

Отчёт по лабораторной работе

Лабораторная №4

Полина Витальевна Барабаш

Содержание

<i>1</i>	<i>Цель работы</i>	<i>4</i>
<i>2</i>	<i>Выполнение работы</i>	<i>5</i>
<i>3</i>	<i>Выполнение самостоятельной работы</i>	<i>10</i>
<i>4</i>	<i>Выводы</i>	<i>15</i>

Список иллюстраций

2.1	Создание необходимого каталога	5
2.2	Переход в каталог	5
2.3	Создание файла и проверка его создания	6
2.4	Компиляция шаблона с помощью make	6
2.5	Созданный объектный файл после компиляции текста программы	7
2.6	Использование команды <code>nasm -o obj.o -f elf -g -l list.lst hello.asm</code> и проверка создания файлов	7
2.7	Создание исполняемого файла и проверка его создания	8
2.8	Создание исполняемого файла и проверка его создания	8
2.9	Выполнение исполняемого файла <code>hello</code>	9
3.1	Копирование файла с другим названием	10
3.2	Изменение текста программы в файле <code>lab4.asm</code>	11
3.3	Запуск программы <code>lab4.asm</code>	11
3.4	Копирование файлов в другой каталог	12
3.5	Компилирование отчёта в других форматах	13
3.6	Загрузка файлов на Github	13
3.7	Загрузка файлов на Github	14

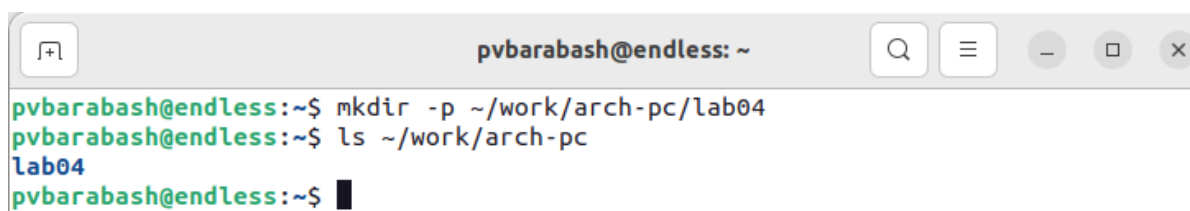
1 Цель работы

Освоение процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Выполнение работы

Задание №1. Создать каталог для работы с программами на языке ассемблера NASM: `mkdir -p ~/work/arch-pc/lab04`

Я создала каталог и проверила, что он создан (рис. 2.1).

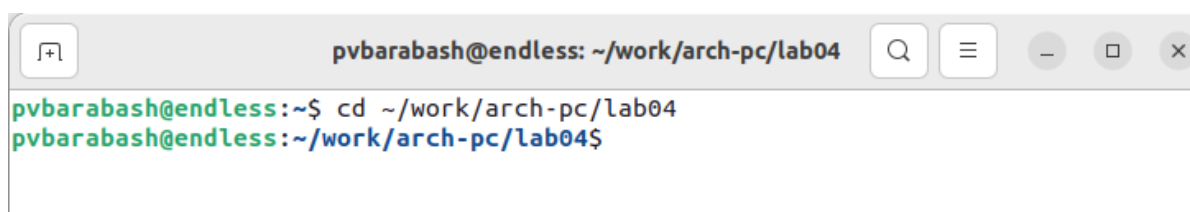
A screenshot of a terminal window titled 'pvbarabash@endless: ~'. The terminal shows the following commands and output:

```
pvbarabash@endless:~$ mkdir -p ~/work/arch-pc/lab04
pvbarabash@endless:~$ ls ~/work/arch-pc
lab04
pvbarabash@endless:~$
```

Рис. 2.1: Создание необходимого каталога

Задание №2. Перейти в созданный каталог.

Я перешла в созданный каталог `~/work/arch-pc/lab04` с помощью `cd` (рис. 2.2).

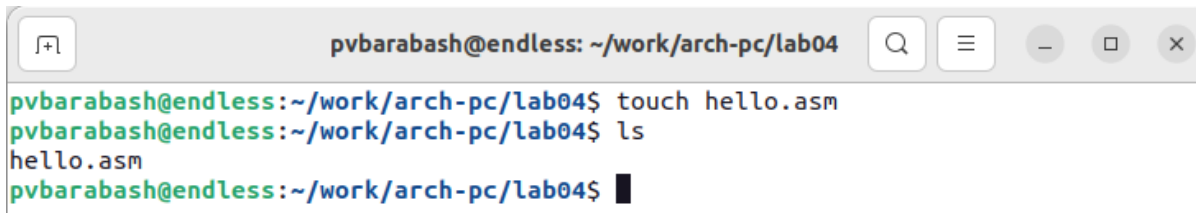
A screenshot of a terminal window titled 'pvbarabash@endless: ~/work/arch-pc/lab04'. The terminal shows the following commands and output:

```
pvbarabash@endless:~$ cd ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.2: Переход в каталог

Задание №3. Создать текстовый файл с именем `hello.asm`.

Я создала текстовый файл с именем `hellp.asm` с помощью `touch` и проверила его создание с помощью `ls` (рис. 2.3).

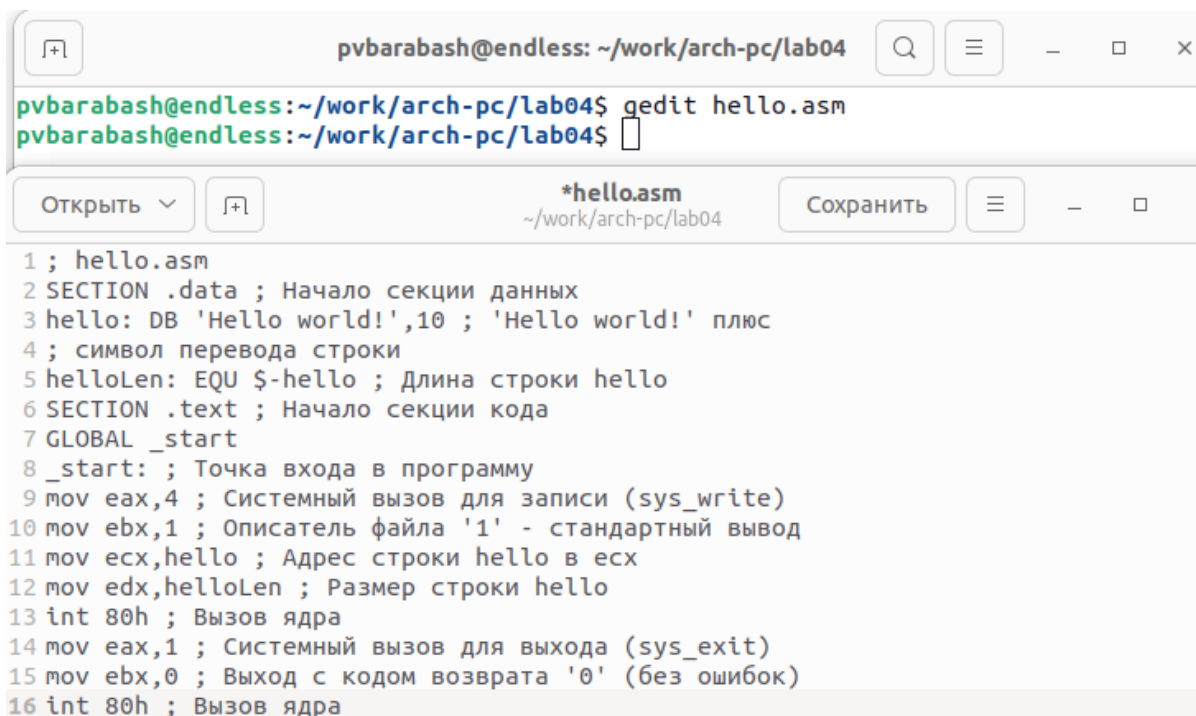


```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ touch hello.asm
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello.asm
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.3: Создание файла и проверка его создания

Задание №4. Открыть этот файл с помощью любого текстового редактора, например, gedit, и ввести в него необходимый текст.

Я открыла этот файл с помощью gedit и ввела необходимый текст (рис. 2.4).



```
pvbarabash@endless:~/work/arch-pc/lab04$ gedit hello.asm
pvbarabash@endless:~/work/arch-pc/lab04$
```

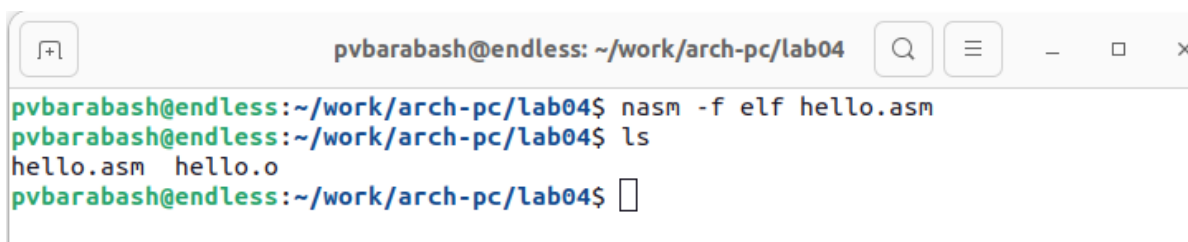
Открыть ▾ | *hello.asm | Сохранить

```
1 ; hello.asm
2 SECTION .data ; Начало секции данных
3 hello: DB 'Hello world!',10 ; 'Hello world!' плюс
4 ; символ перевода строки
5 helloLen: EQU $-hello ; Длина строки hello
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,hello ; Адрес строки hello в ecx
12 mov edx,helloLen ; Размер строки hello
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
```

Рис. 2.4: Компиляция шаблона с помощью make

Задание №5. Скомпилировать текст программы из файла hello.asm и проверить, что был создан объектный файл. Какое имя имеет объектный файл?

Я скомпилировала текст программы с помощью `nasm -f elf hello.asm` и проверила, что был создан файл **hello.o** в текущем каталоге с помощью `ls` (рис. 2.5).

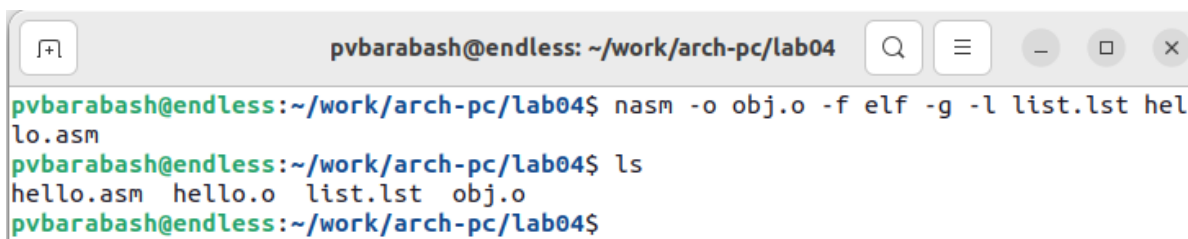


```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ nasm -f elf hello.asm
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.5: Созданный объектный файл после компиляции текста программы

Задание №6. Ввести команду `nasm -o obj.o -f elf -g -l list.lst hello.asm`, которая скомпилирует исходный файл `hello.asm` в `obj.o` (опция `-o` позволяет задать имя объектного файла, в данном случае `obj.o`), при этом формат выходного файла будет `elf`, и в него будут включены символы для отладки (опция `-g`), кроме того, будет создан файл листинга `list.lst` (опция `-l`). Проверить, что файлы были созданы.

Я ввела необходимую команду и проверила, что файлы `list.lst` и `obj.o` были созданы (рис. 2.6).

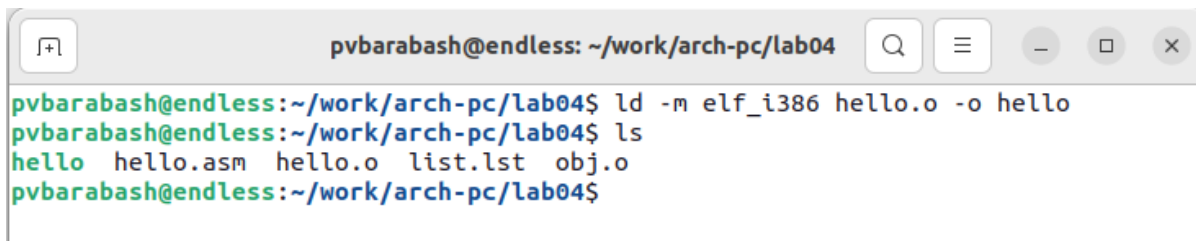


```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hel
lo.asm
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.6: Использование команды `nasm -o obj.o -f elf -g -l list.lst hello.asm` и проверка создания файлов

Задание №7. Передать объектный файл на обработку компоновщика. Проверить, что исполняемый файл `hello` был создан.

Я передала объектный файл с помощью команды `ld -m elf_i386 hello.o -o hello` и проверила, что исполняемый файл `hello` был создан (рис. 2.7).

A terminal window with a title bar showing 'pvbarabash@endless: ~/work/arch-pc/lab04'. The terminal contains the following commands and output:

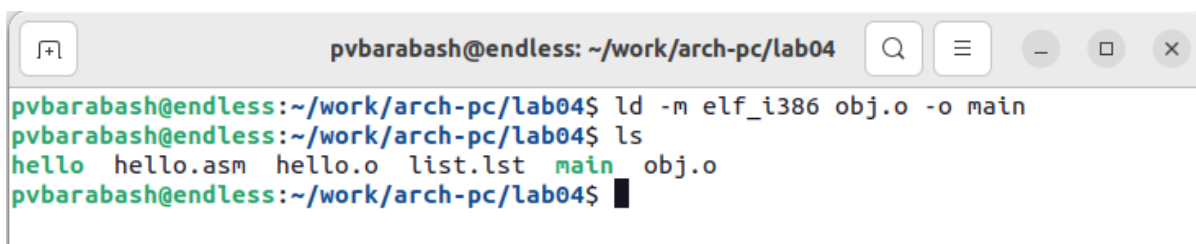
```
pvbarabash@endless:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.7: Создание исполняемого файла и проверка его создания

Задание №8. Выполните следующую команду: `ld -m elf_i386 obj.o -o main`. Какое имя будет иметь исполняемый файл? Какое имя имеет объектный файл из которого собран этот исполняемый файл?

Исходя из текста команды, на обработку компоновщика передаётся объектный файл **obj.o**, а создан будет исполняемый файл с именем **main**.

Я выполнила предложенную команду и посмотрела, какой файл был создан, с помощью `ls` (рис. 2.8).

A terminal window with a title bar showing 'pvbarabash@endless: ~/work/arch-pc/lab04'. The terminal contains the following commands and output:

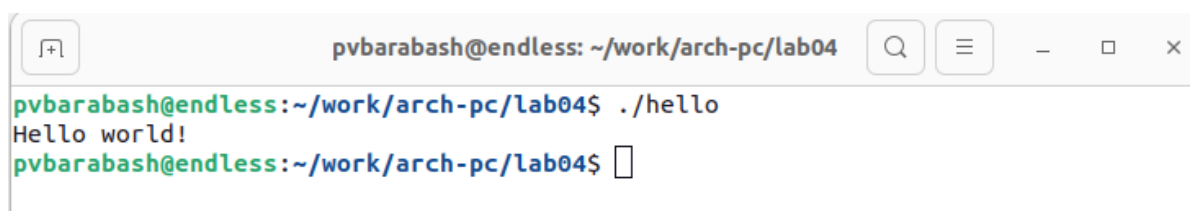
```
pvbarabash@endless:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.8: Создание исполняемого файла и проверка его создания

Действительно, исполняемый файл **main**.

Задание №9. Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге.

С помощью команды `./hello` я запустила на выполнение созданный исполняемый файл `hello`. На экран было выведено *Hello word!* (рис. 2.9).



A terminal window with a title bar containing a maximize button, the text 'pvbarabash@endless: ~/work/arch-pc/lab04', a search icon, a menu icon, and window control buttons (minimize, maximize, close). The terminal content shows a green prompt 'pvbarabash@endless:~/work/arch-pc/lab04\$' followed by the command './hello'. The output 'Hello world!' is displayed on the next line. A second green prompt 'pvbarabash@endless:~/work/arch-pc/lab04\$' is shown with a cursor, indicating the command has completed.

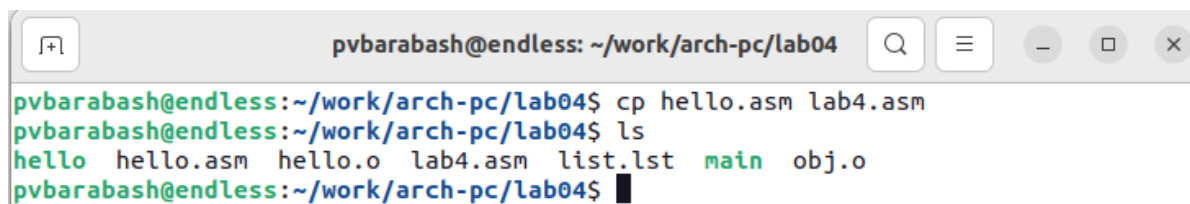
```
pvbarabash@endless:~/work/arch-pc/lab04$ ./hello
Hello world!
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 2.9: Выполнение исполняемого файла hello

3 Выполнение самостоятельной работы

Задание №1. В каталоге `~/work/arch-pc/lab04` с помощью команды `cp` создайте копию файла `hello.asm` с именем `lab4.asm`

Я создала копию файла `hello.asm` с именем `lab4.asm`, введя `cp hello.asm lab4.asm` и проверила, что файл `lab4.asm` был создан (рис. 3.1).

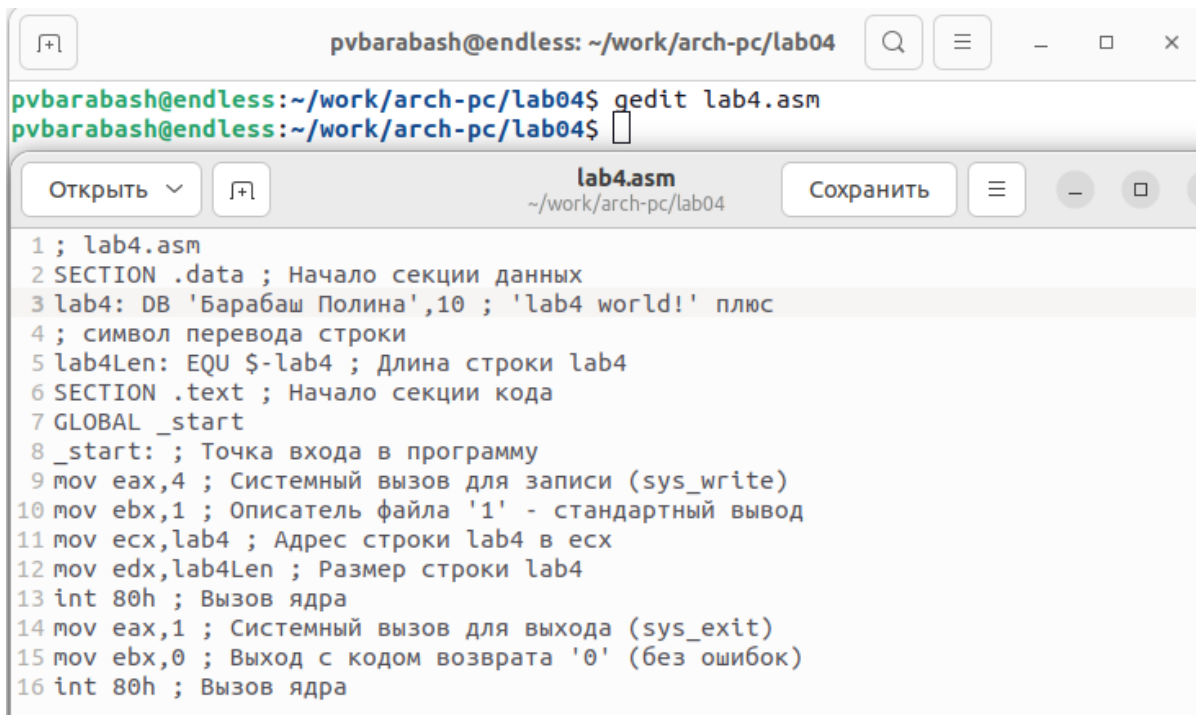
A screenshot of a terminal window. The title bar shows the user 'pvbarabash@endless' and the current directory '~/work/arch-pc/lab04'. The terminal content shows the user entering the command 'cp hello.asm lab4.asm', followed by 'ls', which lists the files: 'hello', 'hello.asm', 'hello.o', 'lab4.asm', 'list.lst', 'main', and 'obj.o'.

```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
pvbarabash@endless:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 3.1: Копирование файла с другим названием

Задание №2. С помощью любого текстового редактора внесите изменения в текст программы в файле `lab4.asm` так, чтобы вместо *Hello world!* на экран выводилась строка с вашими фамилией и именем.

Я открыла файл `lab4.asm` с помощью `gedit` и изменила текст так, чтобы выводилась строка с моими фамилией и именем (рис. 3.2).



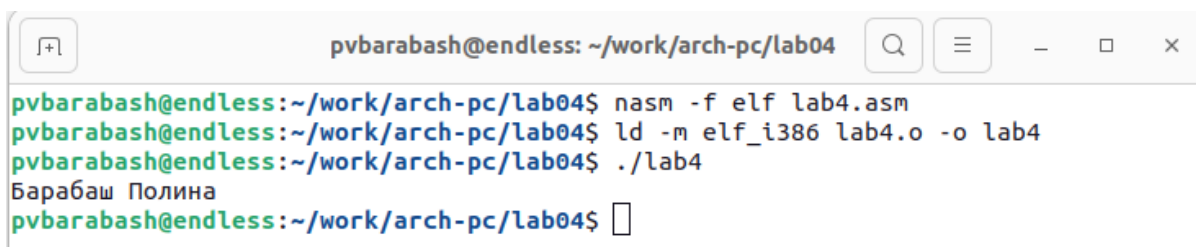
```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ gedit lab4.asm
pvbarabash@endless:~/work/arch-pc/lab04$

1 ; lab4.asm
2 SECTION .data ; Начало секции данных
3 lab4: DB 'Барабаш Полина',10 ; 'lab4 world!' плюс
4 ; символ перевода строки
5 lab4Len: EQU $-lab4 ; Длина строки lab4
6 SECTION .text ; Начало секции кода
7 GLOBAL _start
8 _start: ; Точка входа в программу
9 mov eax,4 ; Системный вызов для записи (sys_write)
10 mov ebx,1 ; Описатель файла '1' - стандартный вывод
11 mov ecx,lab4 ; Адрес строки lab4 в ecx
12 mov edx,lab4Len ; Размер строки lab4
13 int 80h ; Вызов ядра
14 mov eax,1 ; Системный вызов для выхода (sys_exit)
15 mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
16 int 80h ; Вызов ядра
```

Рис. 3.2: Изменение текста программы в файле lab4.asm

Задание №3. Оттранслировать полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл.

С помощью `nasm -f elf lab4.asm` я оттранслировала полученный текст программы lab4.asm в объектный файл. С помощью `ld -m elf_i386 lab4.o -o lab4` я выполнила компоновку объектного файла и с помощью `./lab4` я запустила получившийся исполняемый файл. В терминал действительно вывелись мои фамилия и имя *Барабаш Полина* (рис. 3.3).

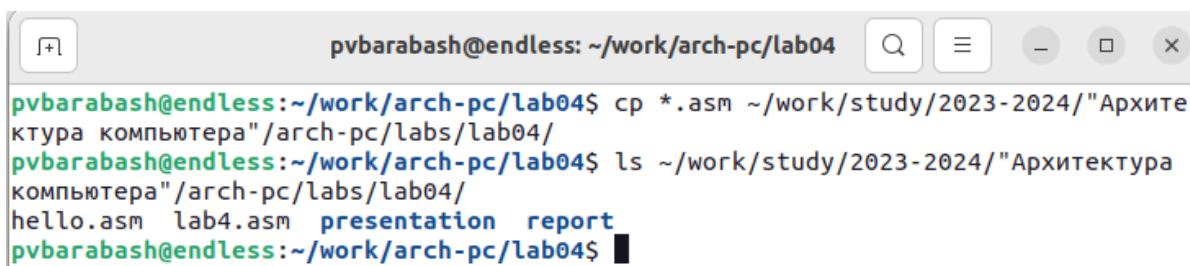


```
pvbarabash@endless: ~/work/arch-pc/lab04
pvbarabash@endless:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
pvbarabash@endless:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
pvbarabash@endless:~/work/arch-pc/lab04$ ./lab4
Барабаш Полина
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 3.3: Запуск программы lab4.asm

Задание №4. Скопируйте файлы `hello.asm` и `lab4.asm` в Ваш локальный репозиторий в каталог `~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/`. Загрузите файлы на Github.

С помощью команды `cp` я скопировала файлы `hello.asm` и `lab4.asm` в мой локальный репозиторий в каталог `~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/`. И с помощью `ls` проверила, что оба файла скопированы в нужный каталог (рис. 3.4).

A screenshot of a terminal window with a title bar that reads 'pvbarabash@endless: ~/work/arch-pc/lab04'. The terminal shows the following commands and output:

```
pvbarabash@endless:~/work/arch-pc/lab04$ cp *.asm ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/
pvbarabash@endless:~/work/arch-pc/lab04$ ls ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab04/
hello.asm  lab4.asm  presentation  report
pvbarabash@endless:~/work/arch-pc/lab04$
```

Рис. 3.4: Копирование файлов в другой каталог

Перед тем, как загружать файлы на Github, я перешла в каталог `~/work/study/2023-2024/“Архитектура компьютера”/arch-pc/labs/lab04/report` с помощью команды `cd` и с помощью команды `make` скомпилировала отчёт также в форматах `.docx` и `.pdf` (рис. 3.5).

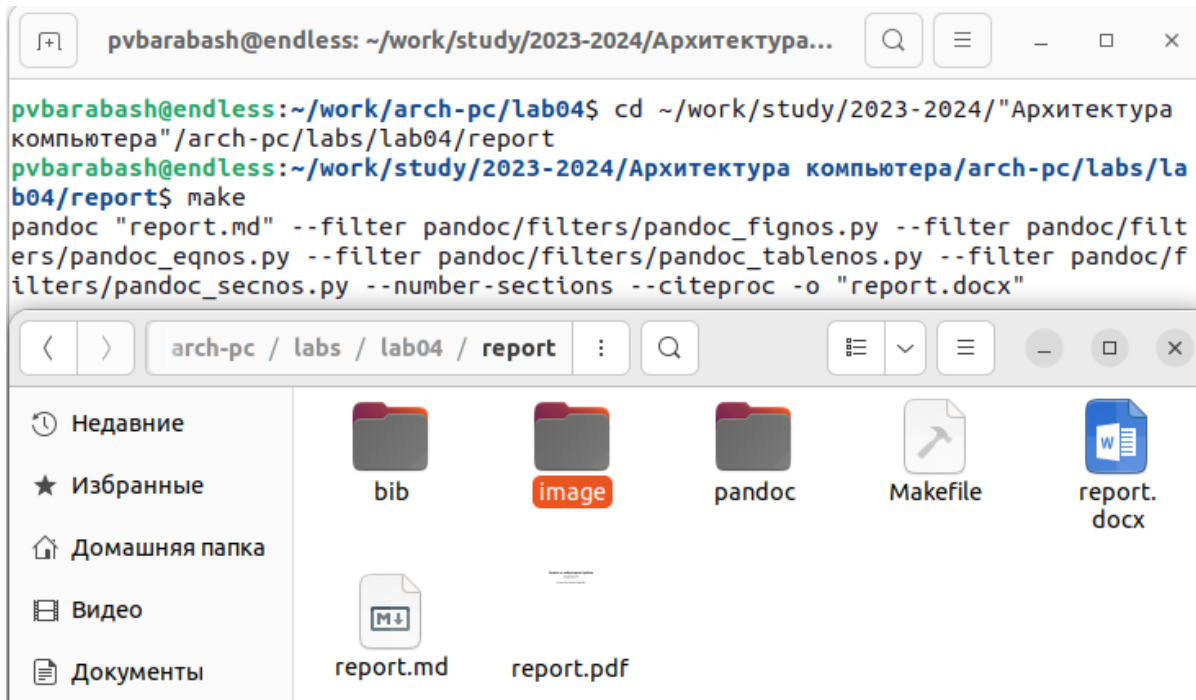


Рис. 3.5: Компилирование отчёта в других форматах

Поднявшись на одну папку вверх, я загрузила файлы на Github с помощью последовательного ввода следующих команд:

```

git add .
git commit -am 'feat(main): add files lab-4, lab4.asm, hello.asm'
git push

```

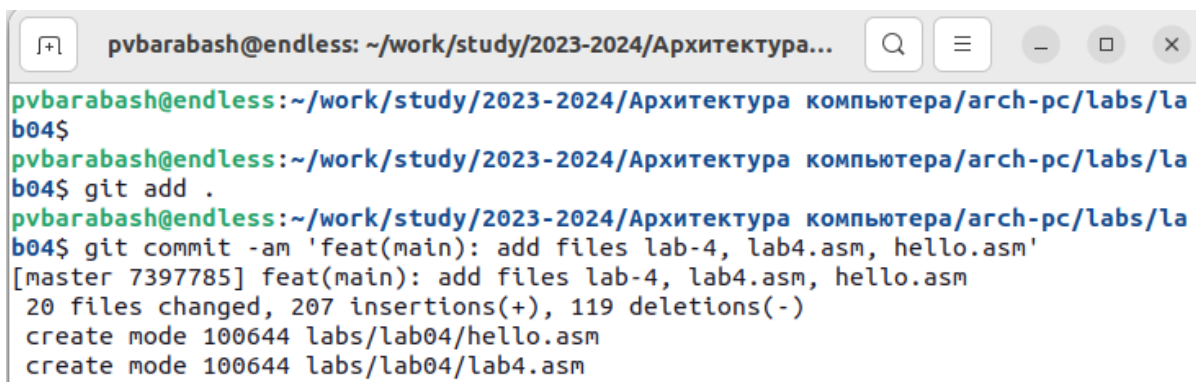


Рис. 3.6: Загрузка файлов на Github

```
pvbarabash@endless: ~/work/study/2023-2024/Архитектура...
create mode 100644 labs/lab04/report/image/fig009.png
create mode 100644 labs/lab04/report/image/fig010.png
create mode 100644 labs/lab04/report/image/fig011.png
create mode 100644 labs/lab04/report/image/fig012.png
create mode 100644 labs/lab04/report/image/fig013.png
create mode 100644 labs/lab04/report/image/fig014.png
delete mode 100644 labs/lab04/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab04/report/report.docx
rewrite labs/lab04/report/report.md (76%)
create mode 100644 labs/lab04/report/report.pdf
pvbarabash@endless:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04$ git push
Перечисление объектов: 31, готово.
Подсчет объектов: 100% (31/31), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (25/25), готово.
Запись объектов: 100% (25/25), 1.47 МиБ | 2.02 МиБ/с, готово.
Всего 25 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To github.com:PBarabash/study_2023-2024_arh--pc.git
  3988f98..7397785 master -> master
pvbarabash@endless:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04$
```

Рис. 3.7: Загрузка файлов на Github

4 Выводы

Я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM. Познакомилась со структурой программы на языке NASM. Вспомнила синтаксис команды `sr`, как выгружать файлы в Github.