

Отчёт по Лабораторной работе №5

дисциплина: Архитектура компьютера

Гончарь Анастасия Александровна

Содержание

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 `mc` 2. Подключение внешнего файла `in_out.asm` 3. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. 1 приведено краткое описание стандартных каталогов Unix.

Таблица 1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую
/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

Имя	
катал	
ога	Описание каталога
<hr/>	
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

4 Выполнение лабораторной работы

4.1 Основы работы с mc

Сначала откроем Midnight Commander, введя в терминал mc (рис. 1).

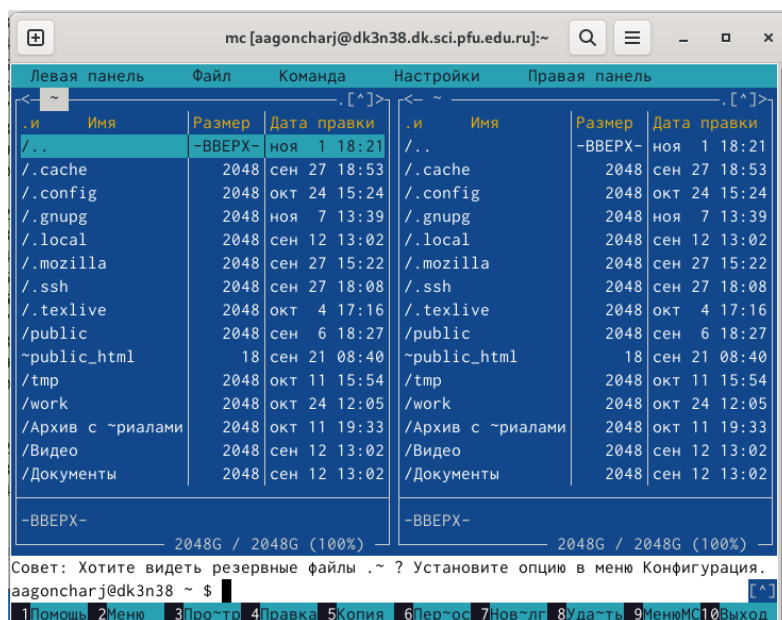


Рис. 1: Открытый mc

Переходим в каталог ~/work/arch-pc (рис. 2)

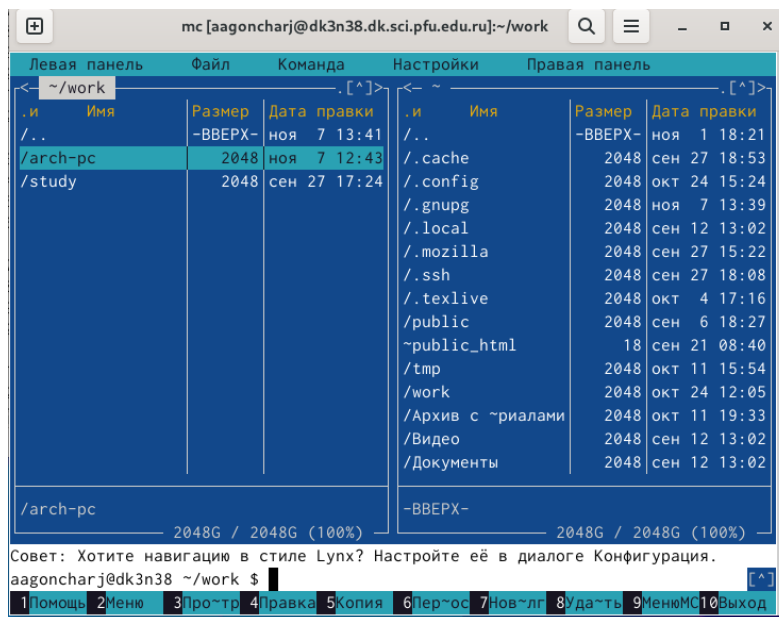


Рис. 2: Перемещение между директориями

С помощью функциональной клавиши F7 создаем каталог lab05 (рис. 3).

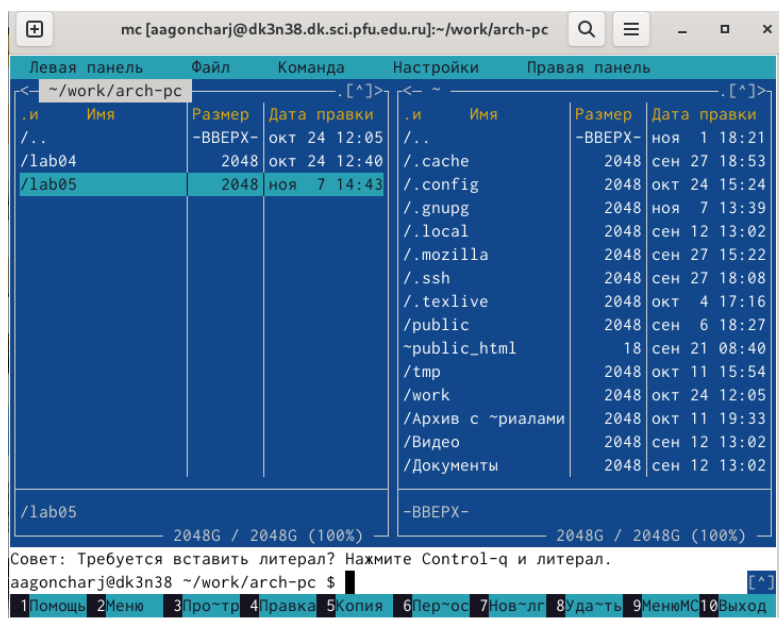


Рис. 3: Создание каталога

Переходим в созданный каталог (рис. 4).

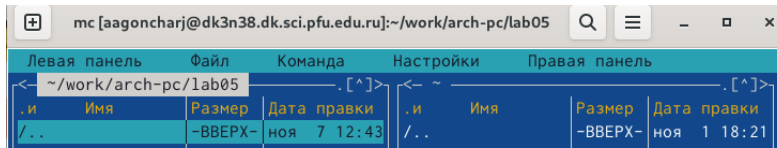


Рис. 4: Перемещение между директориями

Пользуясь строкой ввода и командой `touch lab5-1.asm`, создаю файл `lab5-1.asm` (рис. 5).

```
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ touch lab5-1.asm
```

Рис. 5: Создание файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе `mcedit` (рис. 6).

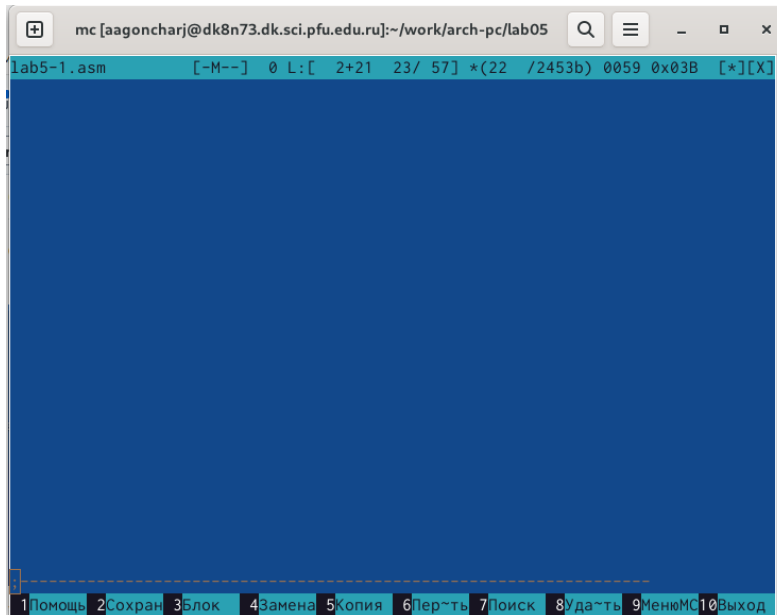
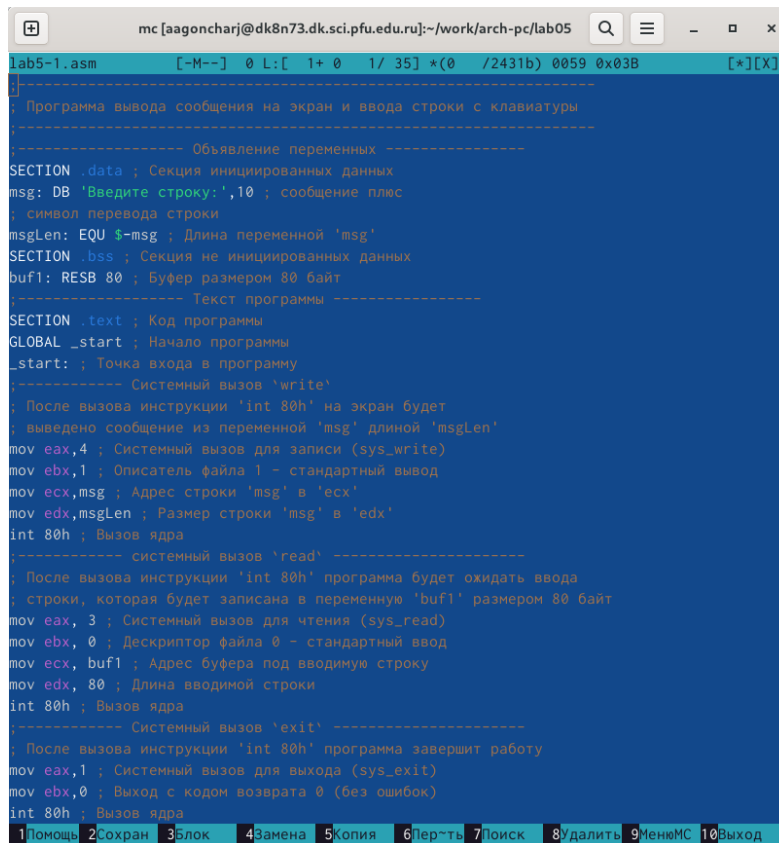


Рис. 6: Открытие файла для редактирования

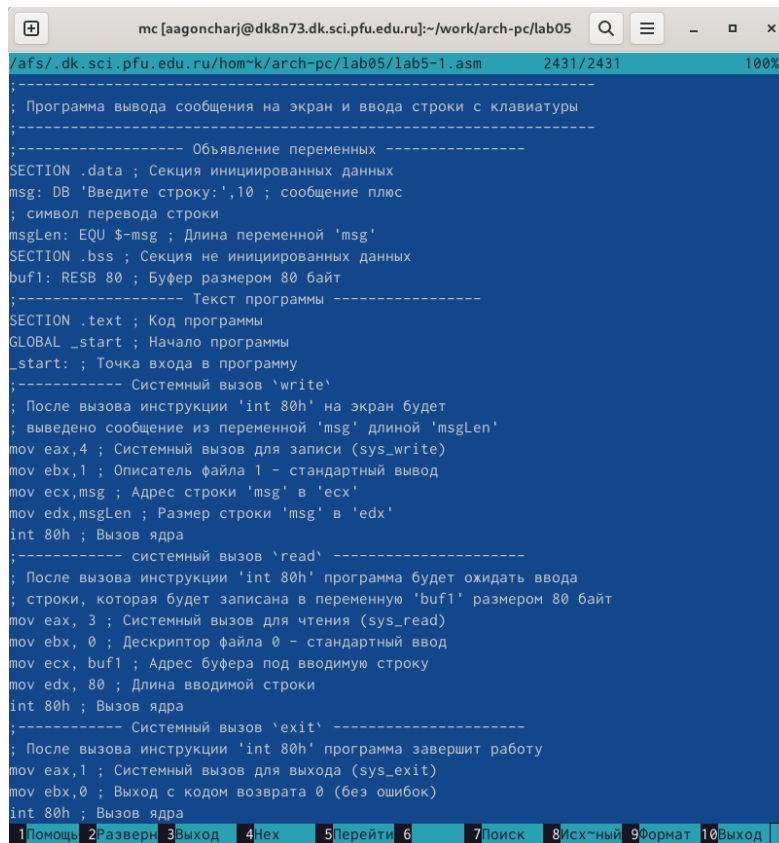
Ввожу в файл код программы вывода сообщения на экран и ввода строки с клавиатуры (рис. 7). Далее выхожу из файла, сохраняя изменения.



```
lab5-1.asm [-M--] 0 L:[ 1+ 0 1/ 35] *(0 /2431b) 0059 0x03B [*][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 '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 6Пер-ть 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 7: Редактирование файла

С помощью функциональной клавиши F3 открываю файл lab5-1.asm для просмотра (рис. 8).



```
mc [aagoncharj@dk8n73.dk.sci.pfu.edu.ru]:~/work/arch-pc/lab05
/afs/.dk.sci.pfu.edu.ru/hom-k/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 2Развернуть 3Выход 4Нех 5Перейти 6 7Поиск 8Исх-ный 9Формат 10Выход
```

Рис. 8: Открытие файла для просмотра

Теперь необходимо оттранслировать текст программы в объектный файл и выполнить компоновку объектного файла с помощью команд `nasm -f elf lab5-1.asm`, `ld -m elf_i386 -o lab5-1 lab5-1.o, ./lab5-1` (рис. 9).

```
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ls
lab5-1.asm  lab5-1.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ls
lab5-1  lab5-1.asm  lab5-1.o
```

Рис. 9: Компиляция файла и передача на обработку компоновщику

После запуска программы я ввожу свои ФИО (рис. 10).

```
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку:
Гончарь Анастасия Александровна
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $
```

Рис. 10: Исполнение файла

4.2 Подключение внешнего файла in_out.asm

Скачиваю файл in_out.asm со страницы курса в ТУИС, который сохранится в каталог “Загрузки”. (рис. 11).

Левая панель	Файл	Команда	Настройки
< ~/Загрузки .[^>			
.и	Имя	Размер	Дата правки
/..		-ВВЕРХ-	ноя 7 13:41
1	курс_НММ_Гол~vNFL](1).xlsx	73214	ноя 7 12:58
1	курс_НММ_Гол~[00vNFL].xlsx	73214	ноя 7 12:58
1	История ЭВМ ~2018_ТУИС.pdf	2581763	ноя 7 12:02
	5249024204576253428.jpg	32228	окт 24 15:51
	hello.asm	338	окт 24 12:08
	in_out.asm	3942	ноя 7 13:45
	report.md	5819	окт 4 16:50

Рис. 11: Скачанный файл

Копирую файл in_out.asm из каталога “Загрузки” в созданный каталог lab05 (рис. 12).

Левая панель	Файл	Команда	Настройки
< ~/work/arch-pc/lab05 .[^>			
.и	Имя	Размер	Дата правки
/..		-ВВЕРХ-	ноя 7 12:43
	in_out.asm	3942	ноя 7 13:45
	*lab5-1	8744	ноя 7 13:43
	lab5-1.asm	2431	ноя 7 12:51
	lab5-1.o	752	ноя 7 13:42
	lab5-2.asm	2431	ноя 7 12:51

Рис. 12: Копирование файла

С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем (рис. 13).

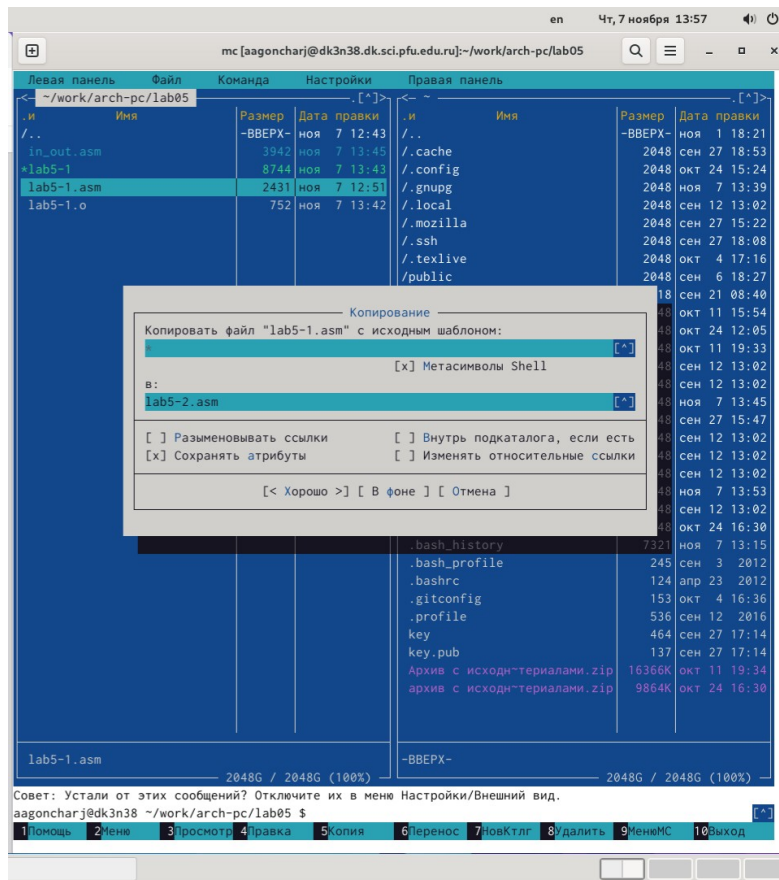


Рис. 13: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе, чтобы в программе использовались подпрограммы из внешнего файла in_out.asm (рис. 14).

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

Также изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий (рис. 15).


```

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. 15: Редактирование файла

Транслирую файл, выполняю компоновку созданного объектного файла и запускаю новый исполняемый файл (рис. 16).

```

aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ls
in_out.asm  lab5-1  lab5-1.asm  lab5-1.o  lab5-2  lab5-2.asm  lab5-2.o  nano.3402.save
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-2
bash: ./lab5-2: Нет такого файла или каталога
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку: Гончарь Анастасия Александровна
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $

```

Рис. 16: Исполнение файла

4.3 Выполнение заданий для самостоятельной работы

Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5 (рис. 17).

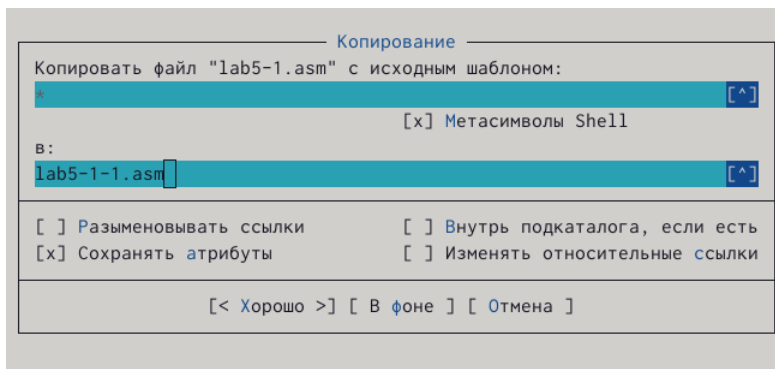


Рис. 17: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования и изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 18).

```
lab5-1-1.asm [-----] 20 L: [ 1+21 22/ 26] *(1290/1521b) 1088 0x440 [*][X]
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10
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 строки buf1 в ecx
mov edx,buf1 ; Размер 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
```

Рис. 18: Редактирование файла

Код программы для файла lab5-1-1.asm:

```
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10
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 строки buf1 в ecx
mov edx,buf1 ; Размер 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
```

Транслирую файл, выполняю компоновку созданного объектного файла и запускаю новый исполняемый файл. Далее ввожу свои ФИО и программа выводит введенные мною данные (рис. 19).

```

aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1-1.asm
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ls
in_out.asm  lab5-1-1      lab5-1-1.o  lab5-1.o  lab5-2.asm  nano.3402.save
lab5-1      lab5-1-1.asm lab5-1.asm  lab5-2     lab5-2.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-1-1
Введите строку:
Гончарь Анастасия Александровна
Гончарь Анастасия Александровна
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ █

```

Рис. 19: Исполнение файла

Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5 (рис. 20).

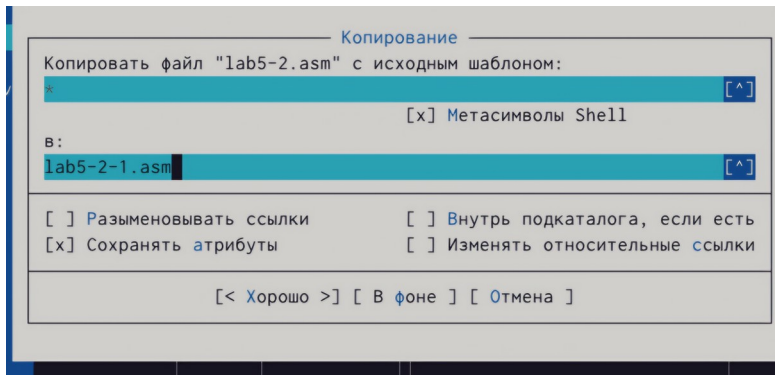


Рис. 20: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 21).

```

lab5-2-1.asm  [-M--] 41 L: [ 1+17 18/ 18] *(1145/1145b) <EOF>  [*][X]
#include 'in_out.asm'
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. 21: Редактирование файла

Код программы для файла lab5-2-1.asm:

```

#include 'in_out.asm'
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

```

Транслирую файл, выполняю компоновку созданного объектного файла и запускаю новый исполняемый файл. Далее ввожу свои ФИО, при этом программа запрашивает ввод без переноса на новую строку, и программа выводит введенные мною данные (рис. 22).

```

aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2-1.asm
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ls
in_out.asm  lab5-1-1  lab5-1-1.o  lab5-1.o  lab5-2-1  lab5-2-1.o  lab5-2.o
lab5-1      lab5-1-1.asm  lab5-1.asm  lab5-2    lab5-2-1.asm  lab5-2.asm  nano.3402.save
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ ./lab5-2-1
Введите строку: Гончарь Анастасия Александровна
Гончарь Анастасия Александровна
aagoncharj@dk3n38 ~/work/arch-pc/lab05 $ █

```

Рис. 22: Исполнение 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.