

Лабораторная работа №4

Отчет

Устинова Виктория Вадимовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	11

Список иллюстраций

3.1	Создаем каталоги с помощью команды <code>mkdir</code> и переходим в него с помощью команды <code>cd</code>	7
3.2	Создаем текстовый файл и открываем данный файл в текстовом редакторе.	7
3.3	Открываем файл в текстовом редакторе.	8
3.4	Используем команду <code>nam</code>	8
3.5	Проверяем создание объектного файла при помощи команды <code>ls</code> . . .	8
3.6	Компилируем файл <code>hello.asm</code> в <code>obj.o</code>	8
3.7	Командой <code>ls</code> проверяем выполнение команды.	9
3.8	Выполняем команду <code>ld</code> и сразу же проверяем создание файла. . . .	9
3.9	Создаем файл <code>main</code> командой <code>ld</code> и командой <code>ls</code> проверяем.	9
3.10	Пишем команду <code>./hello</code>	9
3.11	Пишем команду <code>sr</code>	9
3.12	Открываем и редактируем файл уже со своей фамилией и именем. . .	10
3.13	Прописываем команды для работы файла и запускаем программу. . .	10
3.14	Копируем файлы в <code>lab04</code>	10
3.15	Загружаем все файлы на <code>Gitgub</code>	10

Список таблиц

1 Цель работы

Освоить процедуры компиляции и сборки программ, познакомиться с языком ассемблера NASM.

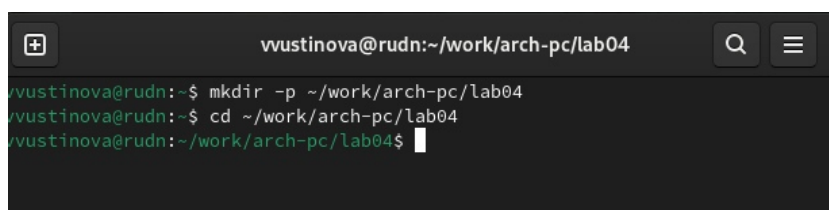
2 Задание

1. Вывести команду “Hello world”.
2. Вывести свое имя и фамилию.

3 Выполнение лабораторной работы

Программа Hello world

Создаем каталог для работы с программами на языке ассемблера NASM.



```
vvustinova@rudn:~/work/arch-pc/lab04
vvustinova@rudn:~$ mkdir -p ~/work/arch-pc/lab04
vvustinova@rudn:~$ cd ~/work/arch-pc/lab04
vvustinova@rudn:~/work/arch-pc/lab04$
```

Рис. 3.1: Создаем каталоги с помощью команды `mkdir` и переходим в него с помощью команды `cd`

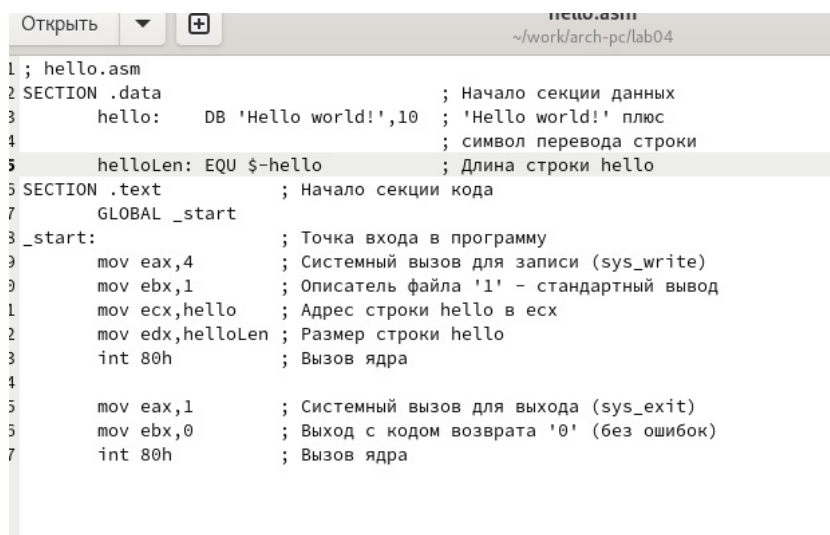
Создаем текстовый файл



```
vvustinova@rudn:~/work/arch-pc/lab04$ touch hello.asm
vvustinova@rudn:~/work/arch-pc/lab04$ gedit hello.asm
```

Рис. 3.2: Создаем текстовый файл и открываем данный файл в текстовом редакторе.

Открыли файл и заполнили его как сказано в примере.

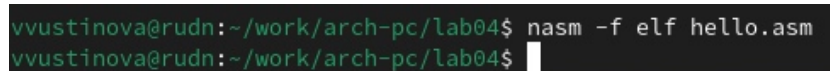


```
1 ; hello.asm
2 SECTION .data                ; Начало секции данных
3     hello:    DB 'Hello world!',10 ; 'Hello world!' плюс
4                                     ; символ перевода строки
5     helloLen: EQU $-hello      ; Длина строки hello
6 SECTION .text                ; Начало секции кода
7     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
15    mov eax,1                ; Системный вызов для выхода (sys_exit)
16    mov ebx,0                ; Выход с кодом возврата '0' (без ошибок)
17    int 80h                  ; Вызов ядра
```

Рис. 3.3: Открываем файл в текстовом редакторе.

Транслятор NASM

Преобразуем текст программы в объектный код.



```
vvustinova@rudn:~/work/arch-pc/lab04$ nasm -f elf hello.asm
vvustinova@rudn:~/work/arch-pc/lab04$
```

Рис. 3.4: Используем команду nasm.

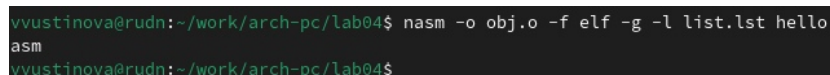


```
vvustinova@rudn:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
vvustinova@rudn:~/work/arch-pc/lab04$
```

Рис. 3.5: Проверяем создание объектного файла при помощи команды ls.

Расширенный синтаксис командной строки NASM.

Компилируем исходный файл.



```
vvustinova@rudn:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello.
asm
vvustinova@rudn:~/work/arch-pc/lab04$
```

Рис. 3.6: Компилируем файл hello.asm в obj.o


```

vvustinova@rudn:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
vvustinova@rudn:~/work/arch-pc/lab04$

```

Рис. 3.7: Командой ls проверяем выполнение команды.

Компановщик ld

Передаем файл на обработку компановщику.

```

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

```

Рис. 3.8: Выполняем команду ld и сразу же проверяем создание файла.

Передаем объектный файл на обработку компановщику.

```

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

```

Рис. 3.9: Создаем файл main командой ld и командой ls проверяем.

Запуск исполняемого файла

Смотрим выполнение команды Hello world!

```

vvustinova@rudn:~/work/arch-pc/lab04$ ./hello
Hello world!
vvustinova@rudn:~/work/arch-pc/lab04$

```

Рис. 3.10: Пишем команду ./hello

Задание для самостоятельной работы

Создаем копию файла hello.asm.

```

vvustinova@rudn:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab04$ cp hello.asm lab4.asm

```

Рис. 3.11: Пишем команду cp

Открываем файл в текстовом редакторе.

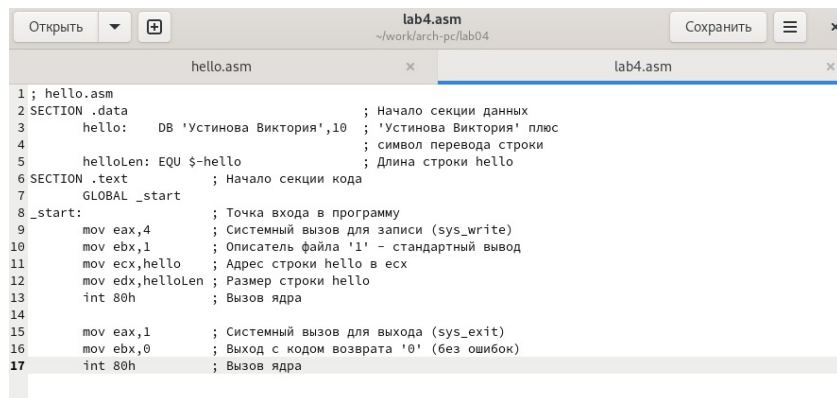


Рис. 3.12: Открываем и редактируем файл уже со своей фамилией и именем.

Используем как пример предыдущие команды.

```

vvustinova@rudn:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
vvustinova@rudn:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst lab4.asm
vvustinova@rudn:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o hello
vvustinova@rudn:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
vvustinova@rudn:~/work/arch-pc/lab04$ ./hello
Устинова Виктория
vvustinova@rudn:~/work/arch-pc/lab04$

```

Рис. 3.13: Прописываем команды для работы файла и запускаем программу.

Копируем файлы в локальный репозиторий.

```

vvustinova@rudn:~/work/arch-pc/lab04$ cp hello.asm ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc2/pc/labs/lab04/
vvustinova@rudn:~/work/arch-pc/lab04$ cp lab4.asm ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc2/pc/labs/lab04/

```

Рис. 3.14: Копируем файлы в lab04.

```

vvustinova@rudn:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git push
Перечисление объектов: 75, готово.
Подсчет объектов: 100% (72/72), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (62/62), готово.
Запись объектов: 100% (62/62), 5.58 МБ | 3.14 МБ/с, готово.
Total 62 (delta 6), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (6/6), completed with 4 local objects.
To github.com:vikauustin/study_2024-2025_arh-pc.git
 e275a2d..8001d12 master -> master

```

Рис. 3.15: Загружаем все файлы на Github.

4 Выводы

Получилось создать две программы, ознакомиться с языком ассемблер.