

# **Отчёт по лабораторной работе №5**

**Дисциплина: Архитектура компьютера**

Буриева Шахзода Акмаловна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>4</b>	<b>Выполнение заданий для самостоятельной работы</b>	<b>15</b>
<b>5</b>	<b>Выводы</b>	<b>19</b>
	<b>Список литературы</b>	<b>20</b>

## Список иллюстраций

3.1	Открытие Midnight Commander . . . . .	8
3.2	Переход в каталог ~/work/arch-pc . . . . .	8
3.3	Создание папки lab05 . . . . .	9
3.4	Переход в каталог lab05 . . . . .	9
3.5	Создание файла lab05-1.asm . . . . .	9
3.6	Открытие файла lab05-1.asm в редакторе nano . . . . .	9
3.7	Ввод текста программы . . . . .	10
3.8	Просмотр программы . . . . .	10
3.9	Выполнение компоновки объектного файла lab05-1.asm и его запуск	11
3.10	Скачивание файла in_out.asm . . . . .	11
3.11	Копирование скачанного файла в каталог lab05 . . . . .	11
3.12	Создание копии файла lab05-1.asm с именем lab05-2.asm . . . . .	12
3.13	Исправление текста программы в файле lab05-2.asm и проверка .	12
3.14	Выполнение компоновки объектного файла lab05-2.asm и его запуск	13
3.15	Замена подпрограммы sprintLF . . . . .	13
3.16	Выполнение компоновки объектного файла lab05-2.asm и его запуск	14
4.1	Создание копии файла lab05-1.asm с именем lab05-3.asm . . . . .	15
4.2	Изменения в программе . . . . .	16
4.3	Выполнение компоновки объектного файла lab05-3.asm и его запуск	16
4.4	Ввод фамилии . . . . .	16
4.5	Создание копии файла lab05-1.asm с именем lab05-3-1.asm . . . . .	17
4.6	Выполнение компоновки объектного файла lab05-3-1.asm и его запуск . . . . .	17
4.7	Отправка на github . . . . .	18

## Список таблиц

# 1 Цель работы

Приобрести практические навыки для работы в Midnight Commander. Освоить инструкции 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. Для активации оболочки Midnight Commander достаточно ввести в командной строке mc и нажать клавишу Enter. В Midnight Commander используются функциональные клавиши F1 — F10, к которым привязаны часто выполняемые операции.

Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция инициированных (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).

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.

Простейший диалог с пользователем требует наличия двух функций — вывода текста на экран и ввода текста с клавиатуры. Простейший способ вывести строку на экран — использовать системный вызов write. Этот системный вызов имеет номер 4, поэтому перед вызовом инструкции int необходимо поместить значение 4 в регистр eax. Первым аргументом write, помещаемым в регистр ebx, задаётся дескриптор файла. Для вывода на экран в качестве дескриптора файла нужно указать 1 (это означает «стандартный вывод», т. е. вывод на экран).

Вторым аргументом задаётся адрес выводимой строки (помещаем его в регистр `ecx`, например, инструкцией `mov ecx, msg`). Строка может иметь любую длину. Последним аргументом (т.е. в регистре `edx`) должна задаваться максимальная длина выводимой строки. Для ввода строки с клавиатуры можно использовать аналогичный системный вызов `read`. Его аргументы – такие же, как у вызова `write`, только для «чтения» с клавиатуры используется файловый дескриптор 0 (стандартный ввод). Системный вызов `exit` является обязательным в конце любой программы на языке ассемблера. Для обозначения конца программы перед вызовом инструкции `int 80h` необходимо поместить в регистр `eax` значение 1, а в регистр `ebx` код завершения 0.





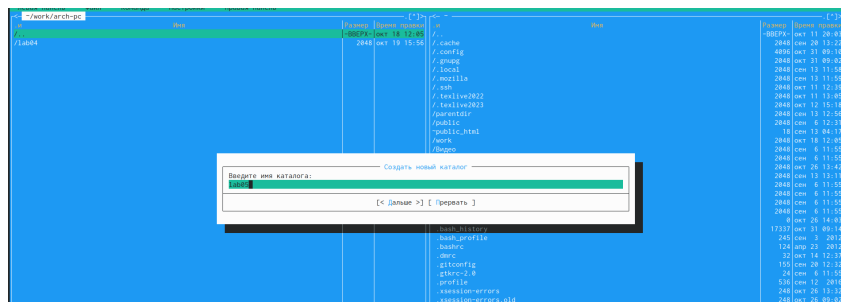


Рис. 3.3: Создание папки lab05

Я перешла в созданный каталог.

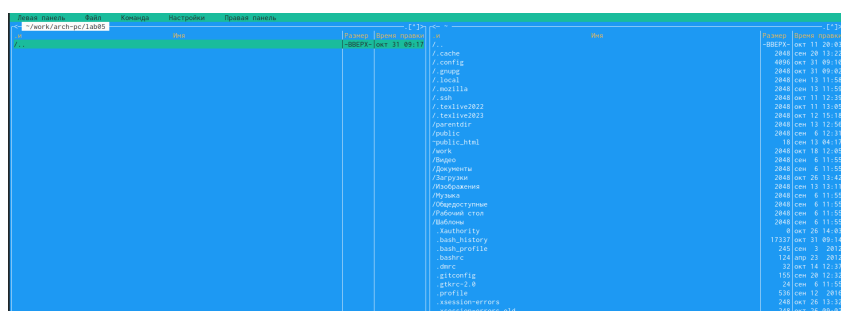


Рис. 3.4: Переход в каталог lab05

Пользуясь строкой ввода и командой touch создала файл lab5-1.asm.

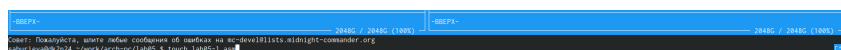


Рис. 3.5: Создание файла lab05-1.asm

С помощью функциональной клавиши F4 открыла файл lab05-1.asm для редактирования во встроенном редакторе nano.

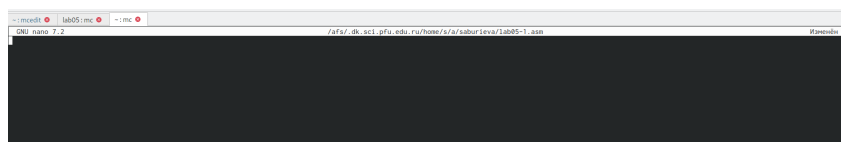
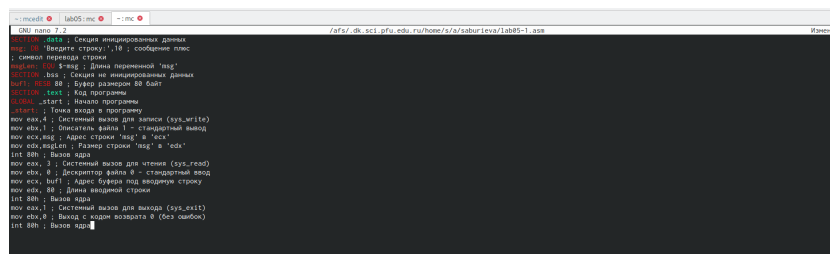


Рис. 3.6: Открытие файла lab05-1.asm в редакторе nano

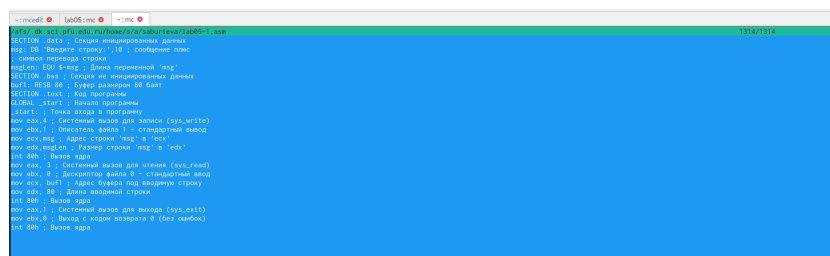
Ввела текст программы из листинга, сохранила изменения и закрыла файл.



```
--mcs51 lab05-1.asm
;lab05-1.asm : Секция инициализации данных
;lab05-1.asm : Введите строку: ,10 : сообщение prompt
;lab05-1.asm : символ перевода строки
;lab05-1.asm : E00: E-рег : Данные переменных 'reg'
;lab05-1.asm : E01: E-рег : Секция не инициализированных данных
;lab05-1.asm : E02: E-рег : Буфер размером 80 байт
;lab05-1.asm : E03: E-рег : Код программы
;lab05-1.asm : E04: E-рег : Начало программы
;lab05-1.asm : E05: E-рег : Точка входа в программу
;lab05-1.asm : E06: E-рег : Системный вызов для вывода (sys_write)
;lab05-1.asm : E07: E-рег : Описание файла 1 - стандартный вывод
;lab05-1.asm : E08: E-рег : Адрес строки 'reg' в 'e00'
;lab05-1.asm : E09: E-рег : Размер строки 'reg' в 'e00'
;lab05-1.asm : E10: E-рег : Выход из программы
;lab05-1.asm : E11: E-рег : Системный вызов для чтения (sys_read)
;lab05-1.asm : E12: E-рег : Описание файла 0 - стандартный ввод
;lab05-1.asm : E13: E-рег : Адрес буфера под входную строку
;lab05-1.asm : E14: E-рег : Длина входной строки
;lab05-1.asm : E15: E-рег : Выход из программы
;lab05-1.asm : E16: E-рег : Системный вызов для вывода (sys_exit)
;lab05-1.asm : E17: E-рег : Выход с кодом возврата 0 (без ошибок)
;lab05-1.asm : E18: E-рег : Выход из программы
```

Рис. 3.7: Ввод текста программы

С помощью функциональной клавиши F3 открыла файл lab05-1.asm для просмотра. Убедилась, что файл содержит текст программы.



```
--mcs51 lab05-1.asm
;lab05-1.asm : Секция инициализации данных
;lab05-1.asm : Введите строку: ,10 : сообщение prompt
;lab05-1.asm : символ перевода строки
;lab05-1.asm : E00: E-рег : Данные переменных 'reg'
;lab05-1.asm : E01: E-рег : Секция не инициализированных данных
;lab05-1.asm : E02: E-рег : Буфер размером 80 байт
;lab05-1.asm : E03: E-рег : Код программы
;lab05-1.asm : E04: E-рег : Начало программы
;lab05-1.asm : E05: E-рег : Точка входа в программу
;lab05-1.asm : E06: E-рег : Системный вызов для вывода (sys_write)
;lab05-1.asm : E07: E-рег : Описание файла 1 - стандартный вывод
;lab05-1.asm : E08: E-рег : Адрес строки 'reg' в 'e00'
;lab05-1.asm : E09: E-рег : Размер строки 'reg' в 'e00'
;lab05-1.asm : E10: E-рег : Выход из программы
;lab05-1.asm : E11: E-рег : Системный вызов для чтения (sys_read)
;lab05-1.asm : E12: E-рег : Описание файла 0 - стандартный ввод
;lab05-1.asm : E13: E-рег : Адрес буфера под входную строку
;lab05-1.asm : E14: E-рег : Длина входной строки
;lab05-1.asm : E15: E-рег : Выход из программы
;lab05-1.asm : E16: E-рег : Системный вызов для вывода (sys_exit)
;lab05-1.asm : E17: E-рег : Выход с кодом возврата 0 (без ошибок)
;lab05-1.asm : E18: E-рег : Выход из программы
```

Рис. 3.8: Просмотр программы

Оттранслировала текст программы lab05-1.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл. Программа вывела строку 'Введите строку:' и ожидала ввода с клавиатуры. На запрос "Введите строку" я ввела свои ФИО.

```

saburievadk8n81 ~/work/arch-pc/lab05 $ nasm -f elf lab05-1.asm
saburievadk8n81 ~/work/arch-pc/lab05 $ d -m elf_i386 -o lab5-1 lab5-1.o
bash: d: команда не найдена
saburievadk8n81 ~/work/arch-pc/lab05 $ d -m elf_i386 -o lab05-1 lab05-1.o
bash: d: команда не найдена
saburievadk8n81 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab05-1 lab05-1.o
saburievadk8n81 ~/work/arch-pc/lab05 $ ./lab05-1
Введите строку:
Буриева Шахзода Акмаловна
saburievadk8n81 ~/work/arch-pc/lab05 $

```

Рис. 3.9: Выполнение компоновки объектного файла lab05-1.asm и его запуск

Скачала файл in\_out.asm со страницы курса в ТУИС. Он сохранился в каталоге “Загрузки”.

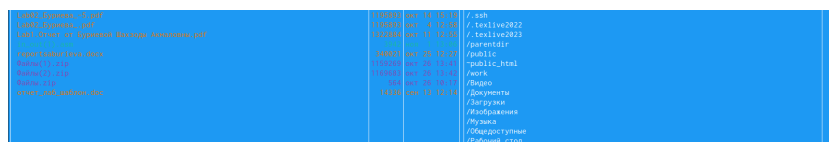


Рис. 3.10: Скачивание файла in\_out.asm

С помощью функциональной клавиши F5 копирую файл in\_out.asm из каталога “Загрузки” в созданный каталог lab05.

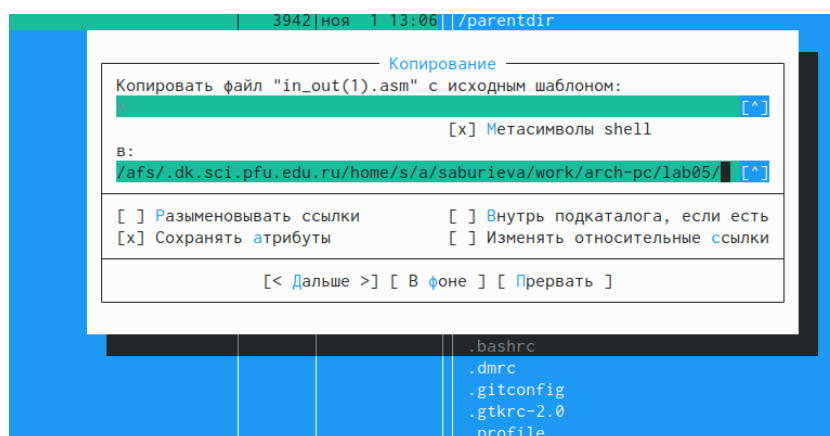


Рис. 3.11: Копирование скачанного файла в каталог lab05

С помощью функциональной клавиши F6 создала копию файла lab05-1.asm с именем lab05-2.asm. Выделила файл lab05-1.asm, нажала клавишу F6, ввела имя файла lab05-2.asm и нажала клавишу Enter.

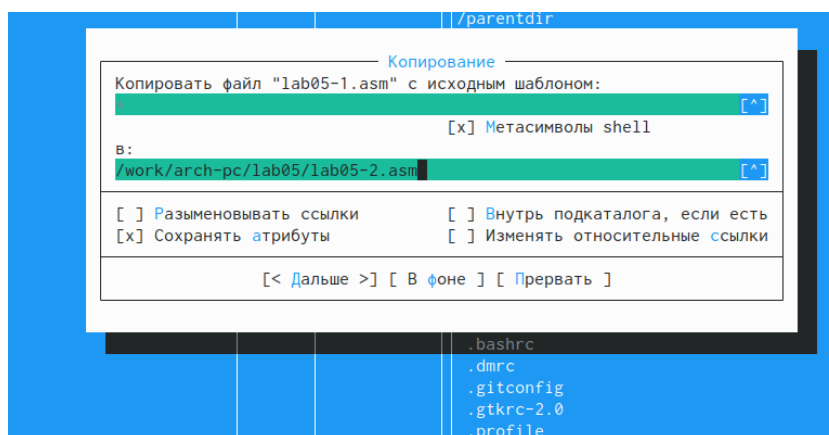
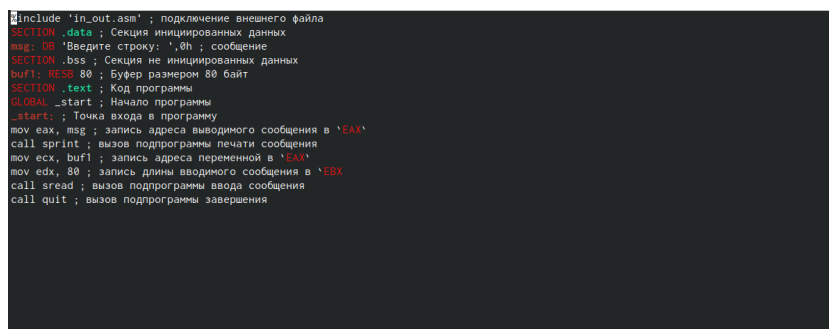


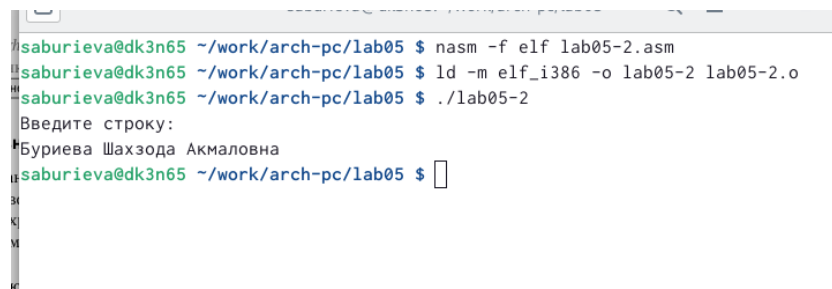
Рис. 3.12: Создание копии файла lab05-1.asm с именем lab05-2.asm

Исправила текст программы в файле lab05-2.asm с использованием подпрограмм из внешнего файла in\_out.asm (использовала подпрограммы sprintf, sread и quit) в соответствии с листингом. Создала исполняемый файл и проверила его работу.



Оттранслировала текст программы lab05-2.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл. Программа вывела строку 'Введите строку:' и ожидала ввода с клавиатуры. На

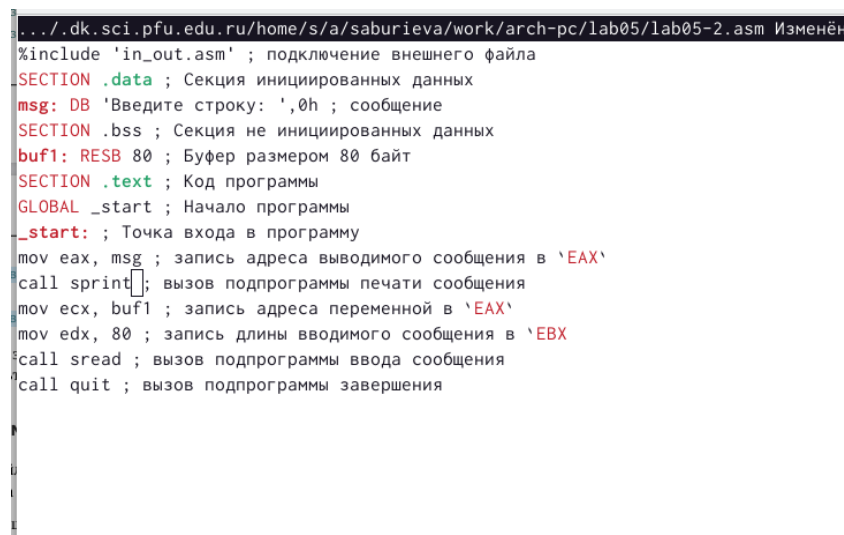
запрос “Введите строку” я ввела свои ФИО.



```
saburieva@dk3n65 ~/work/arch-pc/lab05 $ nasm -f elf lab05-2.asm
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab05-2 lab05-2.o
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ./lab05-2
Введите строку:
Буриева Шахзода Акмаловна
saburieva@dk3n65 ~/work/arch-pc/lab05 $
```

Рис. 3.14: Выполнение компоновки объектного файла lab05-2.asm и его запуск

В файле lab5-2.asm заменила подпрограмму sprintLF на sprint. Создала исполняемый файл и проверила его работу.



```
../../../../dk.sci.pfu.edu.ru/home/s/a/saburieva/work/arch-pc/lab05/lab05-2.asm Изменён
%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 ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 3.15: Замена подпрограммы sprintLF

Снова оттранслировала текст программы lab05-2.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл. Программа вывела строку ‘Введите строку:’ и ожидала ввода с клавиатуры. На запрос “Введите строку” я ввела свои ФИО.

```
saburieva@dk3n65 ~/work/arch-pc/lab05 $ nasm -f elf lab05-2.asm
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab05-2 lab05-2.o
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ./lab05-2
Введите строку: Буриева Шахзода Акмаловна
saburieva@dk3n65 ~/work/arch-pc/lab05 $
```

Рис. 3.16: Выполнение компоновки объектного файла lab05-2.asm и его запуск

Разница между первым исполняемым файлом и вторым исполняемым файлом в том, что запуск первого запрашивает ввод с новой строки, а программа, которая выполняется при запуске второго, запрашивает ввод без переноса а новую строку, потому что в этом именно заключается различие между подпрограммами `sprintf` и `sprint`.

## 4 Выполнение заданий для самостоятельной работы

Создала копию файла lab5-1.asm с именем lab05-3.asm с помощью функциональной строки F5.

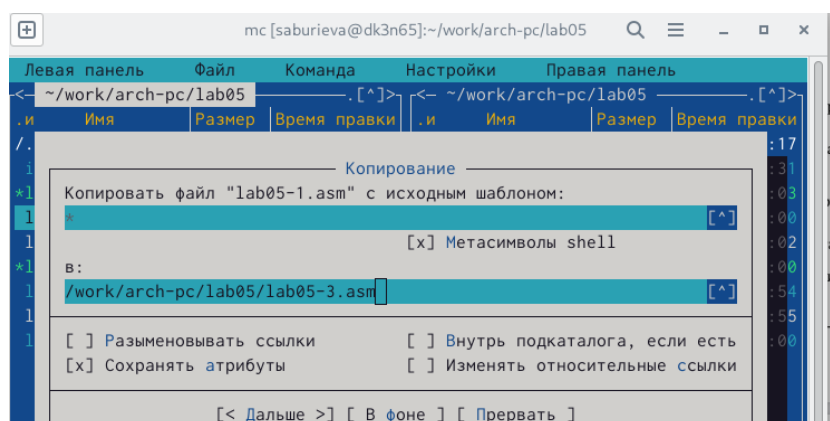
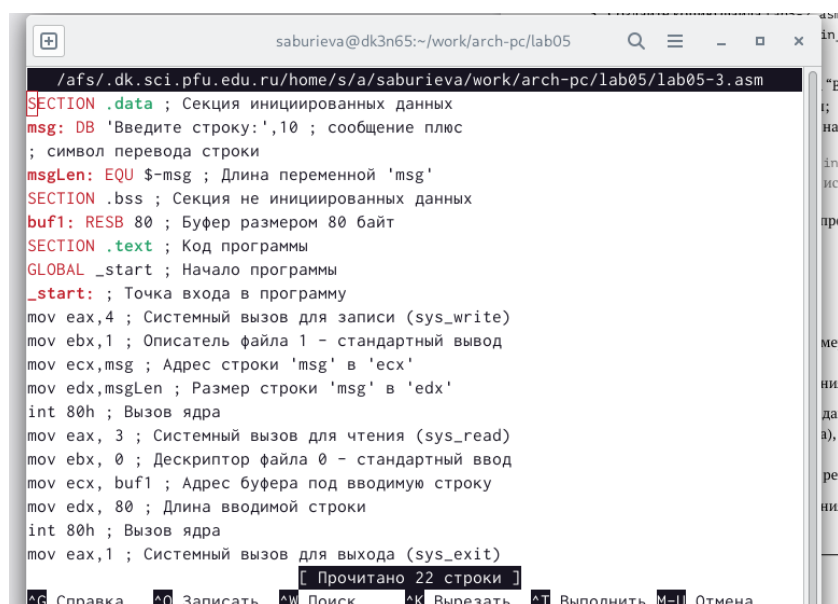


Рис. 4.1: Создание копии файла lab05-1.asm с именем lab05-3.asm


Ввела в эту программу некоторые изменения.



```
/afs/.dk.sci.pfu.edu.ru/home/s/a/saburieva/work/arch-pc/lab05/lab05-3.asm
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
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)
```

Рис. 4.2: Изменения в программе

Оттранслировала текст программы lab05-3.asm в объектный файл. Выполнила компоновку объектного файла и запустила получившийся исполняемый файл.



```
saburieva@dk3n65 ~/work/arch-pc/lab05 $ nasm -f elf lab05-3.asm
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab05-3 lab05-3.o
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ./lab05-3
```

Рис. 4.3: Выполнение компоновки объектного файла lab05-3.asm и его запуск

Программа вывела строку 'Введите строку:' и ожидала ввода с клавиатуры. На запрос "Введите строку" я ввела свою фамилию.



```
Введите строку:
Буриева
saburieva@dk3n65 ~/work/arch-pc/lab05 $
```

Рис. 4.4: Ввод фамилии

Создала копию файла lab5-2.asm с именем lab05-3-1.asm с помощью функциональной строки F5.



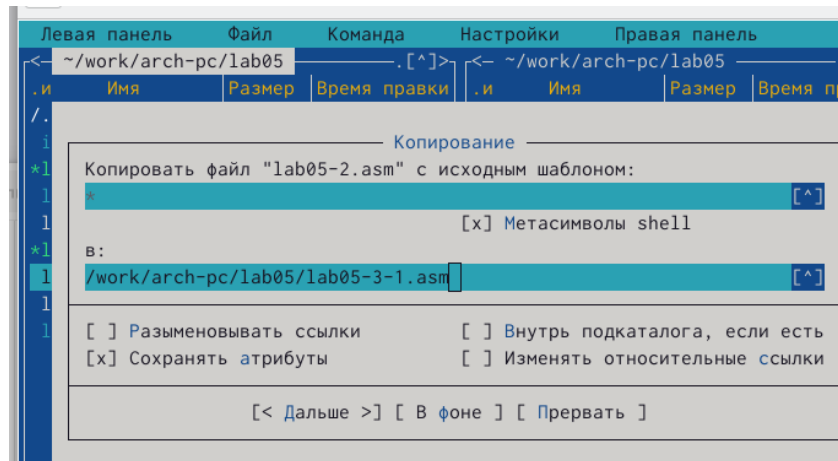


Рис. 4.5: Создание копии файла lab05-1.asm с именем lab05-3-1.asm

Оттранслировала текст программы lab05-3-1.asm в объектный файл. Выпол-  
нила компоновку объектного файла и запустила получившийся исполняемый  
файл. Создала исполняемый файл и проверила его работу.

```
saburieva@dk3n65 ~/work/arch-pc/lab05 $ nasm -f elf lab05-3-1.asm
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab05-3-1 lab05-3-1.o
saburieva@dk3n65 ~/work/arch-pc/lab05 $ ./lab05-3-1
Введите строку: 
```

Рис. 4.6: Выполнение компоновки объектного файла lab05-3-1.asm и его запуск

Отправила лабораторную работу №5 на github.

```

saburieva@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git add .
saburieva@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git commit -am 'feat(main): add files lab-5'
[master 84e80fe] feat(main): add files lab-5
25 files changed, 128 insertions(+), 32 deletions(-)
create mode 100644 labs/lab05/report/image/1.jpg
create mode 100644 labs/lab05/report/image/10.jpg
create mode 100644 labs/lab05/report/image/11.jpg
create mode 100644 labs/lab05/report/image/12.jpg
create mode 100644 labs/lab05/report/image/13.jpg
create mode 100644 labs/lab05/report/image/14.jpg
create mode 100644 labs/lab05/report/image/15.jpg
create mode 100644 labs/lab05/report/image/16.jpg
create mode 100644 labs/lab05/report/image/17.jpg
create mode 100644 labs/lab05/report/image/18.jpg
create mode 100644 labs/lab05/report/image/19.jpg
create mode 100644 labs/lab05/report/image/2.jpg
create mode 100644 labs/lab05/report/image/20.jpg
create mode 100644 labs/lab05/report/image/21.jpg
create mode 100644 labs/lab05/report/image/22.jpg
create mode 100644 labs/lab05/report/image/3.jpg
create mode 100644 labs/lab05/report/image/4.jpg
create mode 100644 labs/lab05/report/image/5.jpg
create mode 100644 labs/lab05/report/image/6.jpg
create mode 100644 labs/lab05/report/image/7.jpg
create mode 100644 labs/lab05/report/image/8.jpg
create mode 100644 labs/lab05/report/image/9.jpg
create mode 100644 labs/lab05/report/report.docx
create mode 100644 labs/lab05/report/report.pdf
saburieva@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (31/31), готово.
Запись объектов: 100% (31/31), 3.55 МБ | 3.49 МБ/с, готово.
Всего 31 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:saburieva/https-github.com-saburieva-study_2023-2024_arh--pc-.git
ac0e47c..84e80fe master -> master
saburieva@dk5n52 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $

```

Рис. 4.7: Отправка на github

## 5 Выводы

Я приобрела практические навыки для работы в Midnight Commander и освоила инструкции языка ассемблера `mov` и `int`.

## **Список литературы**