

Лабораторная работа №5

. Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM. Системные вызовы в ОС GNU Linux

Бочаров Андрей

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Выполнение самостоятельной работы	14
4	Выводы	18

1 Цель работы

Целью работы является приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера `mov` и `int`.

2 Выполнение лабораторной работы

Открыл файловый менеджер mc(рис. 2.1).

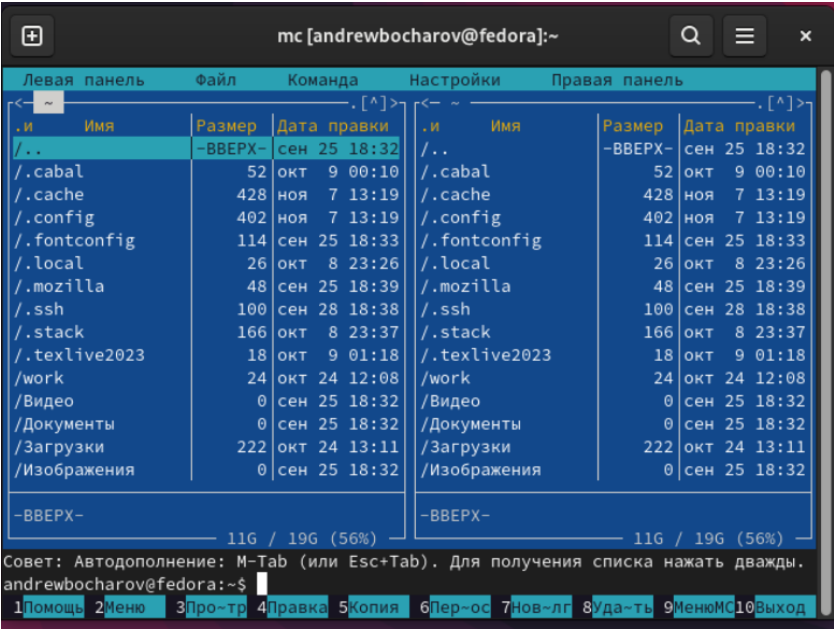


Рис. 2.1: Интерфейс mc

Перешел в папку ~/work/arch-pc при помощи стрелочек и кнопки ввода (рис. 2.2).

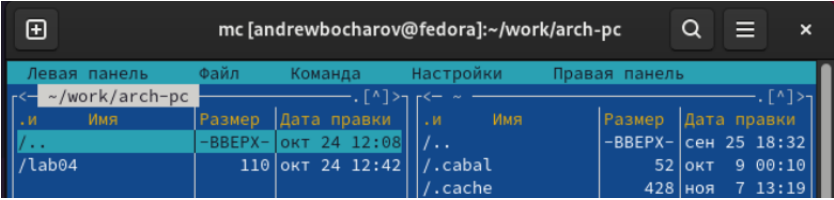


Рис. 2.2: Папка ~/work/arch-pc

Создал папку lab05 (рис. 2.3).

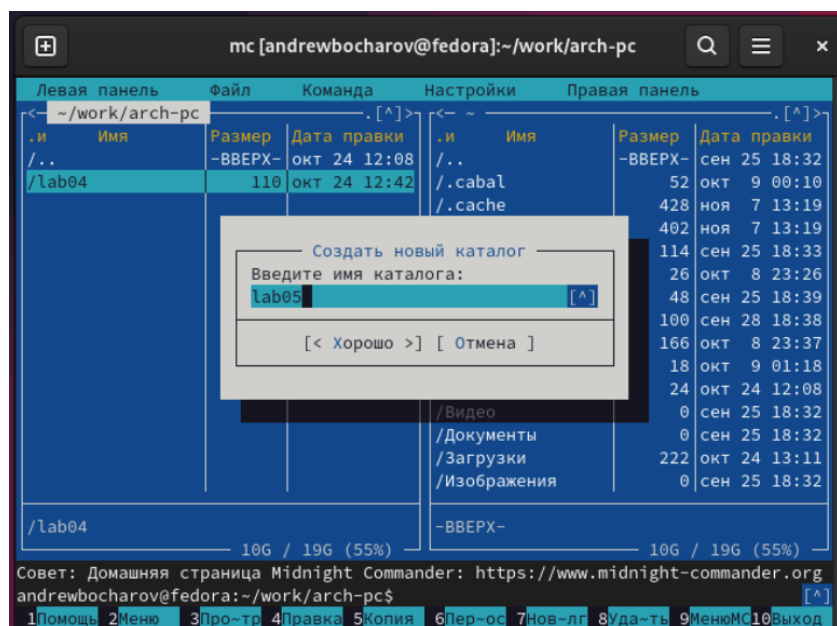


Рис. 2.3: Создание папки lab05

Перешел в новую папку и ввел команду для создания файла lab5-1.asm (рис. 2.4).

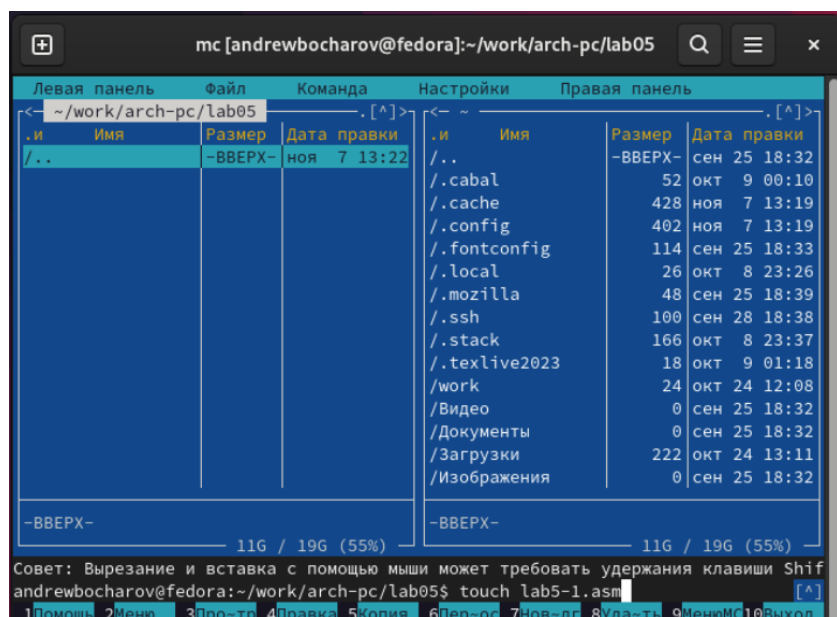


Рис. 2.4: Пустая папка и введенная команда

Открыл редактор mcedit клавишей f4 (рис. 2.5).

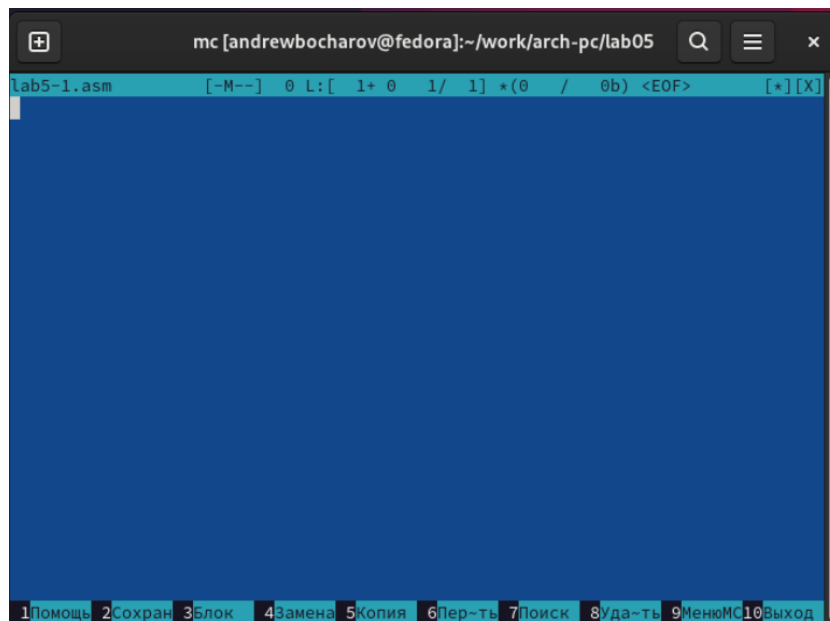


Рис. 2.5: Редактор mcedit

Переписал код для вывода сообщения и ввода строки с клавиатуры (рис. 2.6).

Листинг кода:

```
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
SECTION .data                                ; Секция инициированных данных
msg: DB 'Введите строку: ',10               ; сообщение

msgLen: EQU $-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
```

;----- 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

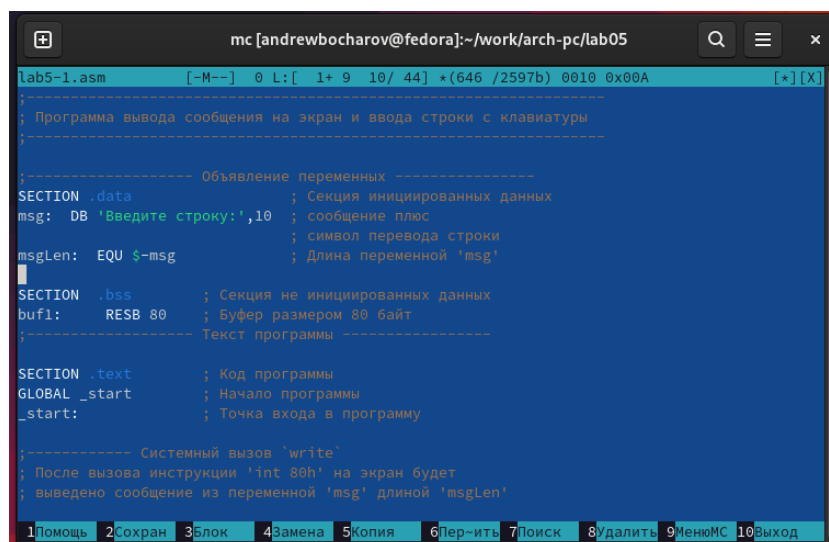
; строки, которая будет записана в переменную 'buf1' размером 80 байт

```
mov eax, 3
mov ebx, 0
mov ecx, buf1 ; запись адреса переменной в `EAX`
mov edx, 80
int 80h ; запись длины вводимого сообщения в `EBX`
```

;----- 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
```

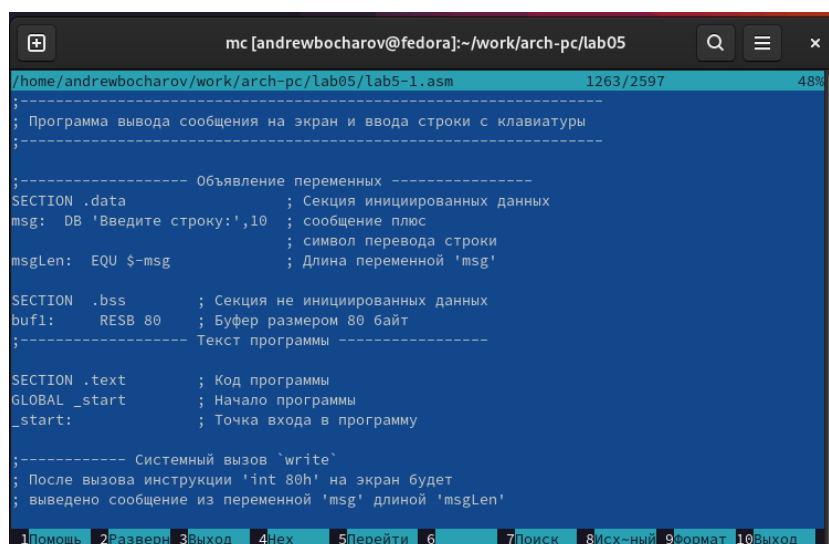


```
lab5-1.asm [-M--] 0 L:[ 1+ 9 10/ 44] *(646 /2597b) 0010 0x00A [*] [X]
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
;-----
SECTION .bss ; Секция не иницированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- 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'
```

Рис. 2.6: Листинг кода

Сохранил и проверил содержимое файла клавишей f3 (рис. 2.7).



```
/home/andrewbocharov/work/arch-pc/lab05/lab5-1.asm 1263/2597 48%
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
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
;----- 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'
```

Рис. 2.7: Содержимое файла

Создал объектный файл lab5-1.o при помощи ассемблера nasm. При помощи объектно-компоновщика ld сделал исполняемый файл lab5-1 (рис. 2.8 и рис. 2.9).


```

lab5-1.asm 11G / 19G (55%) /Документы 11G / 19G (55%)
Совет: Хотите навигацию в стиле Lynx? Настройте её в диалоге Конфигурация.
andrewbocharov@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
1Помощь 2Меню 3Про-отр 4Правка 5Копия 6Перенос 7НовКтлг 8Удалить 9МенюМС 10Выход

```

Рис. 2.8: Трансляция файла

```

Совет: Автодополнение: M-Tab (или Esc+Tab). Для получения списка нажать дважды.
andrewbocharov@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 lab5-1.o -o lab5-1
1Помощь 2Меню 3Про-отр 4Правка 5Копия 6Пер-ос 7Нов-лг 8Удалить 9МенюМС 10Выход

```

Рис. 2.9: Компоновка файла

Проверил наличие файла (рис. 2.10).

```

mc [andrewbocharov@fedora]:~/work/arch-pc/lab05
Левая панель:
<- ~/work/arch-pc/lab05 .[^]>
.и Имя Размер Дата правки
-ВВЕРХ-
*lab5-1 8744 ноя 7 13:43
lab5-1.asm 2597 ноя 7 13:31
lab5-1.o 752 ноя 7 13:35
*lab5-1
11G / 19G (55%)
Правая панель:
<- ~ .[ ^]>
.и Имя Размер Дата правки
-ВВЕРХ-
./.. 52 окт 9 00:10
./cabal 428 ноя 7 13:19
./cache 402 ноя 7 13:19
./fontconfig 114 сен 25 18:33
./local 26 окт 8 23:26
-ВВЕРХ-
11G / 19G (55%)
Совет: Автодополнение: M-Tab (или Esc+Tab). Для получения списка нажать дважды.
andrewbocharov@fedora:~/work/arch-pc/lab05$
1Помощь 2Меню 3Про-отр 4Правка 5Копия 6Пер-ос 7Нов-лг 8Удалить 9МенюМС 10Выход

```

Рис. 2.10: Компоновка файла

Запустил исполняемый файл, он работает (рис. 2.11).

```

andrewbocharov@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Бочаров Андрей

```

Рис. 2.11: Файл работает

Скачал файл in_out.asm (рис. 2.12).

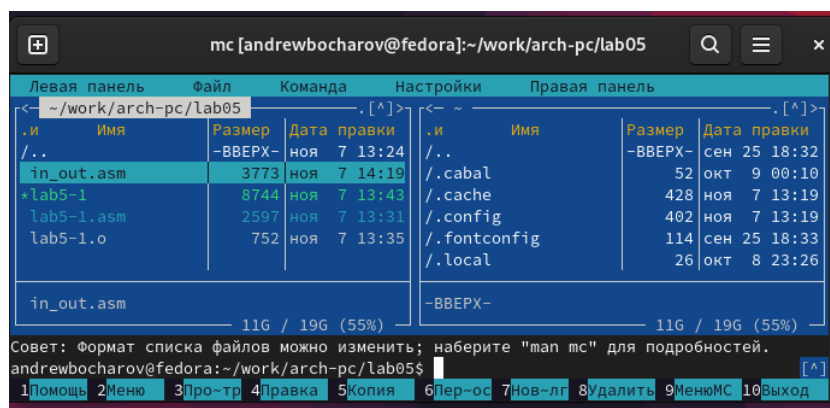


Рис. 2.12: Наличие файла

Сделал копию файла lab5-1.asm при помощи клавиши f5 (рис. 2.13).

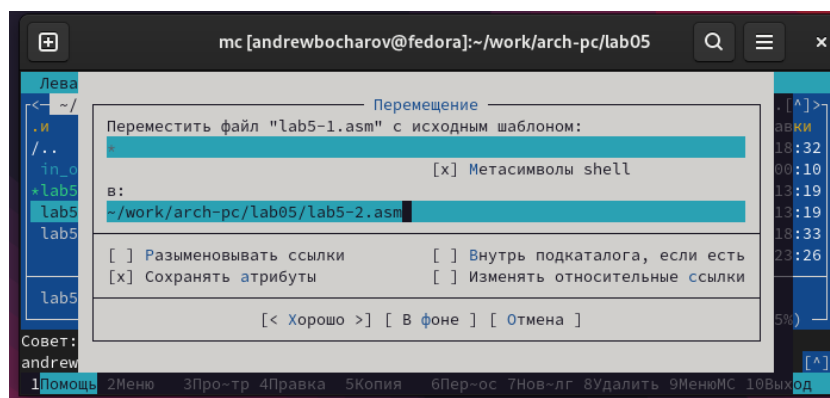


Рис. 2.13: Копирование файла

Переписал код в файле lab5-2.asm (рис. 2.14).

Листинг кода:

```

;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
#include 'in_out.asm'           ; подключение внешнего файла

SECTION .data                  ; Секция инициированных 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 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

```

```

mc [andrewbocharov@fedora]:~/work/arch-pc/lab05
lab5-2.asm [-M--] 0 L:[ 1+ 0 1/ 24] *(0 /1364b) 0059 0x03B [*] [X]
; 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 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

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Удалить 9МенюМС 10Выход

```

Рис. 2.14: Копирование файла

Сохранил изменения в файле lab5-2.asm (рис. 2.15).

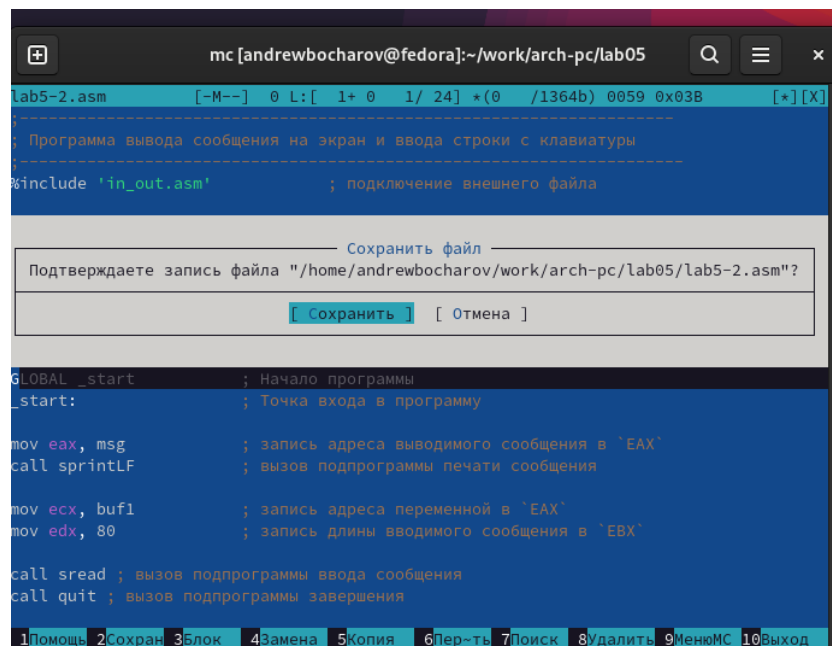


Рис. 2.15: Сохранение изменений

Создал объектный файл lab5-1.o при помощи ассемблера nasm. При помощи объектно компоновщика ld сделал исполняемый файл lab5-2. (рис. 2.16).

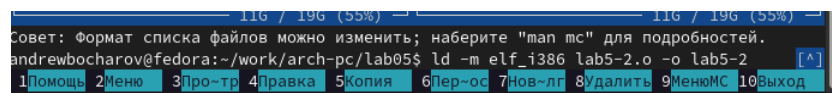


Рис. 2.16: Компоновка файла

Проверил работу исполняемого файла lab5-2 (рис. 2.17).

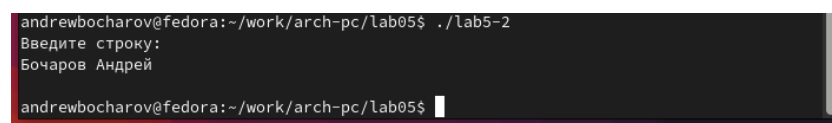


Рис. 2.17: Работает файла

Заменил call sprintf на call string (рис. 2.18).

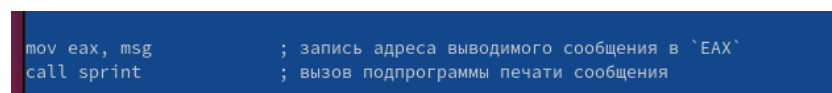
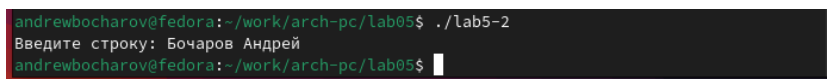


Рис. 2.18: Листинг кода

Посмотрел на результат работы, команда `call stringLF` после вывода строки выводит символ переноса строки, поэтому после изменения команды, Фамилия и имя остались на том же уровне, что и “Введите данные:” (рис. 2.19).

A terminal window with a dark background. The prompt is `andrewbocharov@fedora: ~/work/arch-pc/lab05$`. The command `./lab5-2` has been executed. The output is `Введите строку: Бочаров Андрей`. The prompt is now `andrewbocharov@fedora: ~/work/arch-pc/lab05$` with a cursor at the end.

```
andrewbocharov@fedora: ~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Бочаров Андрей
andrewbocharov@fedora: ~/work/arch-pc/lab05$
```

Рис. 2.19: Листинг кода

3 Выполнение самостоятельной работы

Создал копию файла lab5-1.asm и внес в него изменения, что бы после ввода данных, они выводились на экран. mov eax,4

Листинг кода:

```
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
SECTION .data                                ; Секция инициированных данных
msg: DB 'Введите строку: ',10               ; сообщение

msgLen: EQU $-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

;----- 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
mov ebx, 0
mov ecx, buf1      ; запись адреса переменной в `EAX`
mov edx, 80
int 80h           ; запись длины вводимого сообщения в `EBX`

mov eax, 4          ; Системный вызов для записи (sys_write)
mov ebx, 1          ; Описатель файла 1 - стандартный вывод
mov ecx, buf1      ; Адрес буфера
mov edx, 80        ; Размер буфера
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

```

Сделал трансляцию, компоновку и запустил и проверил код. Код работает (рис.

3.1).

```
andrewbocharov@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Бочаров
Бочаров
```

Рис. 3.1: Результат выполнения

Создал копию файла lab5-2.asm и внес в него изменения, что бы после ввода данных, они выводились на экран с использованием команд из in_out.asm.

Листинг кода:

```
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
%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
```



```
mov eax, buf1          ;запись адреса выводимого сообщения в `EAX`  
call sprint            ;вызов подпрограммы печати сообщения
```

```
call quit ; вызов подпрограммы завершения
```

Сделал трансляцию, компоновку и запустил и проверил код. Код работает (рис. 3.2).



```
andrewbocharov@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm  
andrewbocharov@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 lab5-2.o -o lab5-2  
andrewbocharov@fedora:~/work/arch-pc/lab05$ ./lab5-2  
Введите строку:  
Бочаров  
Бочаров
```

Рис. 3.2: Результат выполнения

4 Выводы

Выполнив данную лабораторную работу я обрел теоретические и практические знания в использовании Midnight Commander. Освоил инструкции языка ассемблера `mov` и `int`.