

# **Отчет по лабораторной работе №5**

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

Мальянц Виктория Кареновна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Выполнение заданий для самостоятельной работы . . . . .	19
<b>3</b>	<b>Выводы</b>	<b>25</b>

# Список иллюстраций

2.1	Окно mc . . . . .	6
2.2	Перемещение в каталог ~/work/arch-pc . . . . .	7
2.3	Создание папки lab05 . . . . .	8
2.4	Переход в папку lab05 . . . . .	8
2.5	Создание файла lab-1.asm . . . . .	9
2.6	Открытие файла lab5-1.asm для редактирования во встроенном редакторе . . . . .	9
2.7	Ввод текста программы . . . . .	10
2.8	Сохранение изменений . . . . .	11
2.9	Выход из редактора . . . . .	12
2.10	Просмотр содержимого файла lab5-1.asm . . . . .	13
2.11	Транслирование текста в объектный файл . . . . .	13
2.12	Компоновка объектного файла . . . . .	13
2.13	Запуск исполняемого файла . . . . .	13
2.14	Скачанный файл in_out_asm . . . . .	14
2.15	Копирование файла in_out.asm в каталог lab05 . . . . .	15
2.16	Создание копии файла lab5-1.asm с именем lab5-2.asm . . . . .	16
2.17	Редактирование текста программы в файле lab5-2.asm . . . . .	17
2.18	Исполнение файла . . . . .	17
2.19	Редактирование текста программы в файле lab5-2.asm . . . . .	18
2.20	Исполнение файла . . . . .	18
2.21	Создание копии файла lab5-1.asm с именем lab5-1-1.asm . . . . .	19
2.22	Ввод текста программы . . . . .	20
2.23	Исполнение файла . . . . .	21
2.24	Создание копии файла lab5-2.asm с именем lab5-2-1.asm . . . . .	22
2.25	Ввод текста программы . . . . .	23
2.26	Исполнение файла . . . . .	24

## **Список таблиц**

# 1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера `mov` и `int`.

# 2 Выполнение лабораторной работы

Открываю Midnight Commander с помощью команды mc (рис. 2.1).

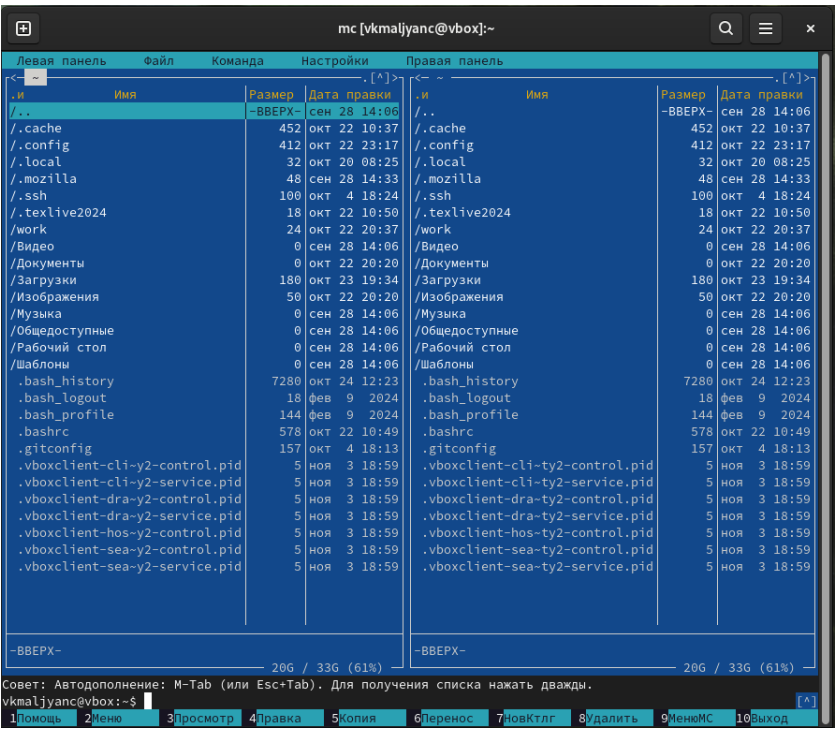


Рис. 2.1: Окно mc

Перехожу в каталог ~/work/arch-рс с помощью файлового менеджера mc (рис. 2.2).

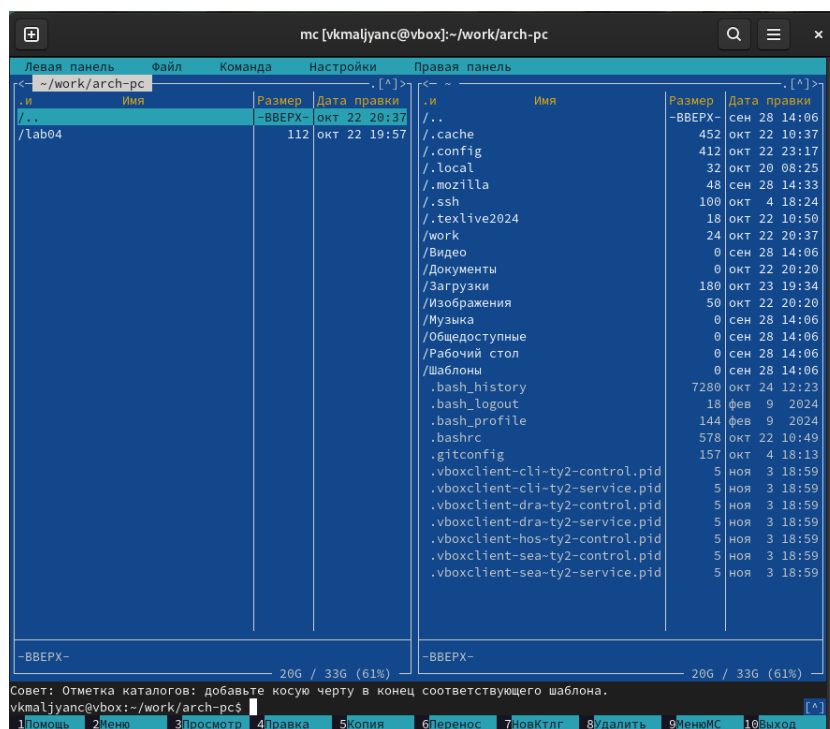


Рис. 2.2: Перемещение в каталог ~/work/arch-pc

Создаю папку lab05 с помощью функциональной клавиши F7 (рис. 2.3) и переможу в нее (рис. 2.4).

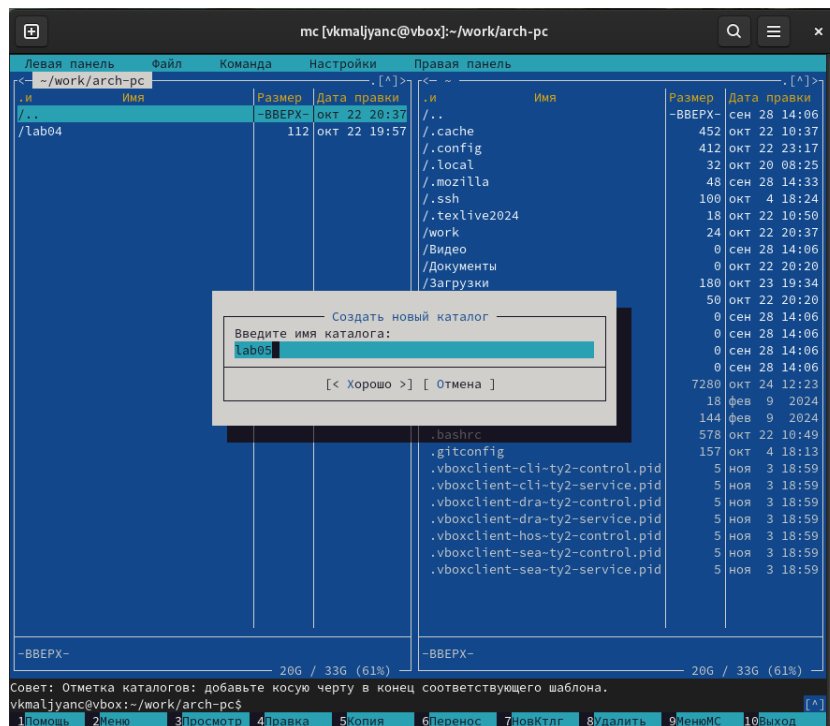


Рис. 2.3: Создание папки lab05

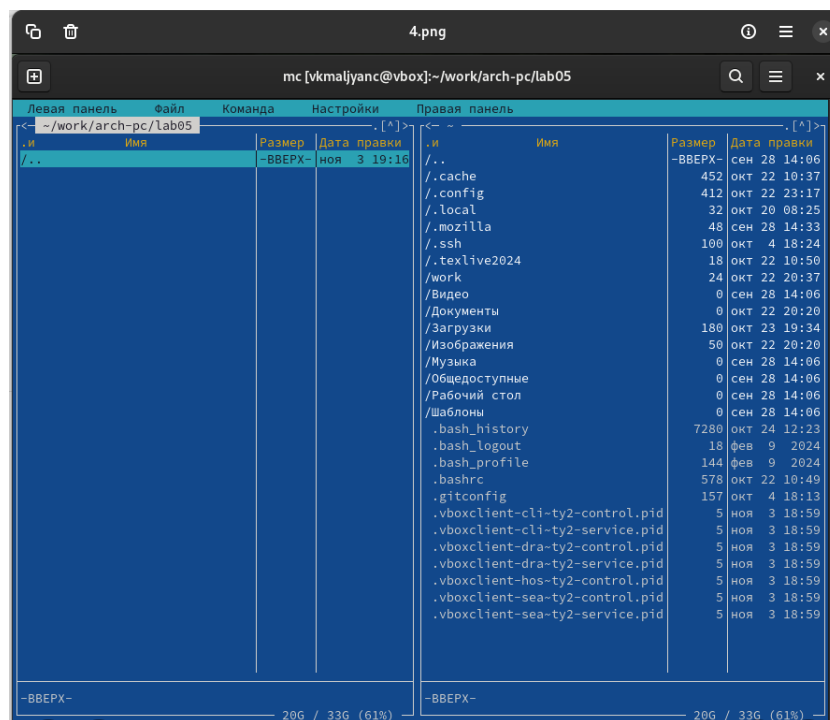


Рис. 2.4: Переход в папку lab05



В строке ввода с помощью команды touch создаю файл lab-1.asm (рис. 2.5).



Рис. 2.5: Создание файла lab-1.asm

Открываю файл lab5-1.asm для редактирования во встроенном редакторе с помощью функциональной клавиши F4 (рис. 2.6).

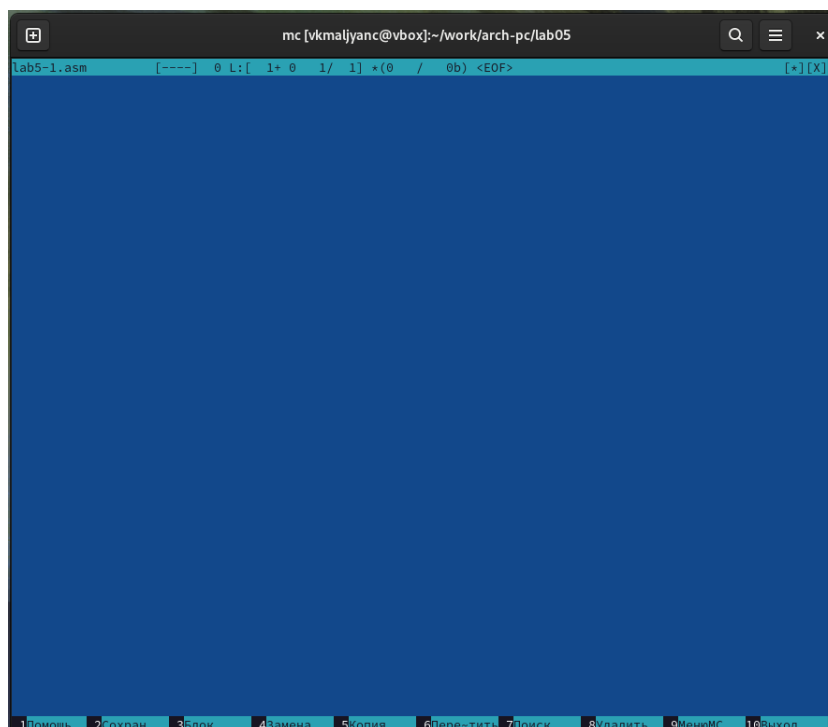
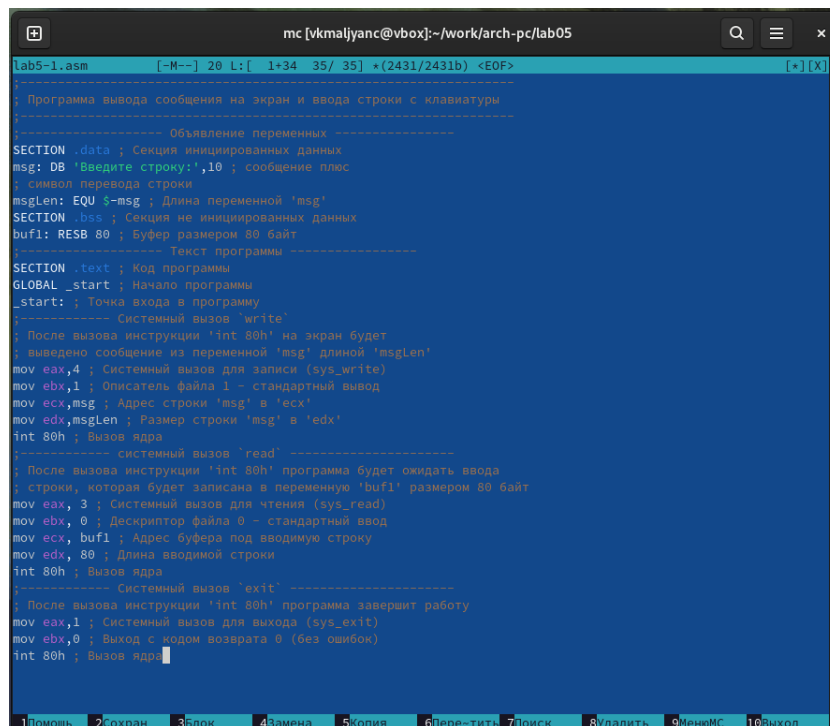


Рис. 2.6: Открытие файла lab5-1.asm для редактирования во встроенном редакторе

Ввожу текст программы вывода сообщения на экран и ввода строки с клавиатуры (рис. 2.7).



The screenshot shows a text editor window titled "mc [vkmaljanc@vbox]:~/work/arch-pc/lab05". The editor displays assembly code for a program. The code includes sections for data, bss, and text. It defines a message "Введите строку:", its length, and a buffer. The program uses system calls to print the message and read user input. The code is as follows:

```
lab5-1.asm [-M--] 20 L: [ 1+34 35/ 35] *(2431/2431b) <EOF> [*] [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 ; Файловый дескриптор 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 ; Файловый дескриптор 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
```

At the bottom of the window, there is a toolbar with icons and labels for various actions: 1Помощь, 2Сохран, 3Блок, 4Замена, 5Копия, 6Пере-тит, 7Поиск, 8/далить, 9МенюМС, 10Выход.

Рис. 2.7: Ввод текста программы

Сохраняю изменения с помощью функциональной клавиши F2 (рис. 2.8).

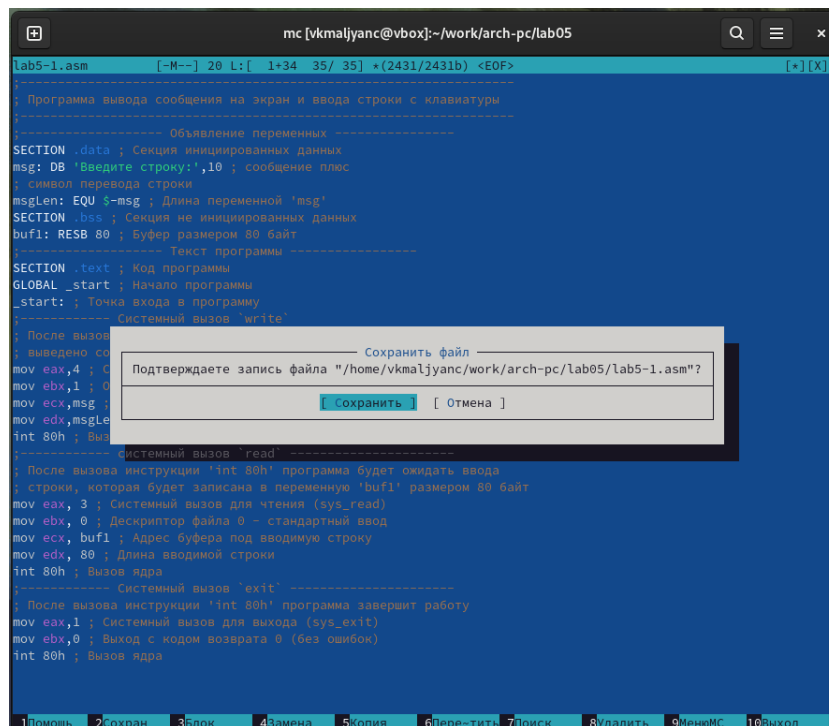


Рис. 2.8: Сохранение изменений

Выхожу из редактора с помощью функциональной клавиши F10 (рис. 2.9).

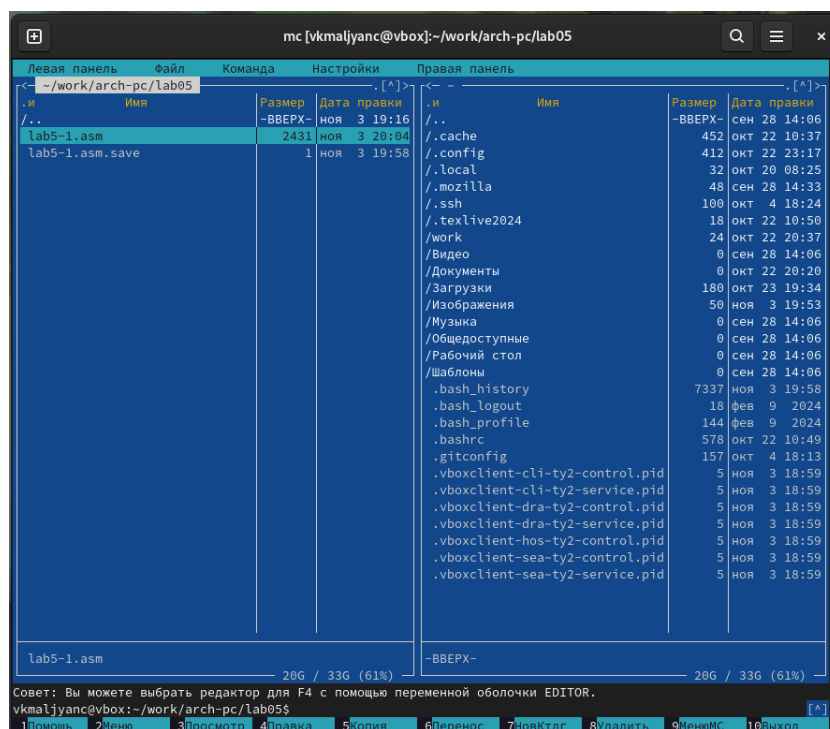
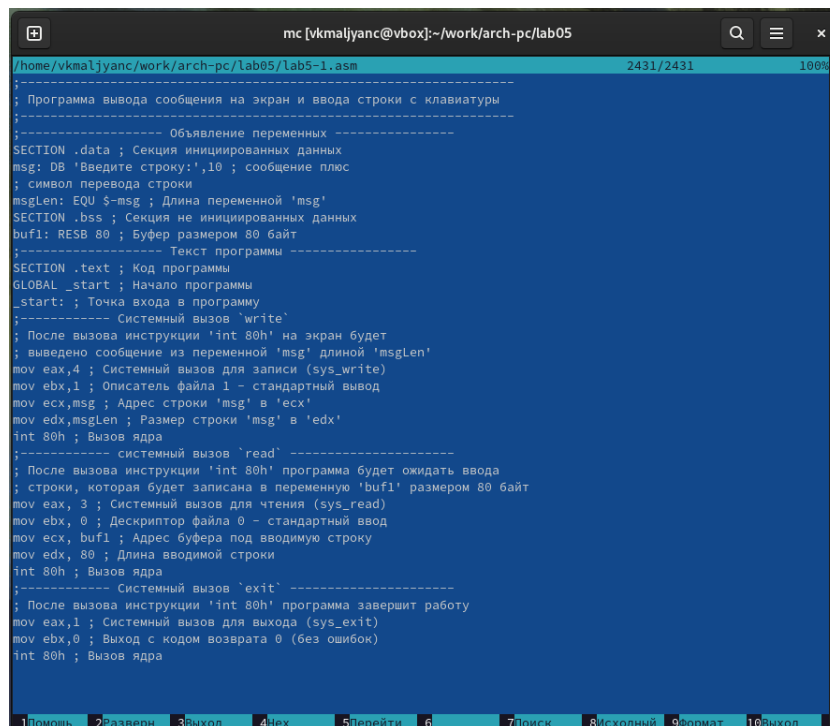


Рис. 2.9: Выход из редактора

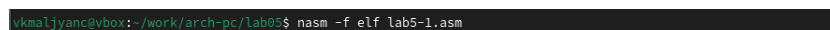
Открываю файл lab5-1.asm для просмотра с помощью функциональной клавиши F3. Убеждаюсь, что файл содержит текст программы (рис. 2.10).



```
mc [vkmal'jyanc@vbox]:~/work/arch-pc/lab05
/home/vkmal'jyanc/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 4Нех 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход
```

Рис. 2.10: Просмотр содержимого файла lab5-1.asm

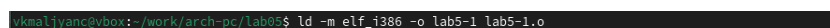
Транслирую текст программы lab5-1.asm в объектный файл (рис. 2.11).



```
vkmal'jyanc@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
```

Рис. 2.11: Транслирование текста в объектный файл

Передаю объектный файл на обработку компоновщику LD (рис. 2.12).



```
vkmal'jyanc@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
```

Рис. 2.12: Компоновка объектного файла

Запускаю получившийся исполняемый файл (рис. 2.13).



```
vkmal'jyanc@vbox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Мальняц Виктория Кареновна
```

Рис. 2.13: Запуск исполняемого файла

Скачиваю файл in\_out\_asm со страницы курса в ТУИС, он сохранился в каталог Загрузки (рис. 2.14).

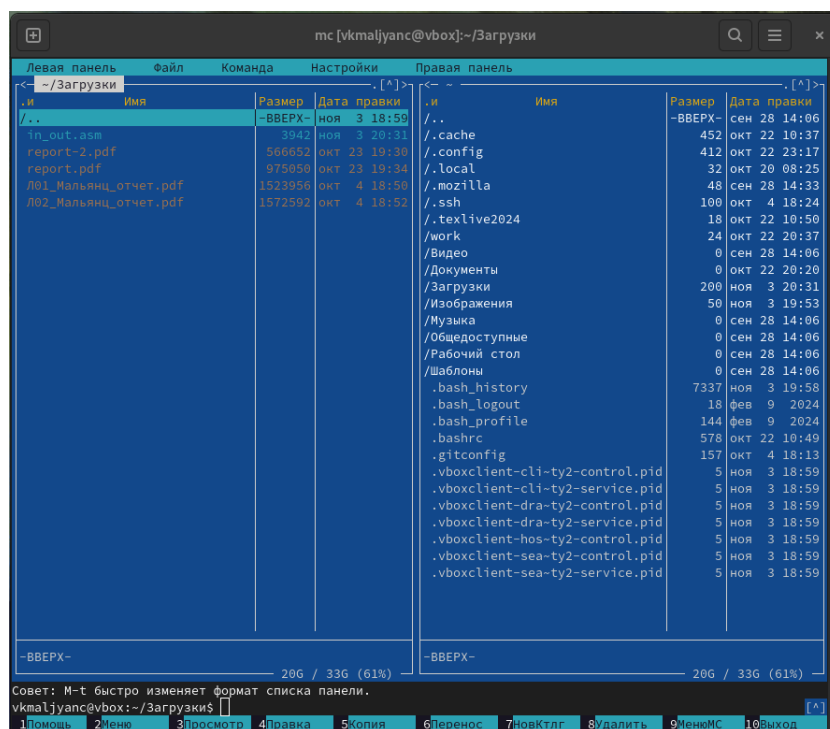


Рис. 2.14: Скачанный файл in\_out\_asm

Копирую файл in\_out.asm из каталога Загрузки в каталог lab05 с помощью функциональной клавиши F5 (рис. 2.15).

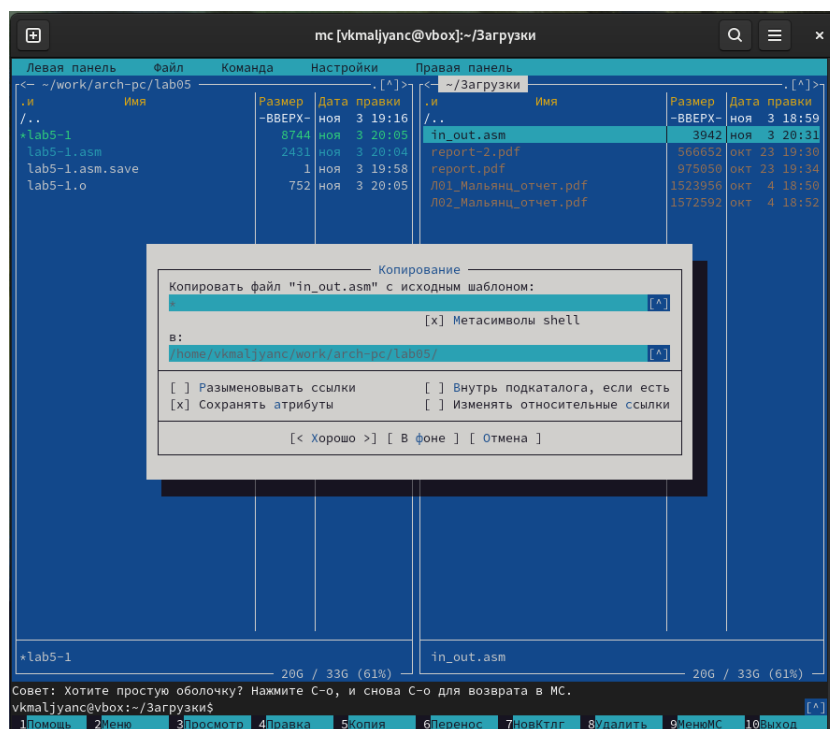


Рис. 2.15: Копирование файла in\_out.asm в каталог lab05

Создаю копию файла lab5-1.asm с именем lab5-2.asm с помощью функциональной клавиши F6 (рис. 2.16).

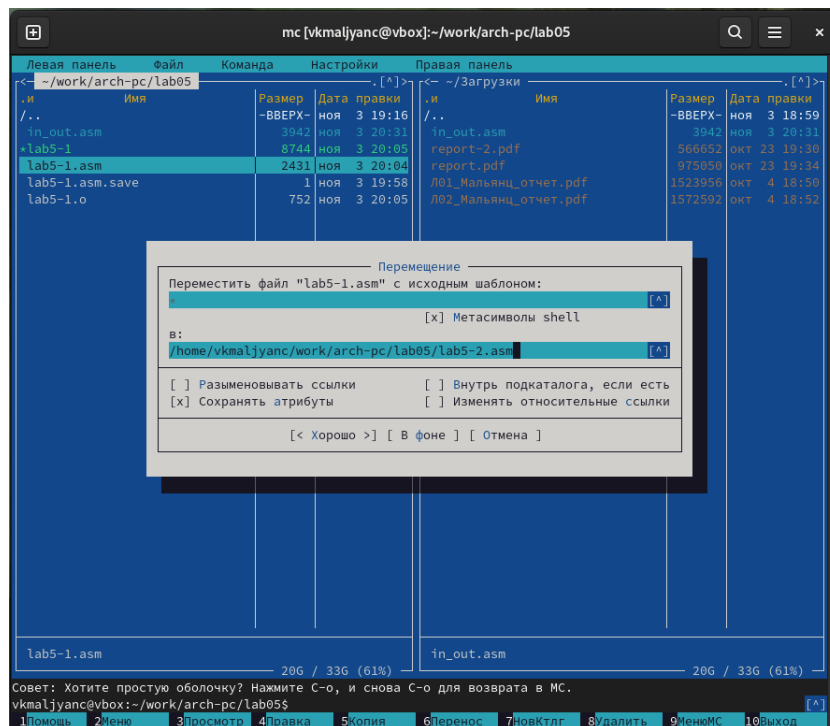


Рис. 2.16: Создание копии файла lab5-1.asm с именем lab5-2.asm

Исправляю текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in\_out.asm (использую подпрограммы sprintLF, sread и quit) (рис. 2.17).



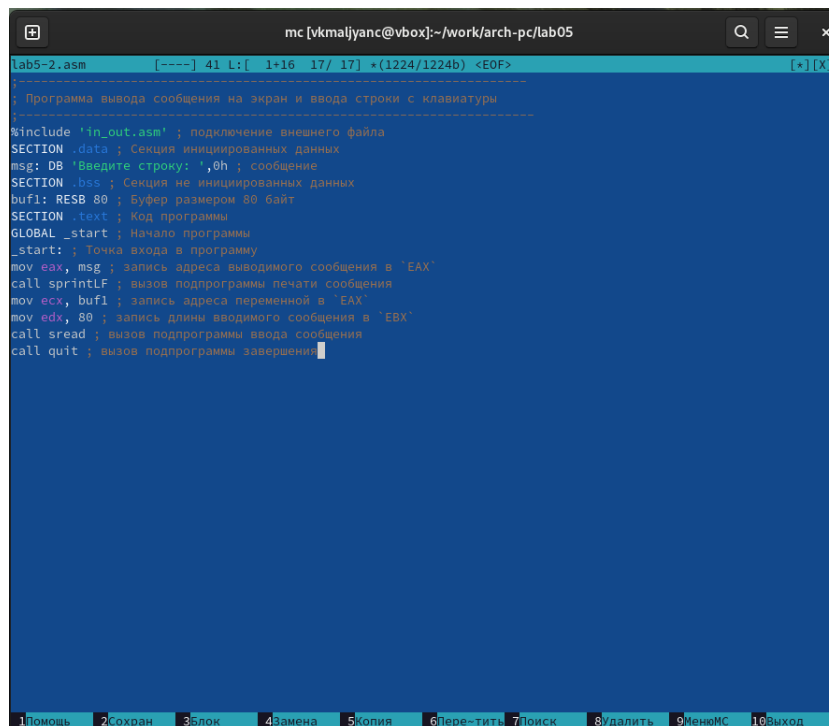


Рис. 2.17: Редактирование текста программы в файле lab5-2.asm

Транслирую текст программы lab5-2.asm в объектный файл, передаю объектный файл на обработку компоновщику LD, запускаю получившийся исполняемый файл (рис. 2.18).

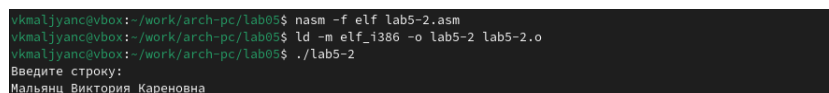


Рис. 2.18: Исполнение файла

Заменяю подпрограмму sprintf на sprint в файле lab5-2.asm (рис. 2.19).

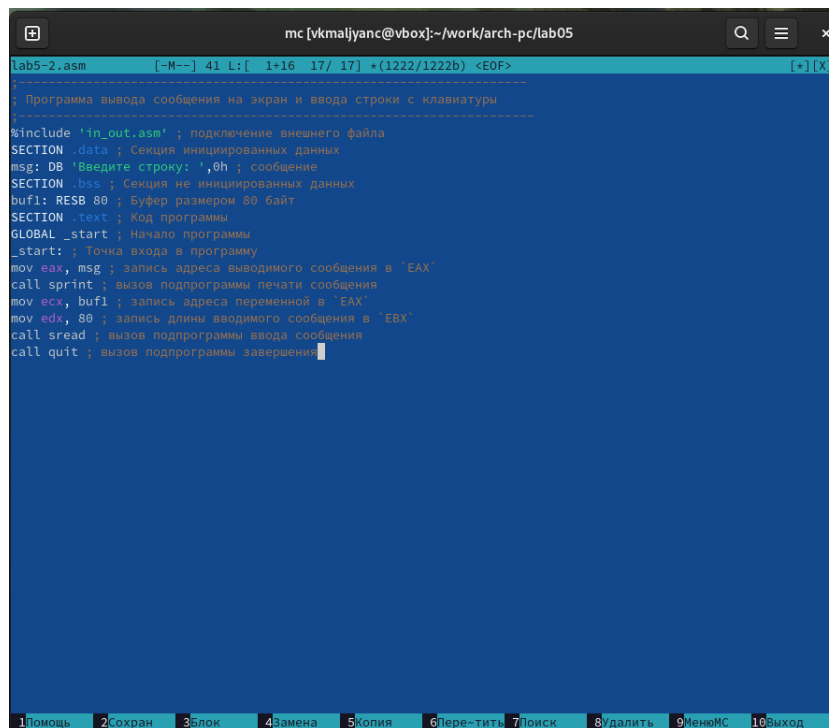


Рис. 2.19: Редактирование текста программы в файле lab5-2.asm

Транслирую текст программы lab5-2.asm в объектный файл, передаю объектный файл на обработку компоновщику LD, запускаю получившийся исполняемый файл (рис. 2.20).

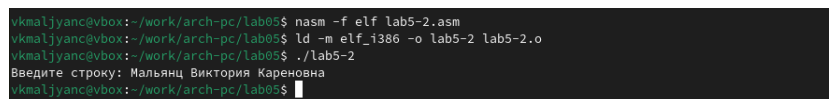


Рис. 2.20: Исполнение файла

Разница между первым (с подпрограммой sprintLF) и вторым (с подпрограммой sprint) исполняемыми файлами заключается в том, что в первом исполняемом файле запуск запрашивает ввод с новой строки, а во втором исполняемом файле запуск запрашивает ввод без переноса на новую строку.

## 2.1 Выполнение заданий для самостоятельной работы

1. Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5 (рис. 2.21).

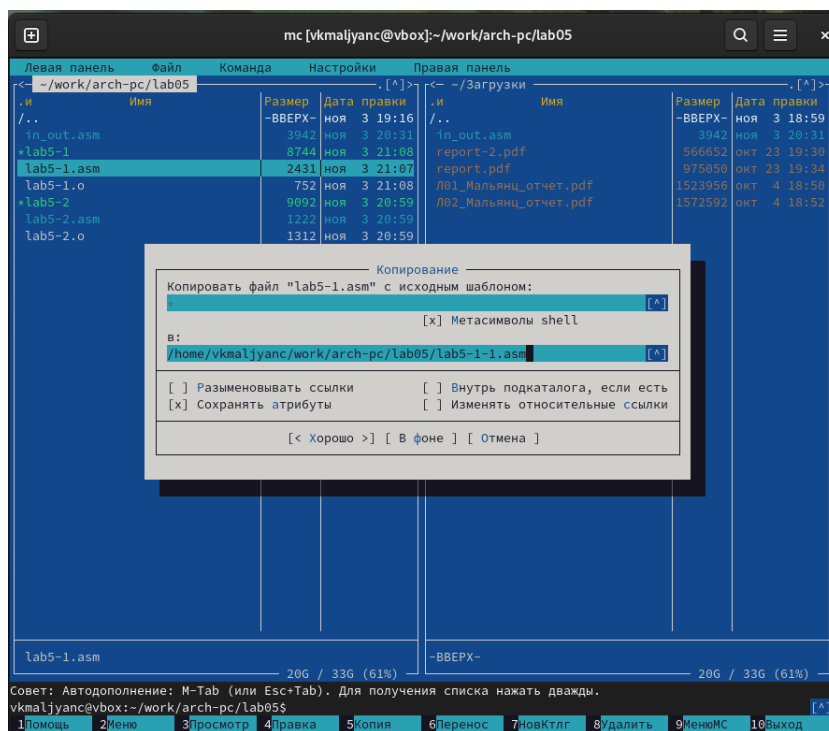
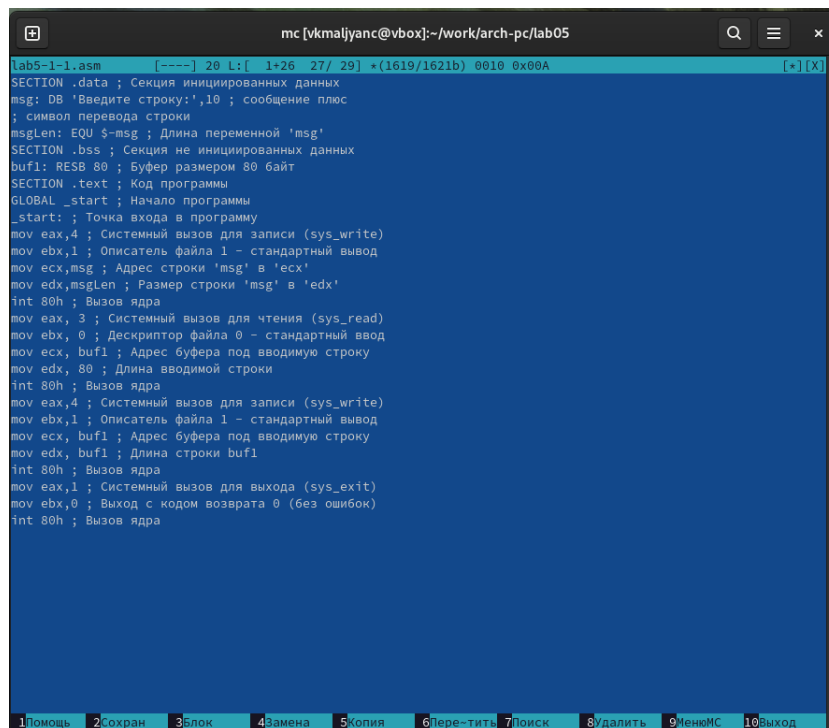


Рис. 2.21: Создание копии файла lab5-1.asm с именем lab5-1-1.asm

Открываю файл lab5-1-1.asm для редактирования во встроенном редакторе с помощью функциональной клавиши F4. Ввожу текст программы вывода сообщения на экран, ввода строки с клавиатуры и вывода введенной строки на экран (рис. 2.22). Сохраняю изменения с помощью функциональной клавиши F2. Выхожу из редактора с помощью функциональной клавиши F10.



```
mc [vkmal]yanc@vbox:~/work/arch-pc/lab05
lab5-1-1.asm [----] 20 L: [ 1+26 27/ 29] *(1619/1621b) 0010 0x00A [+] [X]
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 .text ; 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,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,buf1 ; Адрес буфера под вводимую строку
mov edx,buf1 ; Длина строки buf1
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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 2.22: Ввод текста программы

Листинг программы:

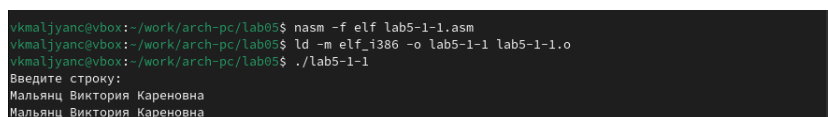
```
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_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, 4 ; Системный вызов для записи (sys_write)
mov ebx, 1 ; Описатель файла 1 - стандартный вывод
mov ecx, buf1 ; Адрес буфера под вводимую строку
mov edx, buf1 ; Длина строки buf1
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

```

2. Транслирую текст программы lab5-1-1.asm в объектный файл, передаю объектный файл на обработку компоновщику LD, запускаю получившийся исполняемый файл. Ввожу свои ФИО. Программа выводит введенные мною данные (рис. 2.23).



```

ykmaljjanc@vbox: ~/work/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
ykmaljjanc@vbox: ~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
ykmaljjanc@vbox: ~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Мальянц Виктория Кареновна
Мальянц Виктория Кареновна

```

Рис. 2.23: Исполнение файла

3. Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5 (рис. 2.24).

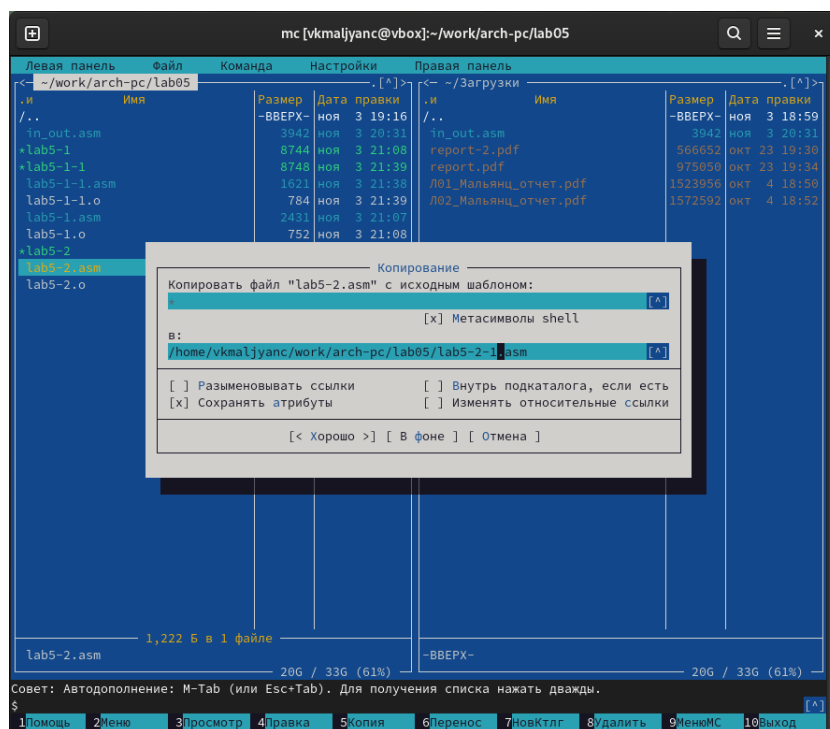


Рис. 2.24: Создание копии файла lab5-2.asm с именем lab5-2-1.asm

Открываю файл lab5-2-1.asm для редактирования во встроенном редакторе с помощью функциональной клавиши F4. Ввожу текст программы вывода сообщения на экран, ввода строки с клавиатуры и вывода введенной строки на экран (рис. 2.25). Сохраняю изменения с помощью функциональной клавиши F2. Выхожу из редактора с помощью функциональной клавиши F10.

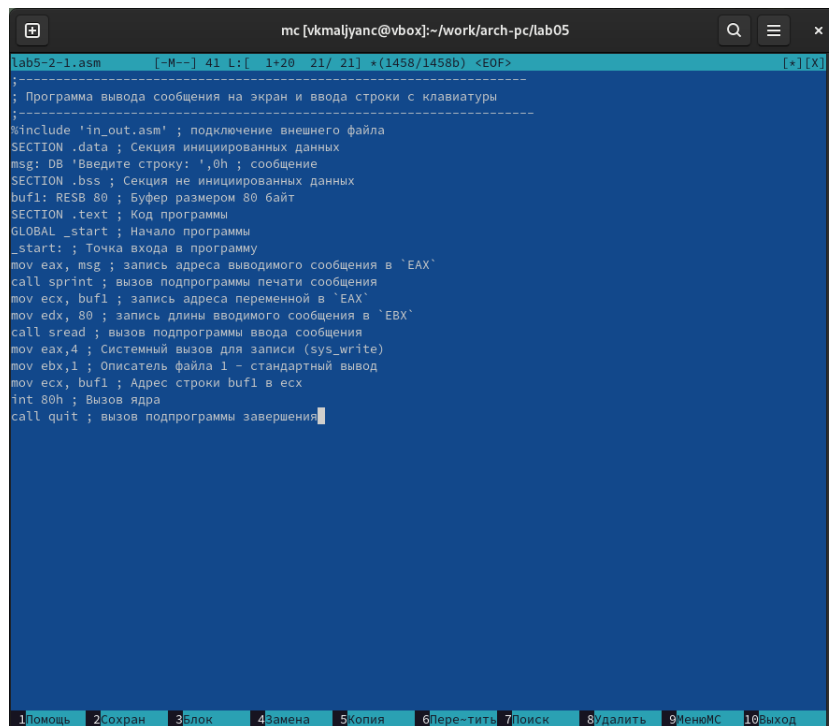


Рис. 2.25: Ввод текста программы

Листинг программы:

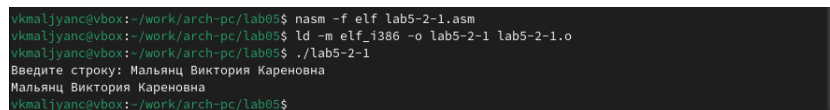
```

;-----
; 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
```

4. Транслирую текст программы lab5-2-1.asm в объектный файл, передаю объектный файл на обработку компоновщику LD, запускаю получившийся исполняемый файл. Ввожу свои ФИО. Программа выводит введенные мною данные (рис. 2.26).



```
vkmaljanc@vbox: ~/work/arch-pc/lab05$ nasm -f elf lab5-2-1.asm  
vkmaljanc@vbox: ~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o  
vkmaljanc@vbox: ~/work/arch-pc/lab05$ ./lab5-2-1  
Введите строку: Мальянц Виктория Кареновна  
Мальянц Виктория Кареновна  
vkmaljanc@vbox: ~/work/arch-pc/lab05$
```

Рис. 2.26: Исполнение файла



## 3 Выводы

Я приобрела практические навыки работы в Midnight Commander. Освоила инструкции языка ассемблера `mov` и `int`.