

# **Отчёт по лабораторной работе**

**НММбд-04-24**

Ракутуманандзара Цантамписедрана Сарубиди

# **Содержание**

|   |           |
|---|-----------|
| <b>1 Цель работы</b>                    | <b>5</b>  |
| <b>2 Задание</b>                        | <b>6</b>  |
| <b>3 Выполнение лабораторной работы</b> | <b>7</b>  |
| <b>4 Выводы</b>                         | <b>18</b> |
| <b>Список литературы</b>                | <b>19</b> |

# **Список иллюстраций**

|                       |    |
|-----------------------|----|
| 3.1 рис 1 . . . . .   | 7  |
| 3.2 рис 2 . . . . .   | 8  |
| 3.3 рис 3 . . . . .   | 8  |
| 3.4 рис 4 . . . . .   | 9  |
| 3.5 рис 5 . . . . .   | 9  |
| 3.6 рис 6 . . . . .   | 10 |
| 3.7 рис 7 . . . . .   | 10 |
| 3.8 рис 8 . . . . .   | 11 |
| 3.9 рис 9 . . . . .   | 11 |
| 3.10 рис 10 . . . . . | 12 |
| 3.11 рис 11 . . . . . | 12 |
| 3.12 рис 13 . . . . . | 13 |
| 3.13 рис 13 . . . . . | 13 |
| 3.14 рис 14 . . . . . | 13 |
| 3.15 рис 15 . . . . . | 14 |
| 3.16 рис 16 . . . . . | 14 |
| 3.17 рис 17 . . . . . | 15 |
| 3.18 рис 18 . . . . . | 16 |
| 3.19 рис 19 . . . . . | 16 |
| 3.20 рис 20 . . . . . | 17 |

# **Список таблиц**

# **1 Цель работы**

Целью данной работы является приобретение практических навыков работы в Midnight Commander. Также освойте инструкции на языке ассемблера mov и int.

## **2 Задание**

1. Основы работы с mc
2. Структура программы на языке ассемблера NASM
3. Подключение внешнего файла
- 4 Выполнение заданий для самостоятельной работы

### **3 Выполнение лабораторной работы**

### 3.1 Основы работы с ms

Я открою Midnight Commander, выполнив команду mc в терминале(рис 1)

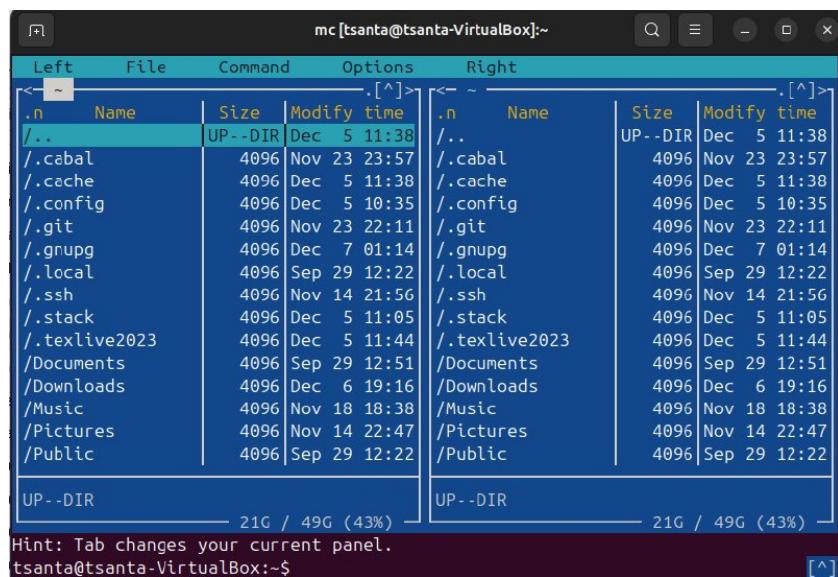


Рис. 3.1: рис 1

Я перейду в каталог `~/work/arch-pc`, который я создал в лаборатории 4 в mc Commander(рис 2)

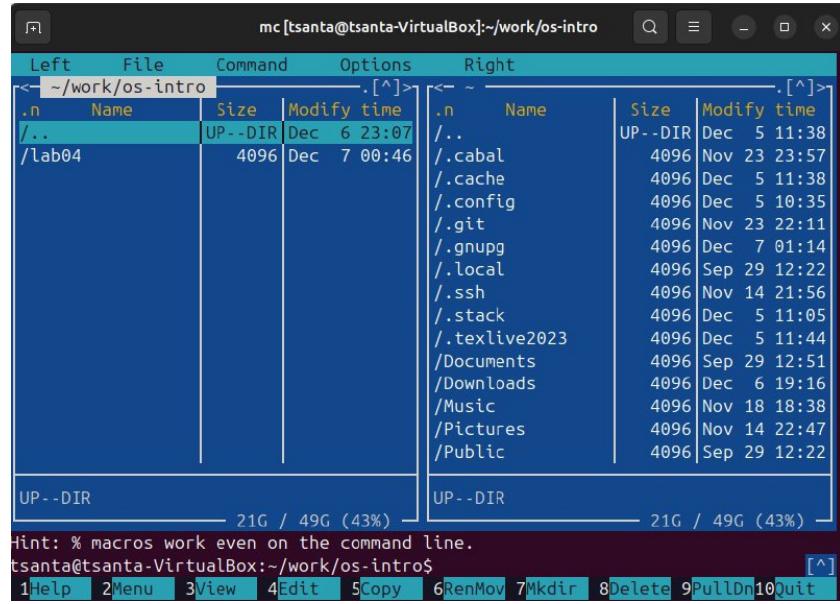


Рис. 3.2: рис 2

Я буду использовать функциональную клавишу F7, чтобы создать папку lab05 и перейти в созданный каталог(рис 3)

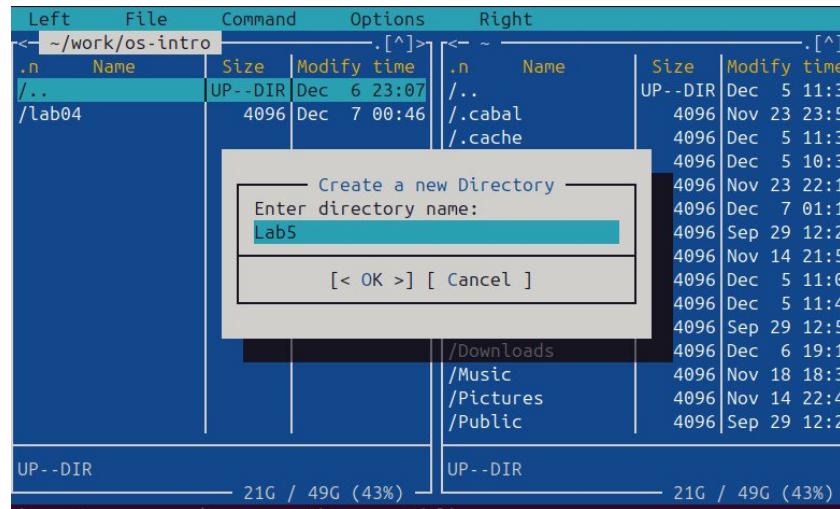


Рис. 3.3: рис 3

В строке ввода я напишу команду touch lab5-1.asm для создания файла, в котором буду работать(рис 4)

```
Hint: % macros work even on the command line.  
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ touch lab-1.asm [^]  
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn 10Quit
```

Рис. 3.4: рис 4

### 3.2 Структура программы на языке ассемблера NASM

Я открываю созданный файл для повторного редактирования. Я введу заданный текст программы, сохраню изменения и закрою файл(рис 5)

The screenshot shows a text editor window with the following details:

- File menu: Open, New.
- Current file: lab5-1.asm (~/work/os-intro/Lab05).
- Recent files: in\_out.asm, tsanta.Lab5.md.
- Tab bar: lab5-1.asm.
- Code content:

```
;-----  
; 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
```

Рис. 3.5: рис 5

С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы(рис 6)

```
/home/tsanta/work/os-intro/Lab05/lab5-1.asm      1186/2499    47%
;----- 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'
1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format 10Quit
```

Рис. 3.6: рис 6

Сначала я переведу текст программы lab5-1.asm в объектный файл. Затем линкую объектный файл и запускаю полученный исполняемый файл. Программа выведет строку “Введите строку:” и ожидает ввода с клавиатуры. При появлении запроса я введу свое полное имя(рис 7)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ cd ~/work/os-intro/Lab05
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ nasm -f elf lab5-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ld -m elf_i386 -o lab5-1 lab5-1.lo
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ./lab5-1
Введите строку:
Цанта Ракуту
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$
```

Рис. 3.7: рис 7

### 3.3.Подключение внешнего файла in\_out.asm

Я скачиваю файл in\_out.asm со страницы курса в ТУИС и сохраняю его в каталоге «Downloads»(рис 8)

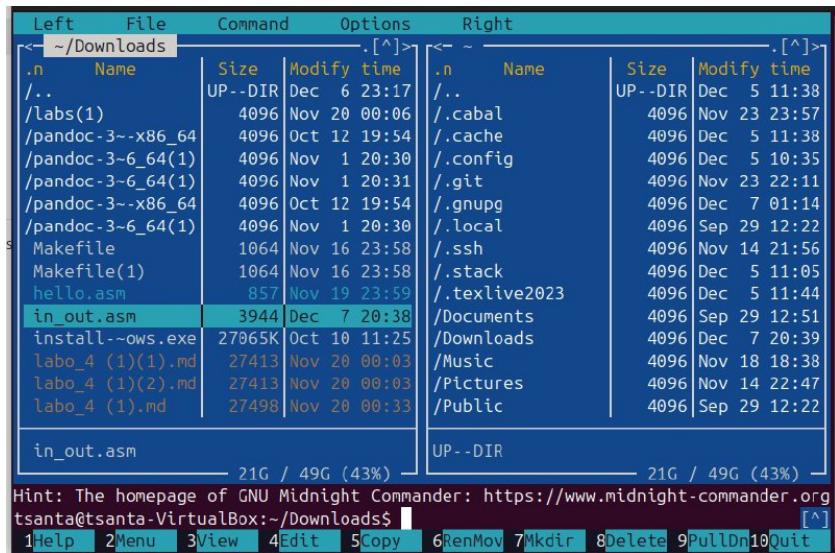


Рис. 3.8: рис 8

Функциональной клавишей F5 скопирую файл in\_out.asm из каталога Downloads в созданный каталог lab05(рис 9)

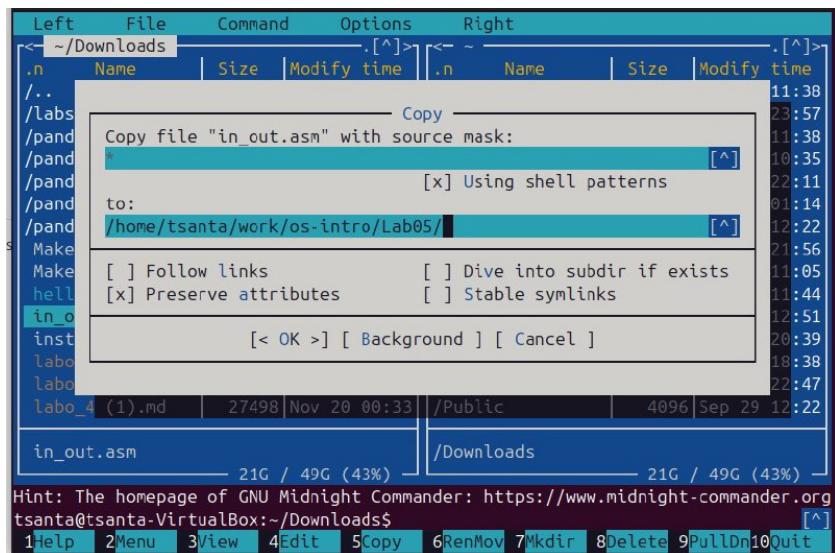


Рис. 3.9: рис 9

Функциональной клавишей F5 я скопирую файл lab5-1.asm в тот же каталог, но с другим именем; для этого в появившемся окне мс я введу имя копии файла(рис 10)

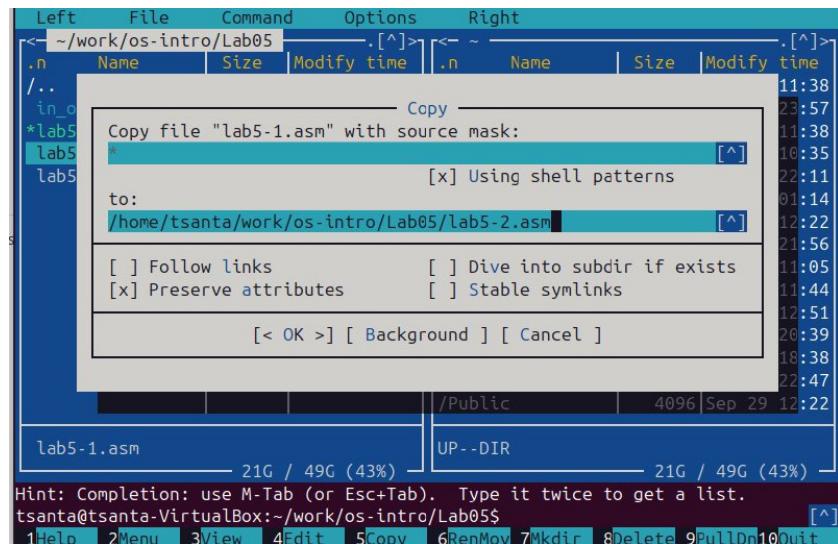


Рис. 3.10: рис 10

Содержимое файла lab5-2.asm я изменю с помощью редактора, чтобы программа использовала подпрограммы из внешнего файла in\_out.asm(рис 11)

```

Open  lab5-2.asm ~/work/os-intro/Lab05
tsanta.Lab5.md   lab5-1.asm   in_out(1).asm   lab5-2.asm x

;
;-----;
; 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

```

Рис. 3.11: рис 11

Я переведу текст программы lab5-2.asm в объектный файл. Затем линкую объектный файл и запускаю полученный исполняемый файл. Программа выведет строку “Введите строку:” и ожидает ввода с клавиатуры. При появлении запроса я введу свое полное имя(рис 12)

```

tsanta@tsanta-VirtualBox:~$ cd ~/work/os-intro/Lab05
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ nasm -f elf lab5-2.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ld -m elf_i386 -o lab5-2 lab5-2.
o
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ./lab5-2
Введите строку:

```

Рис. 3.12: рис 13

Теперь я открою файл lab5-2.asm и поменяю место sprintLF на sprint. После этого я создам его исполняемый файл и запущу его, чтобы увидеть разницу(рис 13 и 14)

```

SECTION .bss
buf1: RESB 80          ; Секция не инициализированных данных
                        ; Буфер размером 80 байт

SECTION .text
GLOBAL _start
_start:
        ; Код программы
        ; Начало программы
        ; 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`
        call sprintLF          ; вызов подпрограммы печати сообщения

        mov edx, 80             ; запись адреса переменной в `EAX`
                                ; запись длины вводимого сообщения в `EBX`

```

Рис. 3.13: рис 13

```

tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ nasm -f elf lab5-2.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ld -m elf_i386 -o lab5-2 lab5-2.
o
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ./lab5-2

```

Рис. 3.14: рис 14

Разница между первым исполняемым файлом lab5-2 и вторым lab5-2 заключается в том, что при запуске первого запрашивается перевод строки, а программа, запускающаяся при запуске второго, запрашивает ввод без перевода строки.

### 3.4 Задание для самостоятельной работы

1. Я создам копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5(рис 15)

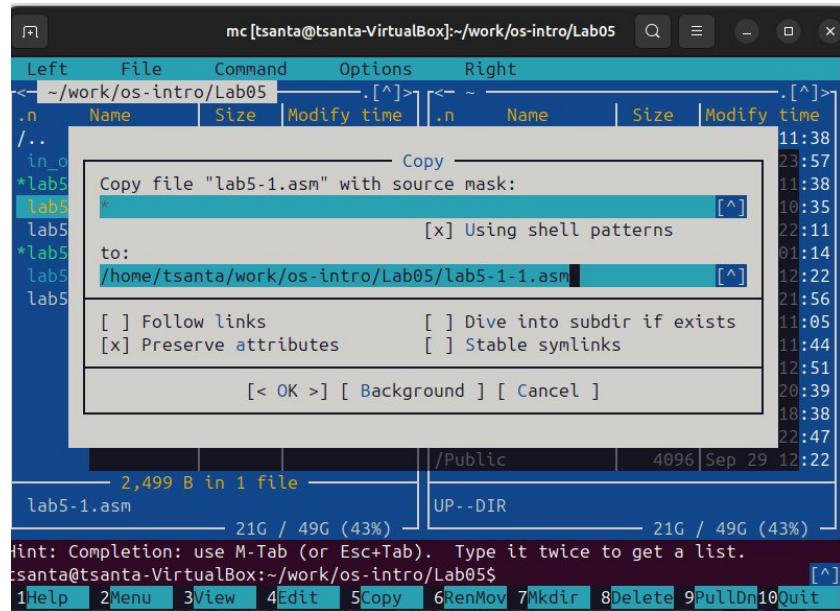


Рис. 3.15: рис 15

Я открою созданный файл для редактирования. Я изменяю программу так, чтобы она помимо отображения подсказки и запроса на ввод отображала строку, введенную пользователем(рис 16)

```

lab5-1-1.asm
~/work/os-intro/Lab05

;----- 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

```

Рис. 3.16: рис 16

2. Я переведу текст программы lab5-2.asm в объектный файл. Затем линкую объ-

ектный файл и запускаю полученный исполняемый файл. Программа выведет строку “Ведите строку:” и ожидает ввода с клавиатуры. При появлении запроса я введу свою фамилию(рис 17)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ nasm -f elf lab5-1-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ld -m elf_i386 -o lab5-1-1 lab5-
1-1.o
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ./lab5-1-1
Введите строку:
Ракуту
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$
```

Рис. 3.17: рис 17

Код программы из пункта 1:

```
SECTION .data ; Секция инициализированных данных msg: DB ‘Введите строку:’,10
; сообщение плюс msgLen: EQU $-msg ; Длина переменной ‘msg’ 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,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 чтения(sys_exit) mov ebx,1 ; Описатель файла ‘1’ - стандартный вывод(без
ошибок) 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
```

3. Я создам копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5(рис 18)

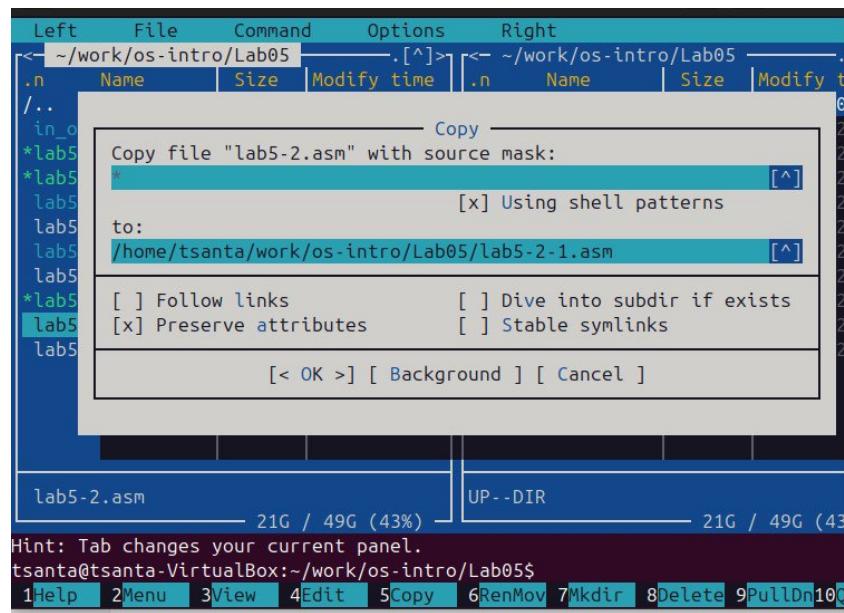


Рис. 3.18: рис 18

Я открою созданный файл для редактирования. Я изменяю программу так, чтобы она помимо отображения подсказки и запроса на ввод отображала строку, введенную пользователем(рис 19)

```

ta.Lab5.md | lab5-1.asm | in_out(1).asm | lab5-2.asm | lab5-1-1.asm | lab5-2-1.as x
-----
;-----;
; 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

```

Рис. 3.19: рис 19

4. Я создаю исполняемый файл и проверьте его работу(рис 20)

```
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ nasm -f elf lab5-2-1.asm
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ld -m elf_i386 -o lab5-2-1 lab5-
2-1.o
tsanta@tsanta-VirtualBox:~/work/os-intro/Lab05$ ./lab5-2-1
Введите строку:
```

Рис. 3.20: рис 20

Код программы из пункта 3:

```
%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 mov eax,4 ; Системный вызов для чтения(sys_exit) mov ebx,1 ; Описатель файла '1' - стандартный вывод(без ошибок) mov ecx, buf1 ; Адрес строки buf1 в ecx mov edx, buf1 ; Размер строки buf1 int 80h ; Вызов ядра call quit ; вызов подпрограммы завершения
```

## **4 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**

Архитектура ЭВМ