

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

дисциплина: Архитектура компьютера

Студент: Лобанова Екатерина Евгеньевна

Группа: НПИбд-01-25

Студ. Билет: 1032252596

МОСКВА

2025 г.

1. Цель работы

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

2. Порядок выполнения лабораторной работы

2.1 Программа Hello world!

Создаем каталог для работы с программами на языке ассемблера NASM и переходим в созданный каталог. Рис 1. 1

```
(eelobanova@kali)-[~]  
$ mkdir -p ~/work/arch-pc/lab04  
  
(eelobanova@kali)-[~]  
$ cd ~/work/arch-pc/lab04  
  
(eelobanova@kali)-[~/work/arch-pc/lab04]  
$
```

Рис 1. 1

Создаем текстовый файл с именем hello.asm Рис 1. 2

```
(eelobanova@kali)-[~/work/arch-pc/lab04]  
$ touch hello.asm
```

Рис 1. 2

Откроем этот файл с помощью любого текстового редактора, например nano и введем заданный текст. Рис 1. 3, Рис 1. 4

```
(eelobanova@kali)-[~/work/arch-pc/lab04]  
$ nano hello.asm
```

Рис 1. 3



```
GNU nano 8.4
; hello.asm
SECTION .data
    hello: DB 'Hello world!', 10
    helloLen: EQU $-hello
SECTION .text
GLOBAL _start
_start:
    mov eax,4
    mov ebx,1
    mov ecx,hello
    mov edx,helloLen
    int 80h

    mov eax,1
    mov ebx,0
    int 80h

^G Help      ^O Write Out  ^F Where Is
^X Exit      ^R Read File  ^\ Replace
```

Рис 1. 4

2.2 Транслятор NASM

NASM превращает текст программы в объектный код. Например, для компиляции приведённого выше текста программы «Hello World» напишем:

`nasm -f elf hello.asm` Рис 1. 5

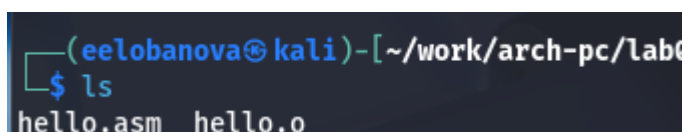


```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ nasm -f elf hello.asm
```

Рис 1. 5

Если текст программы набран без ошибок, то транслятор преобразует текст программы из файла `hello.asm` в объектный код, который запишется в файл `hello.o`. С помощью команды `ls` проверьте, что объектный файл был создан.

Объектный файл имеет название `hello`. Рис 1. 6



```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello.asm  hello.o
```

Рис 1. 6

2.3 Расширенный синтаксис командной строки NASM

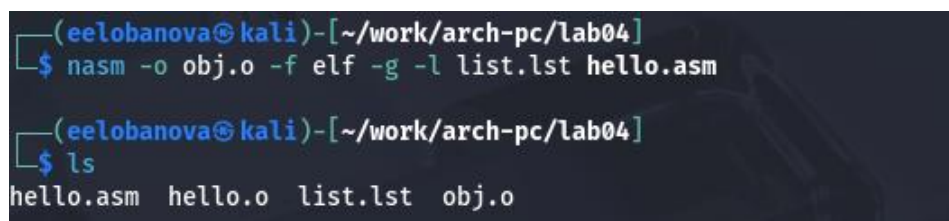
Полный вариант командной строки `nasm` выглядит следующим образом: `nasm [-@ косвенный_файл_настроек] [-o объектный_файл] [-f ↵`

`формат_объектного_файла] [-l листинг] [параметры...] [--] исходный_файл`

Выполним команду. Рис 1.7

Данная команда компилирует исходный файл `hello.asm` в `obj.o` (опция `-o` позволяет задать имя объектного файла, в данном случае `obj.o`), при этом формат выходного файла получается `elf`, и в него будут включены символы для отладки (опция `-g`), кроме того, будет создан файл листинга `list.lst` (опция `-l`).

С помощью команды `ls` проверим, что файлы были созданы. Рис 1. 7



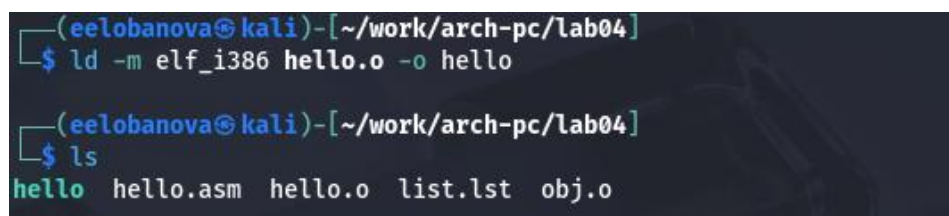
```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ nasm -o obj.o -f elf -g -l list.lst hello.asm

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello.asm  hello.o  list.lst  obj.o
```

Рис 1. 7

2.4 Компоновщик LD

Чтобы получить исполняемую программу, объектный файл необходимо передать на обработку компоновщику. С помощью команды `ls` проверим, что исполняемый файл `hello` был создан. Рис 1.8

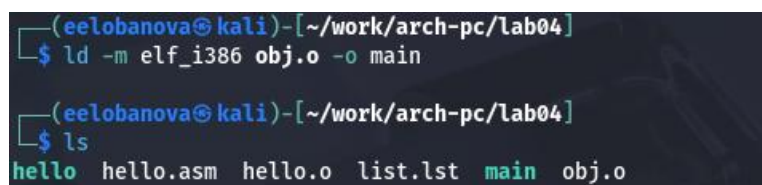


```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 hello.o -o hello

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello  hello.asm  hello.o  list.lst  obj.o
```

Рис 1. 8

Ключ `-o` с последующим значением задаёт в данном случае имя создаваемого исполняемого файла. Выполним команду: Рис 1. 9



```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 obj.o -o main

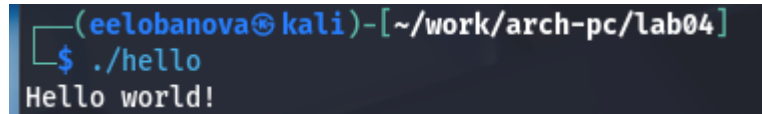
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
```

Рис 1. 9

Исполняемый файл имеет имя main, а объектный файл имя obj.o

2.5 Запуск исполняемого файла

Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге, можно, набрав в командной строке: ./hello Рис 1. 10

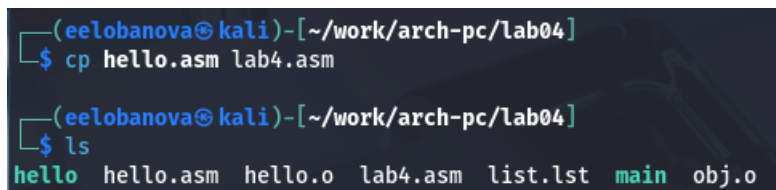


```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ./hello
Hello world!
```

Рис 1. 10

3 Задание для самостоятельной работы

- 1) В каталоге ~/work/arch-pc/lab04 с помощью команды cp создайте копию файла hello.asm с именем lab4.asm Рис 2. 1

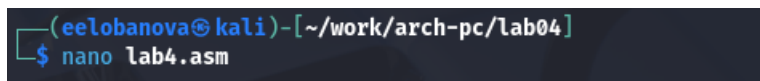


```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ cp hello.asm lab4.asm

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello  hello.asm  hello.o  lab4.asm  list.lst  main  obj.o
```

Рис 2. 1

- 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. Рис 2. 2, Рис 2. 3



```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ nano lab4.asm
```

Рис 2. 2

```
GNU nano 8.4 lab4.asm
; hello.asm
SECTION .data
    hello: DB 'Kate Lobanova', 10
    helloLen: EQU $-hello
SECTION .text
GLOBAL _start
_start:
    mov eax,4
    mov ebx,1
    mov ecx,hello
    mov edx,helloLen
    int 80h

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

Рис 2. 3

- 3) Оттранслируйте полученный текст программы lab4.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл. Рис 2. 4

```
(eelobanova@kali)-[~/work/arch-pc/lab04]
$ nasm -f elf lab4.asm

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o lab4.asm lab4.o list.lst main obj.o

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ld -m elf_i386 lab4.o -o lab4

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ ./lab4
Kate Lobanova
```

Рис 2. 4

- 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. Рис 2. 5, Рис 2. 6 Рис 2. 7

```

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ cp ~/work/arch-pc/lab04/lab4.asm ~/work/study/2025-2026/"Архитектура компьютер
epa"/arch-pc/labs/lab04/lab4.asm

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ cp ~/work/arch-pc/lab04/hello.asm ~/work/study/2025-2026/"Архитектура компью
теpa"/arch-pc/labs/lab04/hello.asm

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ cd ~/work/arch-pc/lab04

(eelobanova@kali)-[~/work/arch-pc/lab04]
$ cd ~/work/study/2025-2026/"Архитектура компьютера"/arch-pc

(eelobanova@kali)-[~/.../study/2025-2026/Архитектура компьютера/arch-pc]
$ git merge
Already up to date.

(eelobanova@kali)-[~/.../study/2025-2026/Архитектура компьютера/arch-pc]
$ git add .

(eelobanova@kali)-[~/.../study/2025-2026/Архитектура компьютера/arch-pc]
$ git commit
[master 97118bf] Добавление файлов на гит
6 files changed, 37 insertions(+), 1 deletion(-)
create mode 100644 labs/lab03/report/.report.qmd.swp
rename labs/lab03/report/{arch-pc--lab03--report.qmd => report.qmd} (100%)
create mode 100644 labs/lab04/hello.asm
create mode 100644 labs/lab04/lab4.asm
create mode 100644 labs/lab83/report/report.md

```

Рис 2. 5


```

(eelobanova@kali)-[~/.../study/2025-2026/Архитектура компьютера/arch-pc]
$ git push
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (13/13), 1.18 KiB | 605.00 KiB/s, done.
Total 13 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:kayetarinii/study_2025-2026_arh-pc.git
b1723ff..97118bf master -> master

```

Рис 2. 6

study_2025-2026_arh-pc / labs / lab04 / 


kayetarinii
Добавление файлов на гит

Name
..
presentation
report
hello.asm
lab4.asm

Рис 2. 7

Заключение

В ходе лабораторной работы были изучены основы работы с ассемблером NASM в операционной системе Linux. Приобретены практические навыки написания, трансляции и выполнения низкоуровневых программ на языке ассемблера.