

Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Царёв Максим Александрович

Содержание

1	Цель работы.....	1
2	Теоретическое введение.....	1
3	Задание	2
4	Выводы.....	11

1 Цель работы

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

2 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция иницированных (известных во время компиляции) данных (SECTION .data) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (SECTION .bss). Для объявления иницированных данных в секции .data используются директивы DB, DW, DD, DQ и DT, которые резервируют память и указывают, какие значения должны храниться в этой памяти: - DB (define byte) — определяет переменную размером в 1 байт; - DW (define word) — определяет переменную размером в 2 байта (слово); - DD (define double word) — определяет переменную размером в 4 байта (двойное слово); - DQ (define quad word) — определяет переменную размером в 8 байт (учетверённое слово); - DT (define ten bytes) — определяет переменную размером в 10 байт. Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву DB в связи с особенностями хранения данных в оперативной памяти. Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике.

```
mov dst,src
```

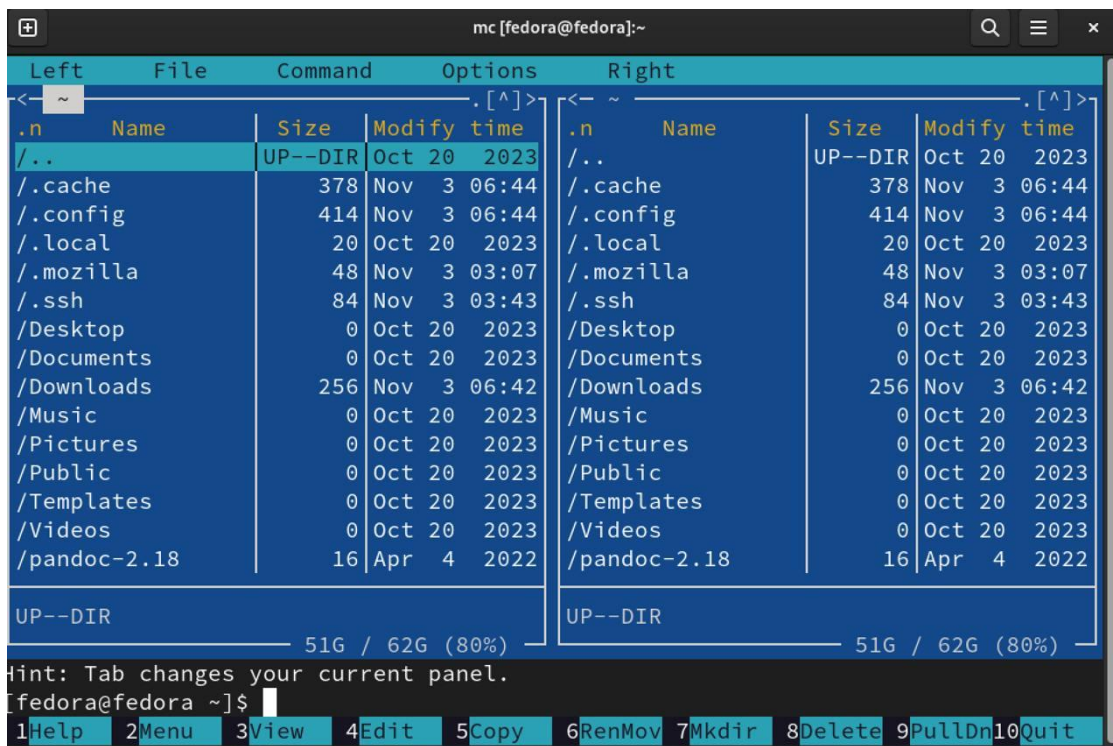
Здесь операнд `dst` — приёмник, а `src` — источник. В качестве операнда могут выступать регистры (`register`), ячейки памяти (`memory`) и непосредственные значения (`const`). Инструкция языка ассемблера `intp` предназначена для вызова прерывания с указанным номером.

```
int n
```

Здесь `n` — номер прерывания, принадлежащий диапазону 0–255. При программировании в Linux с использованием вызовов ядра `sys_calls` `n=80h` (принято задавать в шестнадцатеричной системе счисления).

3 Задание

Открываю Midnight Commander, введя в терминал `mc`



Перехожу в каталог `~/work/study/2023-2024/Архитектура Компьютера/arch-pc`

```
mc [fedora@fedora]:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/labs
```

Left	File	Command	Options	Right			
<- ...dy_2024-2025_ar-h-pc/labs	..[^]>	<- ~	..[^]>				
.n	Name	Size	Modify time	.n	Name	Size	Modify time
/..		UP--DIR	Nov 3 03:43	/..		UP--DIR	Oct 20 2023
/lab01		36	Nov 3 03:43	/.cache		378	Nov 3 06:44
/lab02		36	Nov 3 03:43	/.config		414	Nov 3 06:44
/lab03		36	Nov 3 03:43	/.local		20	Oct 20 2023
/lab04		138	Nov 3 03:43	/.mozilla		48	Nov 3 03:07
/lab05		36	Nov 3 03:43	/.ssh		84	Nov 3 03:43
/lab06		36	Nov 3 03:43	/Desktop		0	Oct 20 2023
/lab07		36	Nov 3 03:43	/Documents		0	Oct 20 2023
/lab08		36	Nov 3 03:43	/Downloads		256	Nov 3 06:42
/lab09		36	Nov 3 03:43	/Music		0	Oct 20 2023
/lab10		36	Nov 3 03:43	/Pictures		0	Oct 20 2023
/lab11		36	Nov 3 03:43	/Public		0	Oct 20 2023
README.md		19	Nov 3 03:43	/Templates		0	Oct 20 2023
README.ru.md		40	Nov 3 03:43	/Videos		0	Oct 20 2023
				/pandoc-2.18		16	Apr 4 2022
/lab04				UP--DIR			
50G / 62G (80%)				51G / 62G (80%)			

Hint: Use C-x t to copy tagged file names to the command line.
[fedora@fedora labs]\$

1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit

С помощью клавиши F7 создаю каталог lab05

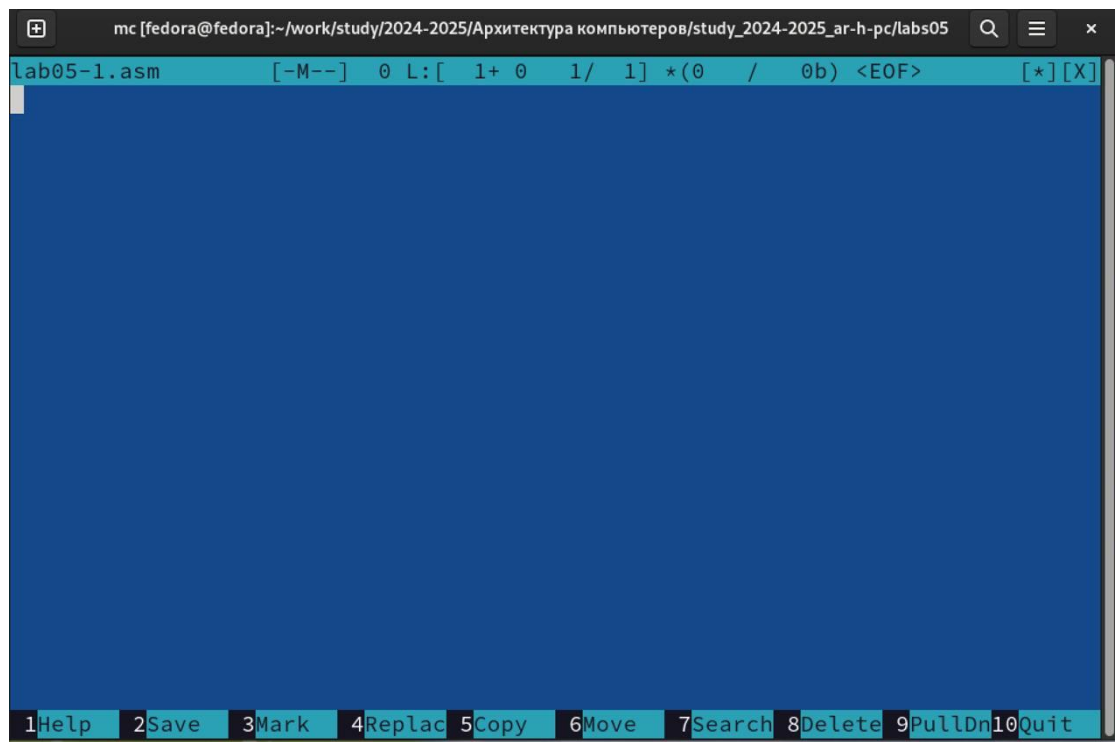
```
mc [fedora@fedora]:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/labs05
```

Left	File	Command	Options	Right			
<- ..._2024-2025_ar-h-pc/labs05	..[^]>	<- ..._2024-2025_ar-h-pc/labs05	..[^]>				
.n	Name	Size	Modify time	.n	Name	Size	Modify time
/..		UP--DIR	Nov 3 06:50	/..		UP--DIR	Nov 3 06:50
lab05-1.asm		0	Nov 3 06:54	lab05-1.asm		0	Nov 3 06:54
UP--DIR				UP--DIR			
48G / 62G (77%)				48G / 62G (77%)			

Hint: Tab changes your current panel.
[fedora@fedora labs05]\$

1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit

Перехожу в созданный каталог, в строке ввода прописываю команду touch lab05-1.asm, чтобы создать файл



4.2 Структура программы на языке ассемблера NASM Превращаю текст программы для вывода "Hello world!" в объектный код с помощью транслятора NASM, используя команду `nasm -f elf hello.asm`, ключ `-f` указывает транслятору `nasm`, что требуется создать бинарный файл в формате ELF (рис. 4.5). Далее проверяю правильность выполнения команды с помощью утилиты `ls`: действительно, создан файл "hello.o"

4.3 Работа с расширенным синтаксисом командной строки NASM С помощью функциональной клавиши F4 открываю файл `lab5-1.asm` для редактирования во встроенном редакторе. Ввожу в файл код программы для запроса строки у пользователя

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/labs05
...5/Архитектура компьютеров/study_2024-2025_ar-h-pc/labs05/lab05-1.asm Modified
msgLen: EQU $-msg ; символ перевода строки
; 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
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

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

выхожу из файла (Ctrl+X), сохраняя изменения (Y, Enter). Открыл файл и убедился, что файл содержит 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`.

```
mc [fedora@fedora]:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/labs05
Left      File      Command      Options      Right
<- ..._2024-2025_ar-h-pc/labs05 -.[^]> <- ..._2024-2025_ar-h-pc/labs05 -.[^]>
.n      Name      Size      Modify time      .n      Name      Size      Modify time
/..      UP--DIR      Nov 3 06:50      /..      UP--DIR      Nov 3 06:50
lab05-1.asm      1314      Nov 3 07:09      lab05-1.asm      1314      Nov 3 07:09
lab05-1.o      752      Nov 3 07:10      lab05-1.o      752      Nov 3 07:10

lab05-1.asm      48G / 62G (77%)      UP--DIR      48G / 62G (77%)

Hint: Tab changes your current panel.
[fedora@fedora labs05]$ $$
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit
```



```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-  
[fedora@fedora lab05]$ nasm -f elf lab5-1.asm  
[fedora@fedora lab05]$ ls  
lab5-1.asm  lab5-1.o  
[fedora@fedora lab05]$ mold -m elf_i386 -o lab5-1 lab5-1.o  
[fedora@fedora lab05]$ ls  
lab5-1  lab5-1.asm  lab5-1.o  
[fedora@fedora lab05]$ ./lab5-1  
Введите строку:  
Царёв Максим Александрович  
[fedora@fedora lab05]$
```

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

##Подключение внешнего файла Скачиваю файл in_out.asm со страницы курса в ТУИС.

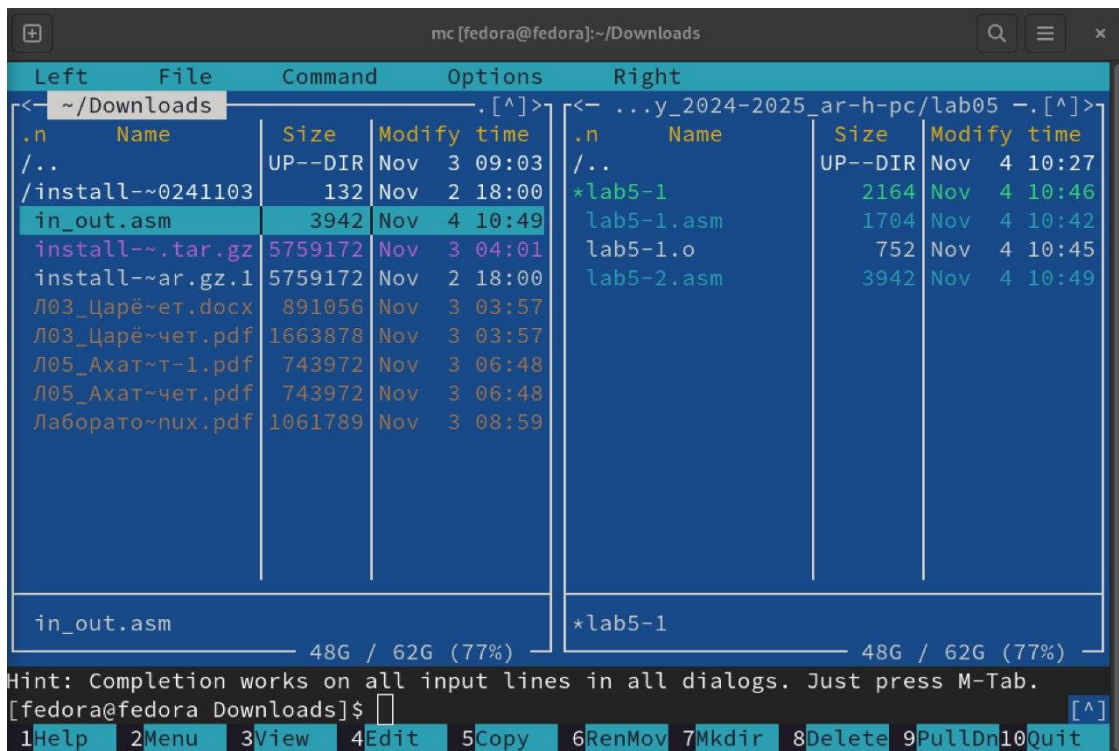
mc [fedora@fedora]:~/Downloads

Left	File	Command	Options	Right			
<- ~/Downloads	.[^]>			<- ../B/study_2024-2025_ar-h-pc			
.n	Name	Size	Modify time	.n	Name	Size	Modify time
/..		UP--DIR	Nov 3 09:03	/..		UP--DIR	Nov 3 03:43
/install--0241103		132	Nov 2 18:00	/.git		138	Nov 3 03:43
in_out.asm		3942	Nov 4 10:49	/config		24	Nov 3 03:43
install--~.tar.gz		5759172	Nov 3 04:01	/lab05		48	Nov 4 10:46
install--ar.gz.1		5759172	Nov 2 18:00	/labs		152	Nov 3 03:43
л03_Царёв-ет.docx		891056	Nov 3 03:57	/presentation		78	Nov 3 03:43
л03_Царёв-чет.pdf		1663878	Nov 3 03:57	/template		36	Nov 3 03:43
л05_Ахат-т-1.pdf		743972	Nov 3 06:48	.gitattributes		1765	Nov 3 03:43
л05_Ахат-чет.pdf		743972	Nov 3 06:48	.gitignore		4637	Nov 3 03:43
Лаборато-nux.pdf		1061789	Nov 3 08:59	.gitmodules		278	Nov 3 03:43
				CHANGELOG.md		4786	Nov 3 03:43
				COURSE		8	Nov 3 03:43
				LICENSE		18657	Nov 3 03:43
				Makefile		980	Nov 3 03:43
				README.en.md		152	Nov 3 03:43
in_out.asm				UP--DIR			
48G / 62G (77%)				48G / 62G (77%)			

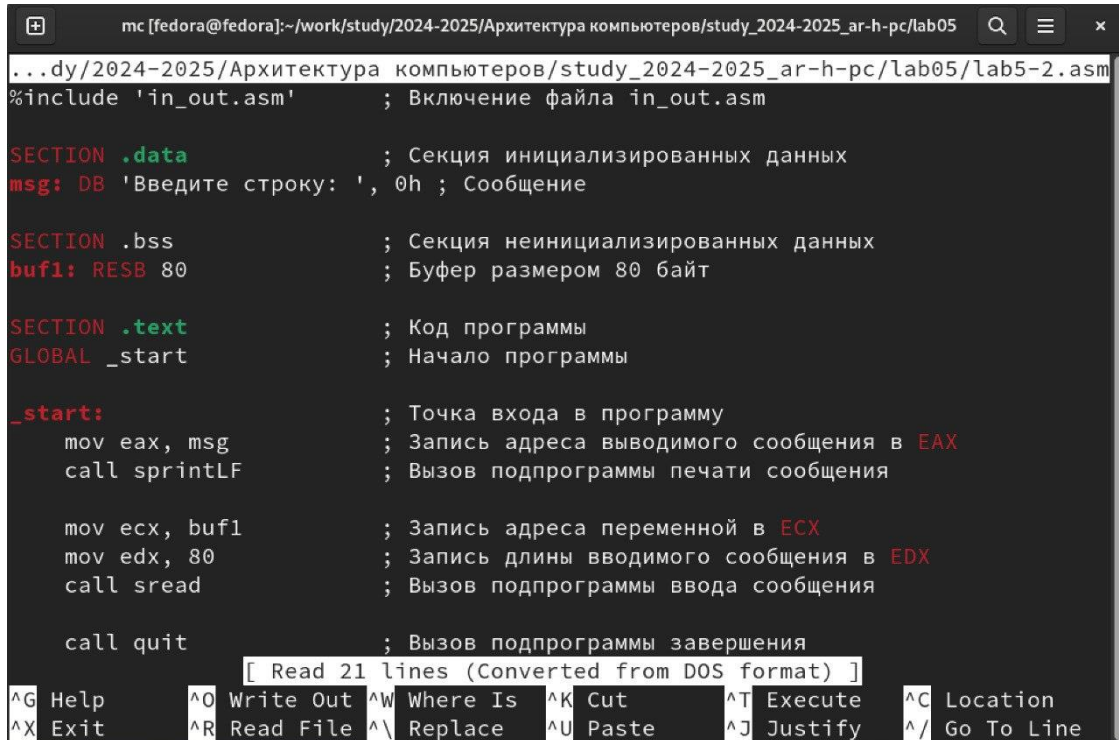
Hint: %D/%T expands to the tagged files in the opposite directory.
[fedora@fedora Downloads]\$

1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn 10Quit

Копирую файл in_out.asm из каталога Загрузки в созданный каталог lab05 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 для копии файла



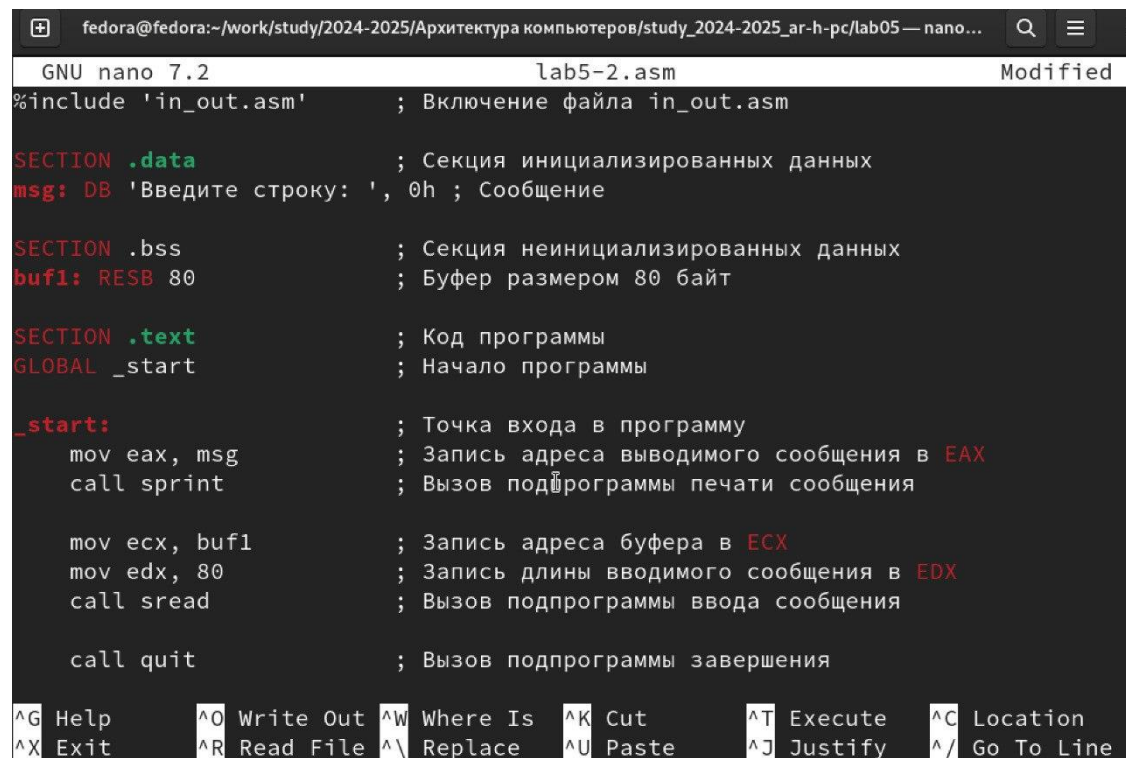
Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano, чтобы в программе использовались подпрограммы из внешнего файла in_out.asm.



Открываю файл lab5-2.asm для редактирования в nano. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий

```
[fedora@fedora lab05]$ nasm -f elf lab5-2.asm
[fedora@fedora lab05]$ mold -m elf_i386 -o lab5-2 lab5-2.o
[fedora@fedora lab05]$ ls
in_out.asm lab5-1 lab5-1.asm lab5-1.o lab5-2 lab5-2.asm lab5-2.o
[fedora@fedora lab05]$ ./lab5-2
Введите строку:
Царёв Максим Александрович
[fedora@fedora lab05]$
```

Открываю файл lab5-2.asm для редактирования в nano. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий



```
GNU nano 7.2 lab5-2.asm Modified
%include 'in_out.asm' ; Включение файла in_out.asm

SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ', 0h ; Сообщение

SECTION .bss ; Секция неинициализированных 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 буфера в ECX
    mov edx, 80 ; Запись длины вводимого сообщения в EDX
    call sread ; Вызов подпрограммы ввода сообщения

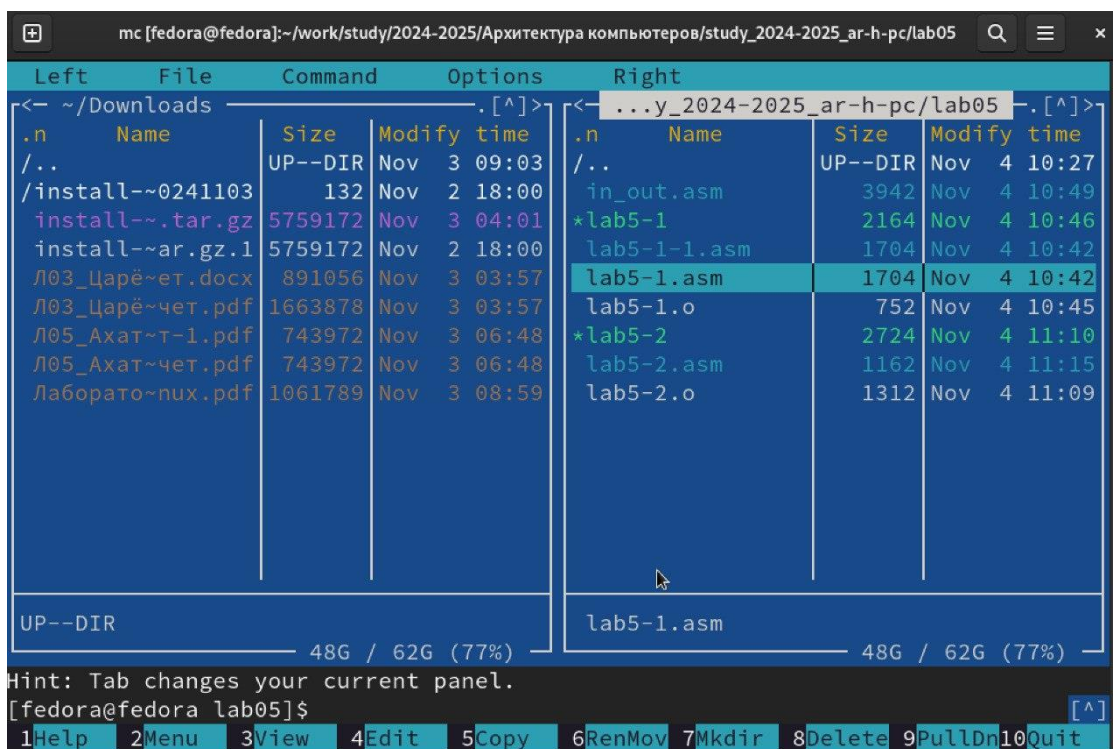
    call quit ; Вызов подпрограммы завершения

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line
```

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл


```
[fedora@fedora lab05]$ nano lab5-2.asm
[fedora@fedora lab05]$ ./lab5-2
Введите строку:
Царёв Максим Александрович
[fedora@fedora lab05]$
```

Разница между первым исполняемым файлом и вторым 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 `sprintf` и `sprint`. #Выполнение заданий для самостоятельной работы Создаю копию файла `lab5-1.asm` с именем `lab5-1-1.asm`. открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку.



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

```

[fedora@fedora lab05]$ nano lab5-1-1.asm
[fedora@fedora lab05]$ nasm -f elf lab5-1-1.asm
[fedora@fedora lab05]$ mold -m elf_i386 -o lab5-1-1 lab5-1-1.o
[fedora@fedora lab05]$ ls
in_out.asm  lab5-1-1      lab5-1-1.o  lab5-1.o  lab5-2.asm
lab5-1      lab5-1-1.asm  lab5-1.asm  lab5-2    lab5-2.o
[fedora@fedora lab05]$ ./lab5-1-1
Введите строку:
Царёв Максим Александрович
Царёв Максим Александрович
[fedora@fedora lab05]$

```

Создаю копию файла lab5-2.asm с именем lab5-2-1.asm

Left	File	Command	Options	Right
<- ~/Downloads				<- ...y_2024-2025_ar-h-pc/lab05
.n	Name	Size	Modify time	.n
UP--DIR	UP--DIR	UP--DIR	UP--DIR	UP--DIR
Nov 3 09:03	Nov 4 10:27	Nov 4 10:49	Nov 4 10:46	Nov 4 11:21
Nov 2 18:00	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 04:01	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 2 18:00	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 03:57	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 03:57	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 06:48	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 06:48	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
Nov 3 08:59	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20	Nov 4 11:20
48G / 62G (77%)	48G / 62G (77%)	48G / 62G (77%)	48G / 62G (77%)	48G / 62G (77%)

Hint: Completion: use M-Tab (or Esc+Tab). Type it twice to get a list.

[fedora@fedora lab05]\$

1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit

Открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку

```
[fedora@fedora lab05]$ nasm -f elf lab5-2-1.asm
[fedora@fedora lab05]$ mold -m elf_i386 -o lab5-2-1 lab5-2-1.o
[fedora@fedora lab05]$ ls
in_out.asm  lab5-1-1.asm  lab5-1.o  lab5-2-1.asm  lab5-2.o
lab5-1      lab5-1-1.o   lab5-2    lab5-2-1.o
lab5-1-1    lab5-1.asm   lab5-2-1  lab5-2.asm
[fedora@fedora lab05]$ ./lab5-2-1
Введите строку:Царёв Максим Александрович
Царёв Максим Александрович
[fedora@fedora lab05]$
```

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 свои ФИО, далее программа выводит введенные мною данные

4 Выводы

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