

# **Отчет по лабораторной работе №5**

**Дисциплина: архитектура компьютера**

Адмиральская Александра Андреевна

# 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>8</b>
<b>5</b>	<b>Выводы</b>	<b>20</b>
	<b>Список литературы</b>	<b>21</b>

# Список иллюстраций

4.1	Midnight Commander . . . . .	8
4.2	Переход в каталоги ~/work/arch-pc . . . . .	8
4.3	Создание папки lab05 . . . . .	9
4.4	Создание файла lab5-1.asm . . . . .	9
4.5	Ввод текста программы . . . . .	10
4.6	Проверка . . . . .	11
4.7	Транслируем текст программы в объектный файл и выполняем его компоновку . . . . .	11
4.8	Запуск получившегося исполняемого файла . . . . .	12
4.9	Файл in_out.asm . . . . .	12
4.10	Копирование файла в созданный каталог . . . . .	13
4.11	Копирование файла с другим именем . . . . .	13
4.12	Изменение содержимого файла . . . . .	14
4.13	Транслируем текст в объектный файл, выполняем компоновку объектного файла и запускаем исполняемый файл . . . . .	14
4.14	Изменение подпрограммы sprintLF на sprint . . . . .	15
4.15	Трансляция файла, его компоновка и запуск . . . . .	15
4.16	Копирование файла с новым именем . . . . .	15
4.17	Изменение программы . . . . .	16
4.18	Запуск полученного исполняемого файла . . . . .	17
4.19	Создание копии файла с новым именем . . . . .	17
4.20	изменение программы . . . . .	18
4.21	Запуск полученного исполняемого файла . . . . .	19

# Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . . .	7
-----	---	---

# 1 Цель работы

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 работы

## 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

## 4 Выполнение лабораторной работы

Для начала открываем Midnight Commander с помощью команды `mc` (рис. 4.1).

```
aaadmiraljskaya@dk3n55 ~ $ mc
```

Рис. 4.1: Midnight Commander

Затем переходим в каталог `~/work/arch-pc` созданный при выполнении лабораторной работы №4 (рис. 4.2).

Левая панель	Файл	Команда	Настройки
< ~/work/arch-pc .[^]>			
.и	Имя	Размер	Дата правки
/..		-ВВЕРХ-	окт 24 12:03
/lab04		2048	окт 24 12:50

Рис. 4.2: Переход в каталог `~/work/arch-pc`

С помощью клавиши `F7` создаем папку `lab05` и переходим в этот каталог (рис. 4.3).



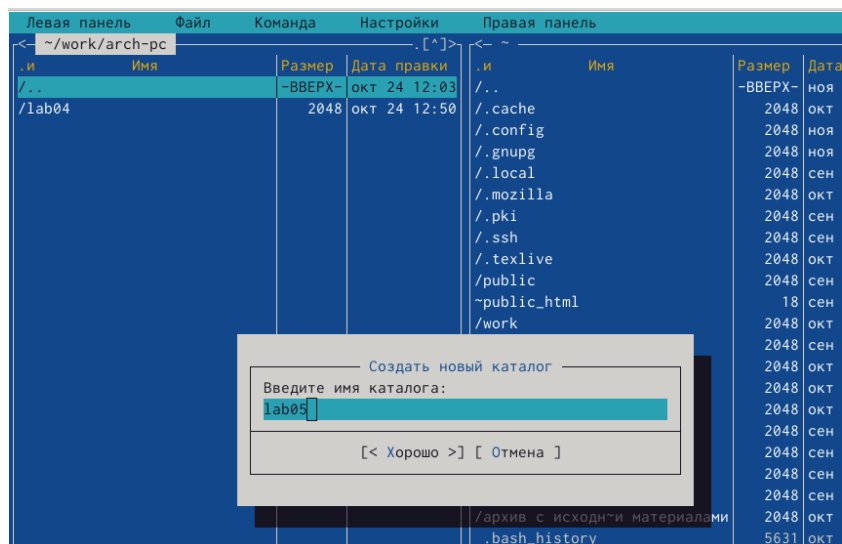


Рис. 4.3: Создание папки lab05

Пользуясь строкой ввода и командой `touch` создаем файл `lab5-1.asm` (рис. 4.4).

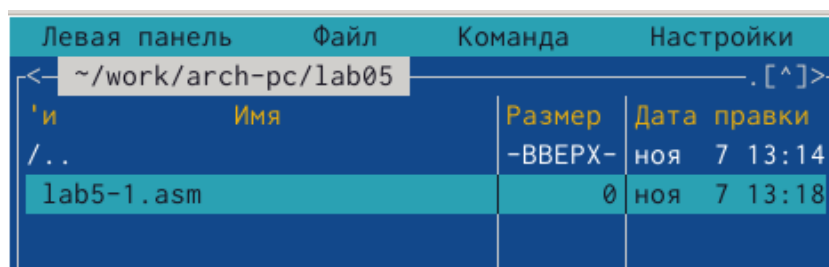


Рис. 4.4: Создание файла lab5-1.asm

С помощью функциональной клавиши F4 открываем созданный файл и вводим текст программы для запроса строки у пользователя (рис. 4.5).

```

lab5-1.asm      [-M--] 20 L: [ 1+34 35/ 35] *(2431/2431b) <EOF>
;-----
; 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
buf1: 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 ; 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
;----- 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
;----- 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. 4.5: Ввод текста программы

При помощи клавиши F3 открываем файл lab5-1.asm для просмотра и убеждаемся, что файл содержит текст программы (рис. 4.6).

```

/afs/.dk.sci.pfu.edu.ru/home/a/a/aa-kaya/work/arch-pc/lab05/lab5-1.asm 2431
;-----
; 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
buf1: 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 ; 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
;----- 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
;----- 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. 4.6: Проверка

Транслируем текст программы файла в объектный файл командой `nasm -f elf lab5-1.asm`. Создался объектный файл `lab5-1.o`. Выполняем компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-1 lab5-1.o`. Создался исполняемый файл `lab5-1` (рис. 4.7).

```

aaadmiraljskaya@dk4n65 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
aaadmiraljskaya@dk4n65 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o

```

Рис. 4.7: Транслируем текст программы в объектный файл и выполняем его компоновку

Запускаем получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и мы вводим свое ФИО (рис. 4.8).

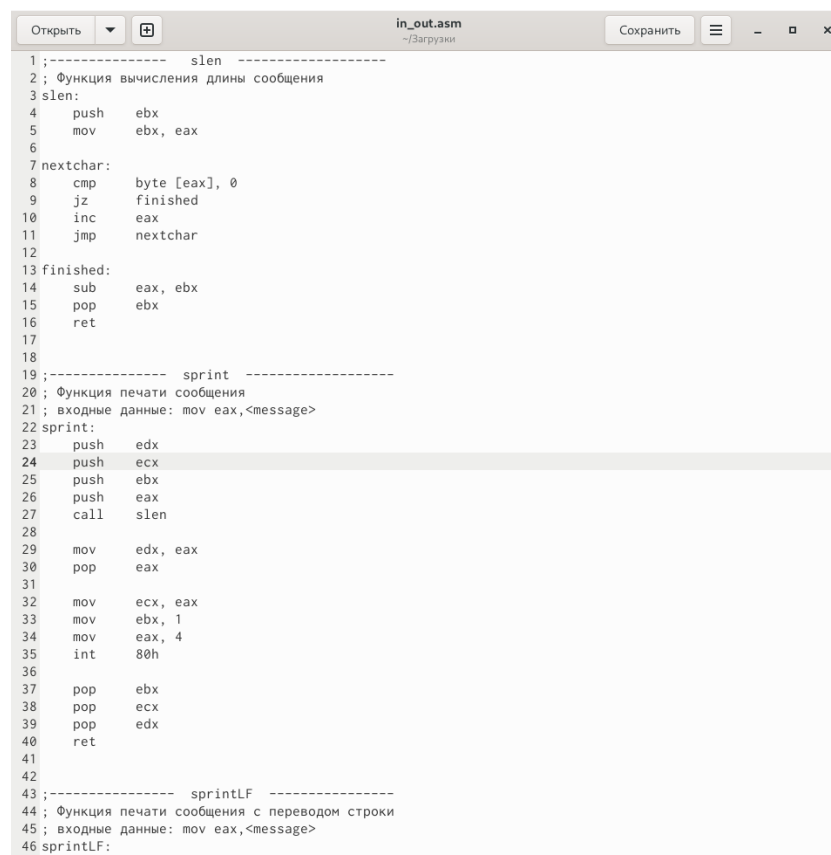
```

aaadmiraljiskaya@dk4n65 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Адмиральская Александра Андреевна
aaadmiraljiskaya@dk4n65 ~/work/arch-pc/lab05 $

```

Рис. 4.8: Запуск получившегося исполняемого файла

Скачиваем файл in\_out.asm со страницы курса в ТУИС. Он сохранился в каталог “Загрузки” (рис. 4.9).



```

1 ;----- slen -----
2 ; Функция вычисления длины сообщения
3 slen:
4     push    ebx
5     mov     ebx, eax
6
7 nextchar:
8     cmp     byte [eax], 0
9     jz      finished
10    inc     eax
11    jmp     nextchar
12
13 finished:
14    sub     eax, ebx
15    pop     ebx
16    ret
17
18
19 ;----- sprint -----
20 ; Функция печати сообщения
21 ; входные данные: mov eax,<message>
22 sprint:
23     push    edx
24     push    ecx
25     push    ebx
26     push    eax
27     call    slen
28
29     mov     edx, eax
30     pop     eax
31
32     mov     ecx, eax
33     mov     ebx, 1
34     mov     eax, 4
35     int     80h
36
37     pop     ebx
38     pop     ecx
39     pop     edx
40     ret
41
42
43 ;----- sprintLF -----
44 ; Функция печати сообщения с переводом строки
45 ; входные данные: mov eax,<message>
46 sprintLF:

```

Рис. 4.9: Файл in\_out.asm

С помощью функциональной клавиши F5 копируем файл in\_out.asm из каталога Загрузки в созданный каталог lab05 (рис. 4.10).

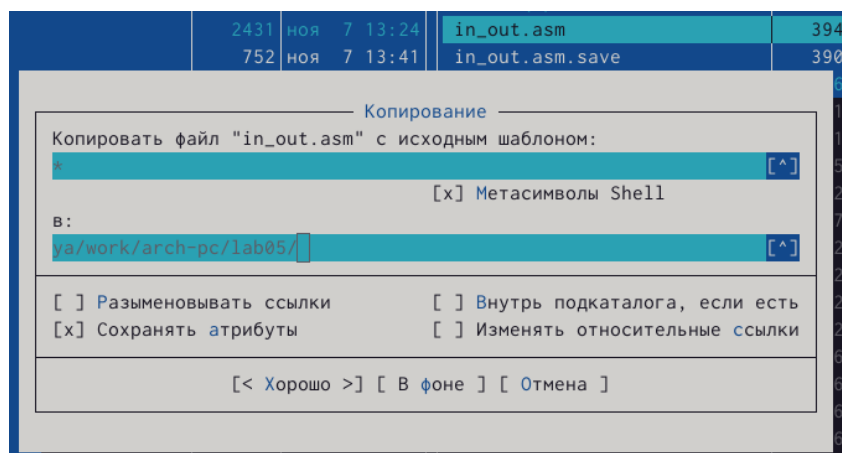


Рис. 4.10: Копирование файла в созданный каталог

С помощью функциональной клавиши F5 копируем файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываем имя для копии файла (рис. 4.11).

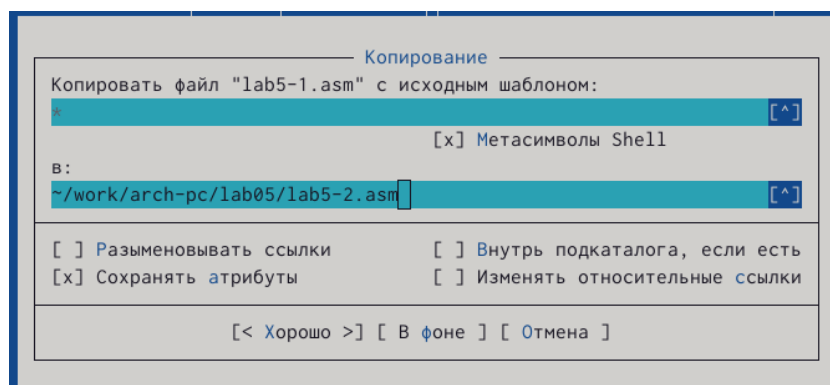
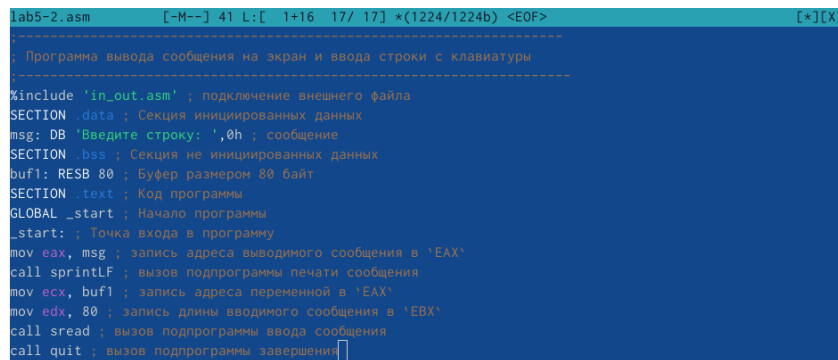


Рис. 4.11: Копирование файла с другим именем

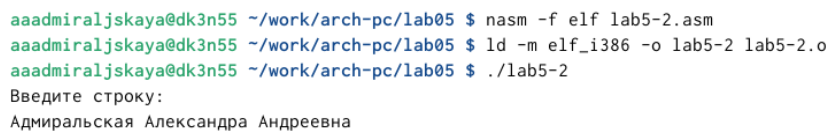
Изменяем содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm (рис. 4.12).



```
lab5-2.asm [-M--] 41 L: [ 1+16 17/ 17] *(1224/1224b) <EOF> [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 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. 4.12: Изменение содержимого файла

Транслируем текст программы файла в объектный файл командой `nasm -f elf lab5-2.asm`. Создался объектный файл `lab5-2.o`. Выполняем компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-2 lab5-2.o`. Создался исполняемый файл `lab5-2`. Запускаем исполняемый файл (рис. 4.13).



```
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
Адмиральская Александра Андреевна
```

Рис. 4.13: Транслируем текст в объектный файл, выполняем компоновку объектного файла и запускаем исполняемый файл

Открываем файл `lab5-2.asm` для редактирования в nano функциональной клавишей F4. Изменяем в нем подпрограмму `sprintf` на `sprint` (рис. 4.14).

```

lab5-2.asm      [-M--] 11 L:[ 1+12 13/ 17] *(847 /1222b) 0032 0x020
;-----
; 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. 4.14: Изменение подпрограммы sprintLF на sprint

Снова транслируем файл, выполняем компоновку созданного объектного файла, запускаем новый исполняемый файл (рис. 4.15). Разница между файлами lab5-2 и lab5-2-2 в том, что запуск первого исполняемого файла запрашивает ввод с новой строки, а программа, используемая при запуске второго файла, запрашивает ввод без переноса на новую строку. В этом и заключается различие между подпрограммами sprintLF и sprint.

```

aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-2 lab5-2.o
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2-2
Введите строку: Адмиральская Александра Андреевна

```

Рис. 4.15: Трансляция файла, его компоновка и запуск

Приступим к выполнению заданий для самостоятельной работы. 1) Создаем копию файла lab5-1.asm с именем lab5-1-1.asm (рис. 4.16).

lab5-1-1.asm	2431	ноя 7 13:24
lab5-1.asm	2431	ноя 7 13:24

Рис. 4.16: Копирование файла с новым именем

С помощью функциональной клавиши F4 открываем созданный файл для редактирования. Изменяем программу так, чтобы кроме вывода приглашения и

запроса ввода, она выводила вводимую пользователем строку (рис. 4.17).

```
lab5-1-1.asm      [-M--] 28 L: [ 1+ 5 6/ 35] *(448 /2401b) 0010 0x00A
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write'
```

Рис. 4.17: Изменение программы

Код измененной программы из пункта 1):

```
;----- ; Программа вывода сообщения на экран
и ввода строки с клавиатуры ;-----
Объявление переменных ----- SECTION .data ; Секция инициированных
данных msg: DB 'Введите строку:',10 ; символ перевода строки msgLen: EQU
$-msg ; Длина переменной 'msg' SECTION .bss ; Секция не инициированных
данных buf1: 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 ; 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
;----- 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
```



;----- 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

- 2) Создаем объектный файл lab5-1-1.o, отдаем его на обработку компоновщи-  
ку, получаем исполняемый файл lab5-1-1, запускаем полученный испол-  
няемый файл. Программа запрашивает ввод, вводим свои ФИО, далее про-  
грамма выводит введенные мною данные (рис. 4.18).

```
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1-1.asm
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-1-1
Введите строку:
Адмиральская Александра Андреевна
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 4.18: Запуск полученного исполняемого файла

- 3) Создаем копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функ-  
циональной клавиши F5 (рис. 4.19).

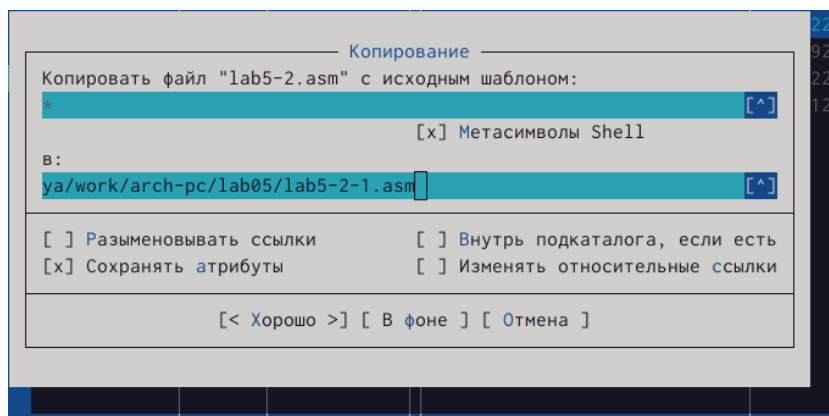


Рис. 4.19: Создание копии файла с новым именем

С помощью функциональной клавиши F4 открываем созданный файл для ре-  
дактирования. Изменяем программу так, чтобы кроме вывода приглашения и  
запроса ввода, она выводила вводимую пользователем строку (рис. 4.20).

```

;-----
; 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
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. 4.20: изменение программы

Код программы из пункта 3):

```

;----- ; Программа вывода сообщения на экран
и ввода строки с клавиатуры ;----- %include
'in_out.asm' ; подключение внешнего файла SECTION .data ; Секция иницииро-
ванных 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

```

- 4) Создаем объектный файл lab5-2-1.o, отдаем его на обработку компоновщи-  
ку, получаем исполняемый файл lab5-2-1, запускаем полученный испол-

няемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. 4.21).

```
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2-1.asm
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2-1
Введите строку: Адмиральская Александра Андреевна
Адмиральская Александра Андреевна
aaadmiraljskaya@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 4.21: Запуск полученного исполняемого файла

## 5 Выводы

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

## Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.
2. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.