

Отчёт по лабораторной работе №5

Дисциплина: архитектура компьютера

Ахмад мд Шешир

Содержание

1	Цель работы	6
2	Задание	7
3	Выполнение лабораторной работы	8
3.0.1	Изучение Midnight Commander	8
3.0.2	Структура программы на языке ассемблера NASM	11
3.0.3	Подключение внешнего файла	13
4	Выполнение заданий для самостоятельной работы	19
4.0.1	1	20
4.0.2	2	24
5	Выводы	27
6	Список литературы	28

Список иллюстраций

3.1	Открытие Midnight Commander	8
3.2	Перехожу в каталог ~/work/arch-pc, используя файловый менеджер mc	9
3.3	С помощью функциональной клавиши F7 создаю каталог lab05 . .	10
3.4	В строке ввода прописываю команду touch lab5-1.asm, чтобы создать файл, в котором буду работать	11
3.5	С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы . .	12
3.6	Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу	12
3.7	Скачиваю файл in_out.asm со страницы курса в ТУИС. Он сохранился в каталог “Загрузки”	13
3.8	С помощью функциональной клавиши F6 переместил файл in_out.asm из каталога Загрузки в созданный каталог lab05	14
3.9	С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя lab5-2.asm для копии файла	15
3.10	16
3.11	Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in_out.asm.	16
3.12	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	17
3.13	Открываю файл lab5-2.asm для редактирования в nano функциональной клавишей F4. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий	17
3.14	Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл	18
4.1	Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5	20
4.2	Проверка	21

4.3	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	22
4.4	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 свои ФИО, далее программа выводит введенные мною данные	23
4.5	Создаю копию файла lab5-2.asm с именем lab5-2-2.asm с помощью функциональной клавиши F5	24
4.6	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	25
4.7	Создаю объектный файл lab5-2-2.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab6-2-2, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные	26

Список таблиц

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. Открыть Midnight Commander.
2. Создать папку lab05, где будут храниться файлы лабораторной работы №5.
3. Создать файл «lab5-1.asm», ввести текст программы. Оттранслировать текст программы, выполнить компоновку объектного файла и запустить получившийся исполняемый файл.
4. Скачать с ТУИС файл «in_out.asm» и переместить его в каталог lab05.
5. Скопировать файл «lab5-1.asm» с именем «lab5-2.asm» и исправить текст программы так чтобы использовались программы из внешнего файла «in_out.asm».
6. Создать исполняемый файл и проверить его работу.
7. Создать копию файла «lab5-1.asm». Внести изменения в программу (без использования внешнего файла «in_out.asm»), так чтобы она работала по определённому алгоритму.
8. Создать копию файла «lab5-2.asm». Также исправить текст программы, но уже с использованием подпрограмм из внешнего файла «in_out.asm», так чтобы она работала по определённому алгоритму.
9. Создать исполняемые файлы и проверить их работу.

3 Выполнение лабораторной работы

3.0.1 Изучение Midnight Commander

Открываем Midnight Commander с помощью команды 'mc'

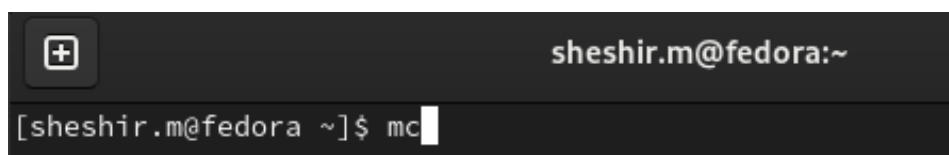


Рис. 3.1: Открытие Midnight Commander

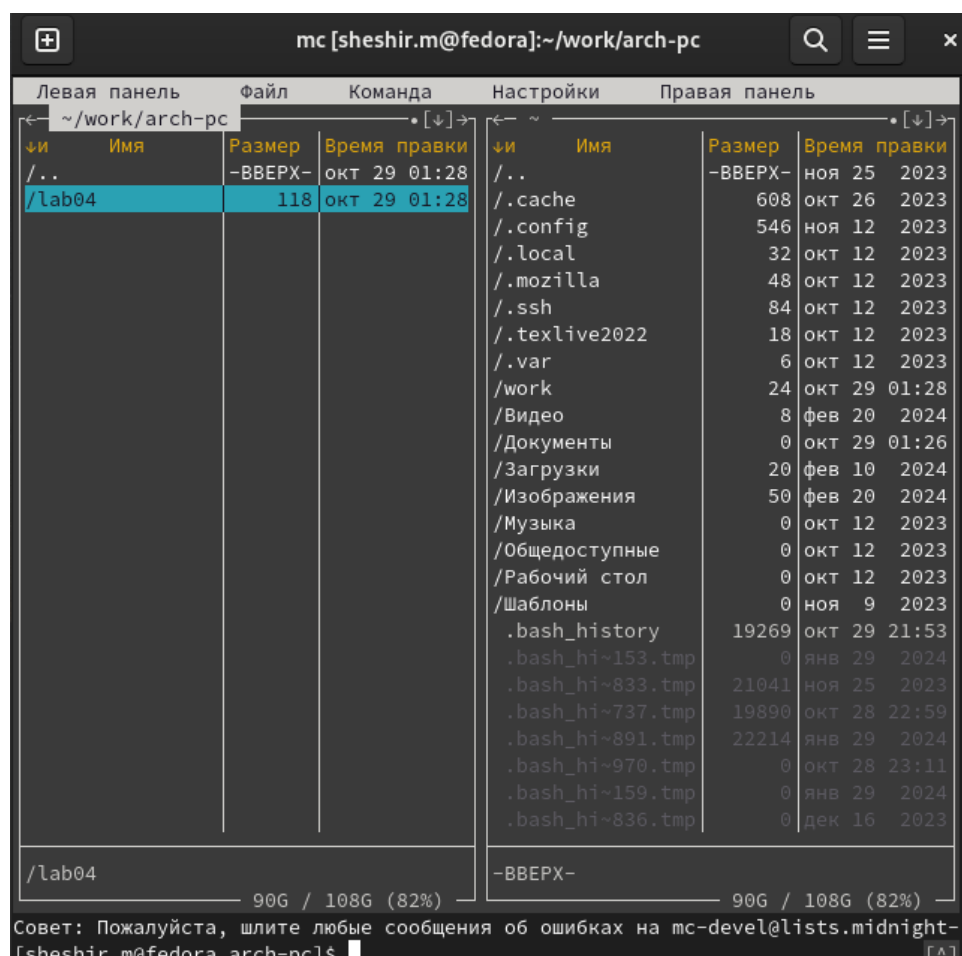


Рис. 3.2: Перехожу в каталог ~/work/arch-pc, используя файловый менеджер mc

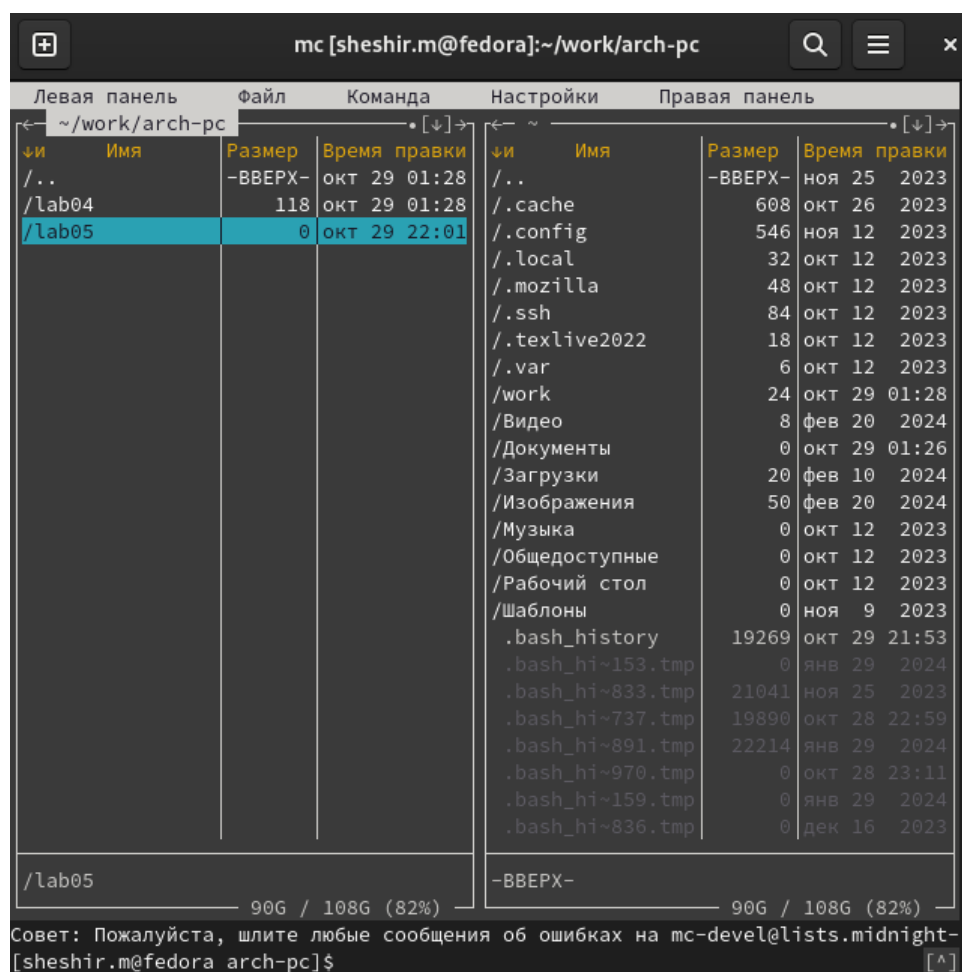


Рис. 3.3: С помощью функциональной клавиши F7 создаю каталог lab05

Перехожу в созданный каталог

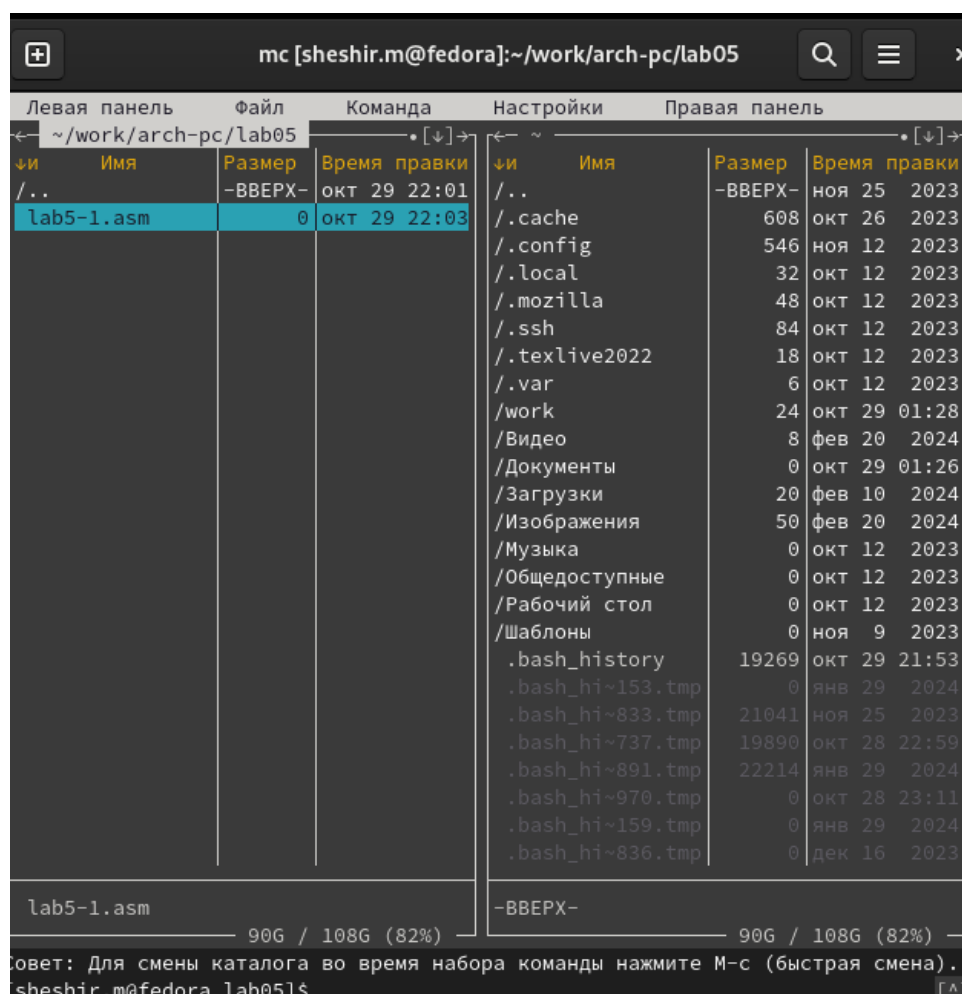
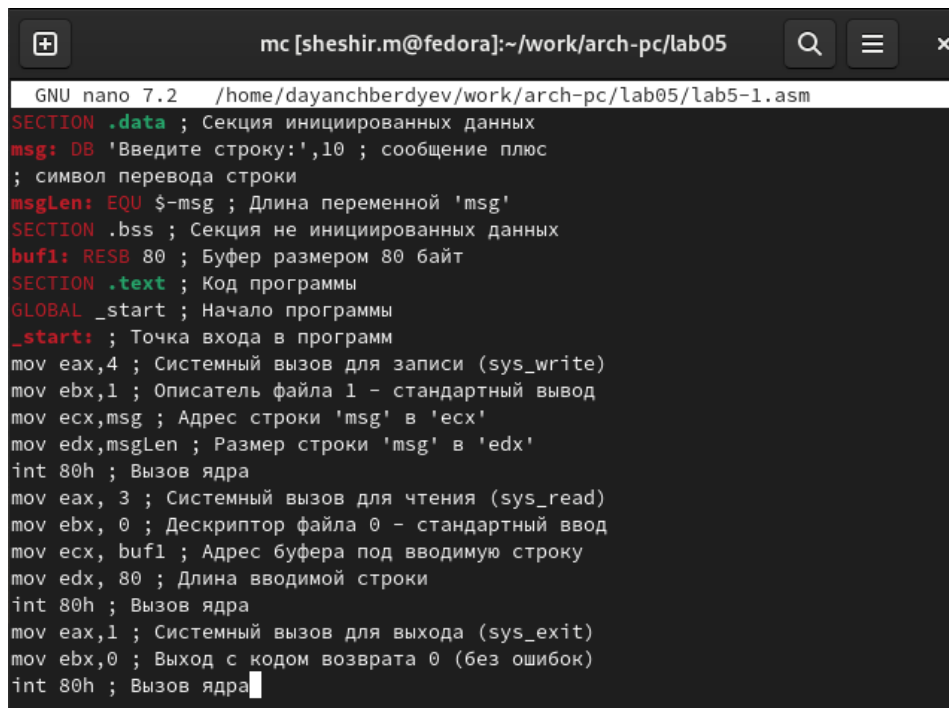


Рис. 3.4: В строке ввода прописываю команду `touch lab5-1.asm`, чтобы создать файл, в котором буду работать

3.0.2 Структура программы на языке ассемблера NASM

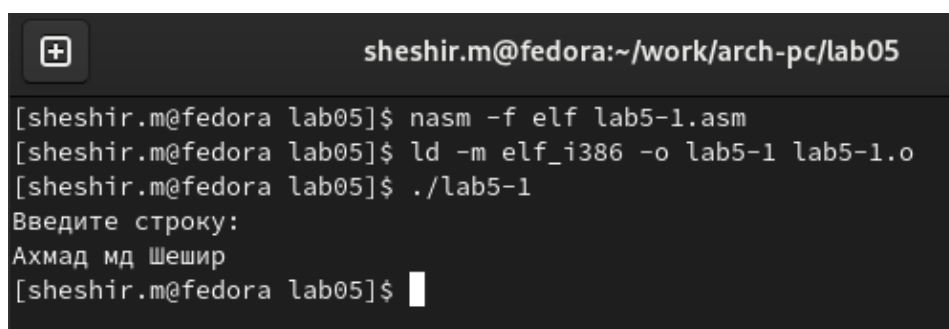
С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano. Ввожу в файл код программы для запроса строки у пользователя. Далее выхожу из файла (Ctrl+X), сохраняя изменения (Y, Enter).



```
GNU nano 7.2 /home/dayanchberdyev/work/arch-pc/lab05/lab5-1.asm
SECTION .data ; Секция инициированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
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 программ
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,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

Рис. 3.5: С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы

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`.



```
sheshir.m@fedora:~/work/arch-pc/lab05
[sheshir.m@fedora lab05]$ nasm -f elf lab5-1.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-1 lab5-1.o
[sheshir.m@fedora lab05]$ ./lab5-1
Введите строку:
Ахмад мд Шешир
[sheshir.m@fedora lab05]$
```

Рис. 3.6: Запускаю исполняемый файл. Программа выводит строку “Введите строку:” и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу

3.0.3 Подключение внешнего файла

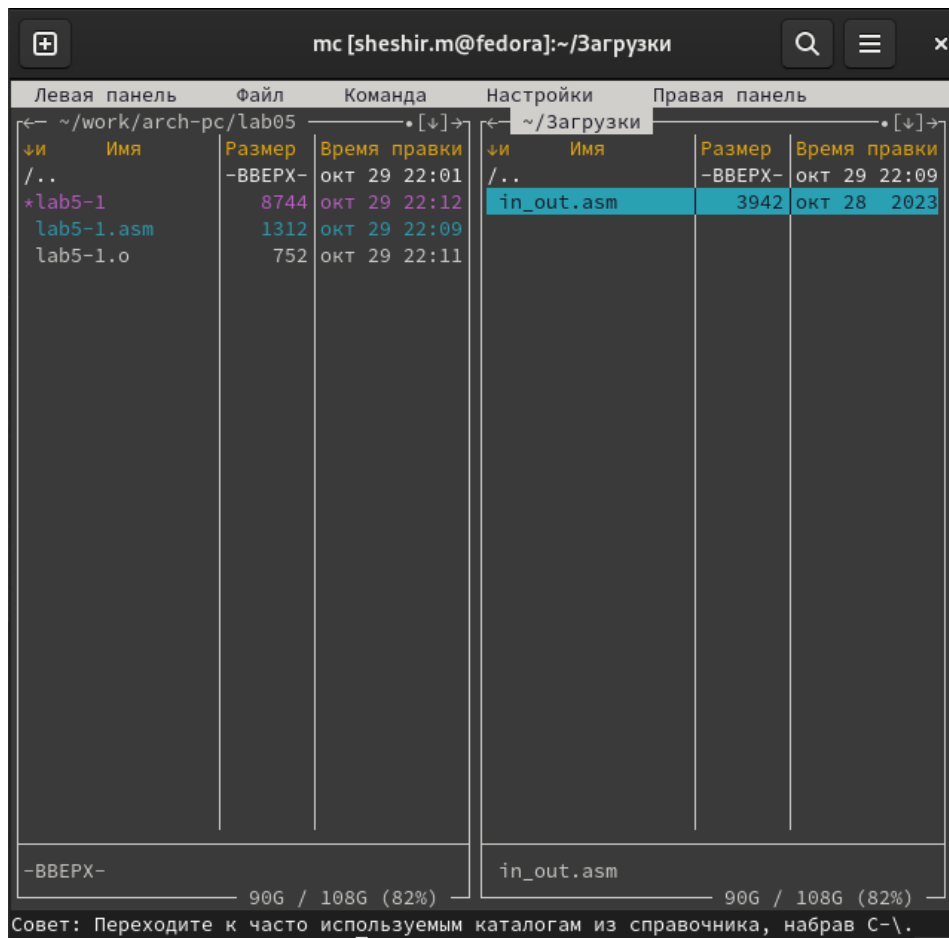


Рис. 3.7: Скачиваю файл `in_out.asm` со страницы курса в ТУИС. Он сохранился в каталог “Загрузки”

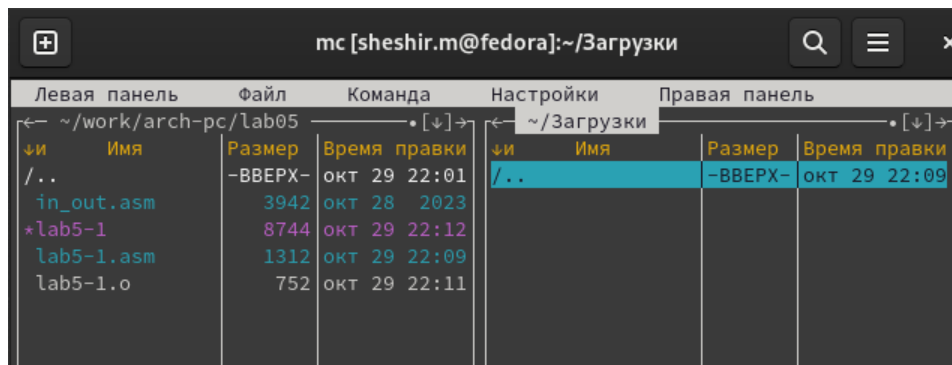


Рис. 3.8: С помощью функциональной клавиши F6 переместил файл in_out.asm из каталога Загрузки в созданный каталог lab05

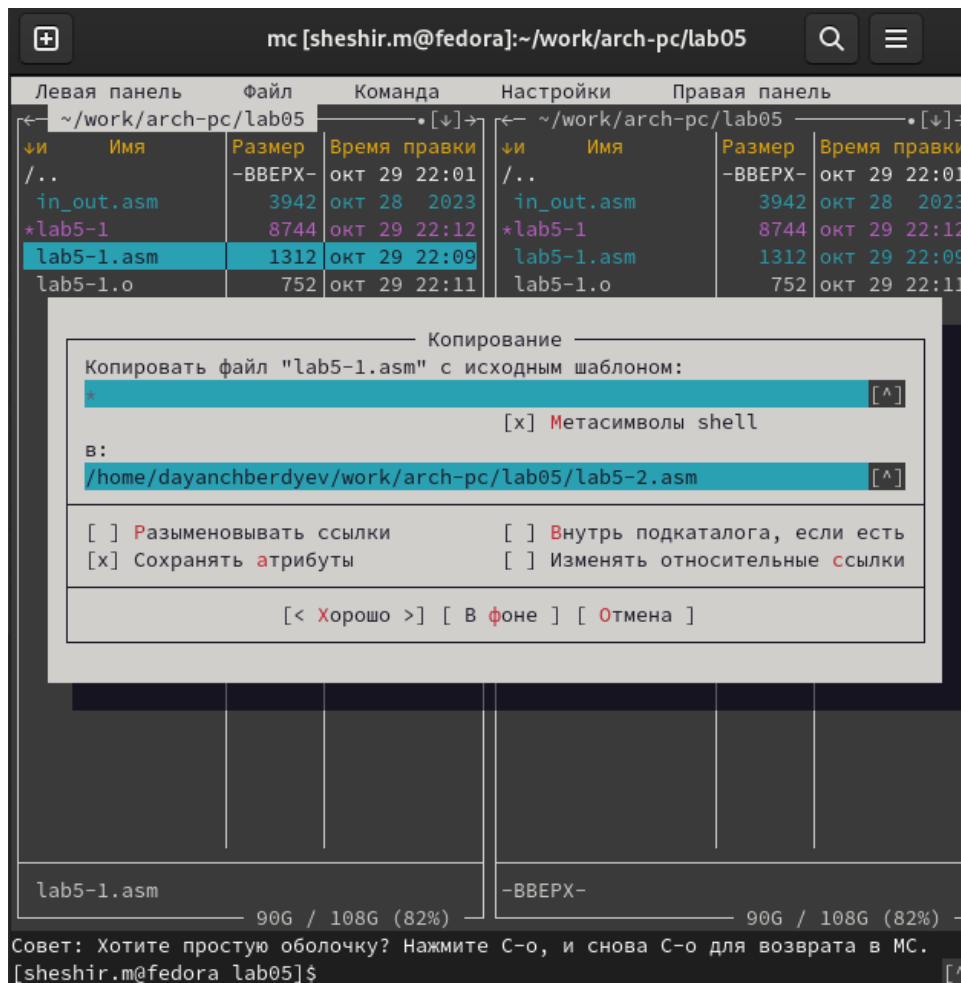


Рис. 3.9: С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя lab5-2.asm для копии файла

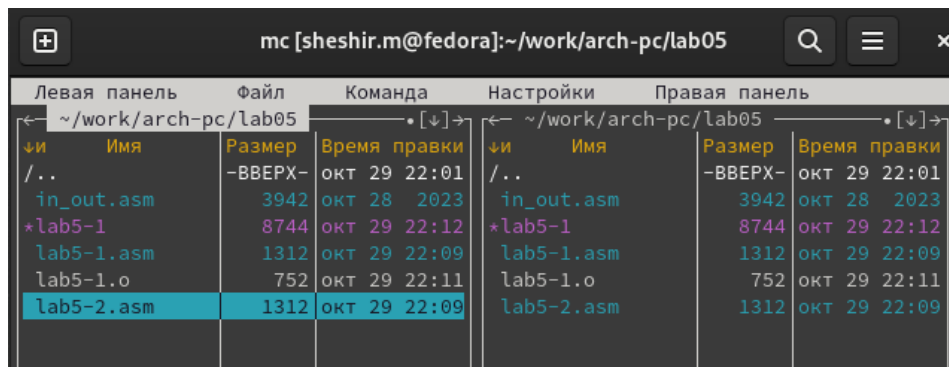


Рис. 3.10:

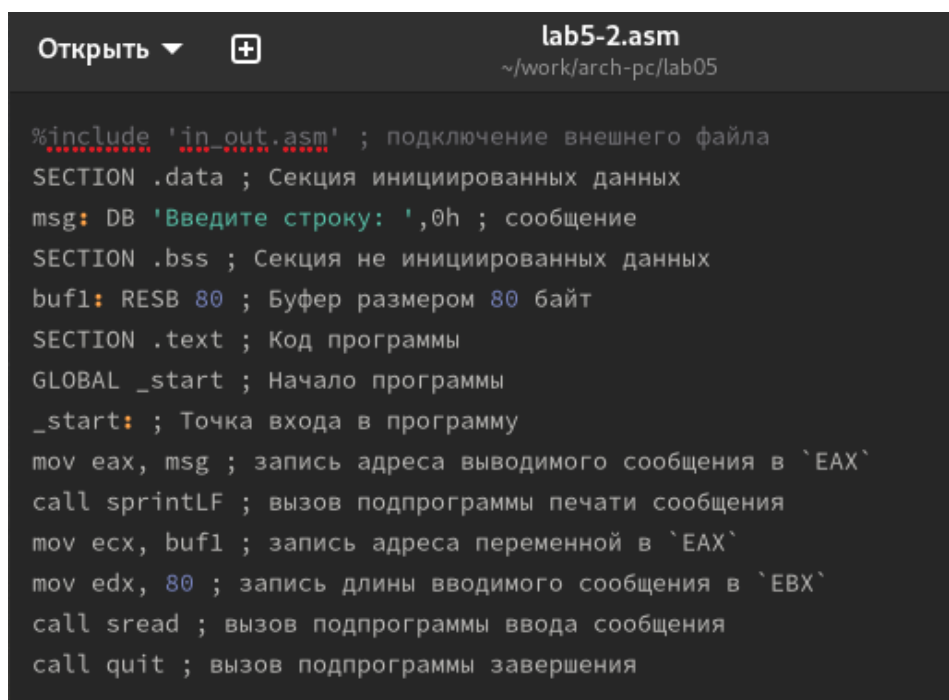
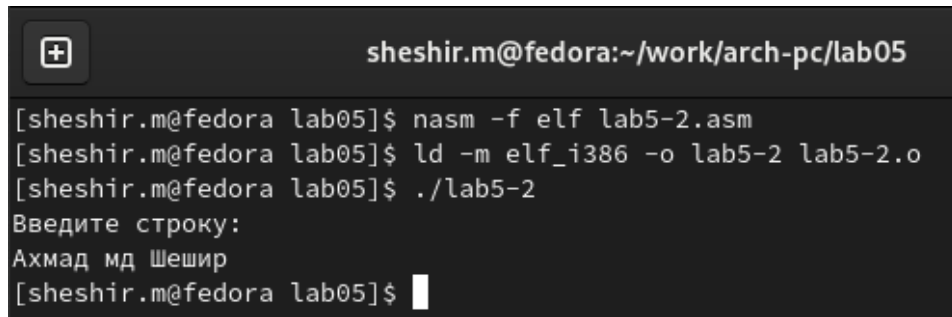
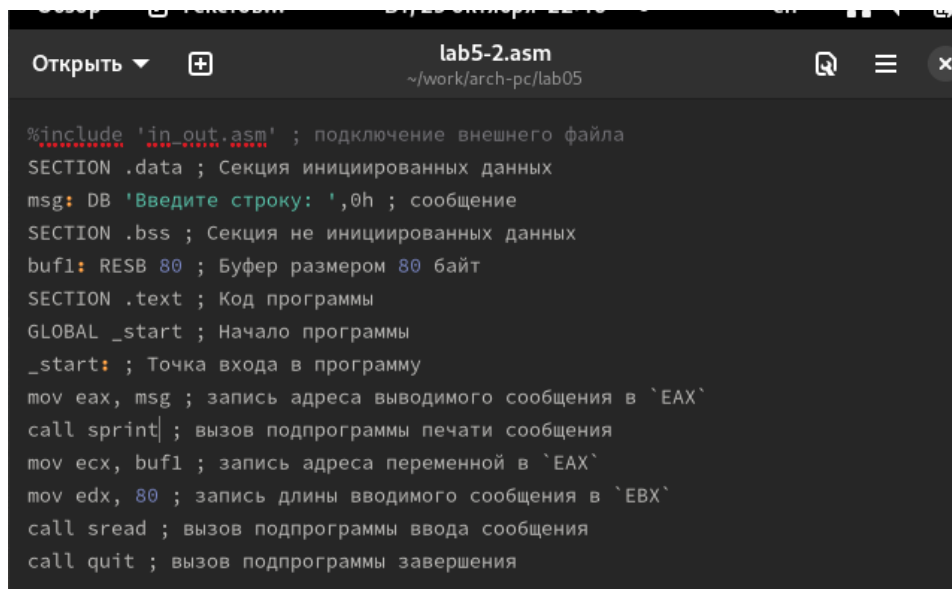


Рис. 3.11: Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in_out.asm.



```
sheshir.m@fedora:~/work/arch-pc/lab05
[sheshir.m@fedora lab05]$ nasm -f elf lab5-2.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-2 lab5-2.o
[sheshir.m@fedora lab05]$ ./lab5-2
Введите строку:
Ахмад мд Шешир
[sheshir.m@fedora lab05]$
```

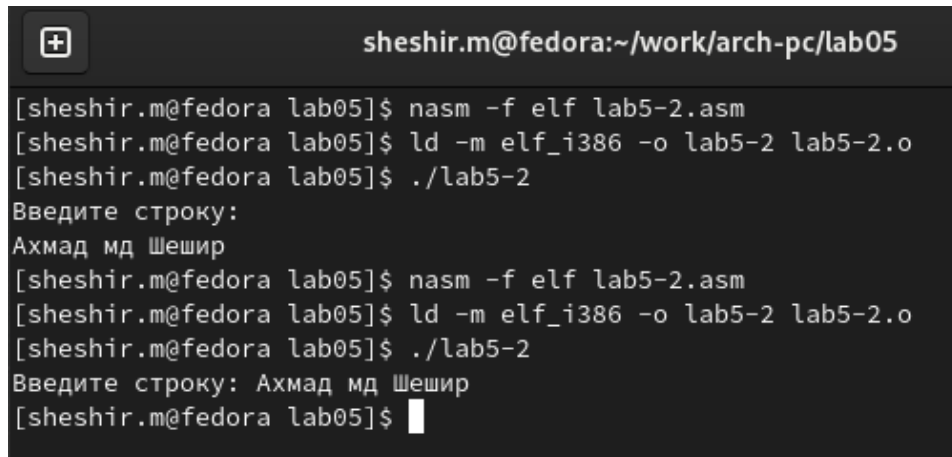
Рис. 3.12: Транслирую текст программы файла в объектный файл командой `nasm -f elf lab5-2.asm`. Создался объектный файл `lab5-2.o`. Выполняю компоновку объектного файла с помощью команды `ld -m elf_i386 -o lab5-2 lab5-2.o`. Создался исполняемый файл `lab5-2`. Запускаю исполняемый файл



```
lab5-2.asm
~/work/arch-pc/lab05

%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 ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 3.13: Открываю файл `lab5-2.asm` для редактирования в nano функциональной клавишей F4. Изменяю в нем подпрограмму `sprintLF` на `sprint`. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий



```
sheshir.m@fedora:~/work/arch-pc/lab05
[sheshir.m@fedora lab05]$ nasm -f elf lab5-2.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-2 lab5-2.o
[sheshir.m@fedora lab05]$ ./lab5-2
Введите строку:
Ахмад мд Шешир
[sheshir.m@fedora lab05]$ nasm -f elf lab5-2.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-2 lab5-2.o
[sheshir.m@fedora lab05]$ ./lab5-2
Введите строку: Ахмад мд Шешир
[sheshir.m@fedora lab05]$
```

Рис. 3.14: Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл

Разница между первым исполняемым файлом lab5-2 и вторым lab5-2 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`.

4 Выполнение заданий для самостоятельной работы

4.0.1 1

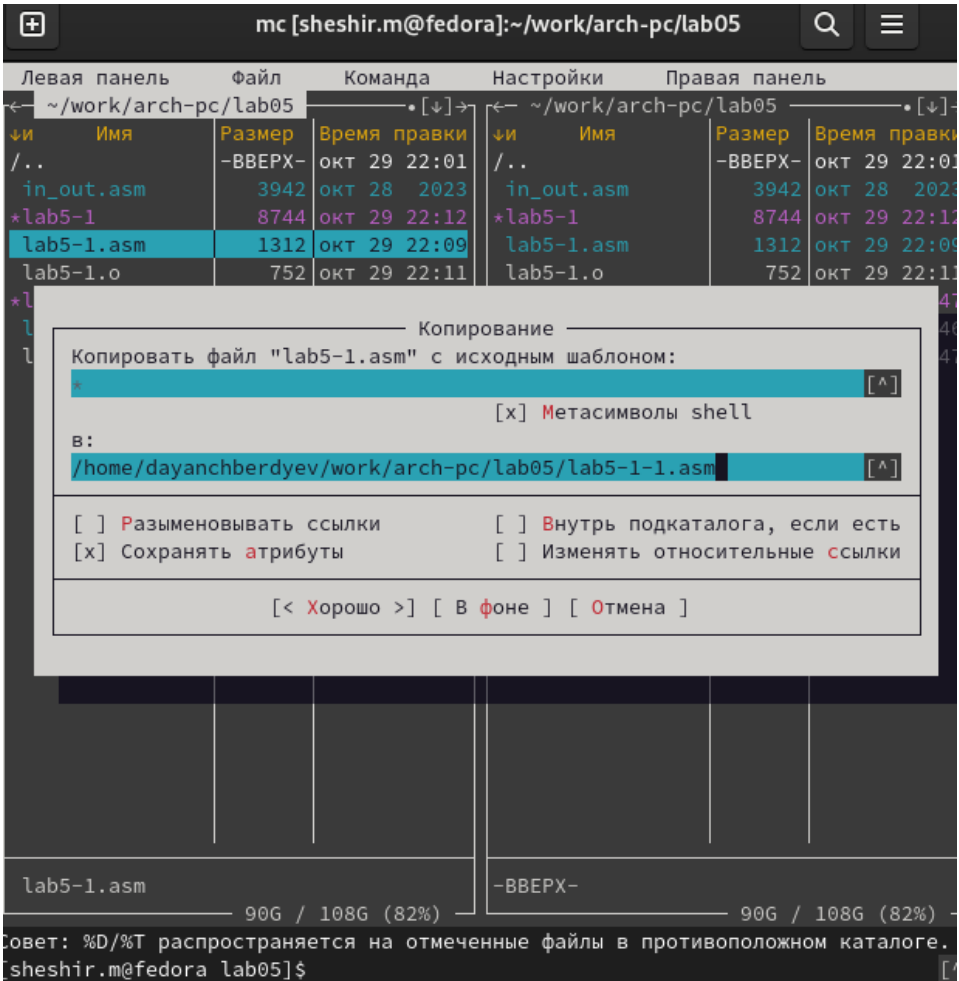


Рис. 4.1: Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F5

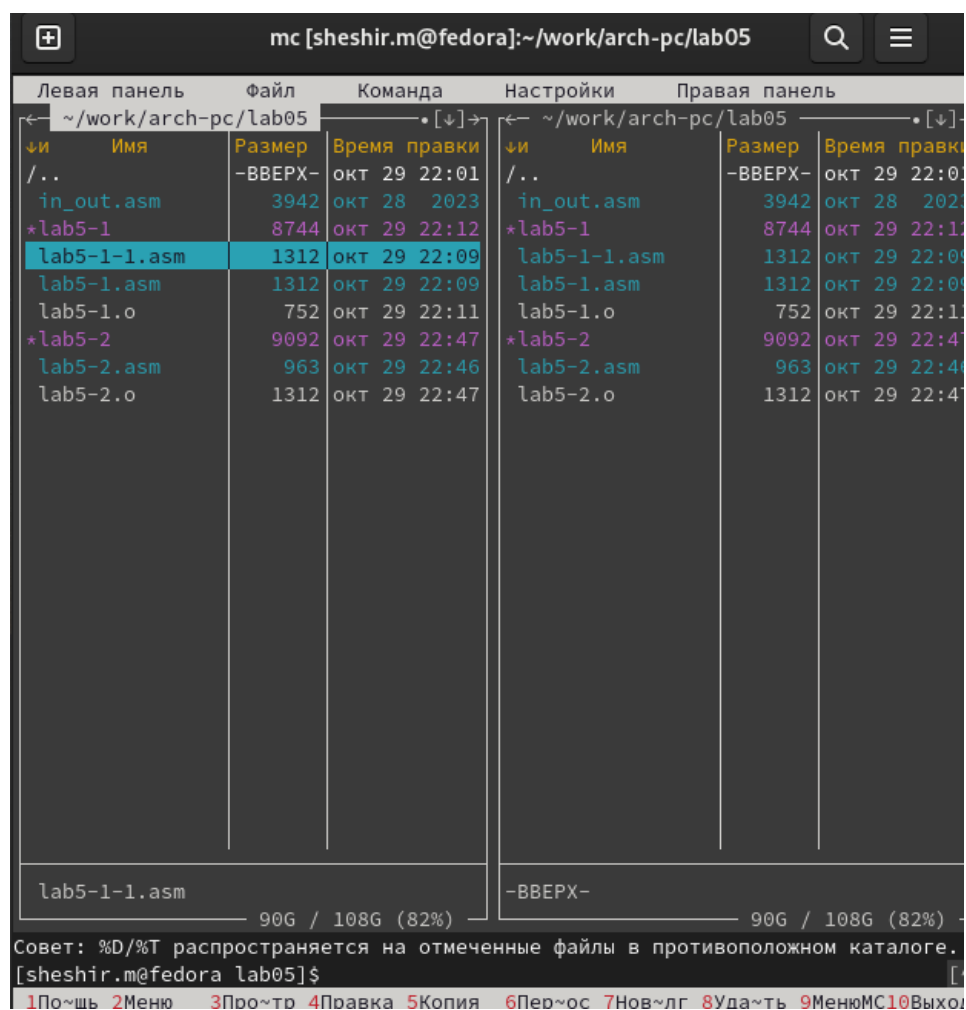
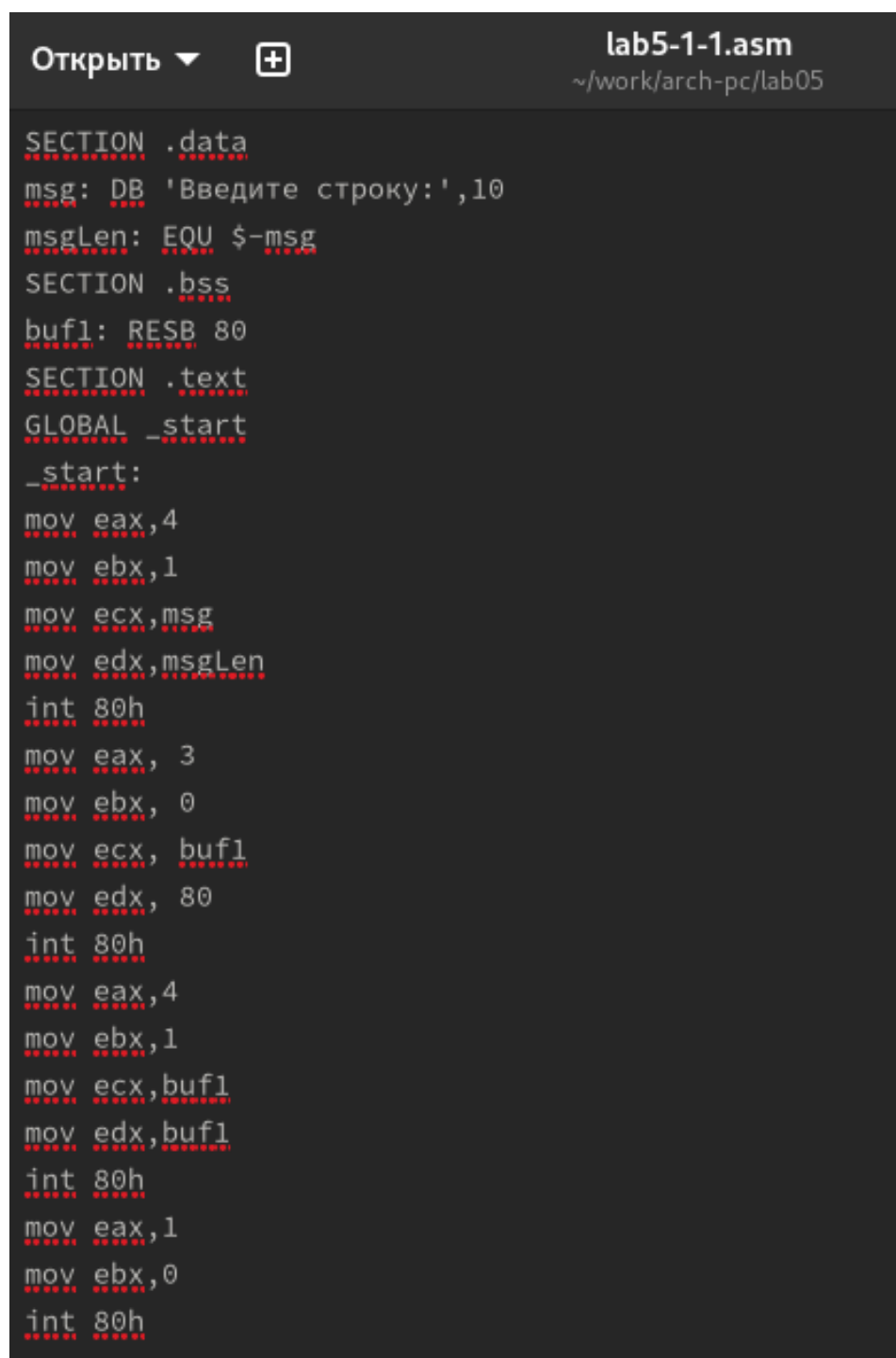



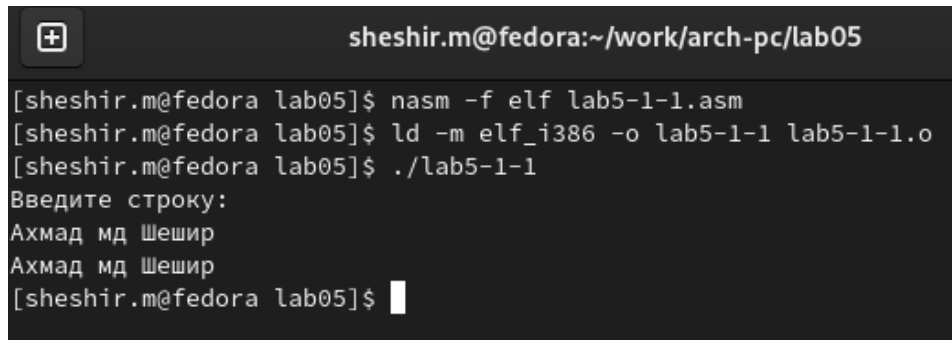
Рис. 4.2: Проверка



```
Открыть ▼  lab5-1-1.asm
~/work/arch-pc/lab05

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
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,buf1
int 80h
mov eax,1
mov ebx,0
int 80h
```

Рис. 4.3: 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



```
sheshir.m@fedora:~/work/arch-pc/lab05
[sheshir.m@fedora lab05]$ nasm -f elf lab5-1-1.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
[sheshir.m@fedora lab05]$ ./lab5-1-1
Введите строку:
Ахмад мд Шешир
Ахмад мд Шешир
[sheshir.m@fedora lab05]$
```

Рис. 4.4: 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 свои ФИО, далее программа выводит введенные мною данные

4.0.2 2

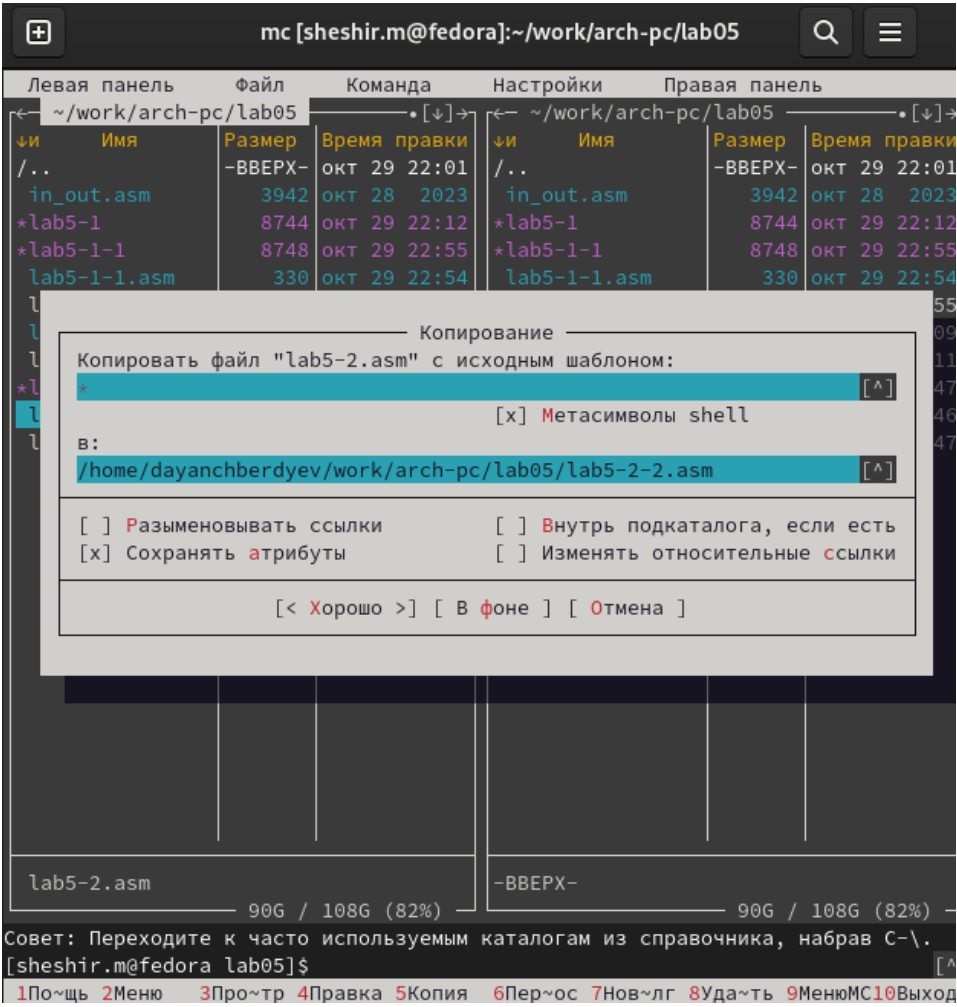
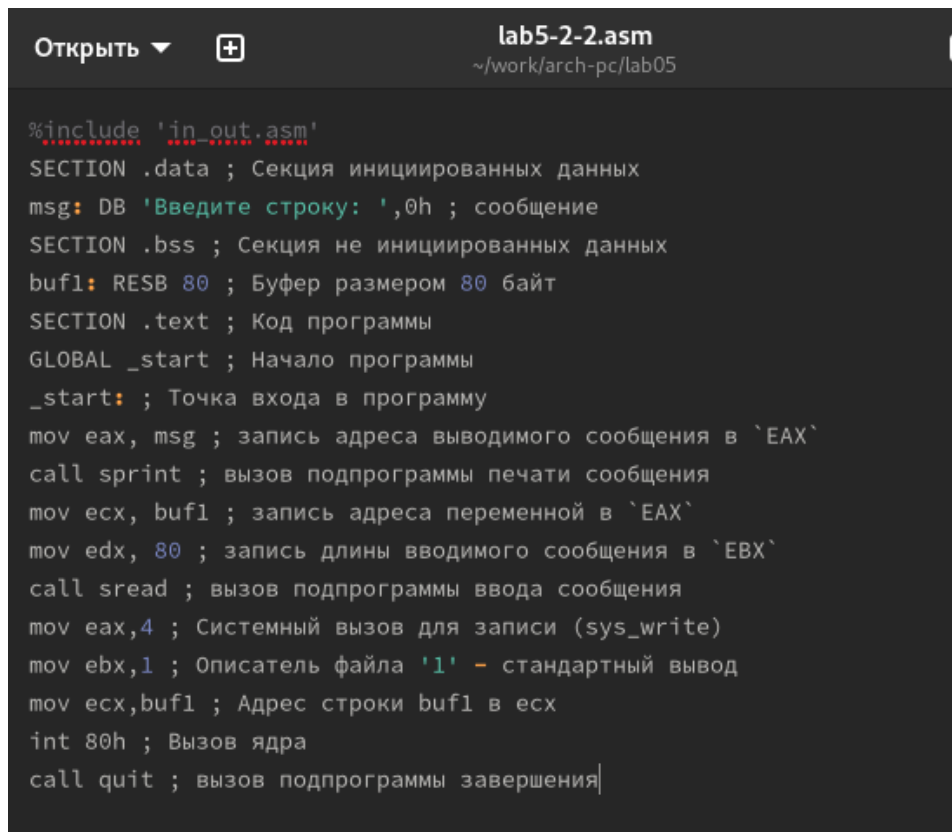


Рис. 4.5: Создаю копию файла lab5-2.asm с именем lab5-2-2.asm с помощью функциональной клавиши F5




```
Открыть ▾  lab5-2-2.asm  
~/work/arch-pc/lab05  
  
%include 'in_out.asm'  
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 ; вызов подпрограммы ввода сообщения  
mov eax, 4 ; Системный вызов для записи (sys_write)  
mov ebx, 1 ; Описатель файла '1' - стандартный вывод  
mov ecx, buf1 ; Адрес строки buf1 в ecx  
int 80h ; Вызов ядра  
call quit ; вызов подпрограммы завершения
```

Рис. 4.6: 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

```
[sheshir.m@fedora lab05]$ nasm -f elf lab5-2-2.asm
[sheshir.m@fedora lab05]$ ld -m elf_i386 -o lab5-2-2 lab5-2-2.o
[sheshir.m@fedora lab05]$ ./lab5-2-2
Введите строку: Ахмад мд Шешир
Ахмад мд Шешир
[sheshir.m@fedora lab05]$
```

Рис. 4.7: Создаю объектный файл lab5-2-2.o, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2-2, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные

5 Выводы

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

6 Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс,
- 11.
12. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
13. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
14. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
15. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-

- е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
16. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
17. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).