

# **Шаблон отчёта по лабораторной работе**

**Простейший вариант**

Дмитрий Сергеевич Кулябов

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Выполнение задания для самостоятельной работы</b>	<b>17</b>
<b>4</b>	<b>Выводы</b>	<b>21</b>

# Список иллюстраций

2.1	Запуск Midnight commander . . . . .	6
2.2	Интерфейс midnight commander . . . . .	7
2.3	Переход в нужный каталог (~/.work/arch-pc) . . . . .	8
2.4	Создание папки . . . . .	9
2.5	Создание файла lab5-1.asm с помощью команды touch прямо в mc	10
2.6	Выбор текстового редактора . . . . .	11
2.7	Редактирование файла lab5-1.asm . . . . .	11
2.8	Проверка успешного редактирования . . . . .	12
2.9	Компиляция файла с помощью nasm . . . . .	12
2.10	Сборка исполняемого файла с помощью ld . . . . .	12
2.11	Запуск исполняемого файла . . . . .	12
2.12	Взаимодействие с программой . . . . .	13
2.13	Открытие папки с файлом in_out.asm в правой панели . . . . .	13
2.14	Копирование файла с помощью F6 . . . . .	14
2.15	Копирование файла с помощью F5 . . . . .	14
2.16	Текущий вид рабочей папки . . . . .	15
2.17	Редактирование файла lab5-2.asm . . . . .	15
2.18	Создание исполняемого файла . . . . .	15
2.19	Запуск исполняемого файла . . . . .	15
2.20	Изменение файла lab5-2.asm . . . . .	16
2.21	Запуск изменённого файла . . . . .	16
3.1	Создание копии файла lab5-1.asm . . . . .	17
3.2	Изменение файла lab5-1-1.asm . . . . .	18
3.3	Создание исполняемого файла . . . . .	18
3.4	Создание копии файла lab5-2.asm . . . . .	19
3.5	Изменение файла lab5-2-1.asm . . . . .	19
3.6	Создание исполняемого файла . . . . .	20

## **Список таблиц**

# 1 Цель работы

Ознакомиться с программой Midnight commander и освоить написание программ на языке ассемблера с помощью инструкций `mov` и `int`

## 2 Выполнение лабораторной работы

Для начала выполнения лабораторной работы нам необходимо открыть Midnight commander с помощью команды `mc`

A terminal window with a dark background. The prompt is 'rapavlichenko@rapavlichenko:~\$' and the command 'mc' has been entered.

```
rapavlichenko@rapavlichenko:~$ mc
```

Рис. 2.1: Запуск Midnight commander

После ввода команды мы увидим такой интерфейс

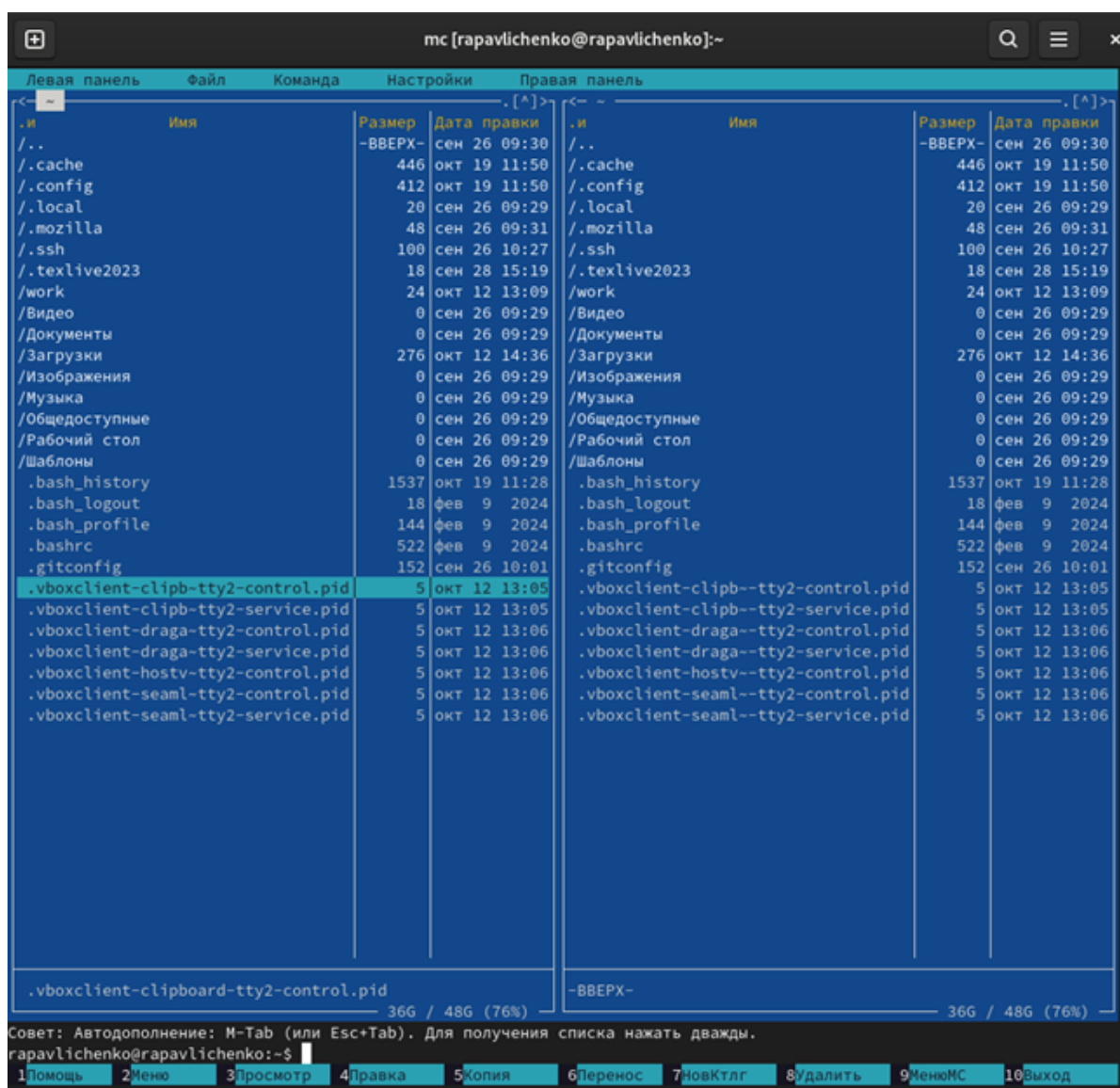


Рис. 2.2: Интерфейс midnight commander

С помощью стрелок и клавиши Enter перейдём в каталог ~/work/arch-pc

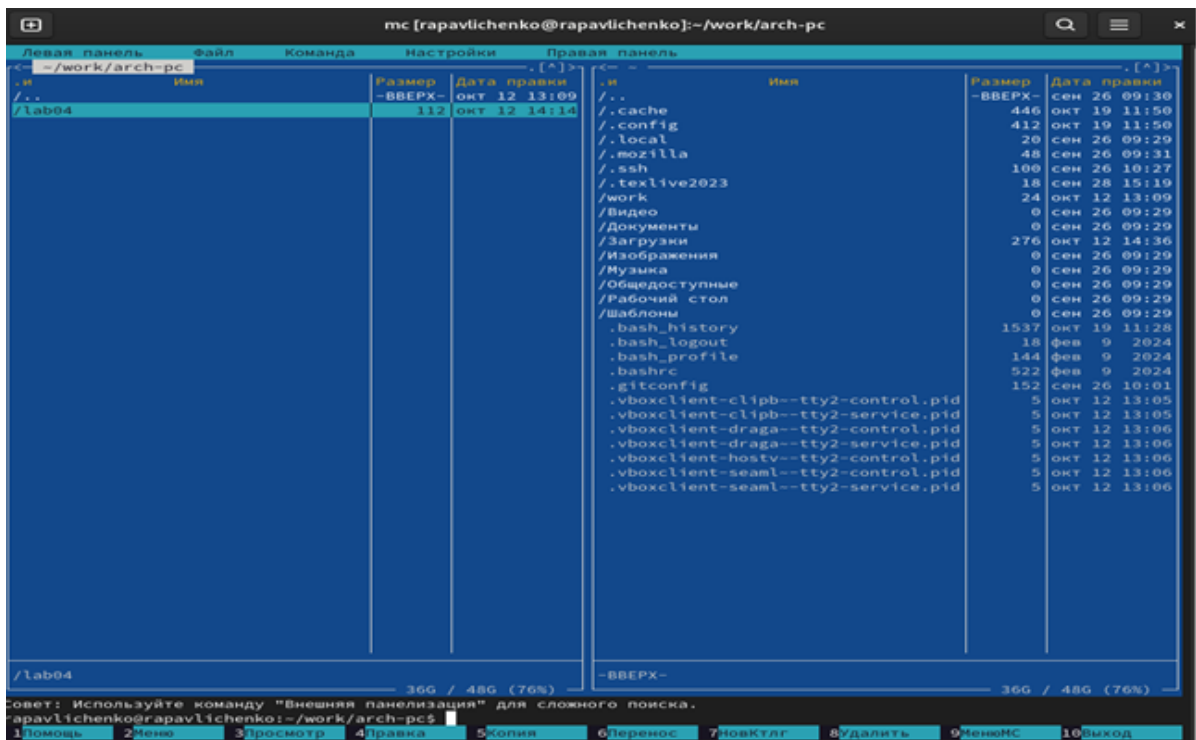


Рис. 2.3: Переход в нужный каталог (~/work/arch-pc)

Создадим папку lab05 с помощью клавиши F7



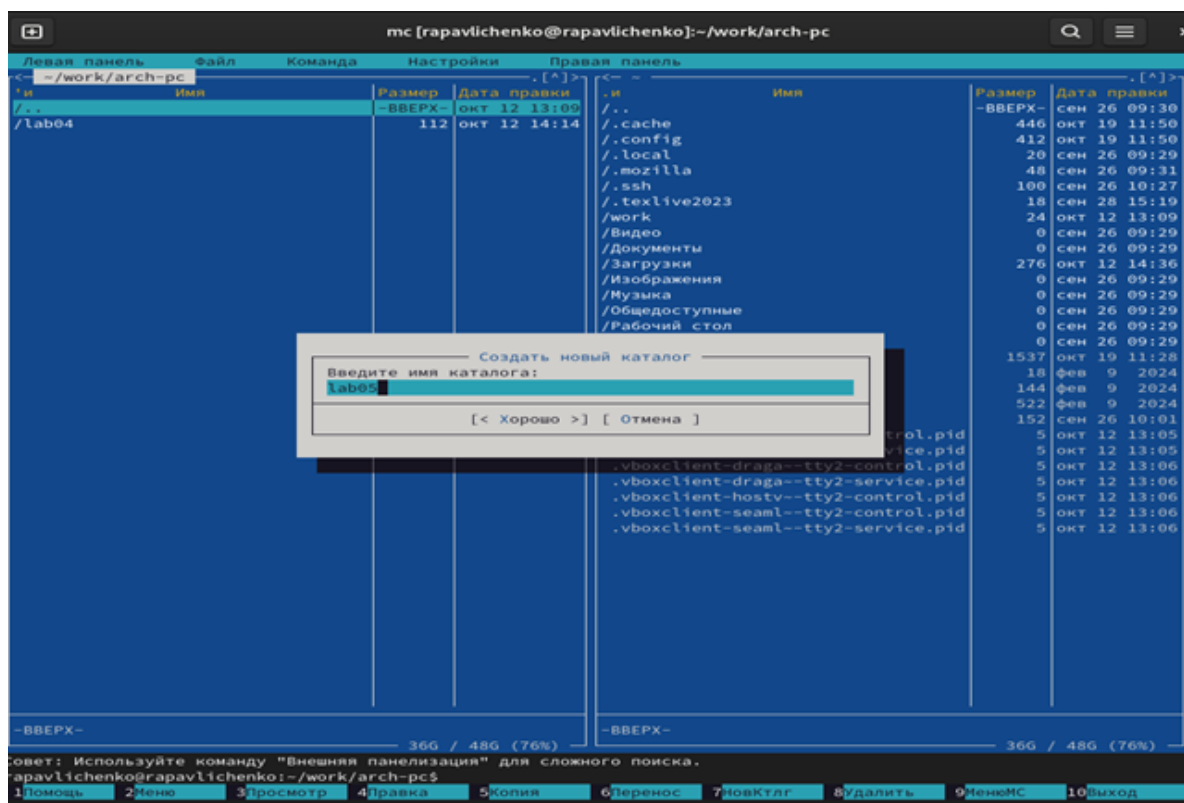


Рис. 2.4: Создание папки

Теперь с помощью команды `touch` создадим файл `lab5-1.asm`

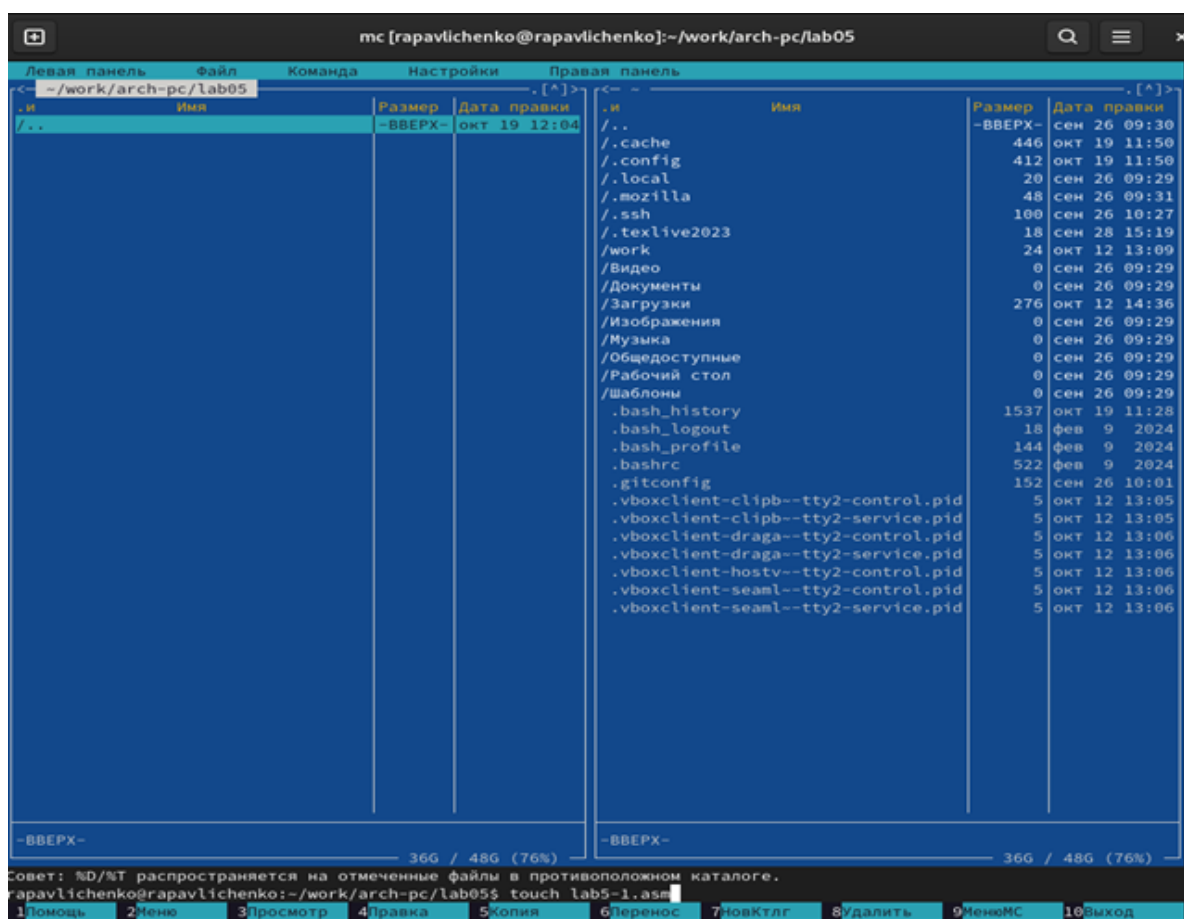
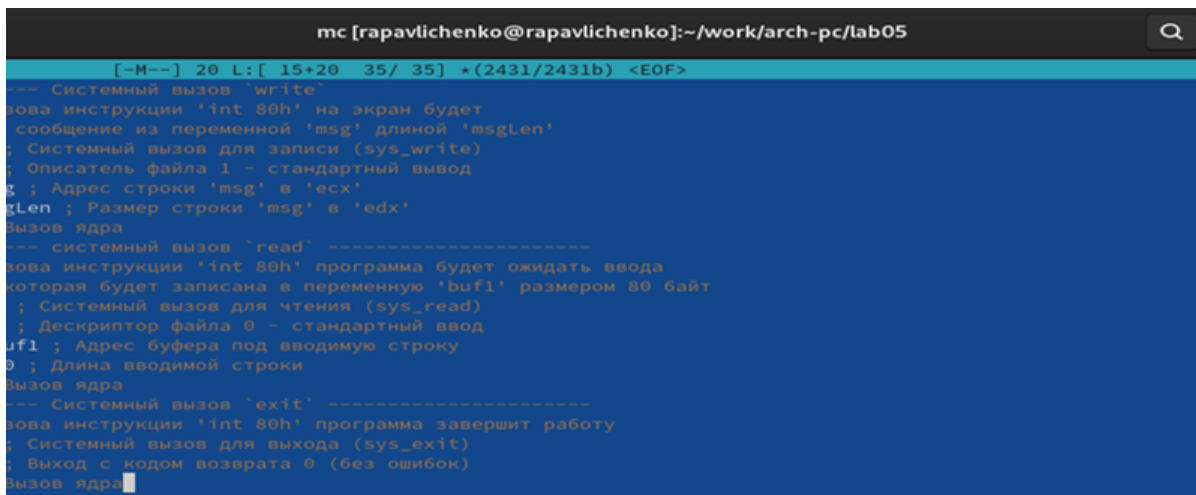


Рис. 2.5: Создание файла lab5-1.asm с помощью команды touch прямо в mc

Теперь с помощью клавиши F4 откроем только что созданный файл и отредактируем файл, поместим в него следующий код



```
mc [rapavlichenko@rapavlichenko]:~/work/arch-pc/lab05
[---] 20 L:[ 15+20 35/ 35] *(2431/2431b) <EOF>
--- Системный вызов 'write'
После вызова инструкции 'int 80h' на экран будет
выведено сообщение из переменной 'msg' длиной 'msgLen'
; Системный вызов для записи (sys_write)
; Описатель файла 1 - стандартный вывод
; Адрес строки 'msg' в 'ecx'
; 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
; Системный вызов для чтения (sys_read)
; Дескриптор файла 0 - стандартный ввод
; Адрес буфера под вводимую строку
; Длина вводимой строки
int 80h ; Вызов ядра
--- Системный вызов 'exit' -----
После вызова инструкции 'int 80h' программа завершит работу
; Системный вызов для выхода (sys_exit)
; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рис. 2.6: Выбор текстового редактора

Теперь сохраним его клавишей F2

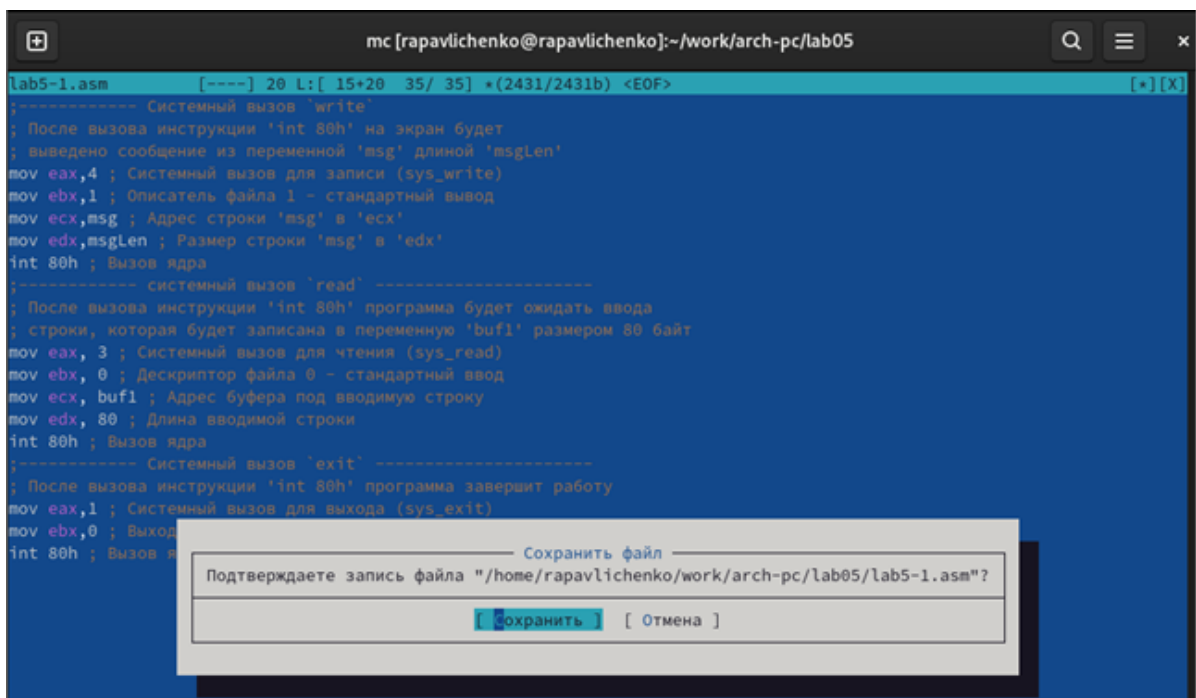
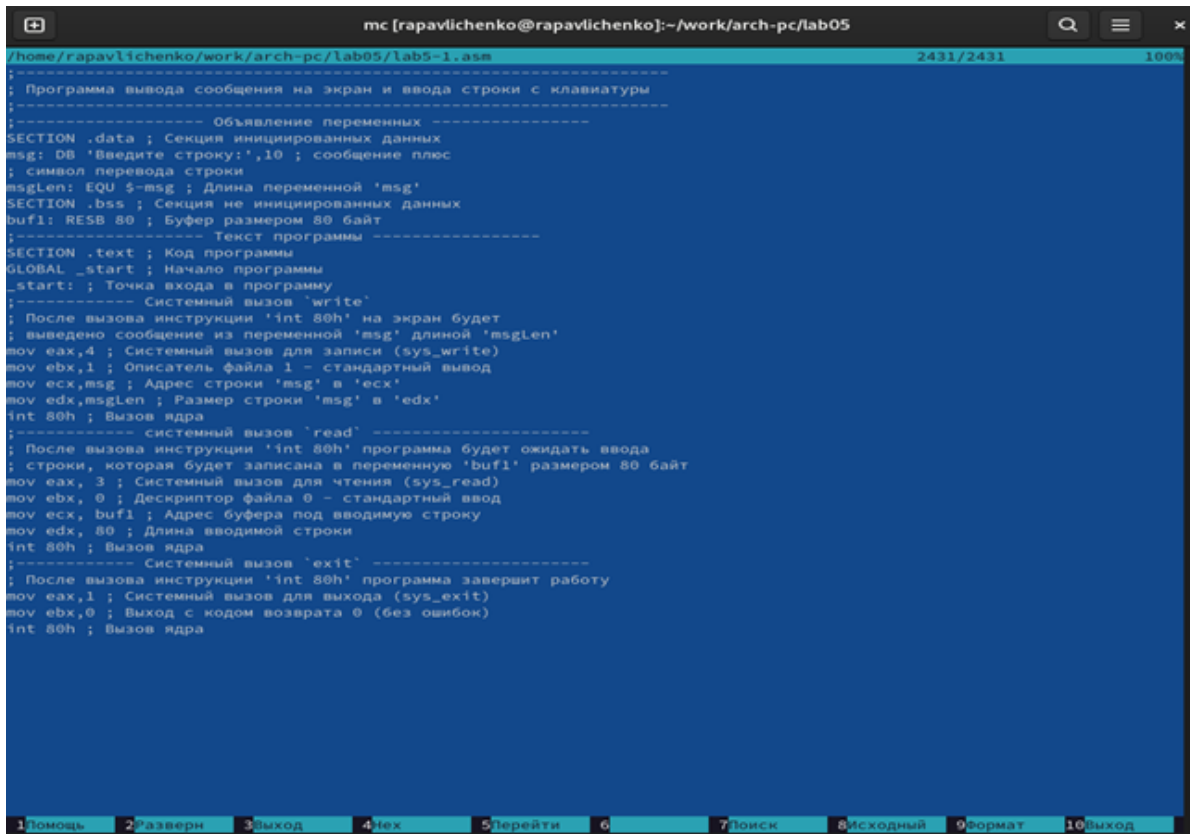


Рис. 2.7: Редактирование файла lab5-1.asm

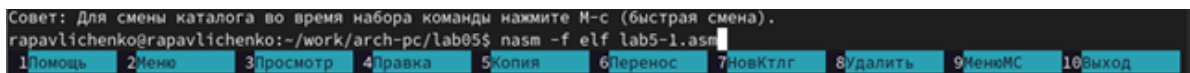
С помощью функциональной клавиши F3 откроем файл lab5-1.asm для просмотра. Убедимся, что файл содержит текст программы.



```
mc [rapavlichenko@rapavlichenko]:~/work/arch-pc/lab05
/home/rapavlichenko/work/arch-pc/lab05/lab5-1.asm 2431/2431 100%
; 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
1Помощь 2Разверн 3Выход 4Тек 5Перейти 6 7Поиск 8Скопировать 9Формат 10Выход
```

Рис. 2.8: Проверка успешного редактирования

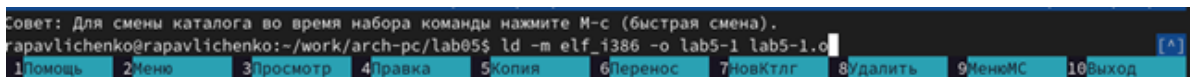
Теперь скомпилируем его



```
Совет: Для смены каталога во время набора команды нажмите M-c (быстрая смена).
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
1Помощь 2Меню 3Просмотр 4Правка 5Копия 6Перенос 7НовКтлг 8Удалить 9МенюМС 10Выход
```

Рис. 2.9: Компиляция файла с помощью nasm

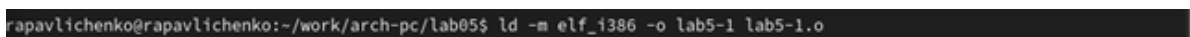
И соберём



```
Совет: Для смены каталога во время набора команды нажмите M-c (быстрая смена).
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
1Помощь 2Меню 3Просмотр 4Правка 5Копия 6Перенос 7НовКтлг 8Удалить 9МенюМС 10Выход
```

Рис. 2.10: Сборка исполняемого файла с помощью ld

После этого запустим получившийся исполняемый файл



```
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
```

Рис. 2.11: Запуск исполняемого файла

Теперь введём ФИО

```
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Павличенко Родион Андреевич
```

Рис. 2.12: Взаимодействие с программой

После нажатия Enter программа завершится и ничего не произойдёт. Теперь скачаем файл `in_out.asm` и откроем папку с ним в правой панели

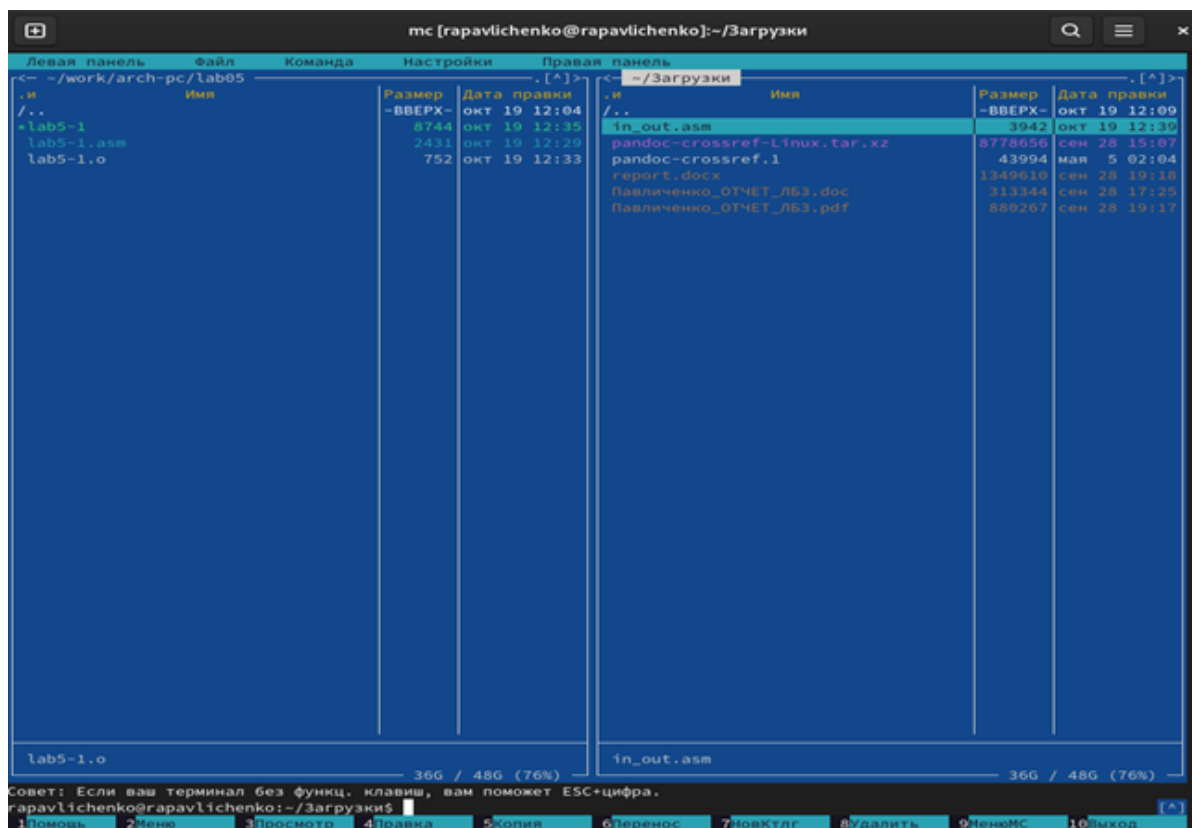


Рис. 2.13: Открытие папки с файлом `in_out.asm` в правой панели

Скопируем его в нашу рабочую папку с помощью F6

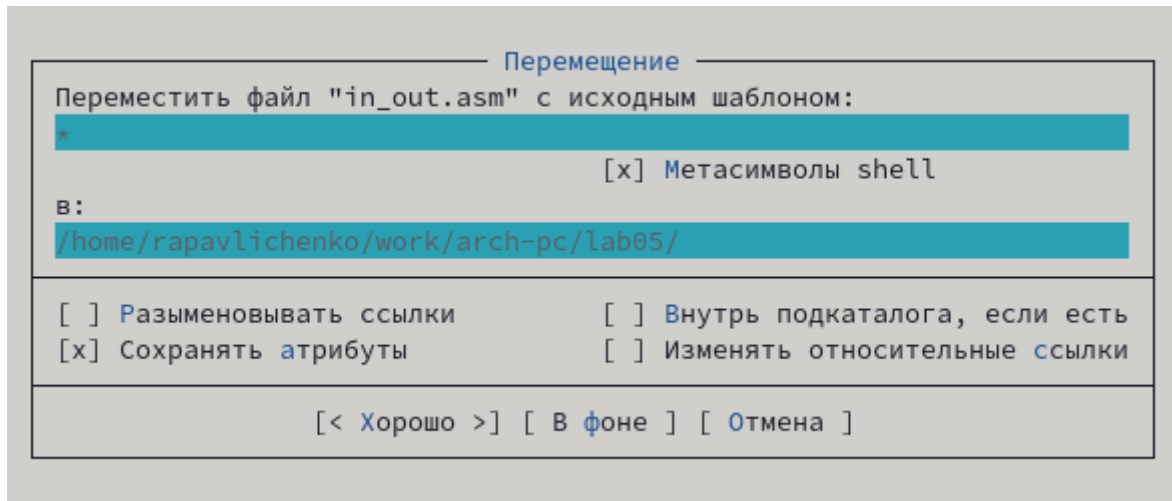


Рис. 2.14: Копирование файла с помощью F6

Теперь сделаем копию файла lab5-1.asm с помощью команды F5. Назовём копию lab5-2.asm

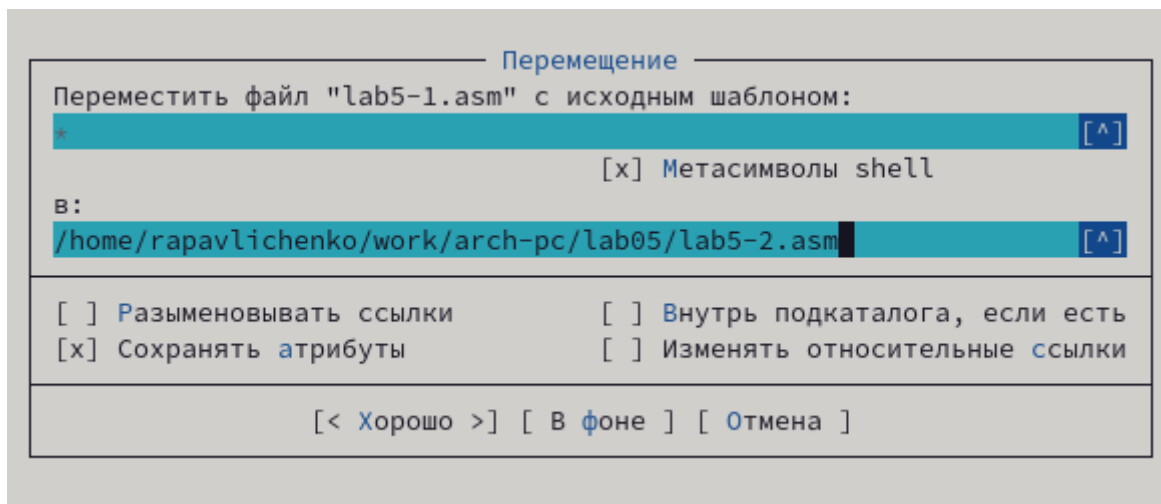


Рис. 2.15: Копирование файла с помощью F5

Теперь наша папка выглядит следующим образом

Имя	Размер	Дата правки
./..	-ВВЕРХ-	окт 19 12:04
in_out.asm	3942	окт 19 12:39
*lab5-1	8744	окт 19 12:35
lab5-1.asm	2431	окт 19 12:29
lab5-1.o	752	окт 19 12:33
lab5-2.asm	2431	окт 19 12:29

Рис. 2.16: Текущий вид рабочей папки

Откроем в текстовом редакторе файл lab5-2.asm и напишем туда следующий код

```

lab5-2.asm  [-M--] 54 L: [ 1+14 15/ 17] *(1076/1224b) 0010 0x00A  [.*] [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. 2.17: Редактирование файла lab5-2.asm

После чего создадим исполняемый файл с помощью nasm и ld

```

rapavlchenko@rapavlchenko:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
rapavlchenko@rapavlchenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
rapavlchenko@rapavlchenko:~/work/arch-pc/lab05$ ./lab5-2

```

Рис. 2.18: Создание исполняемого файла

Запустим созданный файл и введем ФИО

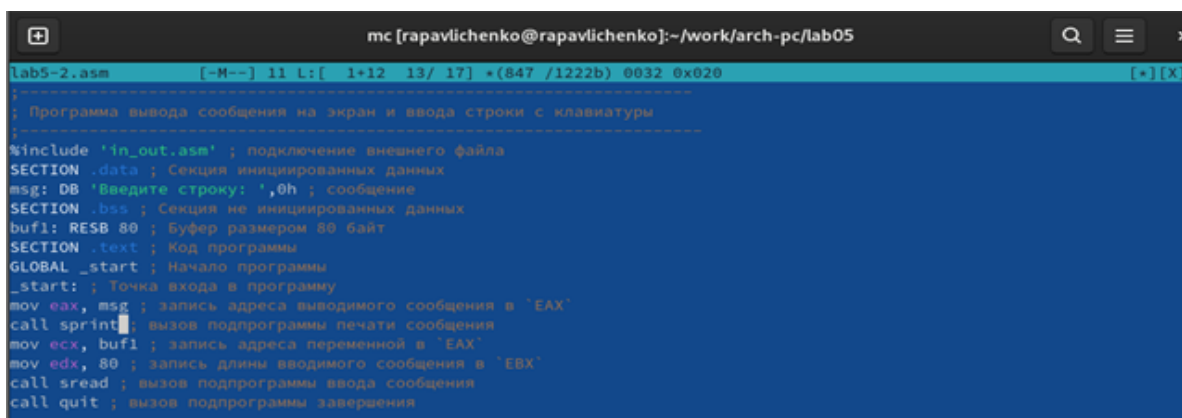
```

rapavlchenko@rapavlchenko:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Павличенко Родион Андреевич

```

Рис. 2.19: Запуск исполняемого файла

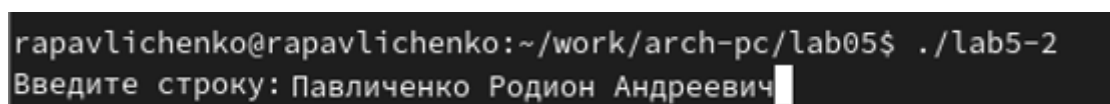
Он работает также, как и файл lab5-1, но использует для работы сторонний файл. Попробуем теперь вместо команды `sprintLF` использовать просто команду `sprint`



```
lab5-2.asm [-M--] 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. 2.20: Изменение файла lab5-2.asm

Точно также соберём исполняемый файл и запустим его



```
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Павличенко Родион Андреевич
```

Рис. 2.21: Запуск изменённого файла

Как мы видим, теперь нет переноса на следующую строку. Этим и отличаются команды `sprintLF` от `sprint`. Первая добавляет перенос после текста, а вторая нет



### 3 Выполнение задания для самостоятельной работы

Теперь создадим с помощью F6 копию файла lab5-1.asm

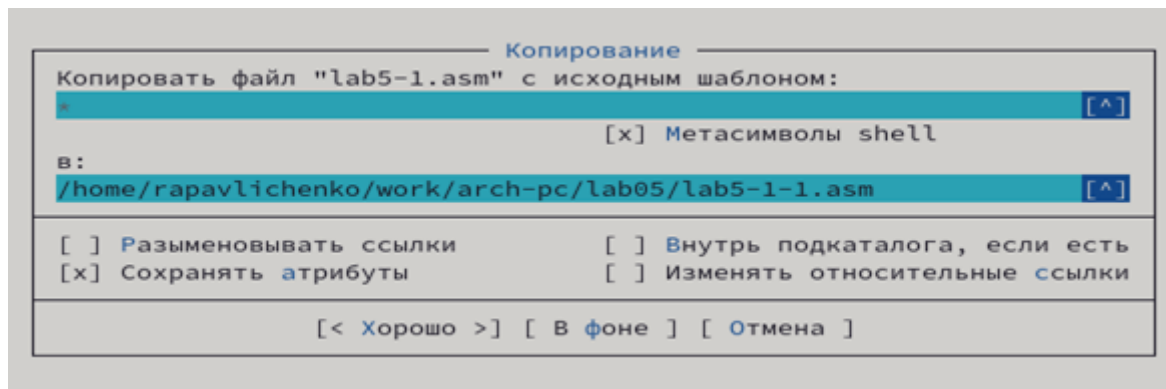


Рис. 3.1: Создание копии файла lab5-1.asm

Изменим копию так, чтобы она выводила тот текст, который получила на ввод. Для этого перед системным вызовом `exit` вставим текст с системным вызовом `write`. Он очень похож на системный вызов `write`, который уже был в коде, но есть несколько отличий. Так, мы перемещаем адрес строки `buf1` в `ecx` и размер строки `buf1` (80) в `edx` (Рис. 3.2):

```

mc [rapavlichenko@rapavlichenko]:~/work/arch-pc/lab05
lab5-1-1.asm [-M--] 20 L: [ 1+12 13/ 43] *(848 /2939b) 1083 0x438 (*) [X]
----- 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 write -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'buf1' длиной 80
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'
mov edx,80 ; 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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Перем-тить 7Поиск 8Удалить 9МенюМС 10Выход

```

Рис. 3.2: Изменение файла lab5-1-1.asm

Сохраним изменения и создадим исполняемый файл. Запустим его и проверим, что всё работает.

```

rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Павличенко Родион Андреевич
Павличенко Родион Андреевич

```

Рис. 3.3: Создание исполняемого файла

Теперь создадим с помощью F5 копию файла lab5-2.asm

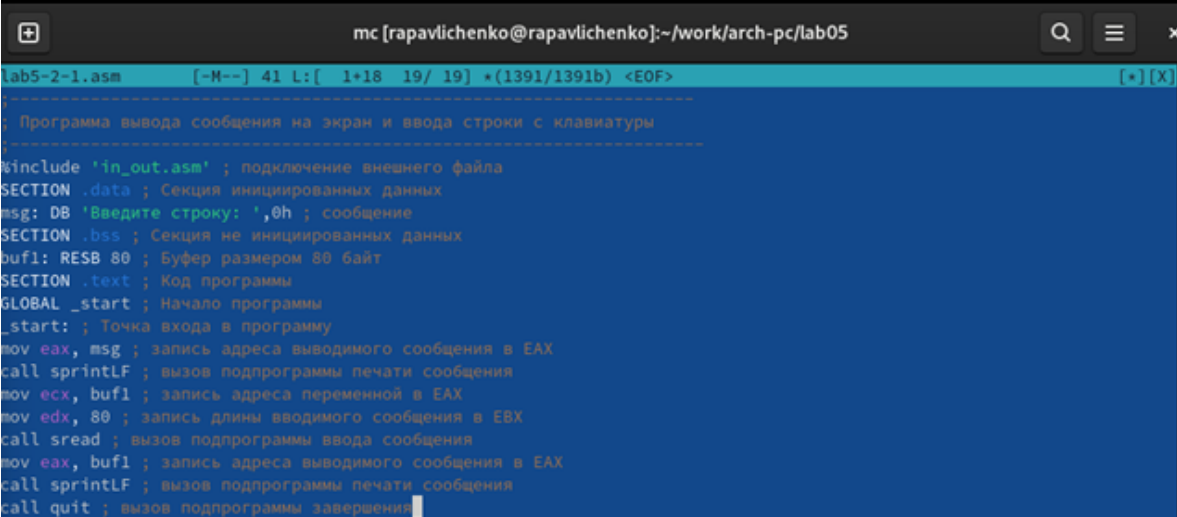
```

rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Павличенко Родион Андреевич
Павличенко Родион Андреевич

```

Рис. 3.4: Создание копии файла lab5-2.asm

Теперь сделаем так, чтобы этот код также выводил тот текст, что получит на ввод. Для этого перед последней строкой добавим строчку, которая записывает в еах адрес buf1, а также строчку, которая вызывает подпрограмму sprintfLF (Рис. 3.6):



```

lab5-2-1.asm  [-M--] 41 L: [ 1+18 19/ 19] *(1391/1391b) <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 sprintfLF ; 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 sprintfLF ; вызов подпрограммы печати сообщения
call quit ; вызов подпрограммы завершения

```

Рис. 3.5: Изменение файла lab5-2-1.asm

Теперь создадим исполняемый файл, запустим программу и убедимся, что она работает

```
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ nasm -f elf lab5-2-1.asm
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
rapavlichenko@rapavlichenko:~/work/arch-pc/lab05$ ./lab5-2-1
Введите строку:
Павличенко Родион Андреевич
Павличенко Родион Андреевич
```

Рис. 3.6: Создание исполняемого файла

## 4 Выводы

В результате выполнения работы были получены навыки работы с Midnight commander, а также навыки написания простых программ ввода-вывода на языке ассемблера