

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Лаптев Тимофей Сергеевич

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Основы работы с mc	6
3.2	Структура программы на языке ассемблера NASM	8
3.3	Подключение внешнего файла	9
3.4	Выполнение заданий для самостоятельной работы	12
4	Выводы	18
5	Список литературы	19

Список иллюстраций

3.1	Установка и открытие mc	6
3.2	Открытый mc	7
3.3	Создание каталога	7
3.4	Перемещение между директориями и создание файла	8
3.5	Открытие файла и его редактирование	8
3.6	Компиляция файла и передача на обработку компоновщику . . .	9
3.7	Изменение файла	10
3.8	Исполнение файла	10
3.9	Отредактированный файл	11
3.10	Исполнение файла	11
3.11	Копирование файла	12
3.12	Редактирование файла	13
3.13	Исполнение файла	13
3.14	Копирование файла и его изменение	15
3.15	Исполнение файла	16

1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в Midnight Commander, освоение инструкций языка ассемблера `mov` и `int`.

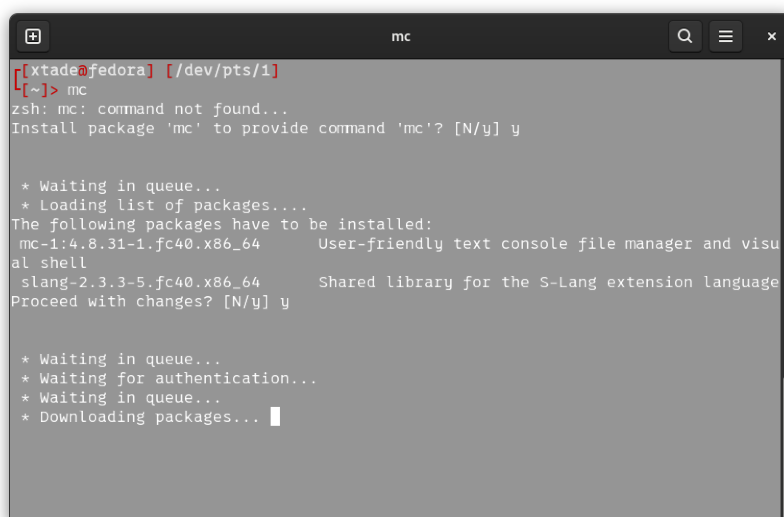
2 Задание

1. Основы работы с тс
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 mc

Устанавливаю и открываю Midnight Commander, введя в терминал mc (рис. 3.1).



```
[xtade@fedora] [/dev/pts/1]
[~]> mc
zsh: mc: command not found...
Install package 'mc' to provide command 'mc'? [N/y] y

* Waiting in queue...
* Loading list of packages...
The following packages have to be installed:
mc-1:4.8.31-1.fc40.x86_64      User-friendly text console file manager and visu
al shell
slang-2.3.3-5.fc40.x86_64    Shared library for the S-Lang extension language
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages... █
```

Рис. 3.1: Установка и открытие mc

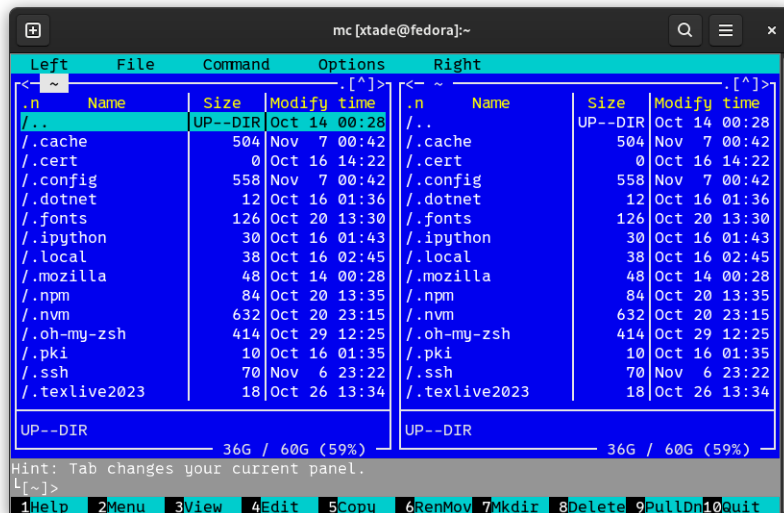


Рис. 3.2: Открытый mc

Перехожу в каталог ~/work/study/2022-2023/Архитектура Компьютера/arch-рс, используя файловый менеджер mc. С помощью функциональной клавиши F7 создаю каталог lab05 (рис. 3.3).

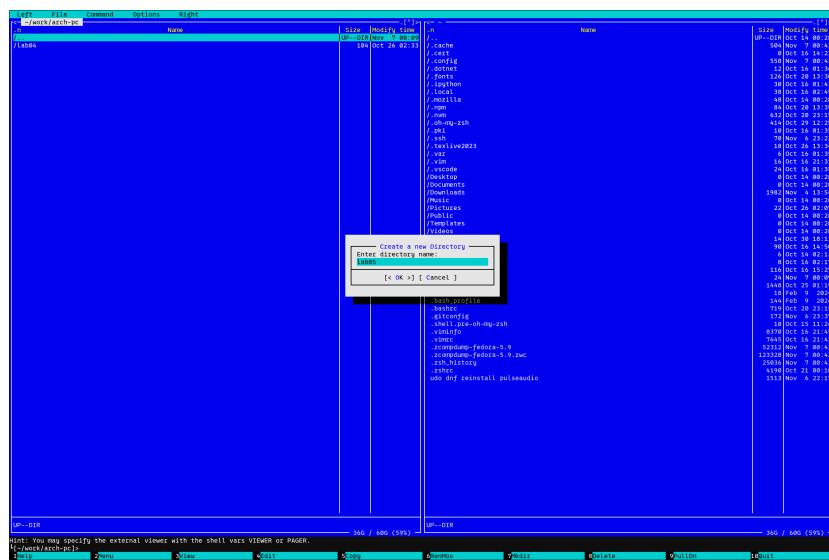


Рис. 3.3: Создание каталога

Перехожу в созданный каталог. В строке ввода прописываю команду touch lab5-1.asm, чтобы создать файл, в котором буду работать (рис. 3.4).

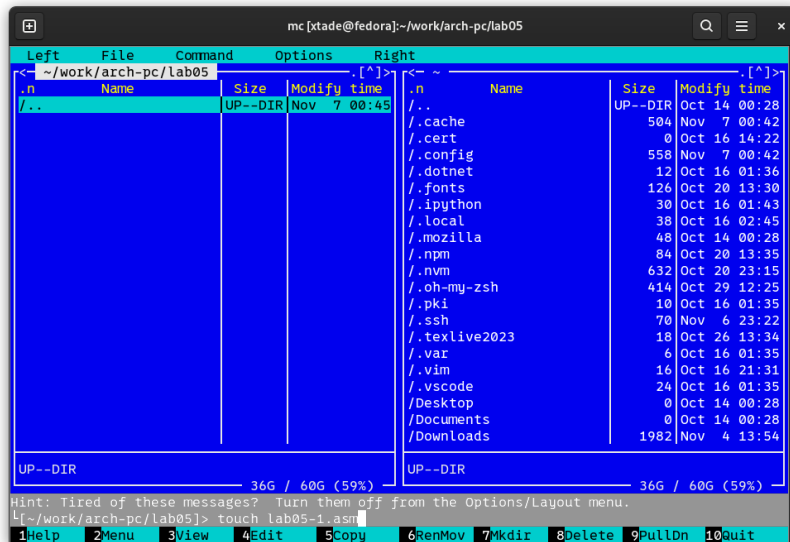


Рис. 3.4: Перемещение между директориями и создание файла

3.2 Структура программы на языке ассемблера NASM

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano. Ввожу в файл код программы для запроса строки у пользователя (рис. 3.5). Далее выхожу из файла, сохраняя изменения.

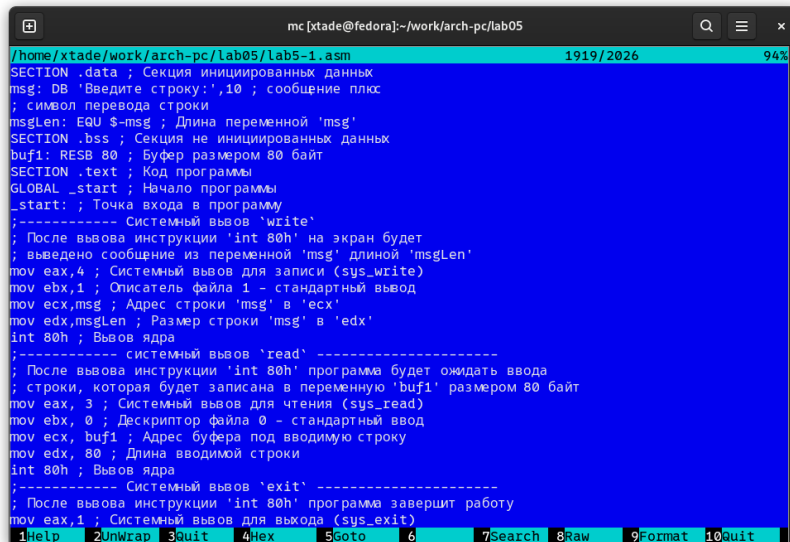
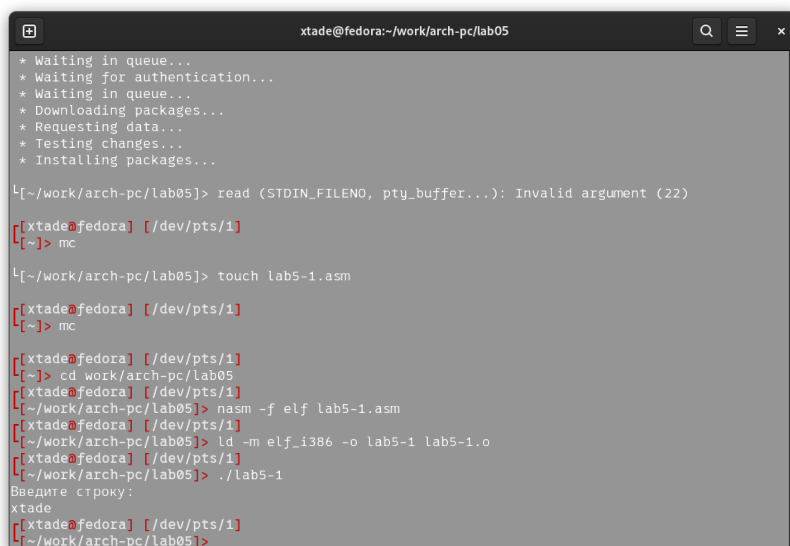


Рис. 3.5: Открытие файла и его редактирование

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-1.asm`. Создался объектный файл `lab5-1.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-1 lab5-1.o`. Создался исполняемый файл `lab5-1`. Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу данные, на этом программа заканчивает свою работу (рис. 3.6).



```
xtade@fedora:~/work/arch-pc/lab05
* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

[~/work/arch-pc/lab05]> read (STDIN_FILENO, pty_buffer...): Invalid argument (22)

[xtade@fedora] [/dev/pts/1]
[~]> mc

[~/work/arch-pc/lab05]> touch lab5-1.asm

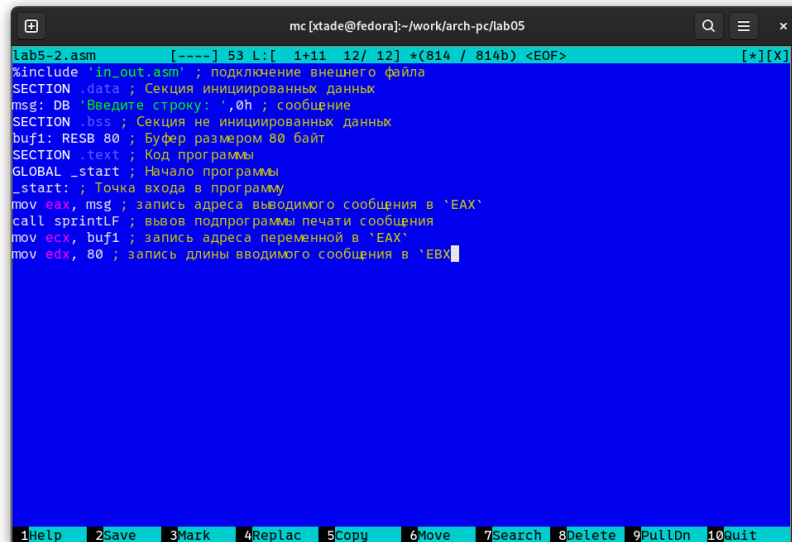
[xtade@fedora] [/dev/pts/1]
[~]> mc

[xtade@fedora] [/dev/pts/1]
[~]> cd work/arch-pc/lab05
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05]> nasm -f elf lab5-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05]> ld -m elf_i386 -o lab5-1 lab5-1.o
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05]> ./lab5-1
Введите строку:
xtade
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05]>
```

Рис. 3.6: Компиляция файла и передача на обработку компоновщику

3.3 Подключение внешнего файла

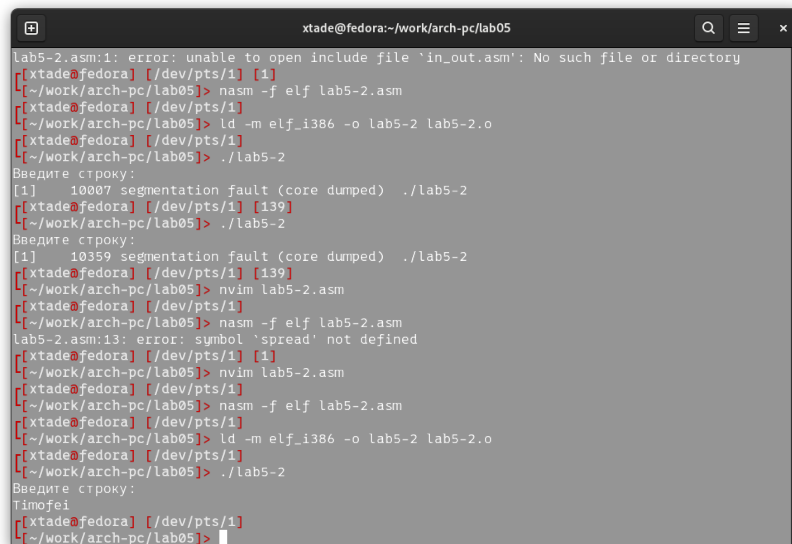
С помощью функциональной клавиши F5 копирую файл `lab6-1` в тот же каталог, но с другим именем, для этого в появившемся окне `mc` прописываю имя для копии файла. Изменяю содержимое файла `lab5-2.asm` (рис. 3.7).



```
lab5-2.asm [----] 53 L: [ 1+11 12/ 12] *(814 / 814b) <EOF> [*][X]
#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 printf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины выводимого сообщения в 'EDX'
```

Рис. 3.7: Изменение файла

Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-2.asm`. Создался объектный файл `lab5-2.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-2 lab5-2.o`. Создался исполняемый файл `lab5-2`. Запускаю исполняемый файл (рис. 3.8).

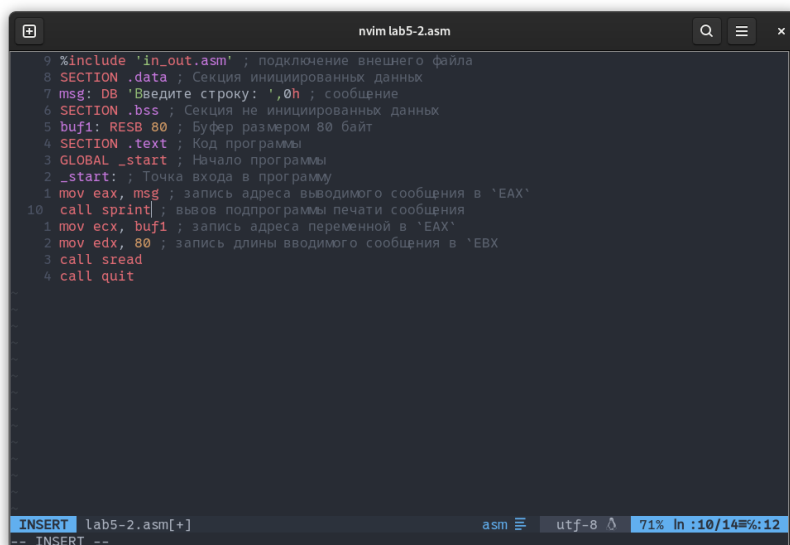


```
lab5-2.asm:1: error: unable to open include file 'in_out.asm': No such file or directory
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
[xtade@fedora] [/dev/pts/1] [139]
[~/work/arch-pc/lab05] ld -m elf_i386 -o lab5-2 lab5-2.o
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку:
[1] 10007 segmentation fault (core dumped) ./lab5-2
[xtade@fedora] [/dev/pts/1] [139]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку:
[1] 10359 segmentation fault (core dumped) ./lab5-2
[xtade@fedora] [/dev/pts/1] [139]
[~/work/arch-pc/lab05] nvim lab5-2.asm
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
lab5-2.asm:13: error: symbol 'spread' not defined
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] nvim lab5-2.asm
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] ld -m elf_i386 -o lab5-2 lab5-2.o
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку:
Timofei
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] [1]
```

Рис. 3.8: Исполнение файла

Открываю файл `lab5-2.asm`. Изменяю в нем подпрограмму `sprintf` на `sprint`.

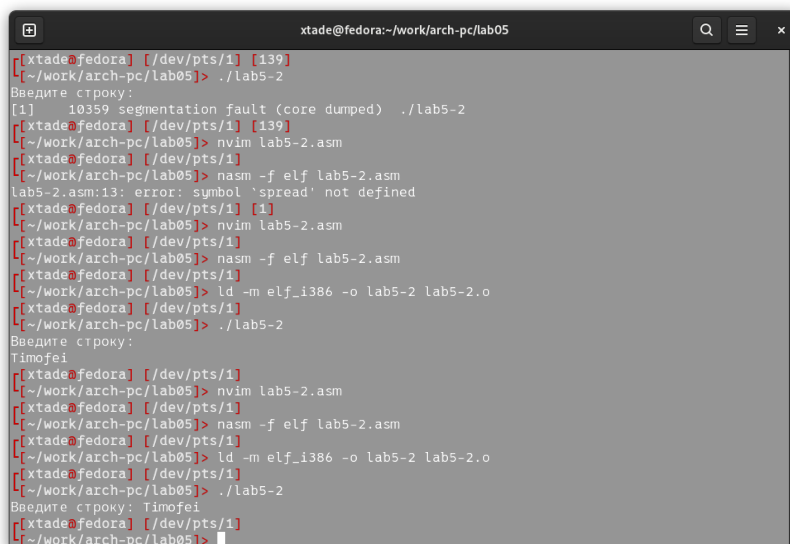
Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий (рис. 3.9).



```
9 %include 'in_out.asm' ; подключение внешнего файла
8 SECTION .data ; Секция инициализированных данных
7 msg: DB 'Введите строку: ',0h ; сообщение
6 SECTION .bss ; Секция не инициализированных данных
5 buf1: RESB 80 ; Буфер размером 80 байт
4 SECTION .text ; Код программы
3 GLOBAL _start ; Начало программы
2 _start: ; Точка входа в программу
1 mov eax, msg ; запись адреса выводимого сообщения в 'EAX'
10 call sprint ; вывод подпрограммы печати сообщения
1 mov ecx, buf1 ; запись адреса переменной в 'EAX'
2 mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
3 call sread
4 call quit
```

Рис. 3.9: Отредактированный файл

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. 3.10).



```
[xtade@fedora] [/dev/pts/1] [139]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку:
[1] 10359 segmentation fault (core dumped) ./lab5-2
[xtade@fedora] [/dev/pts/1] [139]
[~/work/arch-pc/lab05] nasm lab5-2.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
lab5-2.asm:13: error: symbol 'spread' not defined
[xtade@fedora] [/dev/pts/1] [1]
[~/work/arch-pc/lab05] nasm lab5-2.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] ld -m elf_i386 -o lab5-2 lab5-2.o
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку:
Timofei
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] nasm lab5-2.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] nasm -f elf lab5-2.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] ld -m elf_i386 -o lab5-2 lab5-2.o
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] ./lab5-2
Введите строку: Timofei
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05]
```

Рис. 3.10: Исполнение файла

Разница между первым исполняемым файлом lab6-2 и вторым lab5-2-2 в том,

что запуск первого запрашивает ввод с новой строки, а программа, которая выполняется при запуске второго, запрашивает ввод без переноса на новую строку, потому что в этом заключается различие между подпрограммами `sprintLF` и `sprint`.

3.4 Выполнение заданий для самостоятельной работы

1. Создаю копию файла `lab5-1.asm` с именем `lab5-1-1.asm` с помощью функциональной клавиши F5 (рис. 3.11).

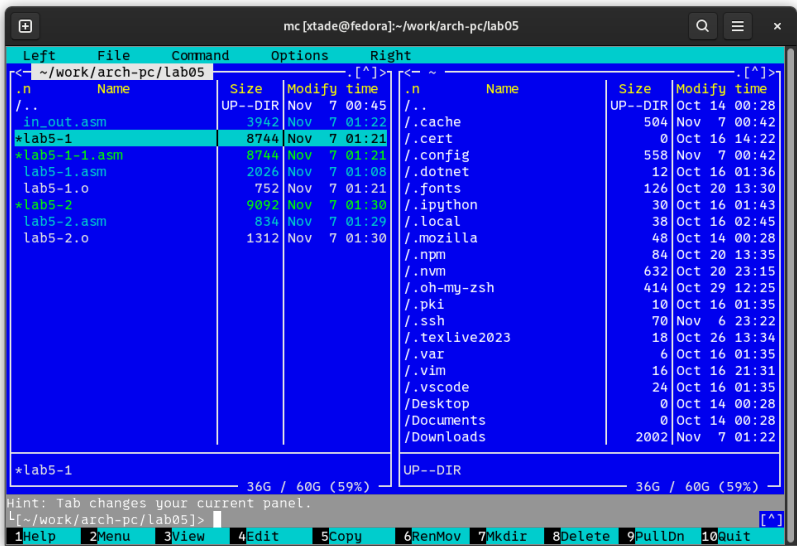
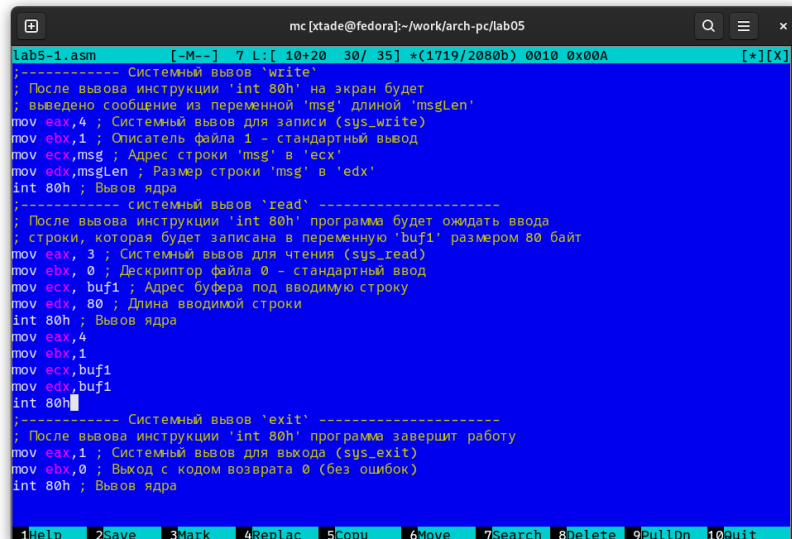


Рис. 3.11: Копирование файла

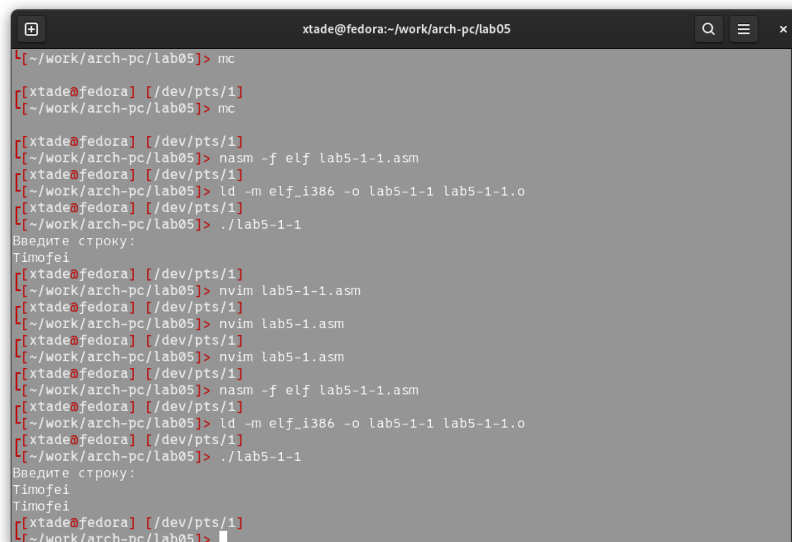
С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 3.12).



```
mc [xtade@fedora] ~/work/arch-pc/lab05
lab5-1.asm [-M--] 7 L: [ 10+20 30/ 35] *(1719/2080b) 0010 0x00A [+] [X]
;----- Системный вызов 'write' -----
; После вывода 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 ; Вывод ядра
;----- системный вызов 'read' -----
; После вывода 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 ; Вывод ядра
mov eax,4
mov ebx,1
mov ecx,buf1
mov edx,buf1
int 80h
;----- Системный вызов 'exit' -----
; После вывода 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)
int 80h ; Вывод ядра
1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Рис. 3.12: Редактирование файла

2. Создаю объектный файл lab5-1-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-1-1, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу имя, далее программа выводит введенные мной данные (рис. 3.13).



```
xtade@fedora: ~/work/arch-pc/lab05
[~/work/arch-pc/lab05] > mc
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > mc
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > nasm -f elf lab5-1-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > ./lab5-1-1
Введите строку:
Timofei
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > nvim lab5-1-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > nvim lab5-1-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > nvim lab5-1-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > nasm -f elf lab5-1-1.asm
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] > ./lab5-1-1
Введите строку:
Timofei
[xtade@fedora] [/dev/pts/1]
[~/work/arch-pc/lab05] >
```

Рис. 3.13: Исполнение файла

Код программы из пункта 1:

```

SSECTION .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
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
mov ebx,1
mov ecx,buf1
mov edx,buf1

```

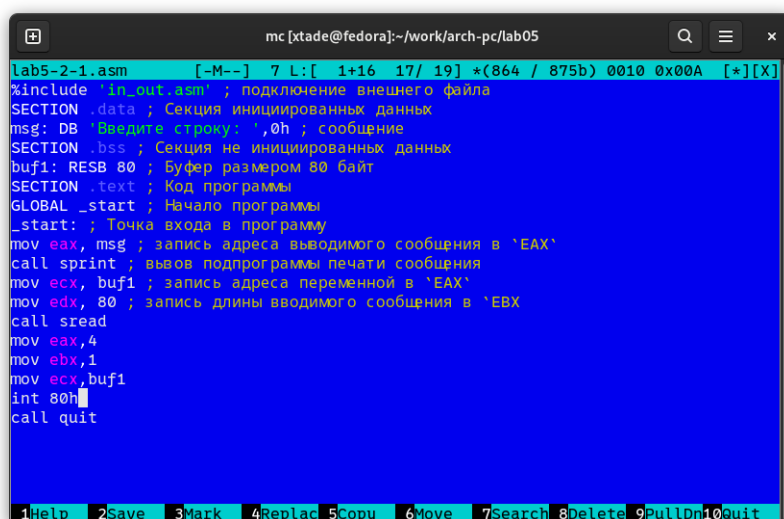
```

int 80h

;----- 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)
int 80h ; Вызов ядра

```

3. Создаю копию файла lab5-2.asm с именем lab5-2-1.asm с помощью функциональной клавиши F5. С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 3.14).



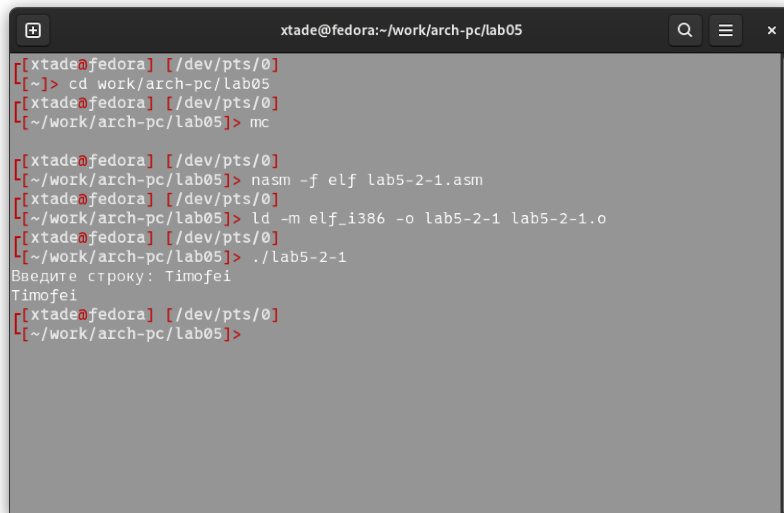
```

lab5-2-1.asm  [-M--]  7 L:[ 1+16 17/ 19] *(864 / 875b) 0010 0x00A [*][X]
%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 sprint ; 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,4
mov ebx,1
mov ecx,buf1
int 80h
call quit

```

Рис. 3.14: Копирование файла и его изменение

4. Создаю объектный файл lab5-2-1.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2-1, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу имя, далее программа выводит введенные мной данные (рис. 3.15).



```
xtade@fedora: ~/work/arch-pc/lab05
[xtade@fedora] [/dev/pts/0]
[~]> cd work/arch-pc/lab05
[xtade@fedora] [/dev/pts/0]
[~/work/arch-pc/lab05]> mc

[xtade@fedora] [/dev/pts/0]
[~/work/arch-pc/lab05]> nasm -f elf lab5-2-1.asm
[xtade@fedora] [/dev/pts/0]
[~/work/arch-pc/lab05]> ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
[xtade@fedora] [/dev/pts/0]
[~/work/arch-pc/lab05]> ./lab5-2-1
Введите строку: Timofei
Timofei
[xtade@fedora] [/dev/pts/0]
[~/work/arch-pc/lab05]>
```

Рис. 3.15: Исполнение файла

Код программы из пункта 3:

```
%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
mov eax,4
mov ebx,1
mov ecx,buf1
```



```
int 80h  
call quit
```

4 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы в Midnight Commander, а также освоила инструкции языка ассемблера `mov` и `int`.

5 Список литературы

1. Лабораторная работа №6