

# **Шаблон отчёта по лабораторной работе**

6

Сильвен Макс Грегор Филс , НКАбд-03-22

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы :</b>	<b>6</b>
2.1	Выводы по результатам выполнения заданий : . . . . .	17
<b>3</b>	<b>Задание для самостоятельной работы :</b>	<b>18</b>
3.1	Создание программы без использования внешнего файла . . . .	18
3.1.1	создание программы с использованием внешнего файла .	19
<b>4</b>	<b>Выводы</b>	<b>20</b>
	<b>Список литературы</b>	<b>21</b>

## Список иллюстраций

2.1	Ресунок 1	6
2.2	Ресунок 2	7
2.3	Ресунок 3	8
2.4	Ресунок 4	9
2.5	Ресунок 5	10
2.6	Ресунок 6	11
2.7	Ресунок 7	12
2.8	Ресунок 8	13
2.9	Ресунок 9	13
2.10	Ресунок 10	14
2.11	Ресунок 11	15
2.12	Ресунок 12	16
3.1	Ресунок 15	18
3.2	Ресунок 16	19

## Список таблиц

# 1 Цель работы

- На шестой лабораторной работе мы научимся использовать “Midnight commander” и освоим инструкции **mov** и **int** языка ассемблера.

## 2 Выполнение лабораторной работы :

- На этом этапе мы запустили mc.(рис. 2.1)

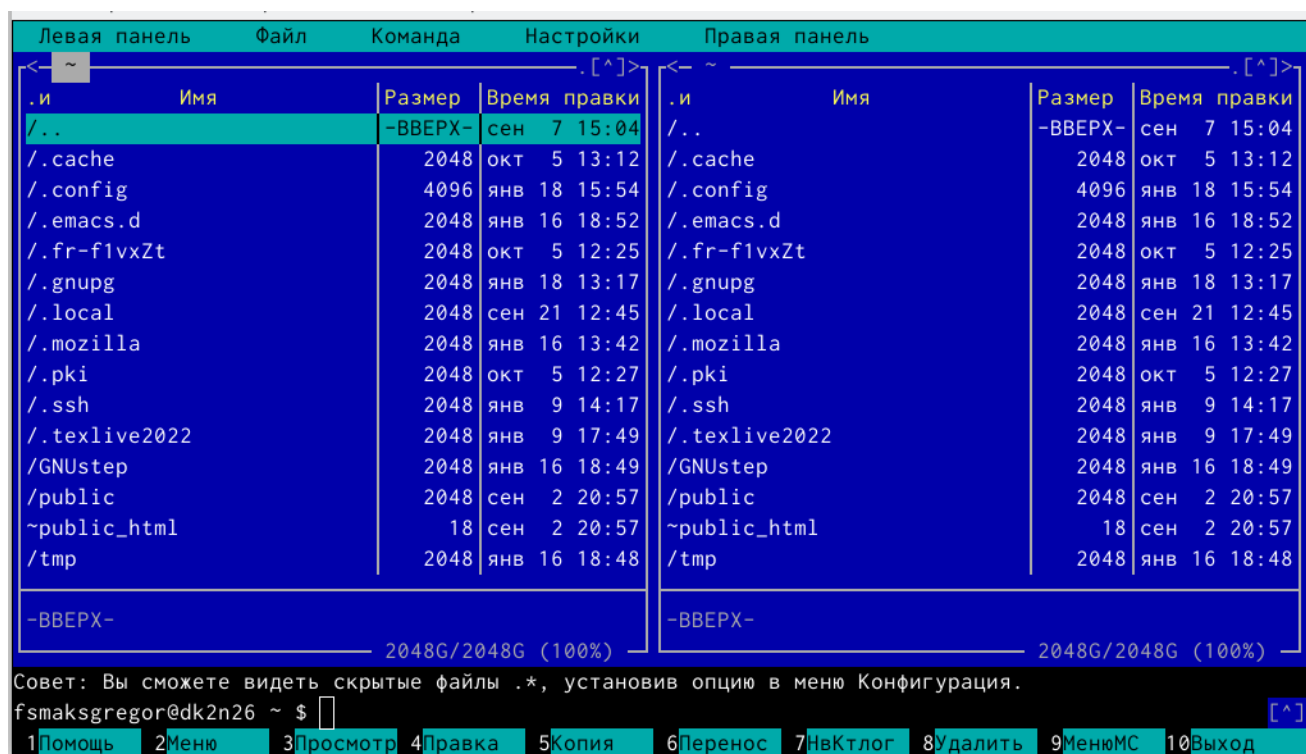


Рис. 2.1: Ресунок 1

- После этого мы переместились в каталог ~/work/arch-рс.(рис. 2.2)

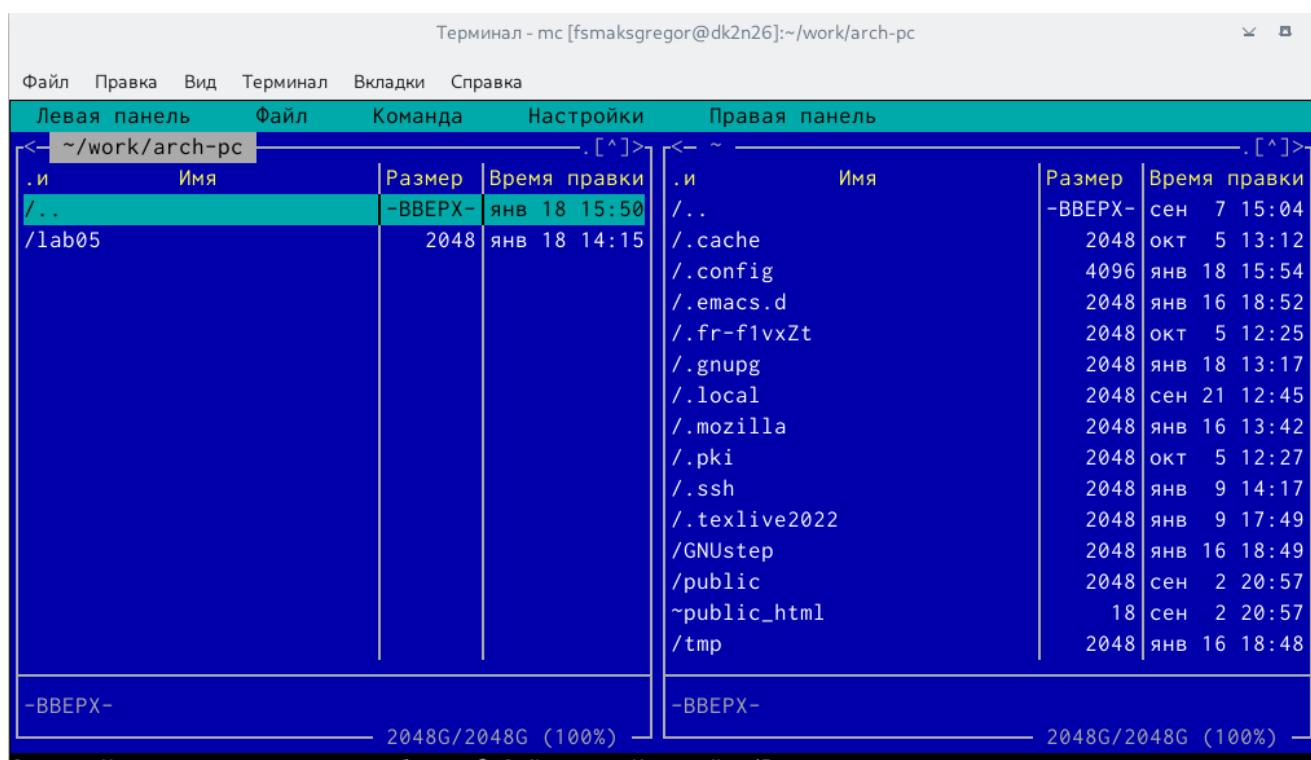


Рис. 2.2: Ресунок 2

- После этого и с помощью клавиши f7 мы создали новую папку lab06.(рис. 2.3)





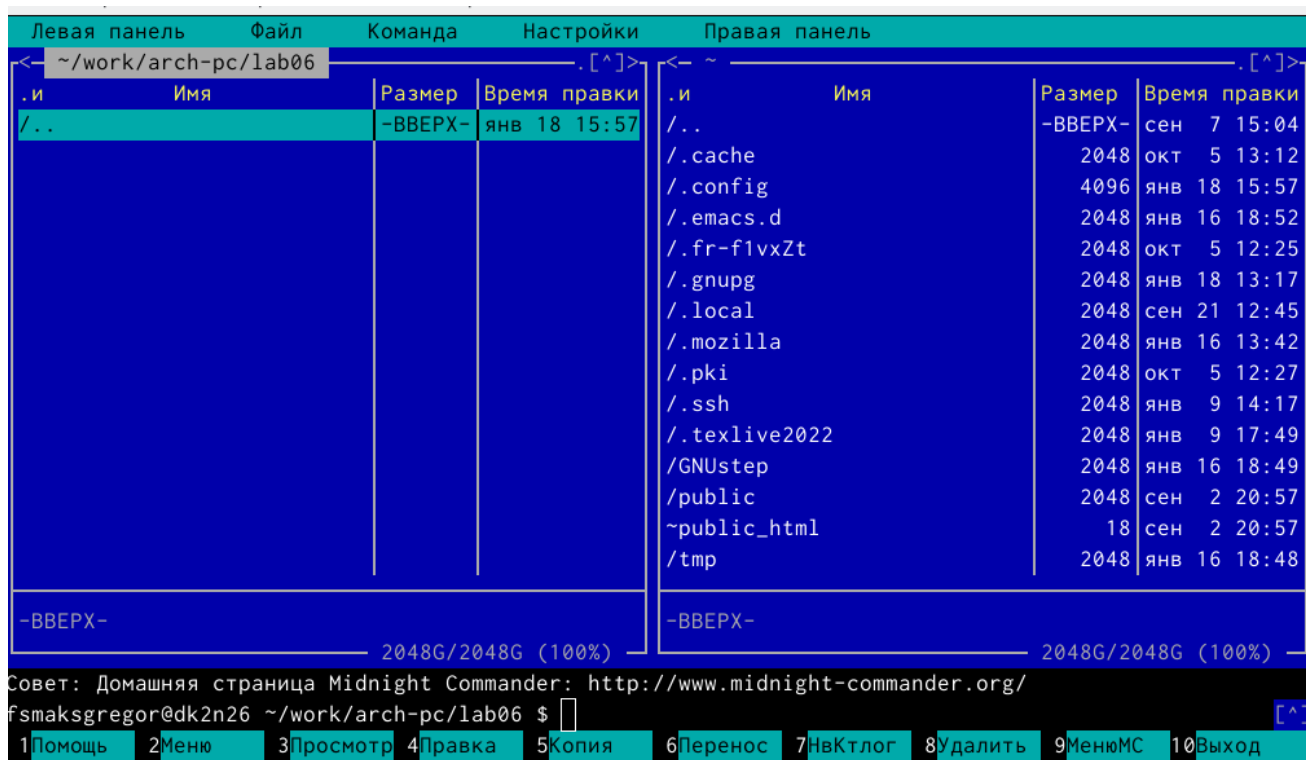


Рис. 2.4: Ресунок 4

- Используя функциональную клавишу F4, мы открыли файл lab6-1.asm. (рис. 2.5)



Рис. 2.5: Ресунок 5

- Мы скопировали текст программы из листинга 6.1 в файл asm, затем сохранили изменения и закрыли файл. (рис. 2.6)

```
GNU nano 6.3                               lab6-1.asm
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

mov eax, 3
mov ebx, 0
mov ecx, buf1
mov edx, 80
int 80h

^G Справка      ^O Записать     ^W Поиск        ^K Вырезать     ^T Выполнить    M-U Отмена
^X Выход        ^R ЧитФайл     ^\ Замена       ^U Вставить     ^C Позиция      M-E Повтор
```

Рис. 2.6: Ресунек 6

- Используя функциональную клавишу F3, мы открыли файл lab6-1.asm для просмотра. и мы проверили, что файл содержит текст программы. (рис. 2.7)

```
/afs/.dk.sci.pfu.edu.ru~ch-pc/lab06/lab6-1.asm 243/243 100%
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

mov eax, 3
mov ebx, 0
mov ecx, buf1
mov edx, 80
int 80h
```

1Помощь 2Раз~рн 3Выход 4Hex 5Пер~ти 6 7Поиск 8Исх~ый 9Формат10Выход

Рис. 2.7: Ресунок 7

- Затем мы перевели текст программы lab6-1.asm в объектный файл . Выполнил разметку объектного файла и запустил полученный исполняемый файл, где программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры. По запросу, в этот момент мы ввели наше имя и фамилию.(рис. 2.8)

```
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:
Sylvain Max
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $
```

Рис. 2.8: Ресунок 8

- После этого мы загрузили файл `in_out.asm` из ТУИСА и с помощью `mc` мы смогли переместить файл в правильный каталог. (рис. 2.9)

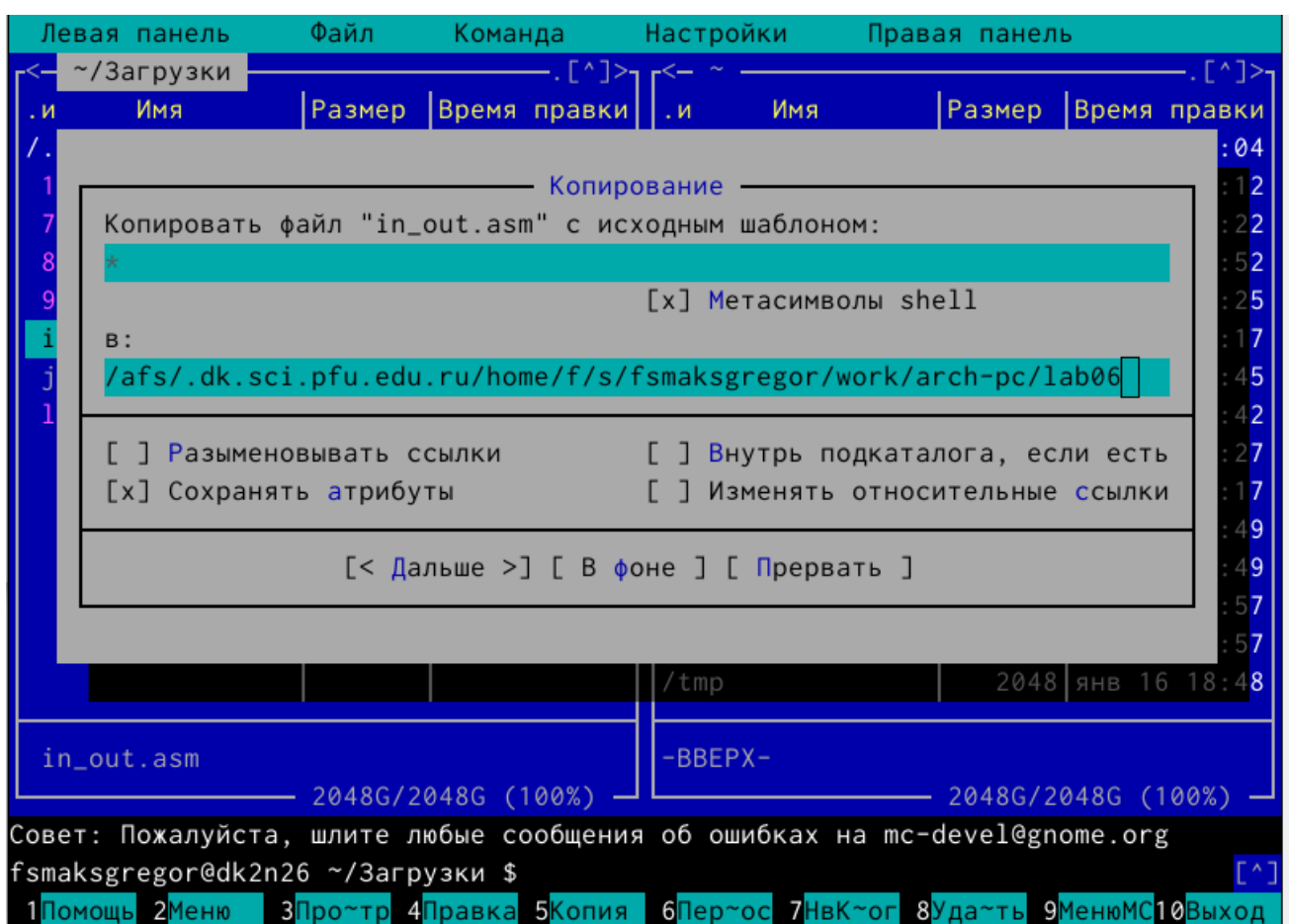


Рис. 2.9: Ресунок 9

- Используя функциональную клавишу `f5`, мы создали копию файла `lab6-1.asm` с именем `lab6-2.asm`. (рис. 2.10)

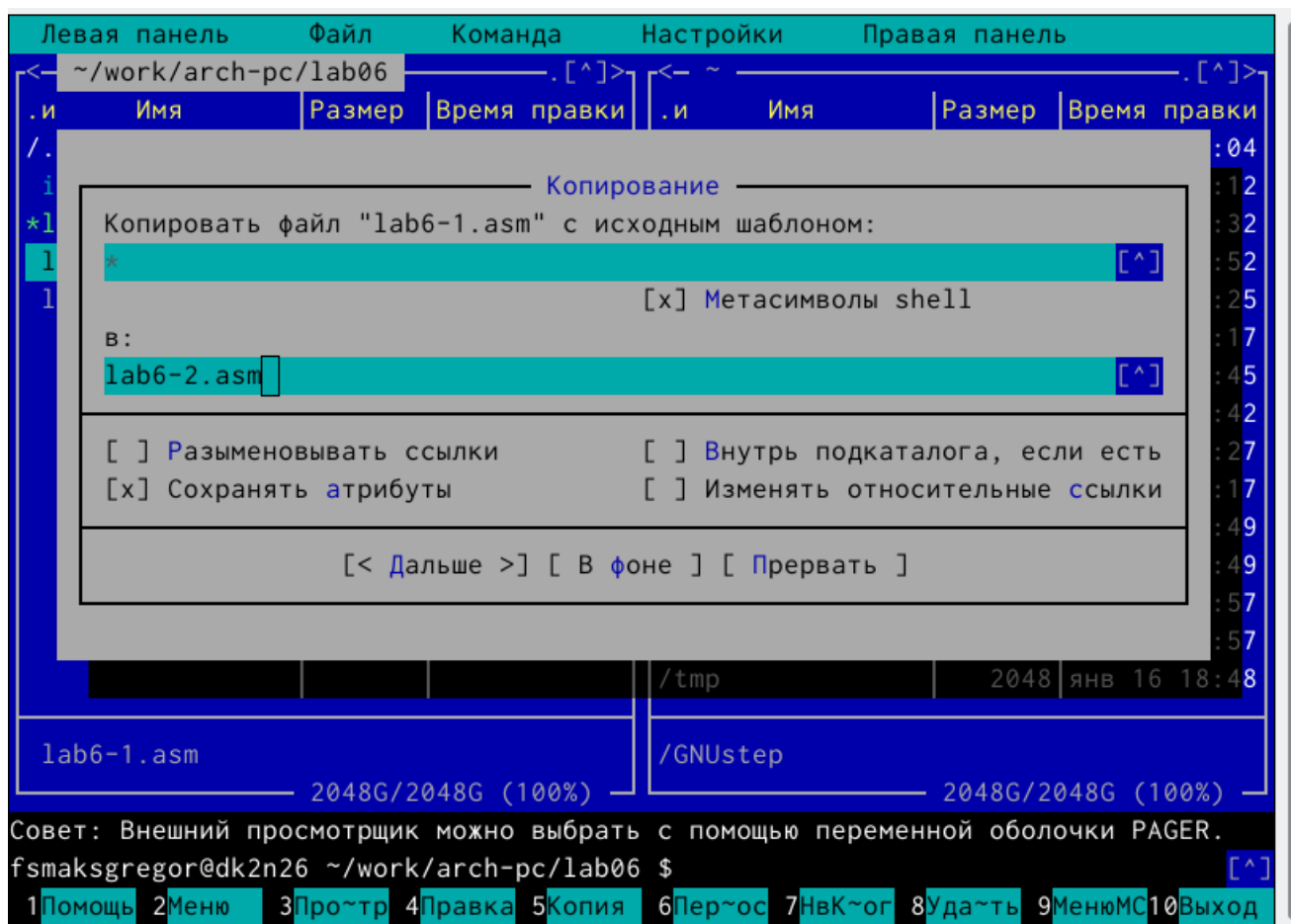


Рис. 2.10: Ресунок 10

- После этого мы исправляем текст программы в файле lab6-2.asm, используя подпрограммы из внешнего файла in\_out.asm.(рис. 2.11)

```
lab6-2.asm      [----] 0 L: [ 6+20 26/ 27] *(1310/1320b) 0010 0x00A [*][X]
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,msg ; Адрес строки 'msg' в 'ecx'
call sprintf

mov ecx, buf1
mov edx, 80
call sread
call quit
```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9МенюМС 10Выход

Рис. 2.11: Ресурсы 11

- Затем мы перевели текст программы lab6-2.asm в объектный файл . Выполнил разметку объектного файла и запустил полученный исполняемый файл.(рис. 2.12)

```
mc [fsmaksgregor@dk2n26]:~/work/arch-pc/lab06 x fsmaksgregor@dk2n26:~/work/arch-pc/lab06 x
fsmaksgregor@dk2n26 ~ $ cd work/arch-pc/lab06
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab6-2
Введите строку:
Max Sylvain
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $
```

Рис. 2.12: Ресунок 12

- На этом шаге мы меняем функцию `sprintf` на функцию `sprint`. Создал исполняемый файл, и разница заключалась в том, что эта функция изменяет входные данные на новую строку. (рис. ??) (рис. ??)

```
lab6-2.asm [-M--] 11 L:[ 6+15 21/ 27] *(1269/1318b) 0010 0x00A [*][X]
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,msg ; Адрес строки 'msg' в 'ecx'
call sprint

mov ecx, buf1
mov edx, 80
call sread

call quit
```



```
mc [fsmaksgregor@dk2n26]:~/work/arch-pc/lab06 x fsmaksgregor@dk2n26:~/work/arch-pc/lab06 x
fsmaksgregor@dk2n26 ~ $ cd work/arch-pc/lab06
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
lab6-2.asm:23: error: symbol 'buf1' not defined
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab6-2
Введите строку:
Max Sylvain
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ █
```

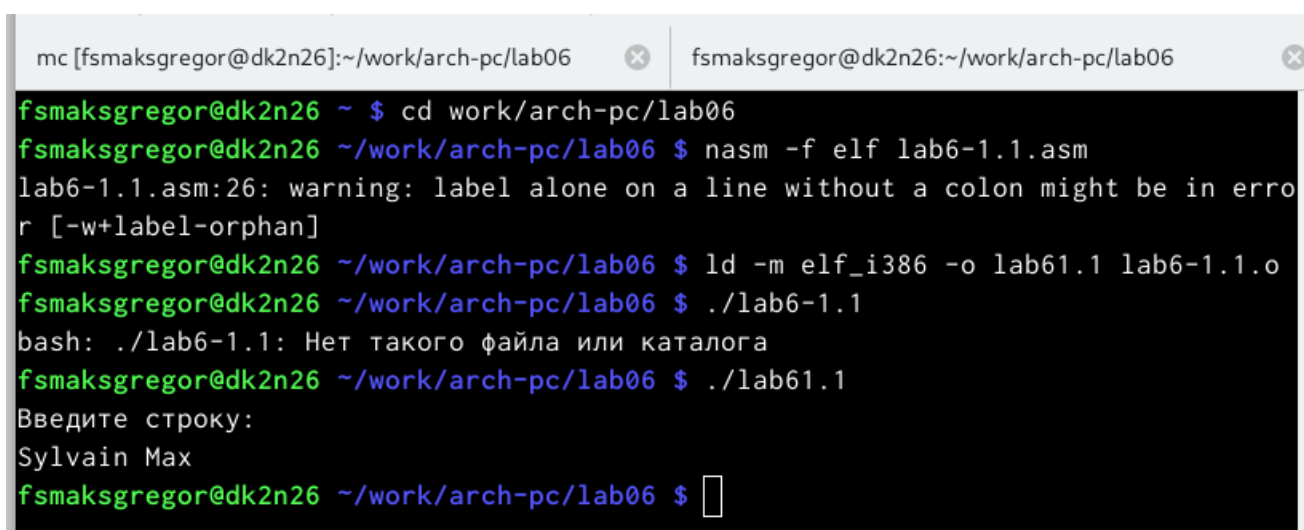
## 2.1 Выводы по результатам выполнения заданий :

- В ходе лабораторных работ мы узнали, как использовать midnight commander, и мы овладели навыками использования инструмента nasm.

## 3 Задание для самостоятельной работы :

### 3.1 Создание программы без использования внешнего файла

- В этой части мы должны были сделать копию файла lab6-1.asm, а затем мы должны были создать программу, которая запрашивает ввод строки, затем позволяет выполнить ввод с клавиатуры и, наконец, отобразить введенную строку, но без использования внешнего файла in\_out.asm.(рис. ?? ) |

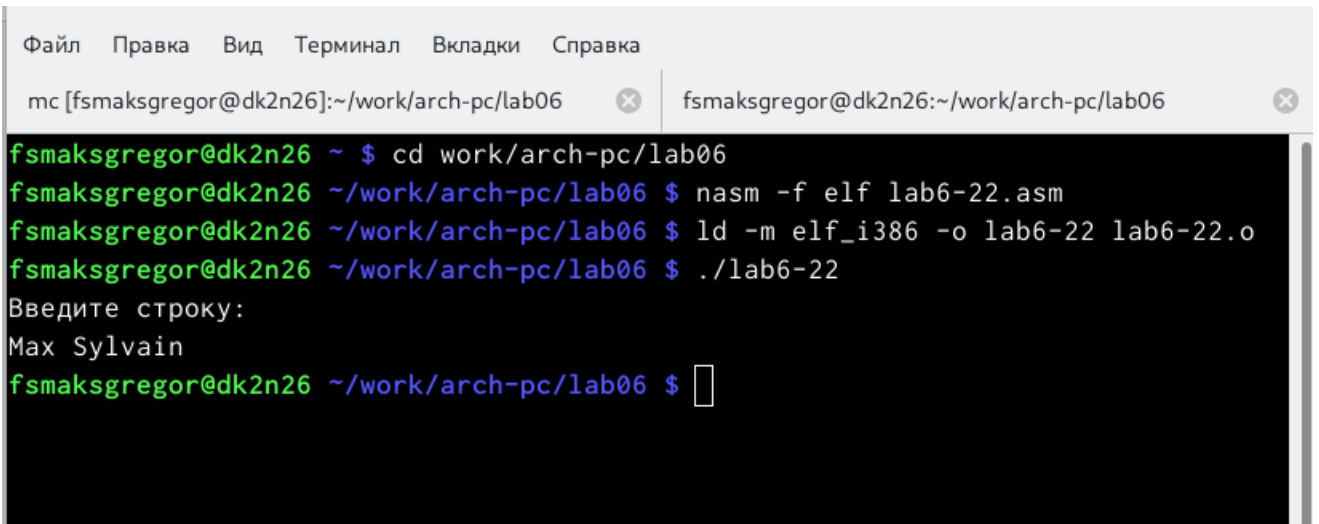


```
mc [fsmaksgregor@dk2n26]:~/work/arch-pc/lab06  fsmaksgregor@dk2n26:~/work/arch-pc/lab06
fsmaksgregor@dk2n26 ~ $ cd work/arch-pc/lab06
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.1.asm
lab6-1.1.asm:26: warning: label alone on a line without a colon might be in error [-w+label-orphan]
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab61.1 lab6-1.1.o
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab6-1.1
bash: ./lab6-1.1: Нет такого файла или каталога
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab61.1
Введите строку:
Sylvain Max
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $
```

Рис. 3.1: Ресунок 15

### 3.1.1 создание программы с использованием внешнего файла

- в этой части мы попытались выполнить ту же программу, но с использованием внешнего файла.(рис. 3.2)



```
Файл  Правка  Вид  Терминал  Вкладки  Справка
mc [fsmaksgregor@dk2n26]:~/work/arch-pc/lab06  fsmaksgregor@dk2n26:~/work/arch-pc/lab06

fsmaksgregor@dk2n26 ~ $ cd work/arch-pc/lab06
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ nasm -f elf lab6-22.asm
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-22 lab6-22.o
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $ ./lab6-22
Введите строку:
Max Sylvain
fsmaksgregor@dk2n26 ~/work/arch-pc/lab06 $
```

Рис. 3.2: Ресунок 16

#### 3.1.1.1 Выводы по результатам выполнения заданий :

- В этой части мы узнали, как создавать и редактировать программы с помощью подпрограмм и как управлять с помощью языка ассемблера.

Более подробно об Unix см. в [1–6].

## 4 Выводы

- На шестой лабораторной работе мы научимся использовать “Midnight commander” и освоим инструкции mov и int языка ассемблера и мы узнали, как создавать и редактировать программы с помощью подпрограмм и как управлять с помощью языка ассемблера.

## Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016.  
URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.