

# **Лабораторная работа №5**

**Отчет**

Устинова Виктория Вадимовна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>15</b>

## Список иллюстраций

3.1	В терминал вводим команду <code>mc</code> . . . . .	7
3.2	Клавишами F7 и FN создаем <code>lab5</code> . . . . .	8
3.3	В нижней строке прописываем команду <code>touch</code> . . . . .	8
3.4	Вводим текст программы из листинга . . . . .	9
3.5	Убеждаемся, что файл содержит текст программы . . . . .	9
3.6	Проверяем правильность выполнения . . . . .	10
3.7	Скачиваем файл . . . . .	10
3.8	Копируем файл <code>in_out.asm</code> . . . . .	10
3.9	Создаем наш файл . . . . .	11
3.10	Смотрим созданся ли файл . . . . .	11
3.11	Изменяем текст . . . . .	11
3.12	Смотрим как она работает . . . . .	12
3.13	Редактируем файл . . . . .	12
3.14	Произошли изменения с переносом строки . . . . .	12
3.15	Создали копию и внесли нужные изменения . . . . .	13
3.16	Запускаем программу . . . . .	13
3.17	Создали копию . . . . .	13
3.18	Редактируем текст, находящийся в ней . . . . .	14
3.19	Запускаем программу . . . . .	14

## **Список таблиц**

# 1 Цель работы

Приобрести практические навыки в работе с Midnight Commander. Освоить инструкцию языка ассемблера `mov` и `int`.

## 2 Задание

1. Выполнение лабораторной работы
2. Подключение внешнего файла in\_out.asm
3. Задание для самостоятельной работы

### 3 Выполнение лабораторной работы

#### Порядок выполнения лабораторной работы

Открываем Midnight Commander(рис. 3.1).

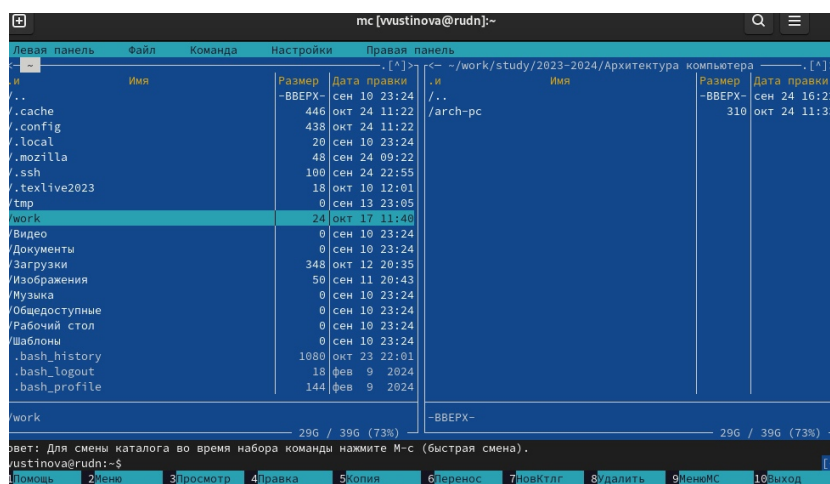


Рис. 3.1: В терминал вводим команду mc

Переходим в каталог arch-rc и создаем в нем файл lab5(рис. 3.2).

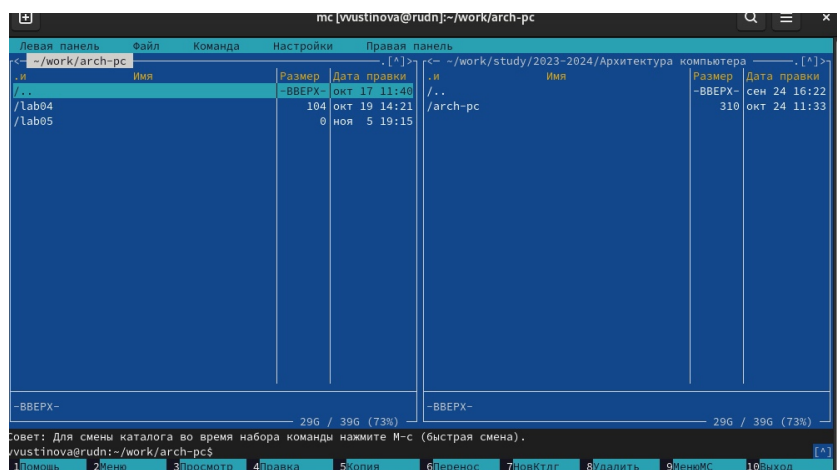


Рис. 3.2: Клавишами F7 и FN создаем lab5

Теперь переходим в созданный каталог и создаем файл lab5-1.asm(рис. 3.3).

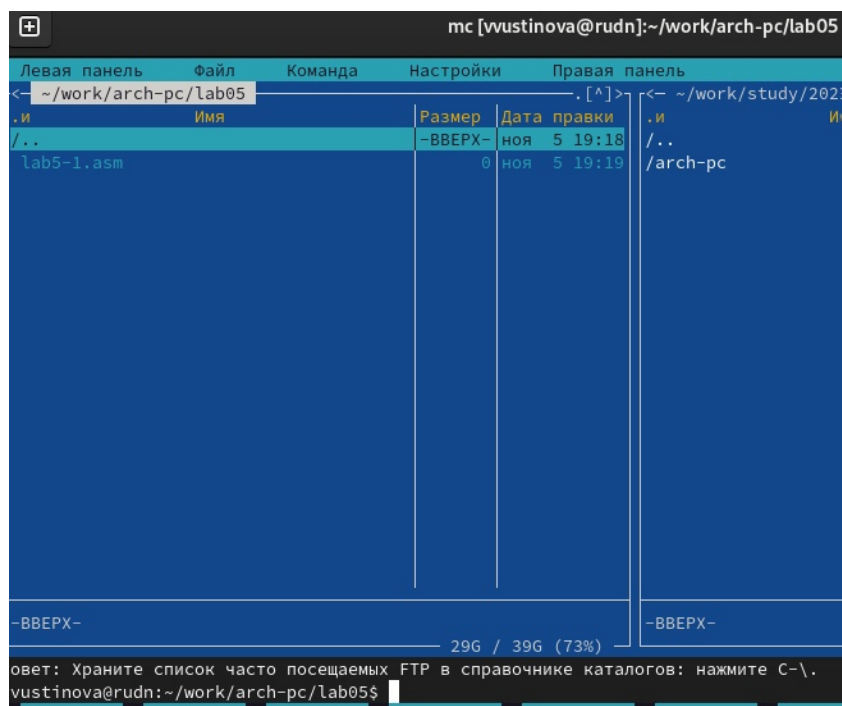
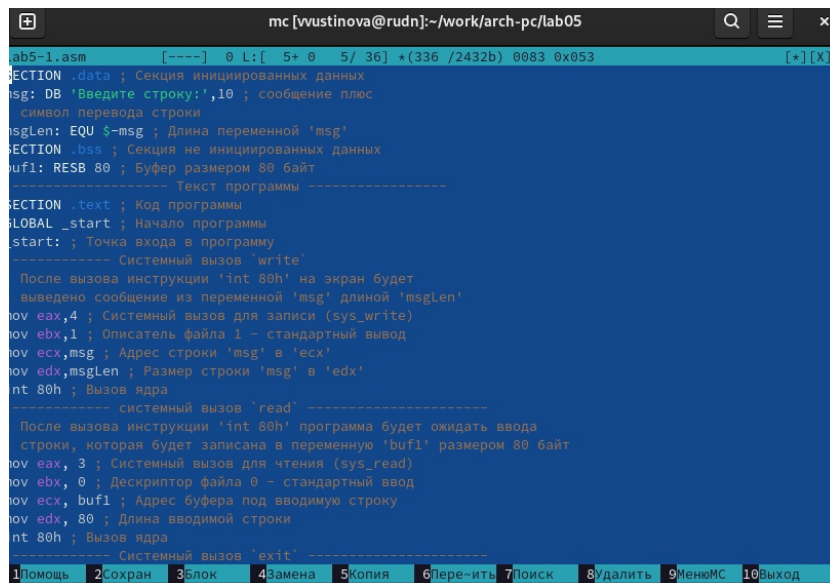


Рис. 3.3: В нижней строке прописываем команду touch

Открываем файл командой F4 и вставляем текст(рис. 3.4).

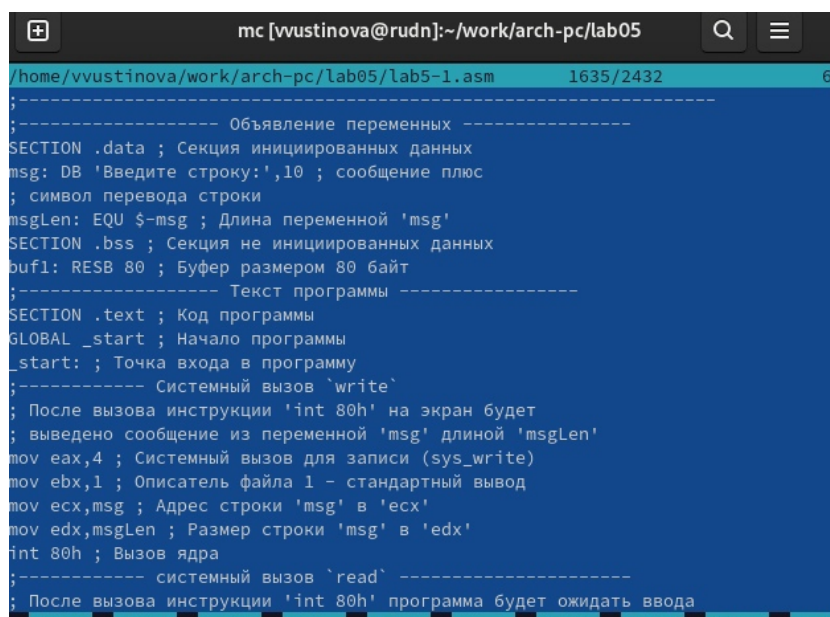




```
lab5-1.asm [----] 0 L: [ 5+ 0 5/ 36] *(336 /2432b) 0083 0x053 [*] [X]
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных данных
buf1: RESB 80 ; Буфер размером 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' -----
; Помощь 2Сохран 3Блок 4Замена 5Копия 6Пере-ить 7Поиск 8/далить 9МенюМС 10Выход
```

Рис. 3.4: Вводим текст программы из листинга

С помощью F3 открываем файл(рис. 3.5).



```
/home/vvustinova/work/arch-pc/lab05/lab5-1.asm 1635/2432 67
;-----
;----- Объявление переменных -----
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных 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
```

Рис. 3.5: Убеждаемся, что файл содержит текст программы

Транслируем текст программы lab5-1.asm в объектный файл.(рис. 3.6).

```

vvustinova@rudn:~$ mc

vvustinova@rudn:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
vvustinova@rudn:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
vvustinova@rudn:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Устинова Виктория Вадимовна
vvustinova@rudn:~/work/arch-pc/lab05$

```

Рис. 3.6: Проверяем правильность выполнения

## Подключение внешнего файла in\_out.asm

Скачиваем файл in\_out.asm со страницы курса в ТУИСе(рис. 3.7).

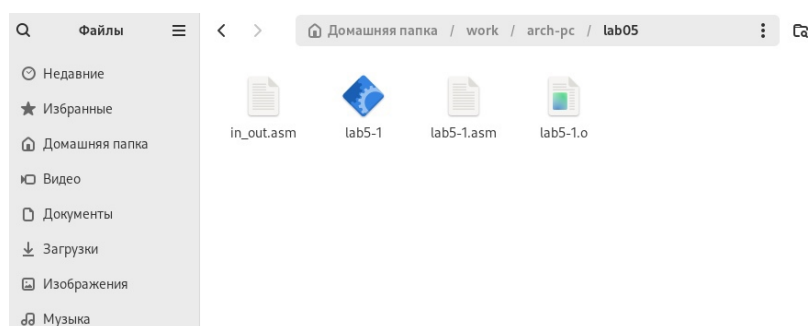


Рис. 3.7: Скачиваем файл

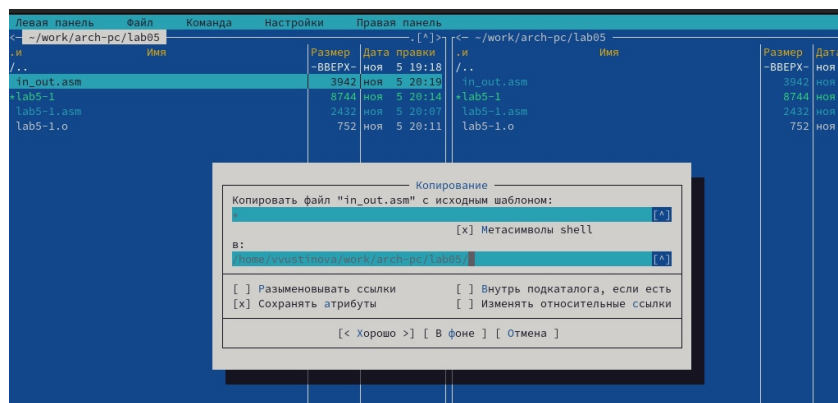


Рис. 3.8: Копируем файл in\_out.asm

С помощью функциональной клавиши F6 создаем копию файла lab5-1.asm с именем lab5-2.asm. (рис. 3.9).

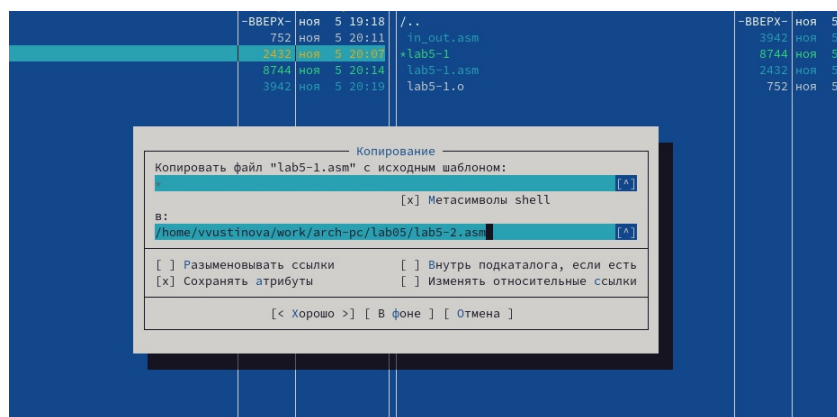


Рис. 3.9: Создаем наш файл

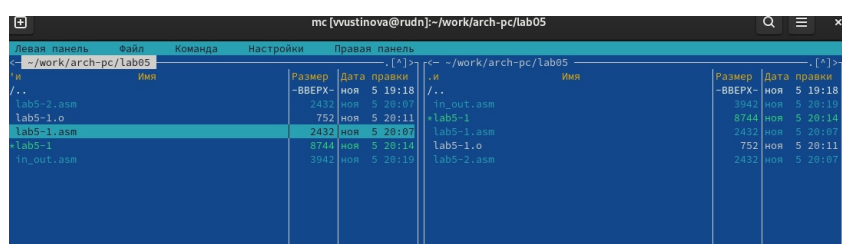


Рис. 3.10: Смотрим созданся ли файл

Исправляем текст программы в новом файле с использованием программы из внешнего файла in\_out.asm(рис. 3.11).

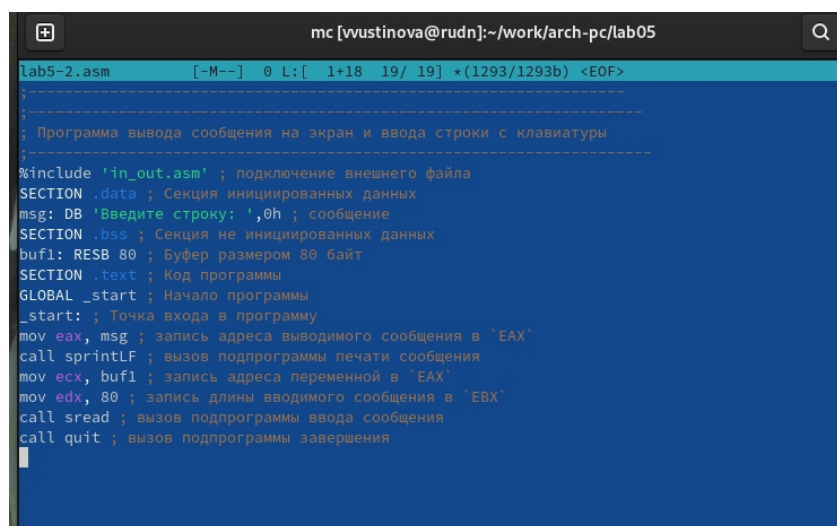


Рис. 3.11: Изменяем текст

Запускаем нашу программу(рис. 3.12).

```
vvustinova@rudn:~/work/arch-pc/lab05$ mc
vvustinova@rudn:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
vvustinova@rudn:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
vvustinova@rudn:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Устинова Виктория Вадимовна
vvustinova@rudn:~/work/arch-pc/lab05$
```

Рис. 3.12: Смотрим как она работает

Открываем тот же файл и меняем `sprintf` на `sprint`(рис. 3.13).

```
lab5-2.asm [-----] 0 L: [ 1+ 0 1/ 19] *(0 /1291b) 0059 0x03B [*]
;
;-----
; 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 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

Запускаем нашу программу(рис. 3.14).

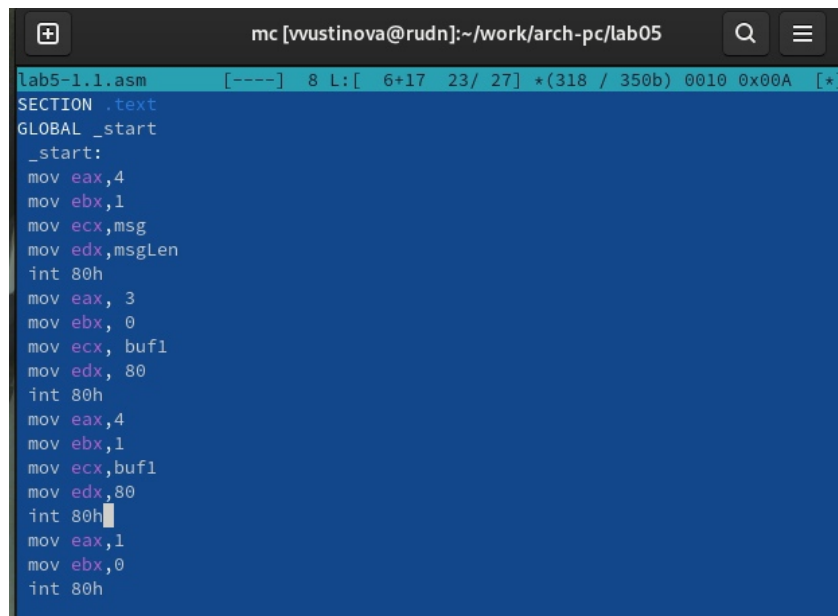
```
vvustinova@rudn:~/work/arch-pc/lab05$ mc
vvustinova@rudn:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
vvustinova@rudn:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
vvustinova@rudn:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Устинова Виктория Вадимовна
vvustinova@rudn:~/work/arch-pc/lab05$
```

Рис. 3.14: Произошли изменения с переносом строки

Разница `sprintf` и `sprint` в том, что строка, где мы вводим данные меняет свое местоположение

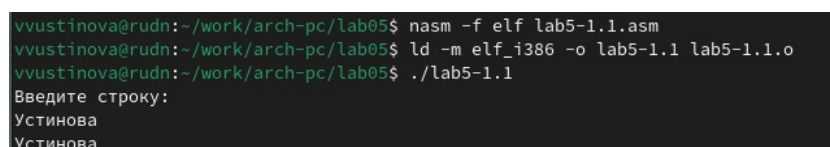
### Задания для самостоятельной работы

Создаем копию файла `lab5-1.asm`(рис. 3.15).



```
lab5-1.1.asm [----] 8 L: [ 6+17 23/ 27] *(318 / 350b) 0010 0x00A [*]
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. 3.15: Создали копию и внесли нужные изменения



```
vvustinova@rudn:~/work/arch-pc/lab05$ nasm -f elf lab5-1.1.asm
vvustinova@rudn:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1.1 lab5-1.1.o
vvustinova@rudn:~/work/arch-pc/lab05$ ./lab5-1.1
Введите строку:
Устинова
Устинова
```

Рис. 3.16: Запускаем программу

Создаем копию файла lab5-2.asm(рис. 3.17).

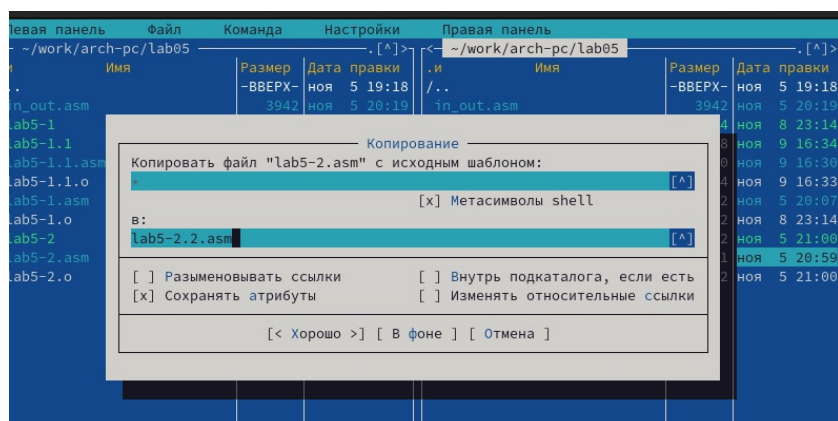


Рис. 3.17: Создали копию

```
mc [vvustinova@rudn]:~/work/arch-pc/lab05
lab5-2.2.asm [-M--] 15 L: [ 1+18 19/ 21] *(1216/1232b) 0010 0x00A
-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
-----
include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не иницированных 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 sprintfL ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
call sread
mov eax,buf1
call sprintf
call quit
```

Рис. 3.18: Редактируем текст, находящийся в ней

```
vvustinova@rudn:~/work/arch-pc/lab05$ nasm -f elf lab5-2.2.asm
vvustinova@rudn:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2.2 lab5-2.2.o
vvustinova@rudn:~/work/arch-pc/lab05$ ./lab5-2.2
Введите строку:
Устинова
Устинова
vvustinova@rudn:~/work/arch-pc/lab05$
```

Рис. 3.19: Запускаем программу

## 4 Выводы

У нас получилось приобрести навыки работы с Midnight Commander и освоить инструкцию языка ассемблера `mov` и `int`.