**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: ТУЙИШИМЕ тьерри

Группа: НКАбд-05-25

Оглавление

[1. Цель работы 3](#_Toc211785078)

[2. Подготовка рабочего каталога 3](#_Toc211785079)

[2.1. Создание программы Hello World 3](#_Toc211785080)

[2.2. Содержание файла hello.asm 3](#_Toc211785081)

[2.3. Трансляция программы с помощью NASM 4](#_Toc211785082)

[2.4. Расширенная компиляция с отладочной информацией 5](#_Toc211785083)

[2.5. Компоновка с помощью LD 5](#_Toc211785084)

[2.6. Запуск исполняемого файла 5](#_Toc211785085)

[3. Самостоятельная работа 5](#_Toc211785086)

[3.1. Создание копии файла 5](#_Toc211785087)

[3.2. Модификация программы 6](#_Toc211785088)

[3.2. Копирование файлов в локальный репозиторий 7](#_Toc211785089)

[4.Результаты выполнения 7](#_Toc211785090)

[5.Выводы 7](#_Toc211785091)

**Создание и процесс обработки программ на языке ассемблера NASM**

# Цель работы

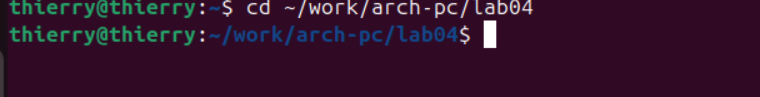
Создание и процесс обработки программ на языке ассемблера

NASM.

# Подготовка рабочего каталога

Команды:

* **mkdir -p**: создает всю иерархию каталогов, даже если родительские каталоги не существуют.
* **Флаг -p** :(parents) гарантирует создание всех промежуточных каталогов.
* **cd**: изменяет текущую рабочую директорию на созданную папку lab04.



## Создание программы Hello World

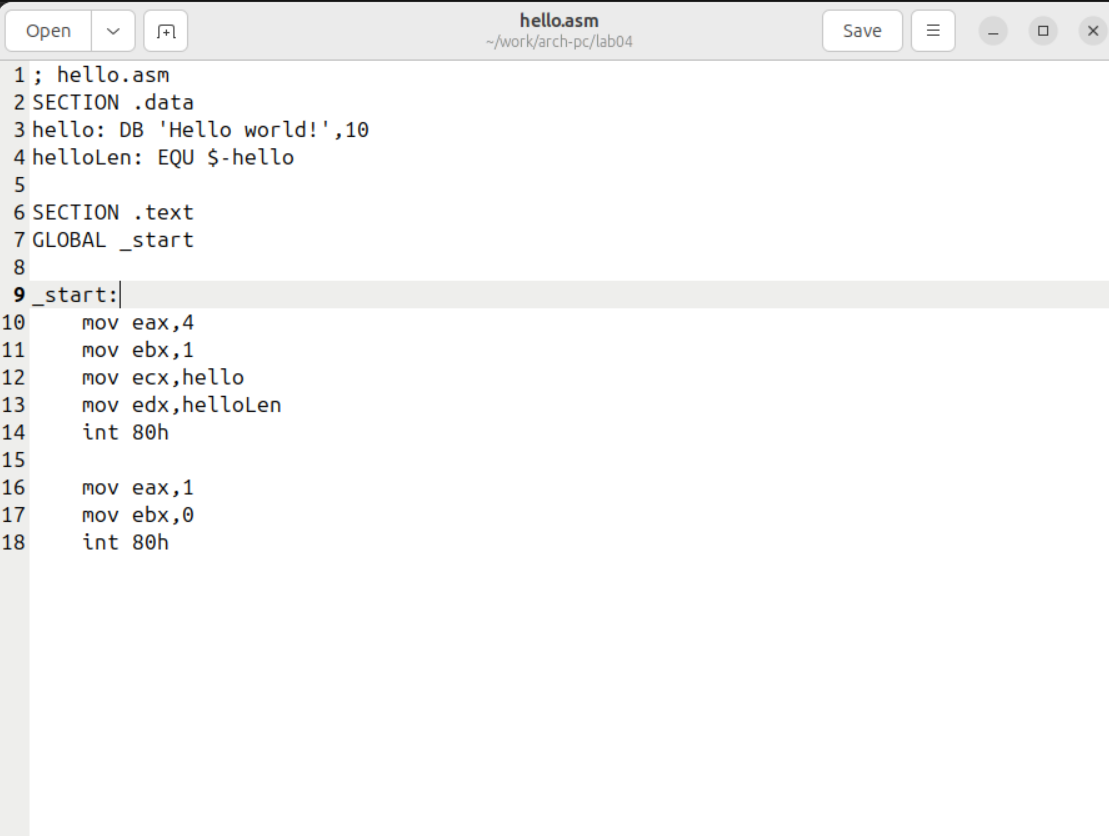
Команды:

**Touch:** создает пустой файл с указанным именем.

**Gedit:** запускает текстовый редактор для редактирования файла.

## Содержание файла hello.asm

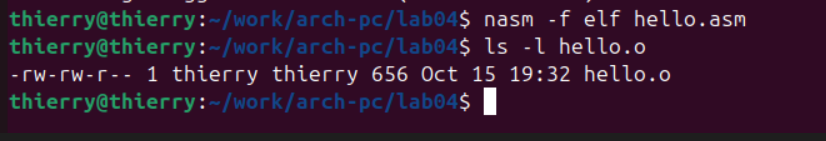
* **SECTION .data**: секция данных для объявления констант
* **DB 'Hello world!',10**: определение байтов: строка + символ новой строки (ASCII 10)
* **EQU $-hello**: вычисление длины строки
* **SECTION .text**: секция кода
* **GLOBAL \_start**: объявление точки входа программы
* **mov eax,4**: системный вызов sys\_write
* **mov ebx,1** - стандартный вывод (stdout)
* **mov ecx, hello**: адрес строки для вывода
* **mov edx, helloLen**: длина строки
* **int 80h**: вызов прерывания ядра Linux



## Трансляция программы с помощью NASM

* **nasm:** ассемблер, преобразующий код в машинные инструкции
* **-f elf**: формат выходного файла (ELF для 32-битных систем)

Создается объектный файл hello. o, содержащий машинный код

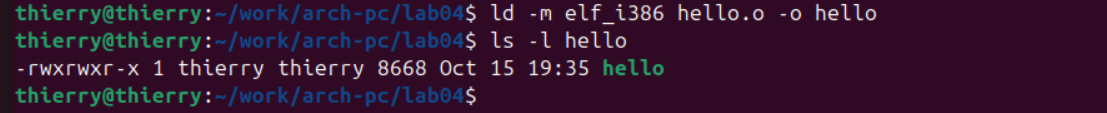


## Расширенная компиляция с отладочной информацией

* **-o obj. o**: задает имя выходного файла
* **-g**: включает отладочную информацию
* **-l list.lst**: создает файл листинга

## Компоновка с помощью LD

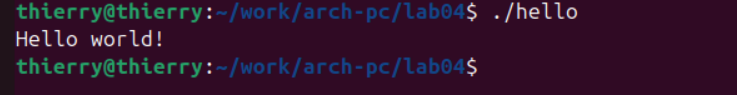
* **ld**: компоновщик, связывающий объектные файлы в исполняемый
* **-m elf\_i386**: целевая архитектура (32-битная x86)
* **-o hello**: имя выходного исполняемого файла



## Запуск исполняемого файла

* **. /** указывает, что программа находится в текущей директории

Программа выполняется и выводит "Hello world!" на экран



# Самостоятельная работа

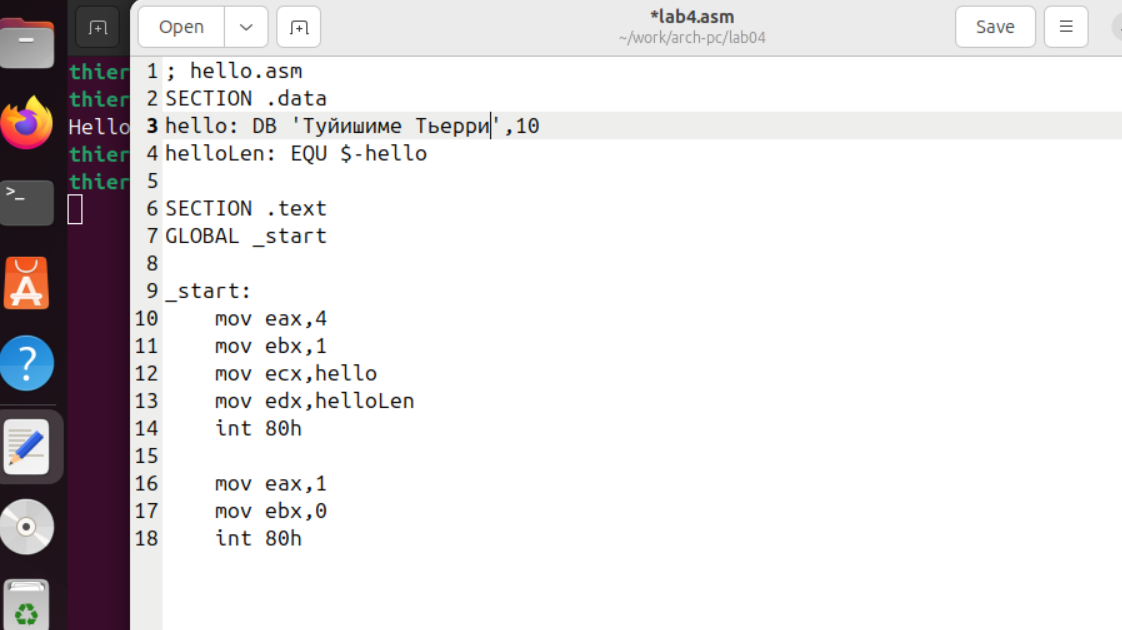
## 3.1. Создание копии файла

Команда:

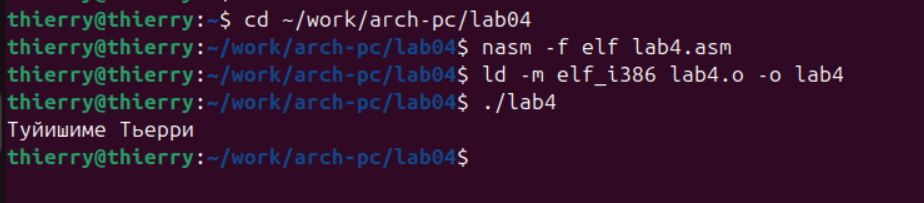
* **cp hello.asm lab4.asm:** Создает копию исходного файла для модификации.

## 3.2. Модификация программы

Редактирование файла lab4.asm:

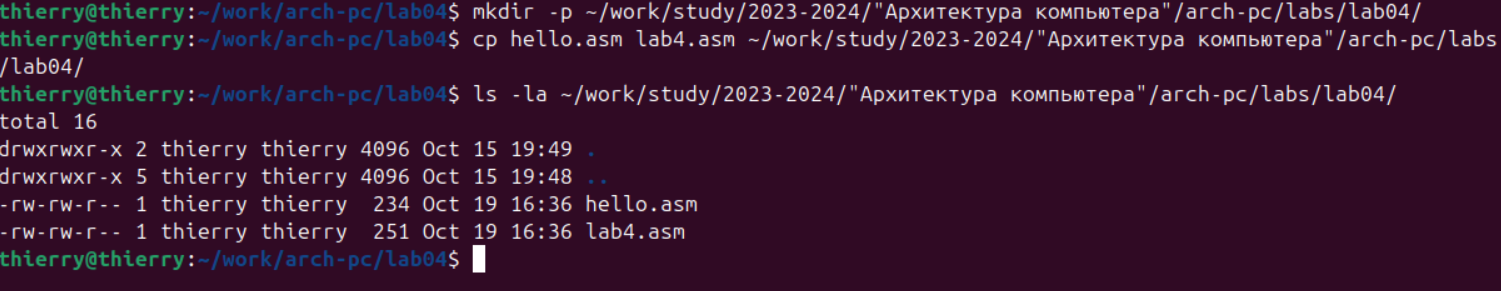


**gedit lab4.asm**: заменена строка "Hello world!" на фамилию и имя студента



* Полный цикл компиляции и запуска измененной программы

## 3.2. Копирование файлов в локальный репозиторий



* Создание структуры каталогов и копирование исходных файлов

# 4.Результаты выполнения

Созданные файлы:

* **hello.asm:** исходный код программы Hello World
* **hello. o:** объектный файл
* **hello:** исполняемый файл
* **lab4.asm:** модифицированная программа
* **lab4.o:** объектный файл модифицированной программы
* **lab4:** исполняемый файл модифицированной программы

# 5.Выводы

Освоен процесс компиляции программ на языке ассемблера NASM от написания кода до получения исполняемого файла

* Изучен pipeline разработки: редактирование → трансляция → компоновка → выполнение
* Практически применены основные директивы и инструкции языка ассемблера:
* Секции данных и кода (.data, .text)
* Объявление данных (DB)
* Системные вызовы через прерывание 80h
* Работа с регистрами процессора