

Отчет по лабораторной работе №5

Основы работы с Midnight Commander (mc)

Югай Александр Витальевич

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Порядок выполнения лабораторной работы	7
3.1.1	Задание для самостоятельной работы	13
4	Выводы	18

Список иллюстраций

3.1	Открываем mc	7
3.2	Переходим в каталог	8
3.3	Создаем каталог функциональной клавишей F7	8
3.4	Создаем файл командой touch	9
3.5	Открываем файл функциональной клавишей, заполняем и сохраняем	9
3.6	Просматриваем файл для проверки	10
3.7	Проверяем как работает программа	10
3.8	Скачиваем файл	11
3.9	Копируем скаченный файл	11
3.10	Создаем копию файла и проверяем	12
3.11	Заполняем файл	12
3.12	Проверяем работу программы	12
3.13	Редактируем файл	13
3.14	Проверяем работу файла и сравниваем с прошлой	13
3.15	Создаем копию файла lab5-1.asm	14
3.16	Редактируем файл	15
3.17	Проверяем правильность написания программы	15
3.18	Создаем копию файла lab5-2.asm	16
3.19	Редактируем файл	16
3.20	Проверяем правильность написания программы	17

Список таблиц

1 Цель работы

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

2 Задание

Написать 2 программы по примеру и внести изменения по их условию

3 Выполнение лабораторной работы

3.1 Порядок выполнения лабораторной работы

Откройте Midnight Commander

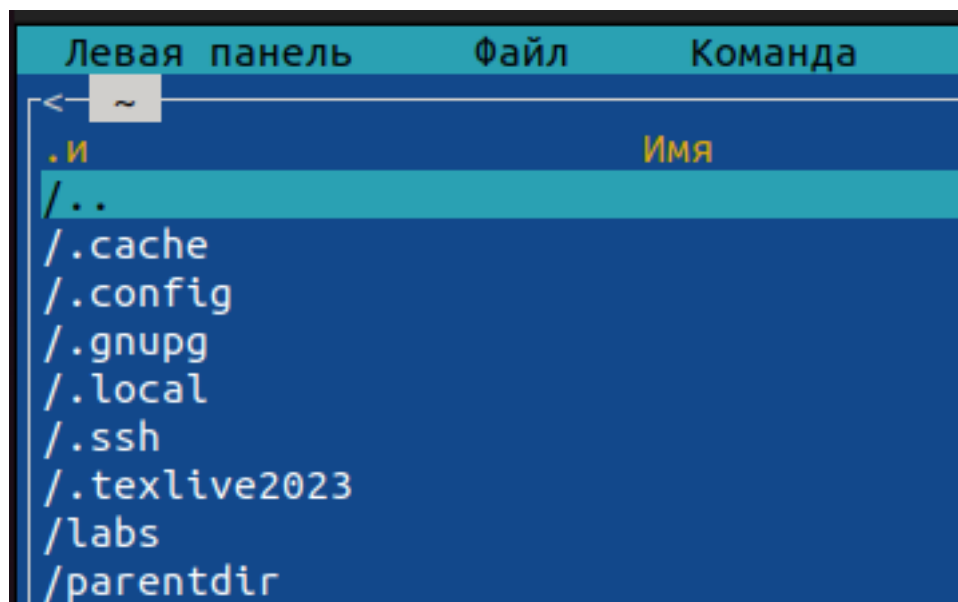


Рис. 3.1: Открываем mc

Переходим в каталог, созданный при выполнении 4 лабораторной работы

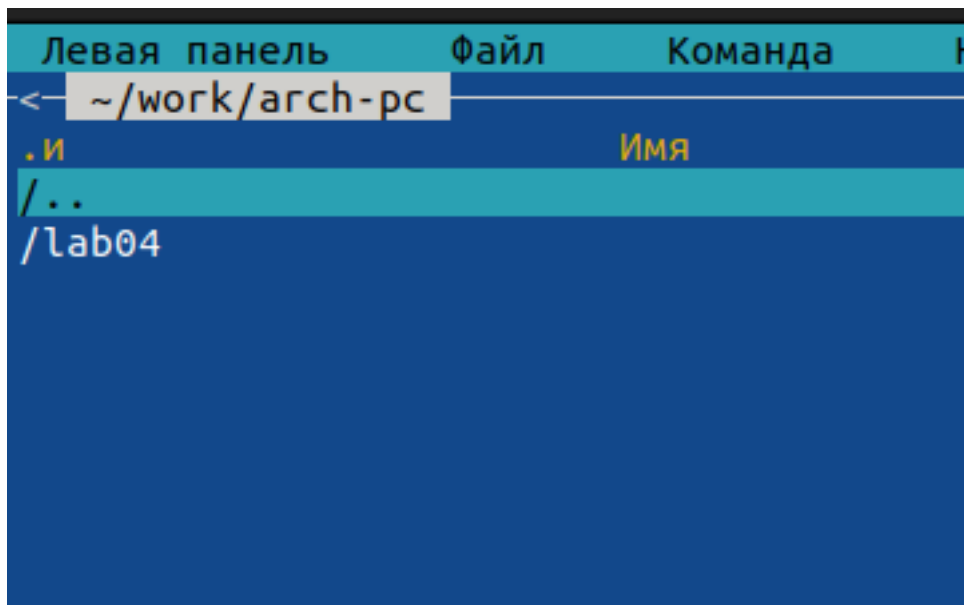


Рис. 3.2: Переходим в каталог

Создаем каталог lab05

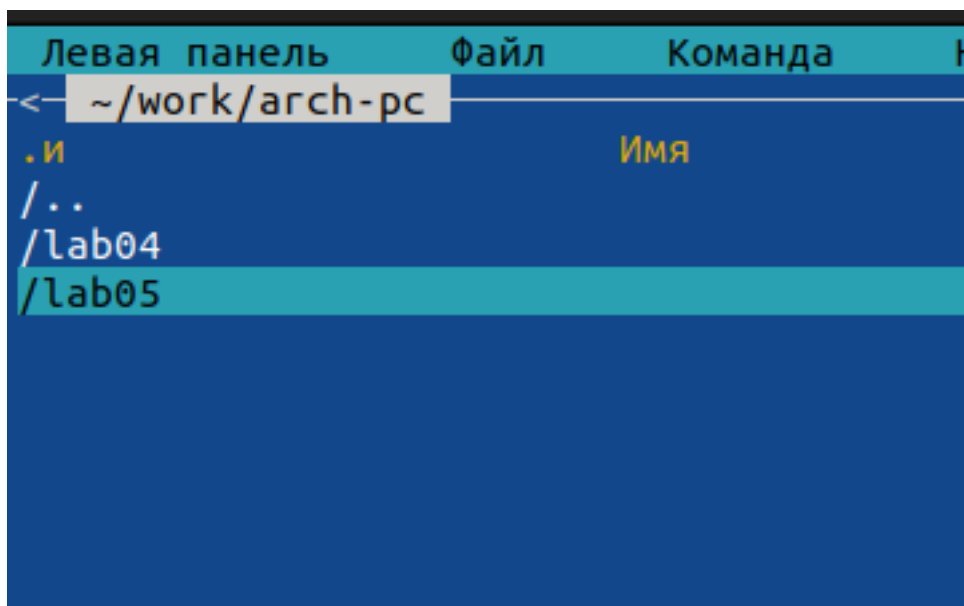


Рис. 3.3: Создаем каталог функциональной клавишей F7

Создаем файл lab5-1.asm

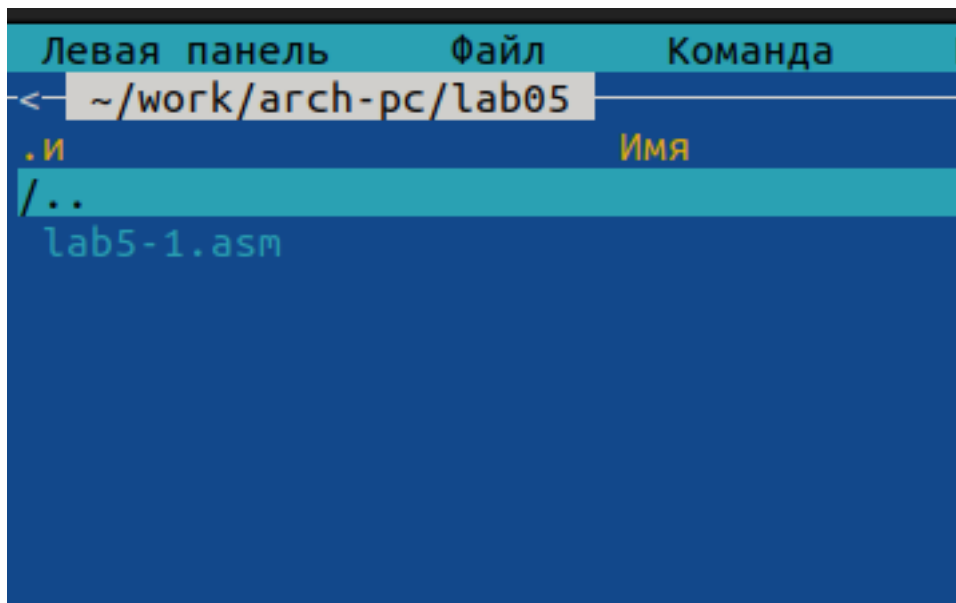


Рис. 3.4: Создаем файл командой touch

Открываем файл для редактирования и заполняем его по листингу

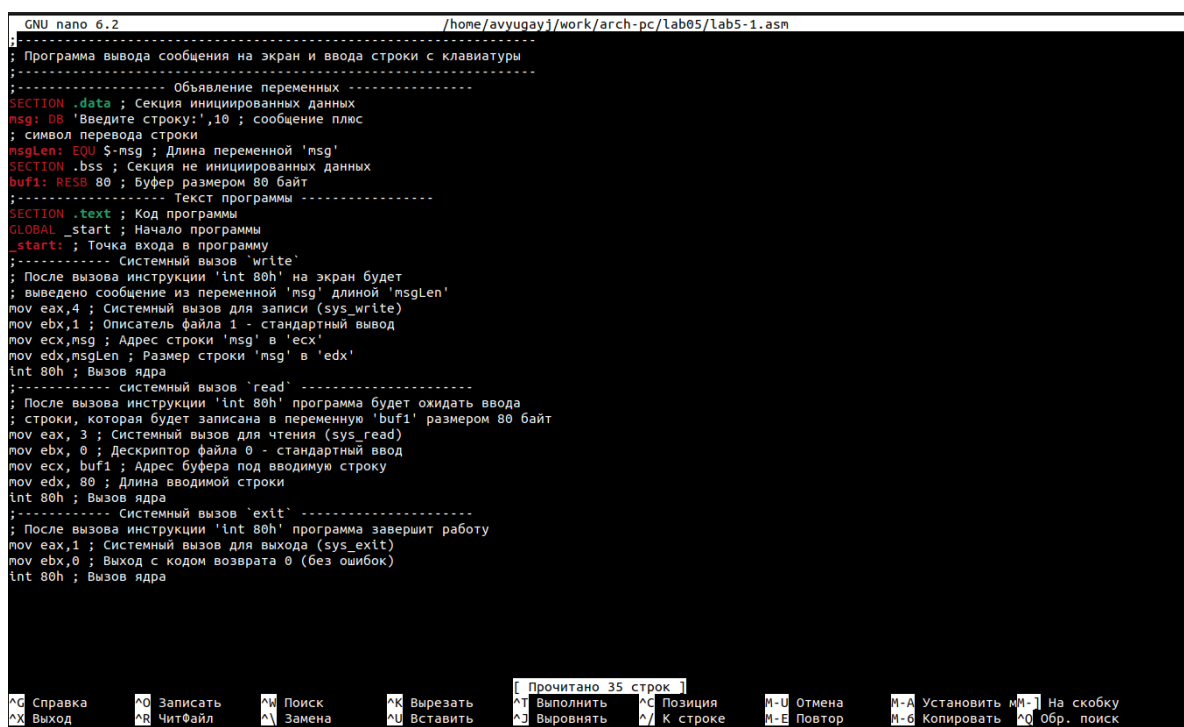
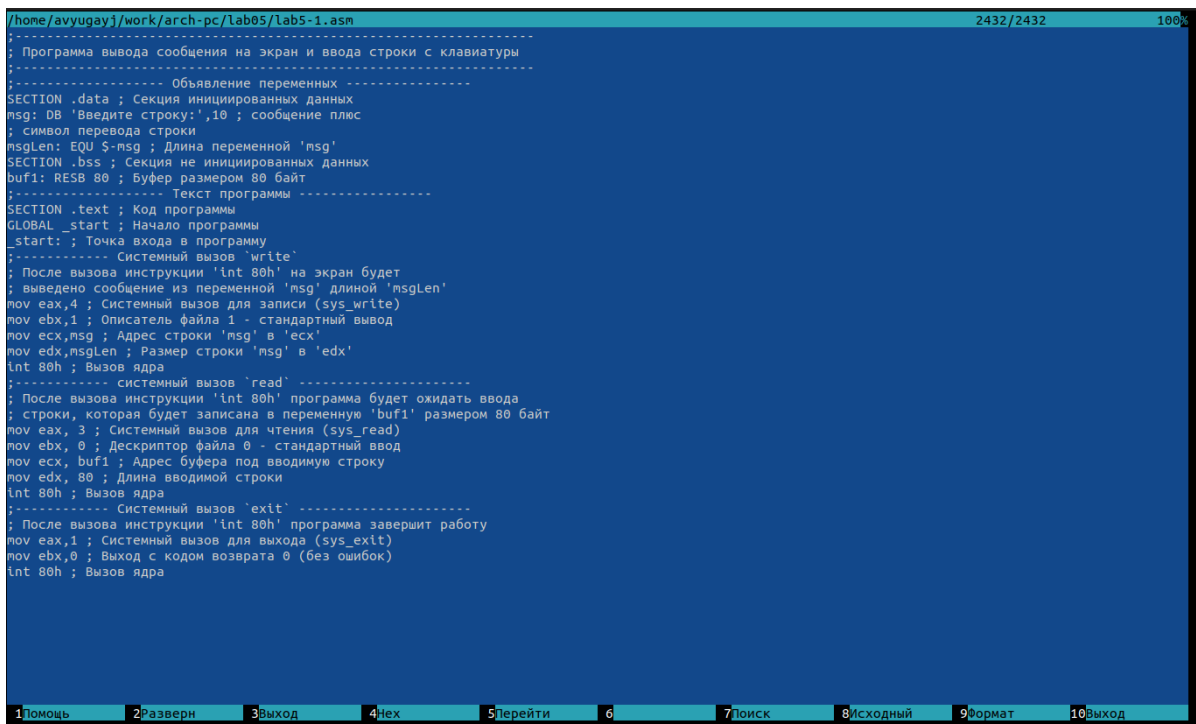


Рис. 3.5: Открываем файл функциональной клавишей, заполняем и сохраняем

Открываем файл для просмотра



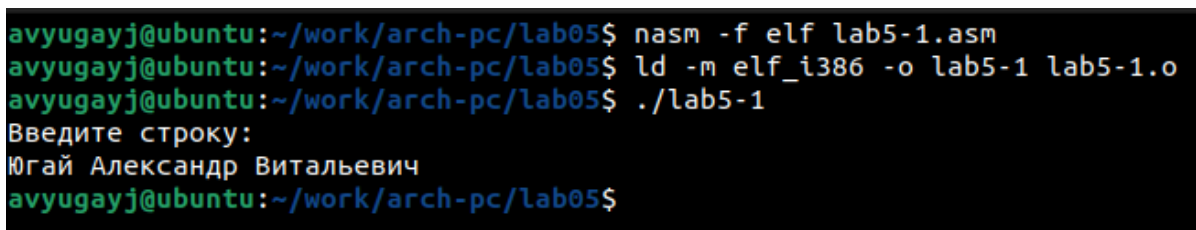
```

/home/avyugayj/work/arch-pc/lab05/lab5-1.asm 2432/2432 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

```

Рис. 3.6: Просматриваем файл для проверки

Транслируем текст программы и запускаем исполняемый файл



```

avyugayj@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
avyugayj@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
avyugayj@ubuntu:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Югай Александр Витальевич
avyugayj@ubuntu:~/work/arch-pc/lab05$

```

Рис. 3.7: Проверяем как работает программа

Скачиваем файл с туиса

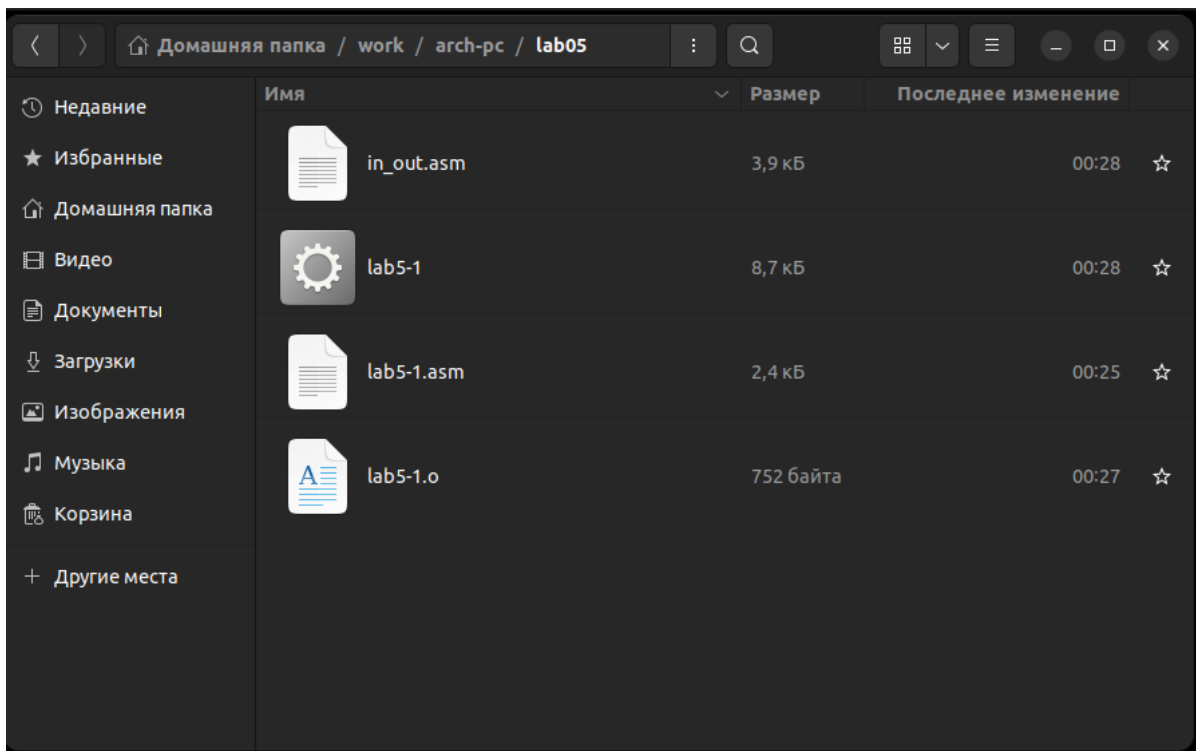


Рис. 3.8: Скачиваем файл

Копируем в нужную директорию

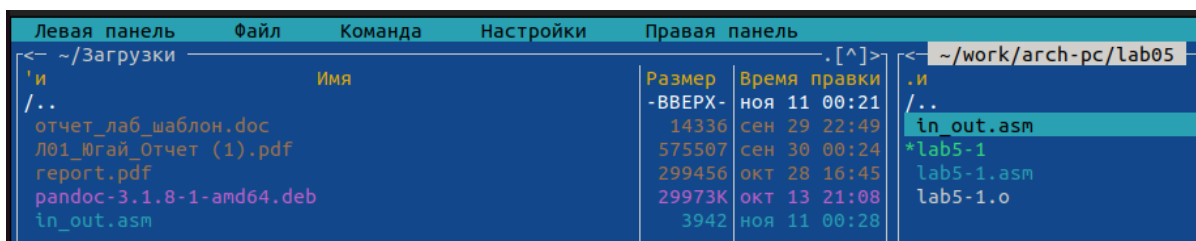


Рис. 3.9: Копируем скаченный файл

Создаем копию файла lab5-1.asm и проверяем

Левая панель	Файл	Команда	Настройки	Правая панель
~/work/arch-pc/lab05				~/. [^]> ~/work/arch-pc/lab05
Имя		Размер	Время правки	
/..		-ВВЕРХ-	ноя 11 00:19	./..
in_out.asm		3942	ноя 11 00:28	in_out.asm
*lab5-1		8744	ноя 11 00:28	*lab5-1
lab5-1.o		752	ноя 11 00:27	lab5-1.o
lab5-2.asm		2432	ноя 11 00:25	lab5-2.asm

Рис. 3.10: Создаем копию файла и проверяем

Открываем файл и заполняем в соответствии с листингом

```

-----
; 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
call quit ; вызов подпрограммы завершения

```

Рис. 3.11: Заполняем файл

Транслируем и запускаем файл

```

avyugayj@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
avyugayj@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
avyugayj@ubuntu:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Югай Александр Витальевич

```

Рис. 3.12: Проверяем работу программы

Снова открываем файл и меняем sprintf на sprint

```

;-----
; 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 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. 3.13: Редактируем файл

Транслируем файл и запускаем

```

avyugayj@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
avyugayj@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
avyugayj@ubuntu:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Югай Александр Витальевич
avyugayj@ubuntu:~/work/arch-pc/lab05$ █

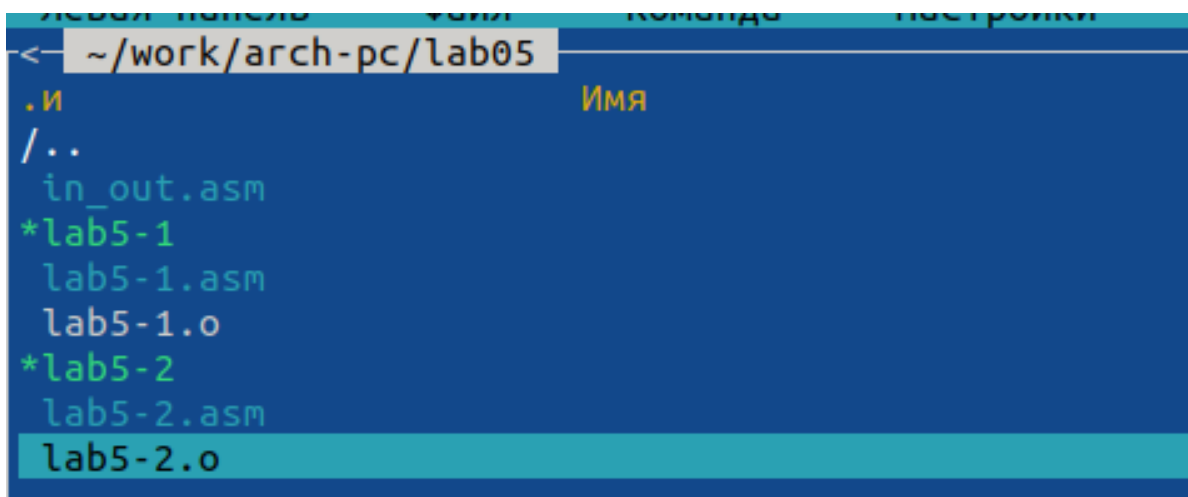
```

Рис. 3.14: Проверяем работу файла и сравниваем с прошлой

Замечаем, что sprint выводит текст в той же строке, а spintLF, выводит с новой

3.1.1 Задание для самостоятельной работы

Создаем копию файла lab5-1.asm и называем его также



The image shows a terminal window with a file manager interface. The top bar has tabs labeled "Лесенка", "панель", "файл", "команда", and "настройка". The address bar shows the path `~/work/arch-pc/lab05`. The main area displays a directory listing:

```
.и  
/..  
in_out.asm  
*lab5-1  
lab5-1.asm  
lab5-1.o  
*lab5-2  
lab5-2.asm  
lab5-2.o
```

Рис. 3.15: Создаем копию файла lab5-1.asm

Редактируем файл, чтобы введенный текст выводился в консоль

```

SECTION .data
msg: DB 'Введите строку:',10
msgLen: EQU $-msg
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start

_start:
mov eax,4
mov ebx,1
mov ecx,msg
mov edx,msgLen
int 80h
mov eax, 3
mov ebx, 0
mov ecx, buf1
mov edx, 80
int 80h

mov eax,4
mov ebx,1
mov ecx,buf1
mov edx,80
int 80h

mov eax,1
mov ebx,0
int 80h

```

Рис. 3.16: Редактируем файл

Транслируем файл и запускаем программу

```

avyugayj@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
avyugayj@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1
avyugayj@ubuntu:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Югай Александр Витальевич
Югай Александр Витальевич
avyugayj@ubuntu:~/work/arch-pc/lab05$

```

Рис. 3.17: Проверяем правильность написания программы

Создаем копию файла lab5-2.asm и называем его также

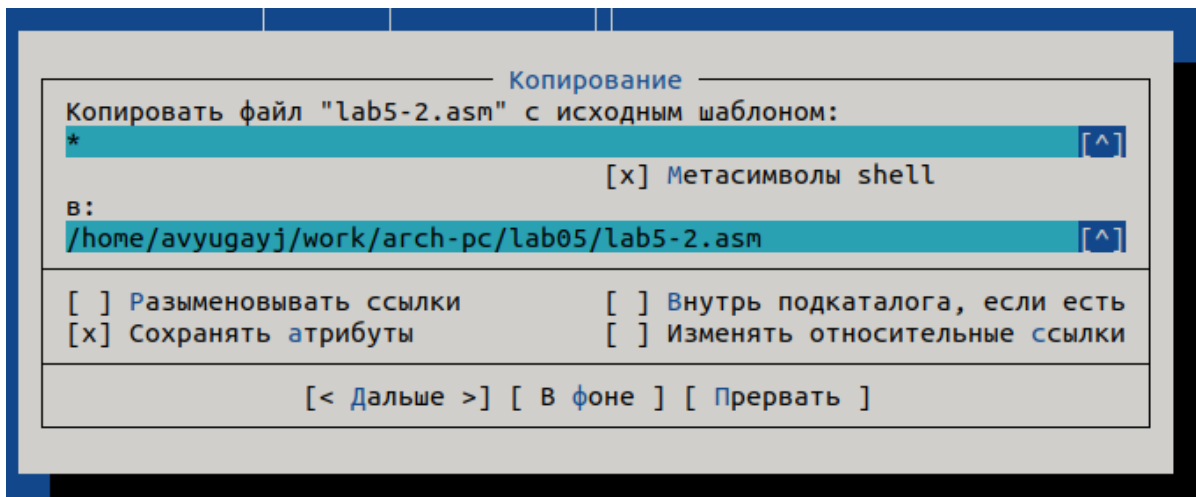


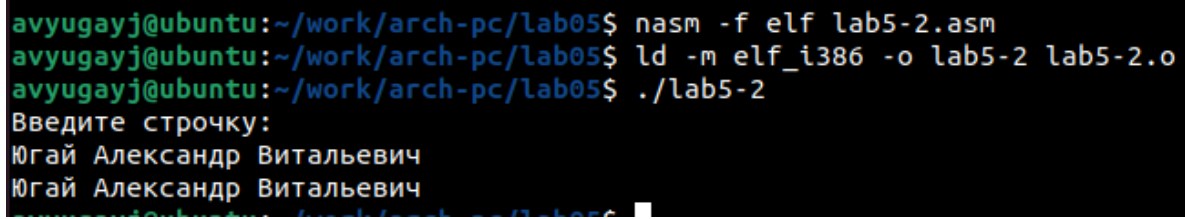
Рис. 3.18: Создаем копию файла lab5-2.asm

Редактируем файл, чтобы введенный текст, выводился в консоль

```
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите строку: ',0h
SECTION .bss
buf1: RESB 80
SECTION .text
    GLOBAL _start
    _start:
    mov eax,msg
    call sprintf
    mov ecx,buf1
    mov edx,80
    call sread
    mov eax,buf1
    call sprint
    call quit
```

Рис. 3.19: Редактируем файл

Транслируем файл и запускаем программу



```
avyugayj@ubuntu:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
avyugayj@ubuntu:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
avyugayj@ubuntu:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Югай Александр Витальевич
Югай Александр Витальевич
```

Рис. 3.20: Проверяем правильность написания программы

4 Выводы

Мы приобрели навыки работы с Midnight Commander и освоили инструкции `mov` и `int`