

Отчет по лабораторной работе №5

Мухин Тимофей Владимирович

Содержание

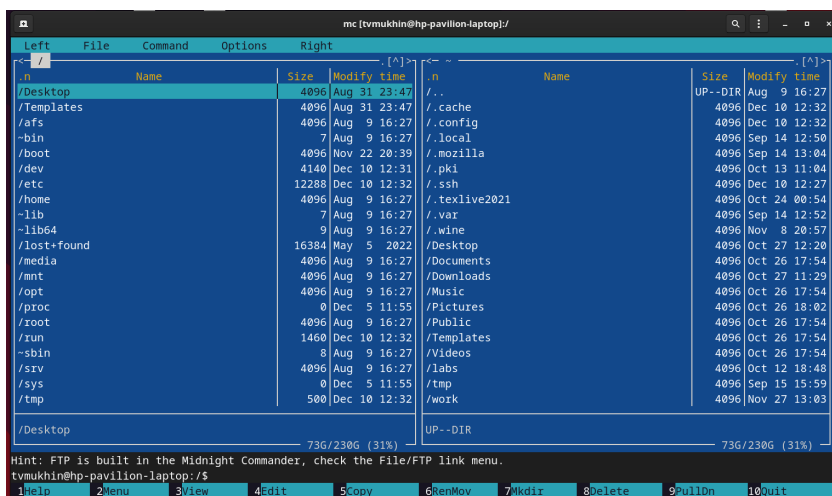
1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Выводы	11

1 Цель работы

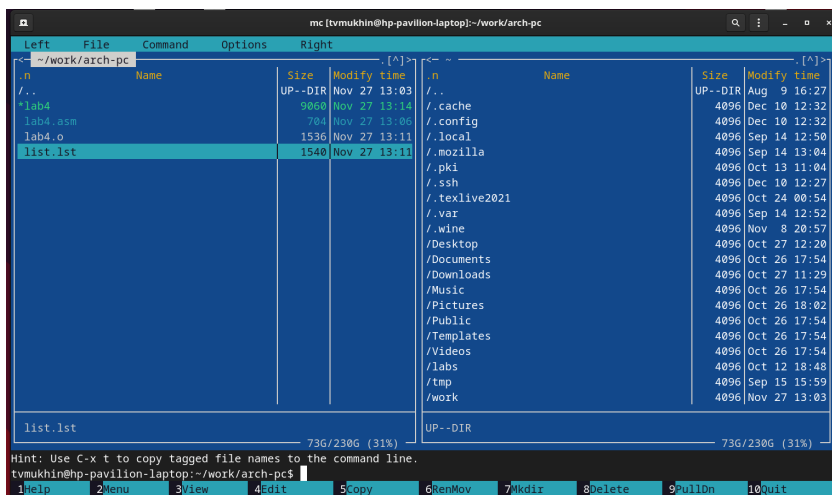
Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера `mov` и `int`.

2 Выполнение лабораторной работы

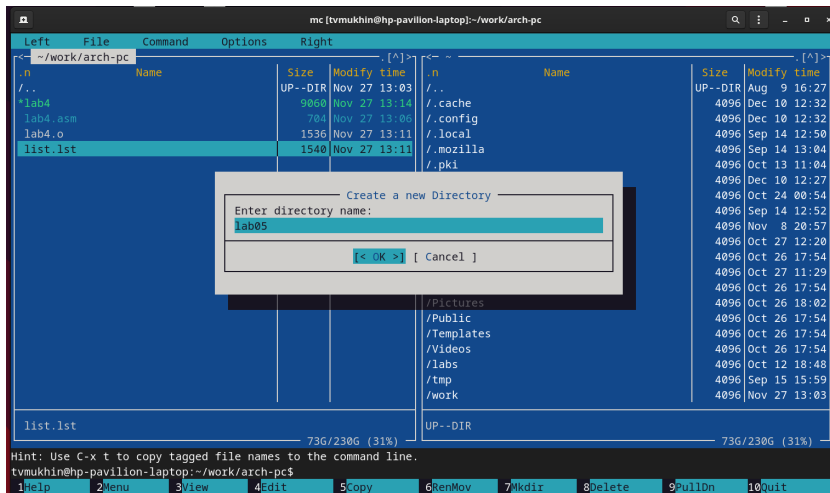
1. Открываем Midnight commander



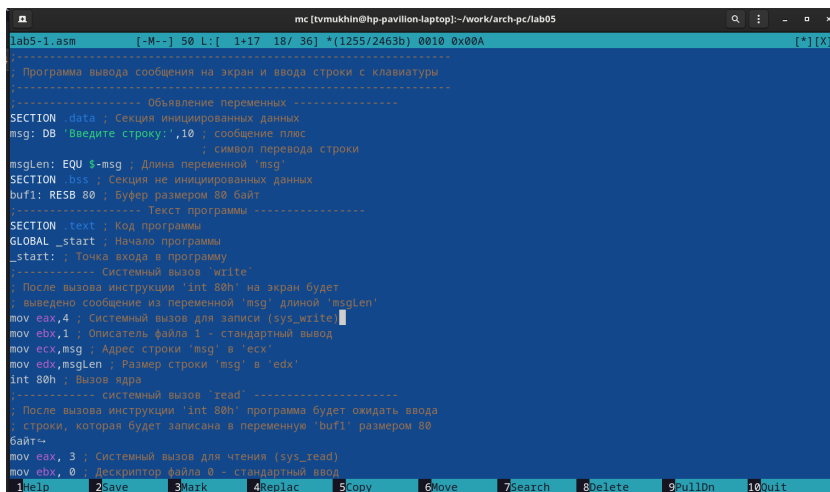
2. Переходим в каталог /work/arch-pc.



3. Создаем папку lab05 и переходим в нее.



4. Пользуясь командой `touch` создаем файл `lab5-1.asm`, открываем его во встроенном редакторе и вводим текст программы из листинга 6.1, сохраняем изменения



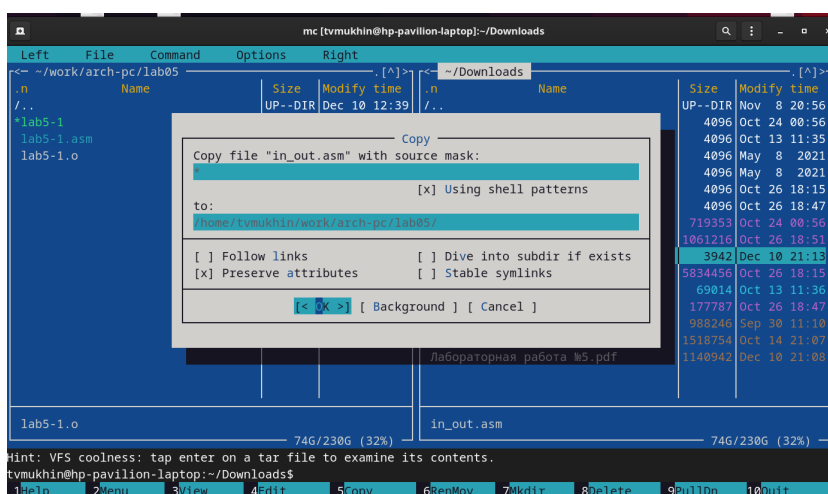
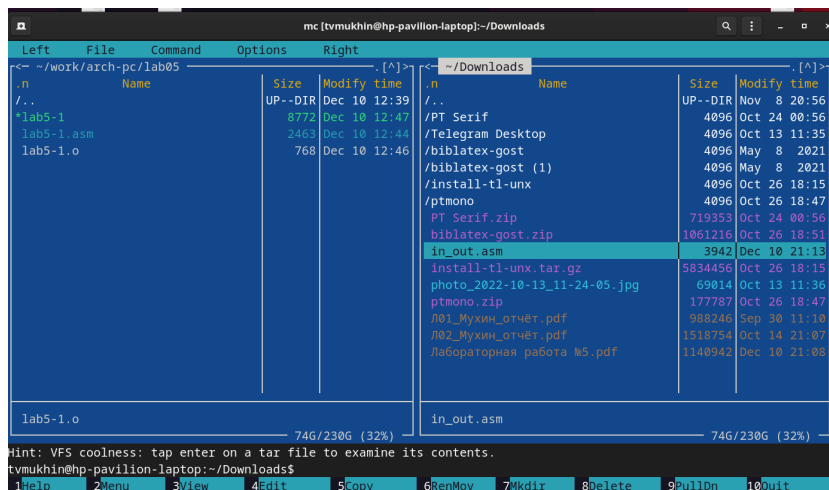
5. Используя функциональные клавиши открываем файл для просмотра. .

```
mc [tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05]
/home/tvmukhin/work/arch-pc/lab05/lab5-1.asm 1941/2463 788
-----
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
-----
1help 2UnWrap 3Quit 4hex 5goto 6 7Search 8Raw 9Format 10Quit
```

6. Транслируем текст программы lab5-1.asm в объектный файл. Выполняем компоновку объектного файла и запускаем получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры.

```
tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05
tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Мухин Тимофей Владимирович
tvmukhin@hp-pavillon-laptop:~/work/arch-pc/lab05$
```

7. Для упрощения написания программ часто встречающиеся одинаковые участки кода можно оформить в виде подпрограмм и сохранить в отдельные файлы, а во всех нужных местах поставить вызов нужной подпрограммы. Скачиваем файл in_out.asm, копируем его в каталог с файлом lab5-1.asm.



9. С помощью функциональной клавиши создаем копию файла lab5-1.asm с именем lab5-2.asm
10. Исправляем текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm (используем подпрограммы sprintf, sread и quit) в соответствии с листингом 6.2)

```
GNU nano 6.4 lab5-2.asm
#include 'in_out.asm'
SECTION .data ;
msg: DB 'Введите строку:',10 ;
msgLen: EQU $-msg ;
SECTION .bss ;
buf1: RESB 80 ;
SECTION .text ;
GLOBAL _start ;
_start: ;

mov eax,msg ;
call sprint ;

mov ecx,buf1 ;
mov edx, 80 ;
call sread

call quit
```

11. Создаём исполняемый файл и проверяем его работу

```
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
1234
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$
```

12. Создаём копию файла lab5-1.asm (с названием lab5-11.asm). Вносим изменения в программу (без использования внешнего файла in_out.asm), так чтобы она работала по следующему алгоритму:

- 1) вывести приглашение типа “Введите строку:”
- 2) ввести строку с клавиатуры
- 3) вывести введённую строку на экран


```

GNU nano 6.4 /home/sev_chik/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab05/lab5-11.asm
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

mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,buf1 ;
mov edx,80 ;
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)

```

13. Получаем исполняемый файл и проверяем его работу.

```

tvmukhin@hp-pavillon-laptop: /work/arch-pc/lab05$ nasm -f elf lab5-11.asm
lab5-11.asm:25: warning: label alone on a line without a colon might be in error
[-wlabel-orphan]
tvmukhin@hp-pavillon-laptop: /work/arch-pc/lab05$ ld -m elf_i386 -o lab5-11 lab5-11.o
tvmukhin@hp-pavillon-laptop: /work/arch-pc/lab05$ ./lab5-11
Введите строку:
Тимофей
Тимофей
tvmukhin@hp-pavillon-laptop: /work/arch-pc/lab05$

```

14. Создаем копию файла lab5-2.asm (с названием lab5-22.asm). Исправляем текст программы с использованием подпрограмм из внешнего файла in_out.asm, так чтобы она работала по следующему алгоритму: 1) вывести приглашение типа “Введите строку:”

2) ввести строку с клавиатуры

3) вывести введенную строку на экран.

```

GNU nano 6.4 /home/seychik/work/study/2022-2023/
%include 'in_out.asm'
SECTION .data ;
msg: DB 'Введите строку:',10 ;
msgLen: EQU $-msg ;
SECTION .bss ;
buf1: RESB 80 ;

SECTION .text ;
GLOBAL _start ;
_start: ;

mov eax,msg ;
call sprint;

mov ecx,buf1 ;
mov edx, 80 ;
call sread

mov eax,buf1;
call sprint;

call quit

```

15. Создаем исполняемый файл и проверяем его работу.

```

tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ nasm -f elf lab5-22.asm
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-22 lab5-22.o
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$ ./lab5-22
Введите строку:
Мухин Тимофей
Мухин Тимофей
tvmukhin@hp-pavilion-laptop:~/work/arch-pc/lab05$

```

3 Выводы

В ходе выполнения лабораторной работы я приобрел практические навыки работы в Midnight Commander, изучил инструкции языка ассемблера `mov` и `int`.