

Лабораторная работа №5

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

Богату Ирина Владимировна

Содержание

Цель работы	5
Выполнение лабораторной работы	6
Выполнение задания для самостоятельной работы	19
Выводы	23

Список иллюстраций

1	Запуск Midnight commander	6
2	Интерфейс midnight commander	7
3	Переход в нужный каталог (~/.work/arch-pc)	8
4	Создание папки	9
5	Создание файла lab5-1.asm с помощью команды touch прямо в mc	10
6	Выбор текстового редактора	11
7	Редактирование файла lab5-1.asm	12
8	Проверка успешного редактирования	13
9	Компиляция файла с помощью nasm	13
10	Сборка исполняемого файла с помощью ld	13
11	Запуск исполняемого файла	14
12	Взаимодействие с программой	14
13	Открытие папки с файлом in_out.asm в правой панели	14
14	Копирование файла с помощью F6	15
15	Копирование файла с помощью F5	15
16	Текущий вид рабочей папки	16
17	Редактирование файла lab5-2.asm	16
18	Создание исполняемого файла	16
19	Запуск исполняемого файла	17
20	Изменение файла lab5-2.asm	17
21	Запуск изменённого файла	17
1	Создание копии файла lab5-1.asm	19
2	Изменение файла lab5-1-1.asm	20
3	Создание исполняемого файла	20
4	Проверка работы программы	20
5	Создание копии файла lab5-2.asm	21
6	Изменение файла lab5-2-1.asm	21
7	Создание исполняемого файла	21
8	Проверка работы программы	22

Список таблиц

Цель работы

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

Выполнение лабораторной работы

Для начала выполнения лабораторной работы нам необходимо открыть Midnight commander с помощью команды mc (Рис. 2.1):

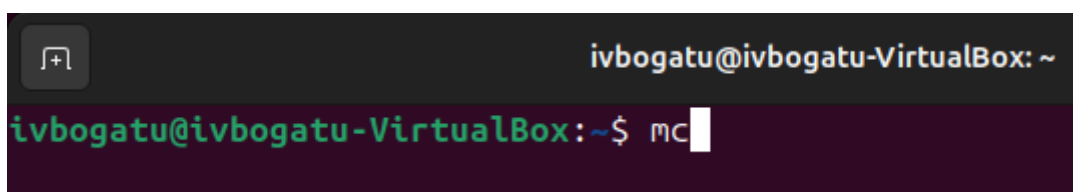


Рис. 1: Запуск Midnight commander

После ввода команды мы увидим такой интерфейс (Рис. 2.2):

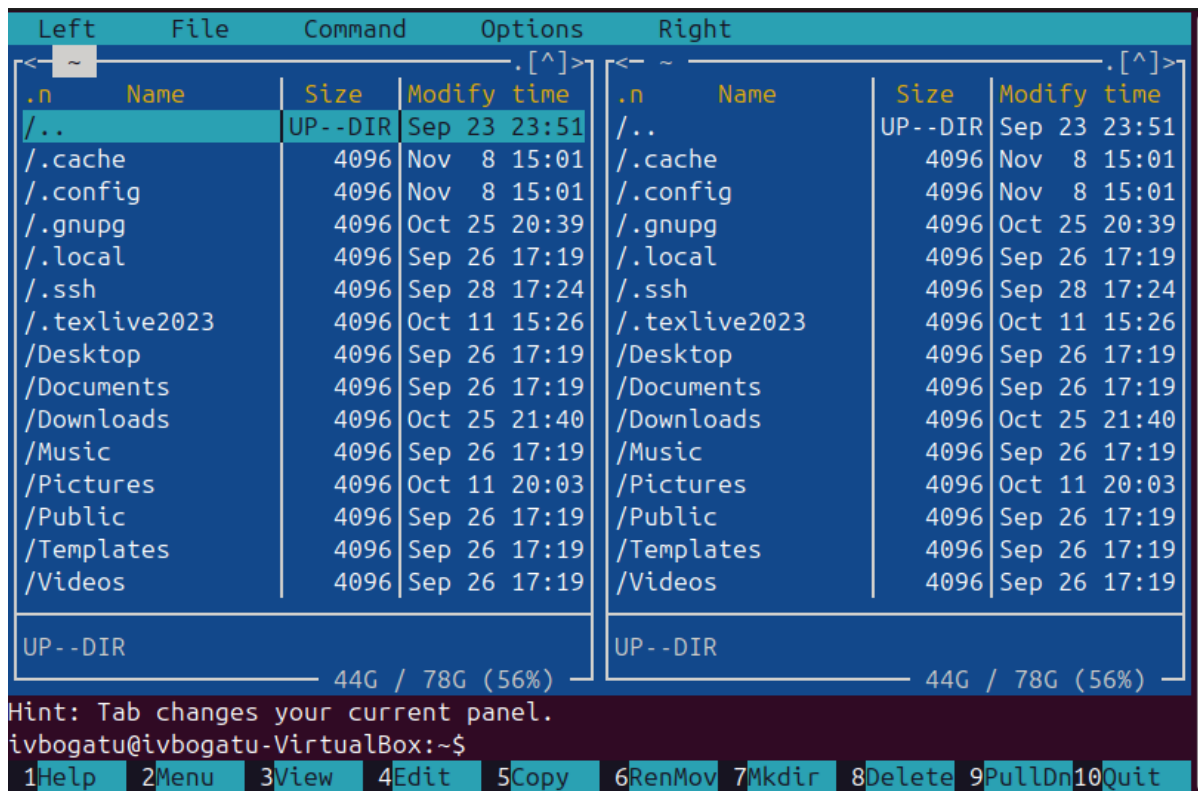


Рис. 2: Интерфейс midnight commander

С помощью стрелок и клавиши Enter перейдём в каталог ~/work/arch-rc (Рис. 2.3):

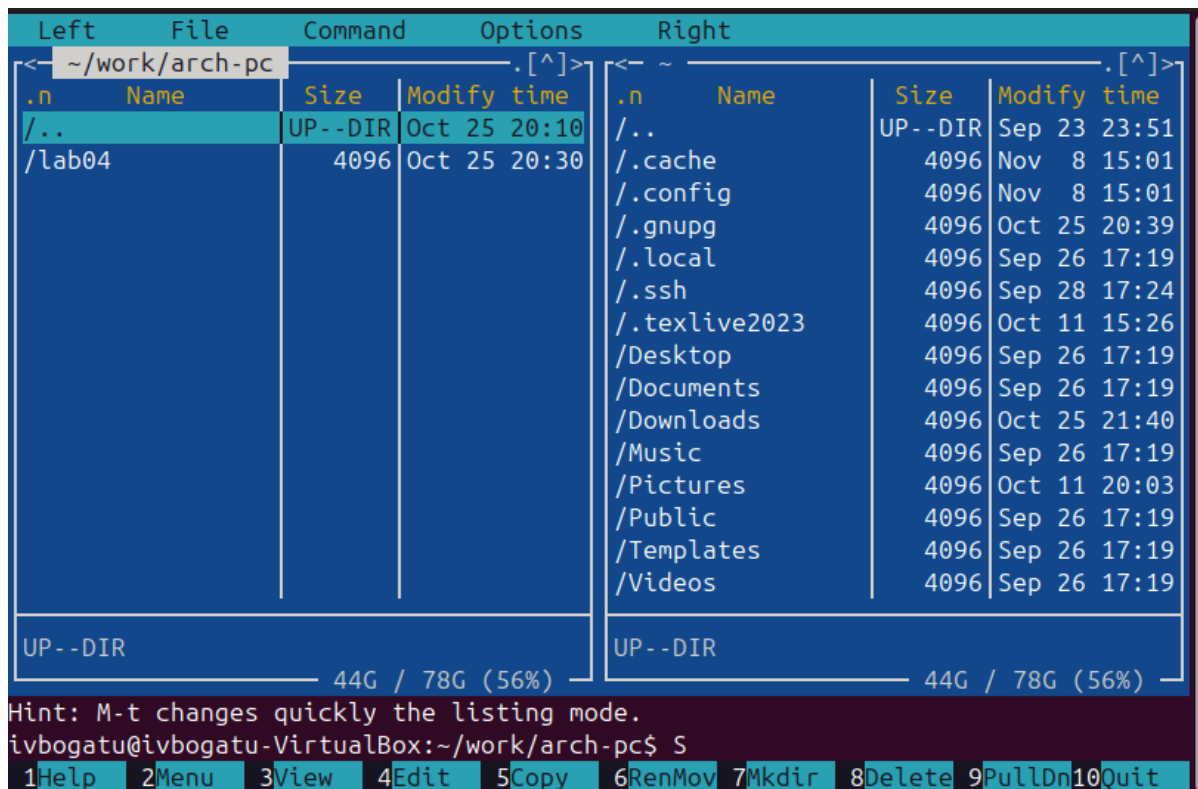


Рис. 3: Переход в нужный каталог (~/work/arch-pc)

Создадим папку lab05 с помощью клавиши F7 (Рис. 2.4):

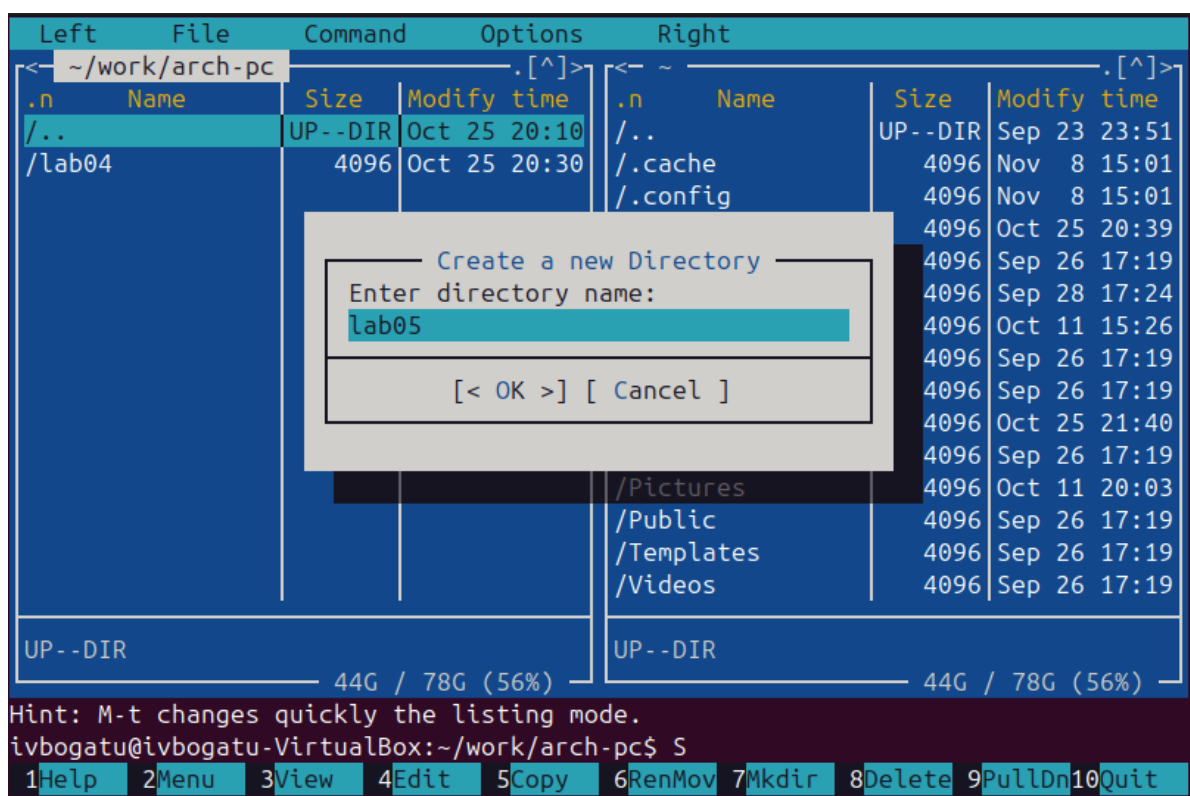


Рис. 4: Создание папки

Теперь с помощью команды `touch` создадим файл `lab5-1.asm` (Рис. 2.5):


```
ivbogatu@ivbogatu-VirtualBox:~$ mc
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc$ S
S: command not found

ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ touch lab5-1.asm

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano          <---- easiest
 2. /usr/bin/mcedit
 3. /usr/bin/vim.tiny
 4. /bin/ed

Choose 1-4 [1]:
```

Рис. 6: Выбор текстового редактора

Теперь отредактируем файл и поместим в него следующий код (Рис. 2.7):

```
GNU nano 7.2 /home/ivbogatu/work/arch-pc/lab05/lab5-1.asm *
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициированных 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'

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Рис. 7: Редактирование файла lab5-1.asm

Теперь сохраним его (сочетанием клавиш ctrl+x и согласившись с сохранением) и с помощью F3 откроем для просмотра, чтобы убедиться, что он сохранился корректно (Рис. 2.8):

```
/home/ivbogatu/work/arch-pc/lab05/lab5-1.asm 1448/2432 59%
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициированных 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
1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format10Quit
```

Рис. 8: Проверка успешного редактирования

Теперь скомпилируем его (Рис. 2.9):

```
Hint: We also have a nice manual page.
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm [^]
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit
```

Рис. 9: Компиляция файла с помощью nasm

И соберём (Рис. 2.10):

```
Hint: We also have a nice manual page.
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ lab5-1 lab5-1.o [^]
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit
```

Рис. 10: Сборка исполняемого файла с помощью ld

После этого запустим получившийся исполняемый файл (Рис. 2.11):

```

Hint: We also have a nice manual page.
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-1
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit

```

Рис. 11: Запуск исполняемого файла

Теперь введём ФИО (Рис. 2.12):

```

ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Богату Ирина Владимировна

```

Рис. 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):

Left	File	Command	Options	Right
<pre> <- ~/work/arch-pc/lab05 .[^]> .n Name Size Modify time /.. UP - -DIR Nov 8 15:04 *lab5-1 8744 Nov 8 15:10 lab5-1.asm 2432 Nov 8 15:08 lab5-1.o 752 Nov 8 15:10 </pre>		<pre> <- ~/Downloads .[^]> .n Name Size Modify time /.. UP - -DIR Nov 8 15:05 /pandoc-c~-master 4096 Oct 13 08:30 /pandoc-c~-ter (2) 4096 Oct 13 08:30 in_out.asm 3942 Nov 8 15:12 pandoc-c~-ter.zip 1325638 Oct 25 21:29 report.pdf 51093 Oct 25 21:40 </pre>		
lab5-1.asm		in_out.asm		
44G / 78G (56%)		44G / 78G (56%)		

```

Hint: Leap to frequently used directories in a single bound with C-\.
ivbogatu@ivbogatu-VirtualBox:~/Downloads$
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit

```

Рис. 13: Открытие папки с файлом `in_out.asm` в правой панели

Скопируем его в нашу рабочую папку с помощью F6 (Рис. 2.14):

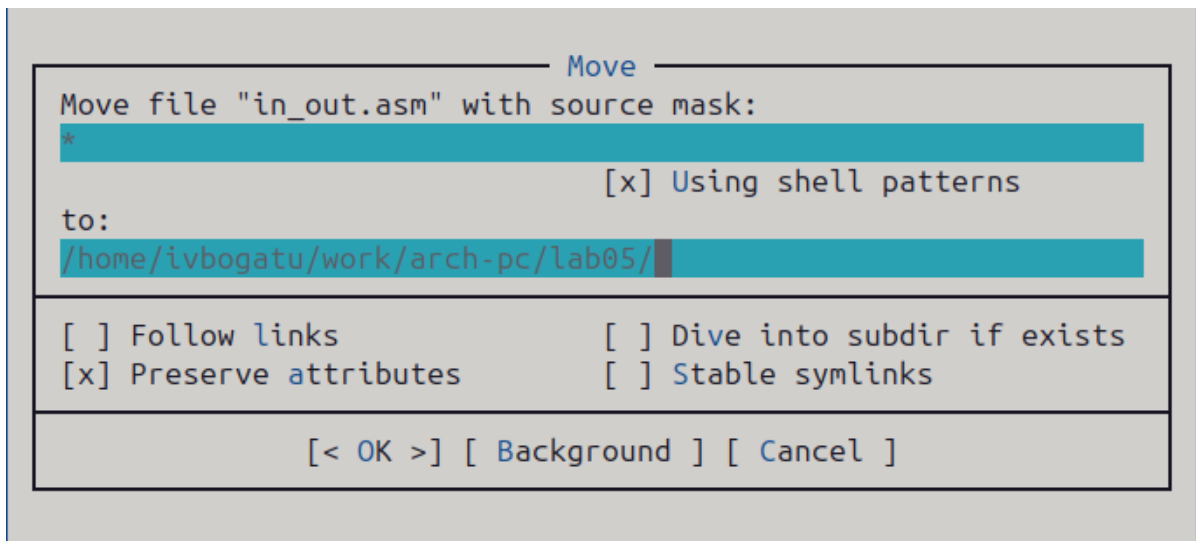


Рис. 14: Копирование файла с помощью F6

Теперь сделаем копию файла lab5-1.asm с помощью команды F5. Назовём копию lab5-2.asm (Рис. 2.15):

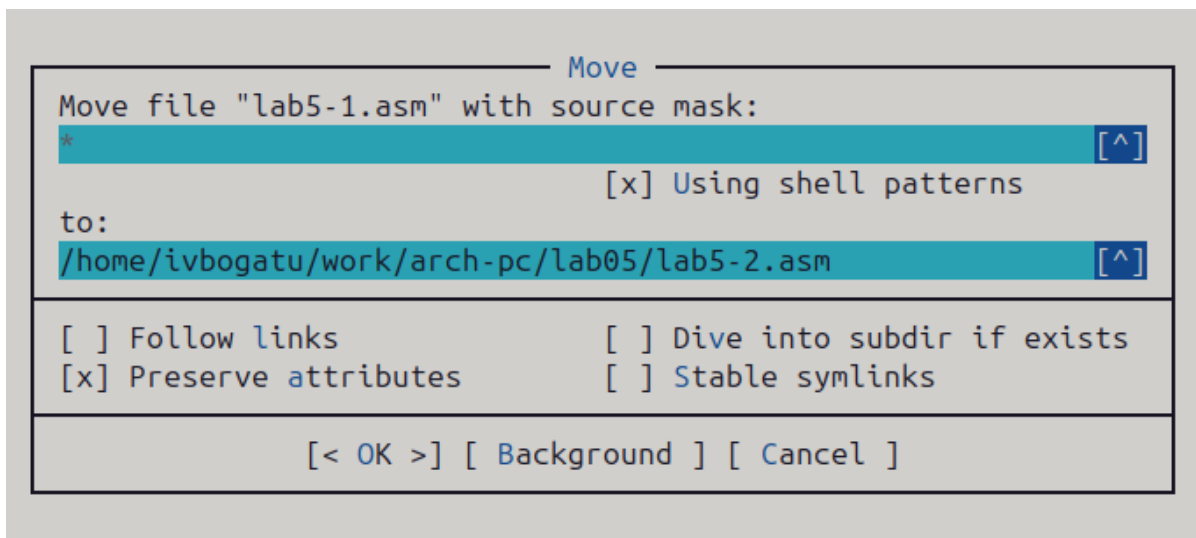


Рис. 15: Копирование файла с помощью F5

Теперь наша папка выглядит следующим образом (Рис. 2.16):

NAME	SIZE	MODIFY TIME
./..	UP - -DIR	Nov 8 15:04
in_out.asm	3942	Nov 8 15:12
*lab5-1	8744	Nov 8 15:10
lab5-1.o	752	Nov 8 15:10
lab5-2.asm	2432	Nov 8 15:08

Рис. 16: Текущий вид рабочей папки

Откроем в текстовом редакторе файл lab5-2.asm и напишем туда следующий код (Рис. 2.17):

```

;-----
; 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. 17: Редактирование файла lab5-2.asm

После чего создадим исполняемый файл с помощью nasm и ld (Рис. 2.18):

```

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

```

Рис. 18: Создание исполняемого файла

Запустим созданный файл (Рис. 2.19):

```
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Богату Ирина Владимировна
```

Рис. 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` (Рис. 2.20):

```
;-----
; 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. 20: Изменение файла lab5-2.asm

Точно также соберём исполняемый файл и запустим его (Рис. 2.21):

```
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Богату Ирина Владимировна
```

Рис. 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

Выполнение задания для самостоятельной работы

Теперь создадим с помощью F6 копию файла lab5-1.asm (Рис. 3.1):

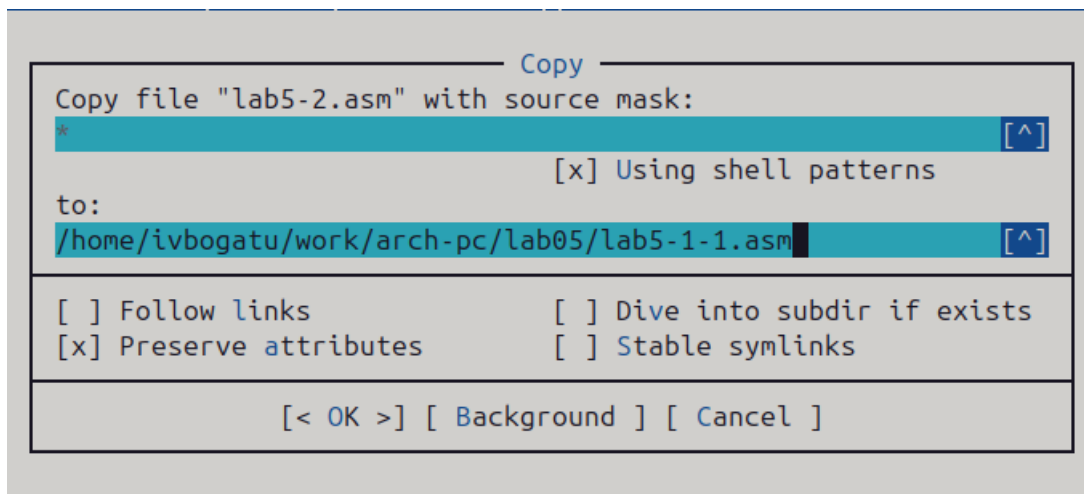


Рис. 1: Создание копии файла lab5-1.asm

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):

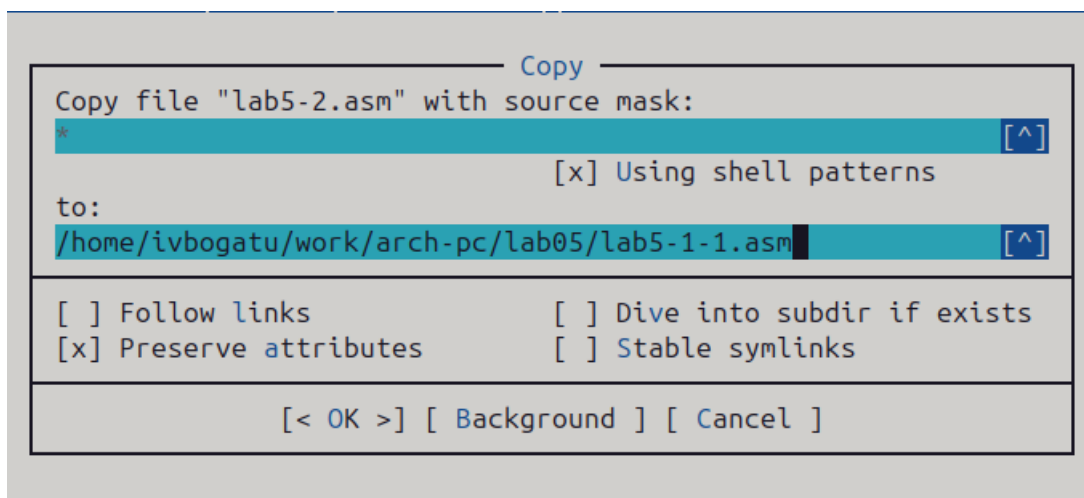


Рис. 2: Изменение файла lab5-1-1.asm

Сохраним изменения и создадим исполняемый файл (Рис. 3.3):

```
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)
```

Рис. 3: Создание исполняемого файла

Запустим его и проверим, что всё работает (Рис. 3.4):

```
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab5-1-1.asm
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
```

Рис. 4: Проверка работы программы

Теперь создадим с помощью F5 копию файла lab5-2.asm (Рис. 3.5):

```
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Богату Ирина Владимировна
Богату Ирина Владимировна
```

Рис. 5: 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 eax адрес buf1, а также строчку, которая вызывает подпрограмму sprintLF (Рис. 3.6):

```
ivbogatu@ivbogatu-VirtualBox:~/work/arch-pc/lab05$ ./lab5-1-1
Введите строку:
Богату Ирина Владимировна
Богату Ирина Владимировна
```

Рис. 6: Изменение файла lab5-2-1.asm

Теперь создадим исполняемый файл (Рис. 3.7):

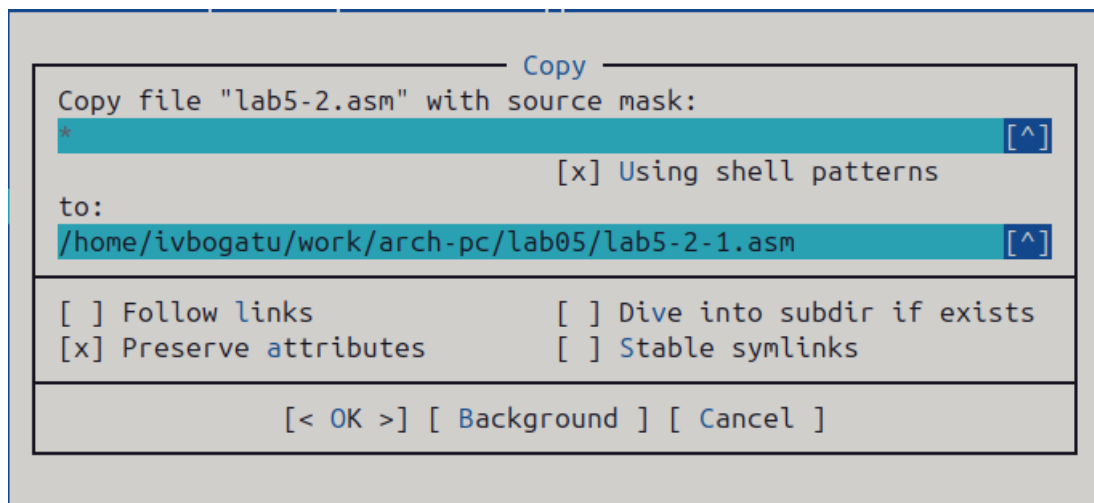


Рис. 7: Создание исполняемого файла

Теперь запустим программу и убедимся, что она работает (Рис. 3.8):

```

;-----
; 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 sprintf ; вызов подпрограммы печати сообщения
call quit ; вызов подпрограммы завершения

```

Рис. 8: Проверка работы программы

Выводы

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