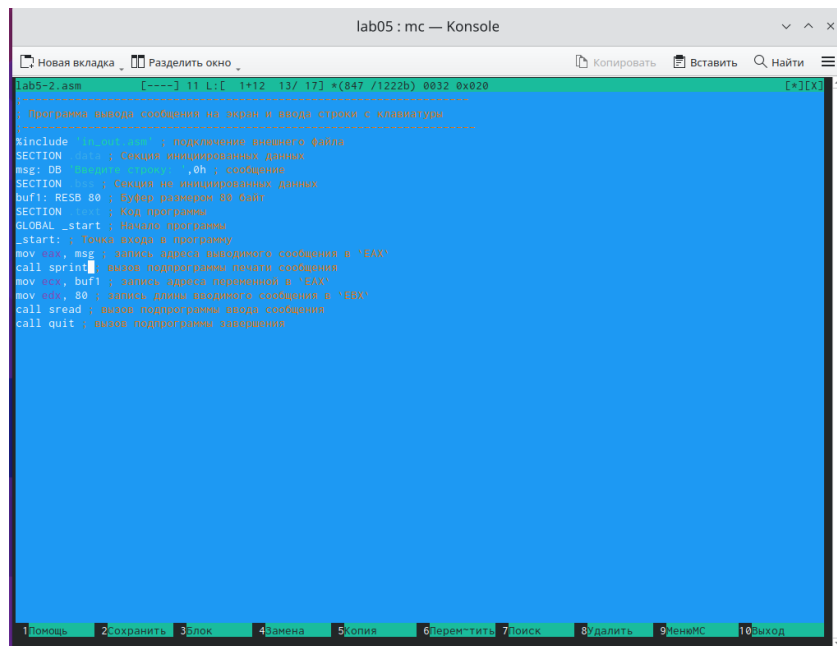


Рис. 4.15: Компиляция, компоновка и исполнение файла

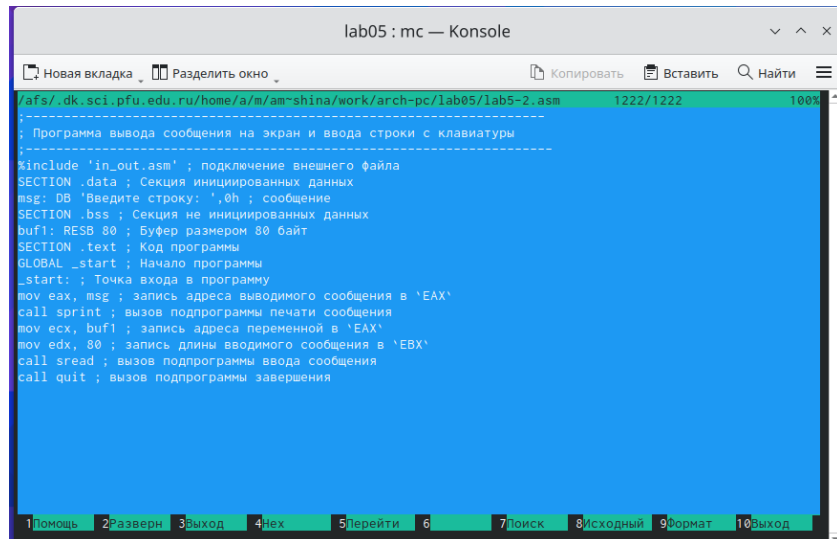
Открываю файл lab5-2.asm для редактирования в mcedit функциональной клавишей F4. Изменяю в нем подпрограмму `sprintf` на `sprint` (рис. 4.16).



```
lab5-2.asm [----] 11 L: [ 1*12 13/ 17] *(847 /1222b) 0032 0x020 [*][X]
; 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: Изменение подпрограммы программы файла lab5-2.asm

Сохраняю изменения и открываю файл для просмотра, чтобы проверить корректность сохранения моих действий (рис. 4.17).



```
lab05: mc — Konsole
/afs/.dk.sci.pfu.edu.ru/home/a/m/am-shina/work/arch-pc/lab05/lab5-2.asm 1222/1222 100%
; 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. 4.17: Просмотр отредактированного файла lab5-2.asm

Снова транслирую текст программы файла в объектный файл, выполняю ком-

поновку созданного объектного файла, получаю исполняемый файл и запускаю его (рис. 4.18).

```
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку: Черкашина Ангелина Максимовна
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $
```

Рис. 4.18: Исполнение отредактированного файла

Разница между первым и вторым исполняемыми файлами в том, что запуск первого запрашивает ввод с новой строки, а программа, которая выполняется при запуске второго, запрашивает ввод без переноса на новую строку. В этом заключается различие между подпрограммами `sprintLF` и `sprint`.

### 4.4 Выполнение заданий для самостоятельной работ

- 1. Создаю копию файла `lab5-1.asm` с именем `lab5-1-1.asm` с помощью функции-ональной клавиши F5 (рис. 4.19).

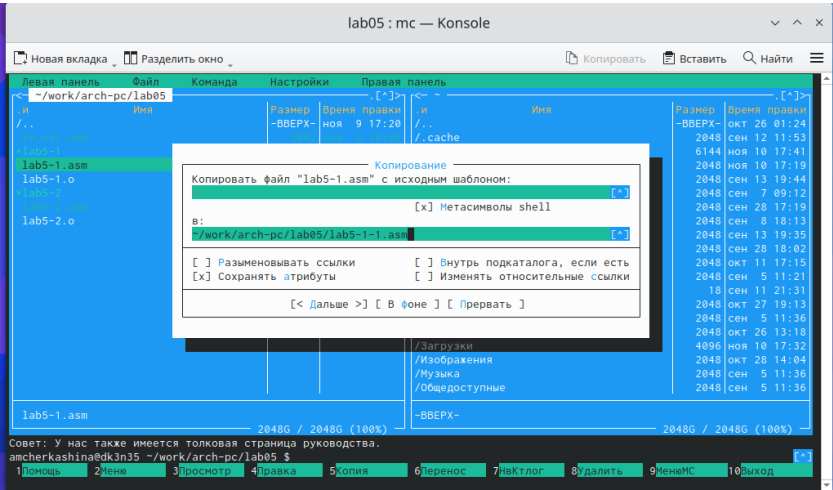


Рис. 4.19: Копирование файла с новым именем

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу (без использования внешнего файла in\_out.asm) так, чтобы кроме вывода приглашения типа “Введите строку:” и запроса ввода строки с клавиатуры она выводила на экран вводимую пользователем строку (рис. 4.20).

```

lab5-1-1.asm  [-M--] 20 L: [ 1+39 40/ 40] *(2773/2773b) <EOF>

; программа вывода сообщения на экран, ввода строки с клавиатуры и вывода введенной строки
;----- Объявление переменных -----
SECTION      ; Секция инициализированных данных
msg: DB "Введите строку: ",10 ; сообщение плюс
; символ перевода строки
msglen: EQU -msg ; Длина переменной 'msg'
SECTION      ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION      ; 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 ; Описание файла 1 - стандартный вывод
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 'buf1' размером 80 байт
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 ; Описание файла 1 - стандартный вывод
mov ecx,buf1 ; Адрес строки 'buf1' в 'ecx'
mov edx,buf1 ; Размер строки 'buf1' в 'edx'
int 80h ; Вызов ядра
;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

```

Рис. 4.20: Редактирование файла lab5-1-1.asm

Открываю файл lab5-1-1.asm для просмотра, чтобы убедиться в корректности внесенных мной изменений (рис. 4.21).



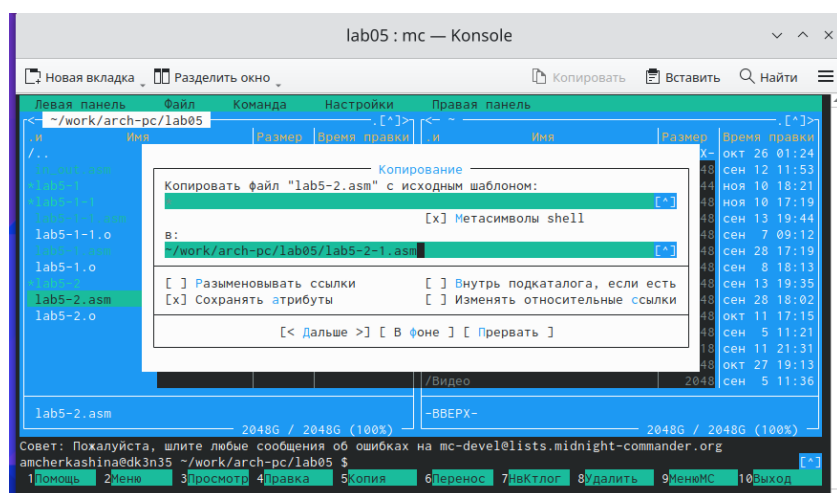


Рис. 4.23: Копирование файла с новым именем

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу, используя подпрограммы из внешнего файла `in_out.asm` так, чтобы кроме вывода приглашения типа “Введите строку:” и запроса ввода строки с клавиатуры она выводила на экран вводимую пользователем строку (рис. 4.24).

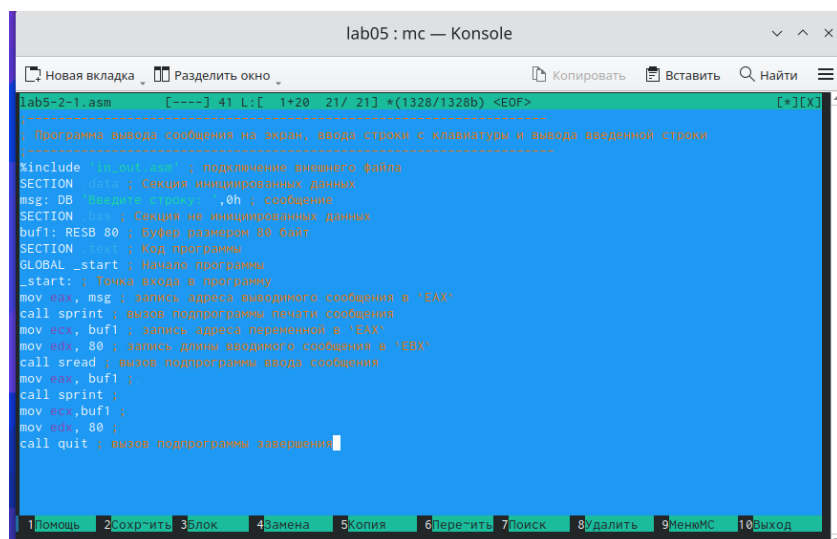
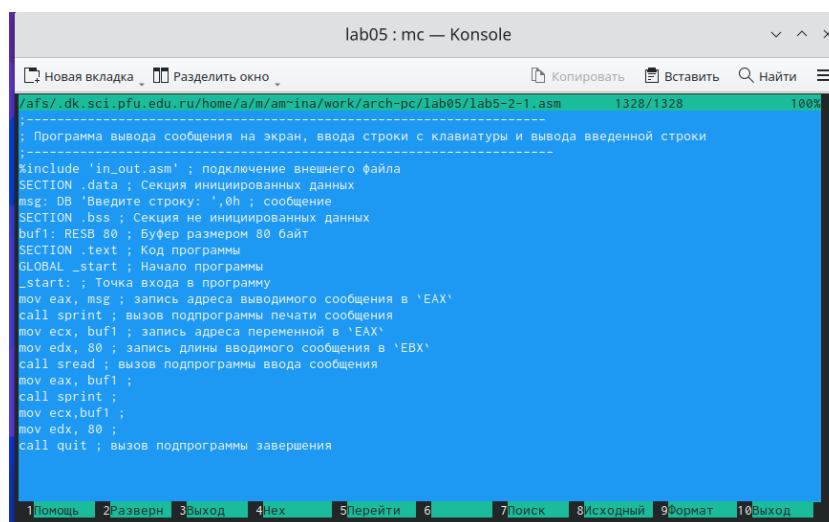


Рис. 4.24: Редактирование файла `lab5-2-1.asm`

Открываю файл `lab5-2-1.asm` для просмотра, чтобы убедиться в корректности

внесенных мной изменений (рис. 4.25).

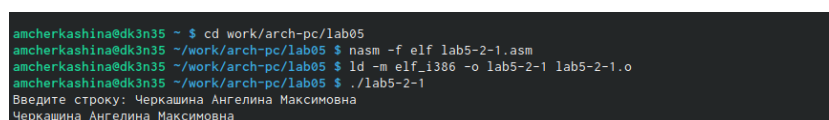


```
lab05 : mc — Konsole
/afs/.dk.sci.pfu.edu.ru/home/a/m/amrina/work/arch-pc/lab05/lab5-2-1.asm 1328/1328 100%
; Программа вывода сообщения на экран, ввода строки с клавиатуры и вывода введенной строки
-----
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных 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
mov eax, buf1 ;
call sprint ;
mov ecx, buf1 ;
mov edx, 80 ;
call quit ; вызов подпрограммы завершения

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

Рис. 4.25: Просмотр отредактированного файла lab5-2-1.asm

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 мной данные на экран (рис. 4.26).



```
amcherkashina@dk3n35 ~ $ cd work/arch-pc/lab05
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2-1.asm
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
amcherkashina@dk3n35 ~/work/arch-pc/lab05 $ ./lab5-2-1
Введите строку: Черкашина Ангелина Максимовна
Черкашина Ангелина Максимовна
```

Рис. 4.26: Исполнение файла lab5-2-1

## 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. Архитектура ЭВМ**