

Отчёт по лабораторной работе №4

Дисциплина: Архитектура компьютеров

Исупова Кристина Павловна

Содержание

1 Цель работы	3
2 Задание	4
3 Теоретическое введение	5
4 Выполнение лабораторной работы	7
4.1 Программа Hello world!	7
4.2 Транслятор NASM	9
4.3 Расширенный синтаксис командной строки NASM	10
4.4 Компоновщик LD	11
4.5 Запуск исполняемого файла	13
4.6 Задания для самостоятельной работы	14
5 Выводы	18
6 Список литературы	19

1 Цель работы

Цель данной лабораторной работы - освоить процедуры компиляции и сборки программ, написанных на ассемблере NASM.

2 Задание

1. Создание программы Hello world!
2. Работа с транслятором NASM
3. Работа с расширенным синтаксисом командной строки NASM
4. Работа с компоновщиком LD
5. Запуск исполняемого файла
6. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Основными функциональными элементами любой ЭВМ являются центральный процессор, память и периферийные устройства. Взаимодействие этих устройств осуществляется через общую шину, к которой они подключены. Физически шина представляет собой большое количество проводников, соединяющих устройства друг с другом. В современных компьютерах проводники выполнены в виде электропроводящих дорожек на материнской плате. Основной задачей процессора является обработка информации, а также организация координации всех узлов компьютера. В состав центрального процессора входят следующие устройства: - арифметико-логическое устройство (АЛУ) — выполняет логические и арифметические действия, необходимые для обработки информации, хранящейся в памяти; - устройство управления (УУ) — обеспечивает управление и контроль всех устройств компьютера; - регистры

— сверхбыстрая оперативная память небольшого объёма, входящая в состав процессора, для временного хранения промежуточных результатов выполнения инструкций; регистры процессора делятся на два типа: регистры общего назначения и специальные регистры. Для того, чтобы писать программы на ассемблере, необходимо знать, какие регистры процессора существуют и как их можно использовать. Большинство команд в программах написанных на ассемблере используют регистры в качестве операндов. Практически все команды представляют собой преобразование данных хранящихся в регистрах процессора, это например пересылка данных между регистрами или между регистрами и памятью, преобразование (арифметические или логические операции) данных хранящихся в регистрах. Доступ к регистрам осуществляется не по адресам, как

к основной памяти, а по именам. Каждый регистр процессора архитектуры x86 имеет свое название, состоящее из 2 или 3 букв латинского алфавита. В качестве примера приведем названия основных регистров общего назначения (именно эти регистры чаще всего используются при написании программ): - RAX, RCX, RDX, RBX, RSI, RDI — 64-битные - EAX, ECX, EDX, EBX, ESI, EDI — 32-битные - AX, CX, DX, BX, SI, DI — 16-битные - AH, AL, CH, CL, DH, DL, BH, BL — 8-битные

Другим важным узлом ЭВМ является оперативное запоминающее устройство (ОЗУ). ОЗУ — это быстродействующее энергозависимое запоминающее устройство, которое напрямую взаимодействует с узлами процессора, предназначенное для хранения программ и данных, с которыми процессор непосредственно работает в текущий момент. ОЗУ состоит из одинаковых пронумерованных ячеек памяти. Номер ячейки памяти — это адрес хранящихся в ней данных. Периферийные устройства в составе ЭВМ: - устройства внешней памяти, которые предназначены для долговременного хранения больших объёмов данных. - устройства ввода-вывода, которые обеспечивают взаимодействие ЦП с внешней средой.

В основе вычислительного процесса ЭВМ лежит принцип программного управления. Это означает, что компьютер решает поставленную задачу как последовательность действий, записанных в виде программы.

Коды команд представляют собой многоразрядные двоичные комбинации из 0 и 1. В коде машинной команды можно выделить две части: операционную и адресную. В операционной части хранится код команды, которую необходимо выполнить. В адресной части хранятся данные или адреса данных, которые участвуют в выполнении данной операции. При выполнении каждой команды процессор выполняет определённую последовательность стандартных действий, которая называется командным

циклом процессора. Он заключается в следующем: 1. формирование адреса в памяти очередной команды; 2. считывание кода команды из памяти и её дешифрация; 3. выполнение команды; 4. переход к

следующей команде.

Язык ассемблера (assembly language, сокращённо asm) — машинно-ориентированный язык низкого уровня. NASM — это открытый проект ассемблера, версии которого доступны под различные операционные системы и который позволяет получать объектные файлы для этих систем. В NASM используется Intel-синтаксис и поддерживаются инструкции x86-64.

4 Выполнение лабораторной работы

4.1 Программа Hello world!

В домашней директории создаю каталог, в котором буду хранить файлы для текущей лабораторной работы. (Рис 4. 1)

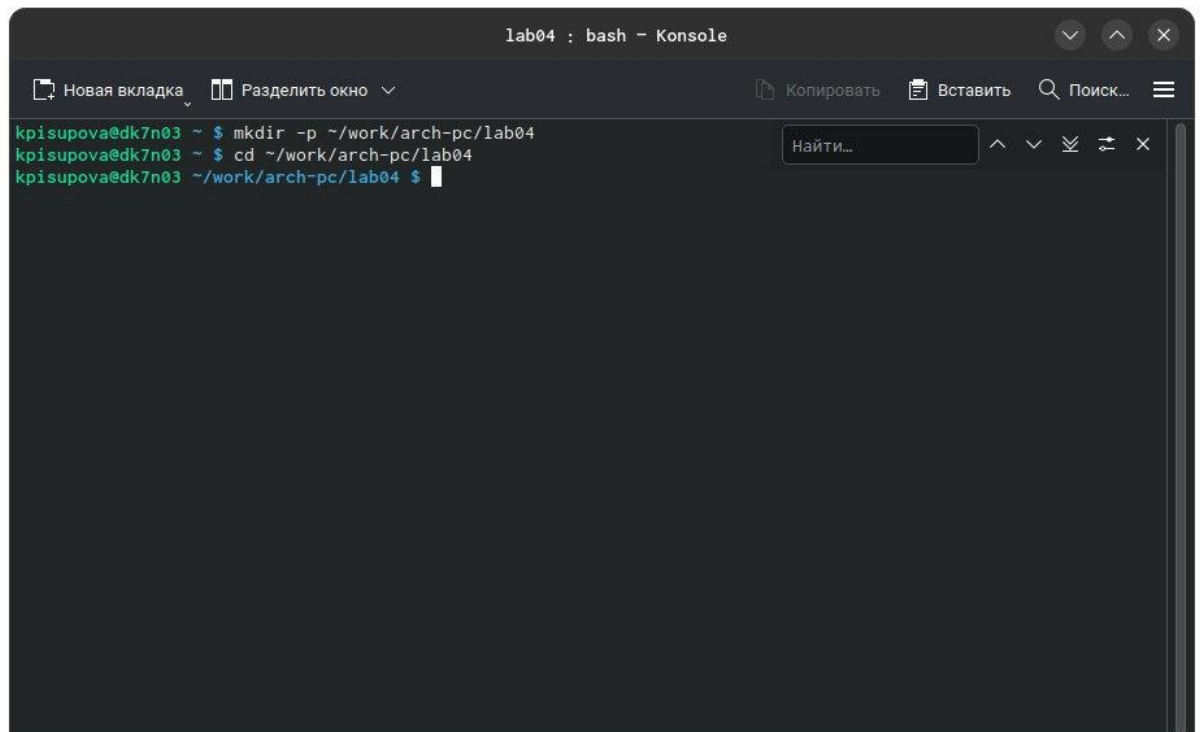
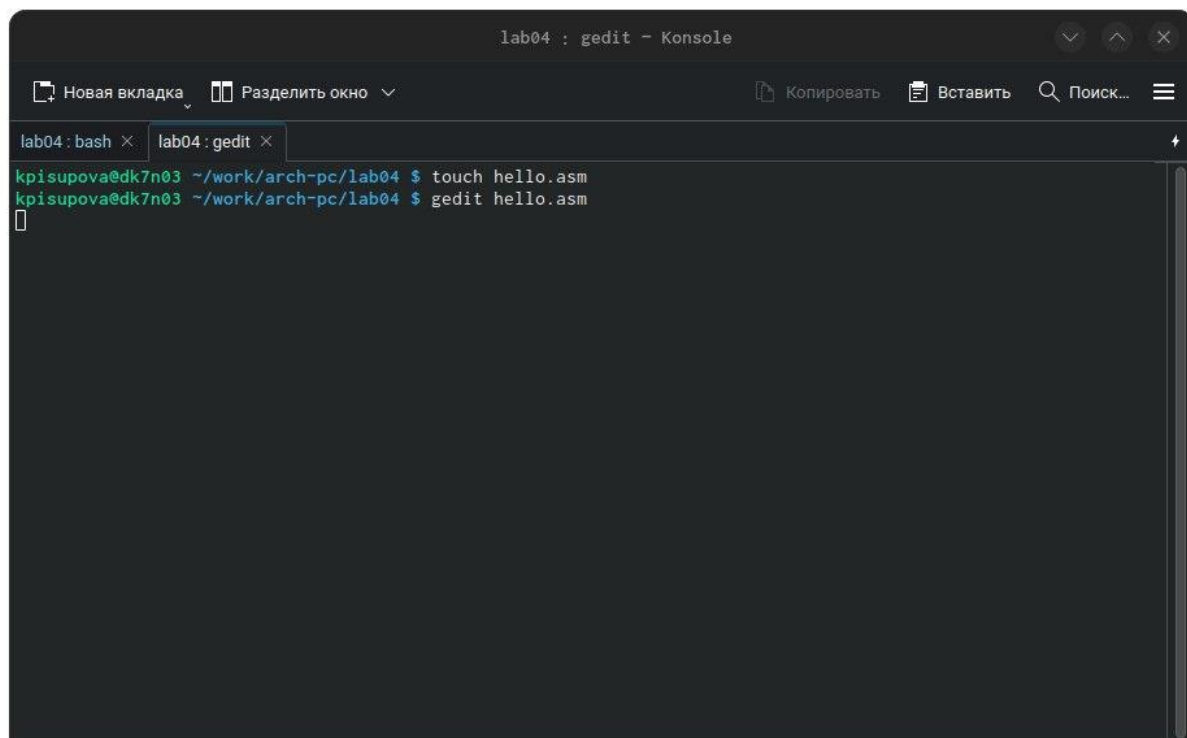


Рис 4. 1 Создание рабочей директории.

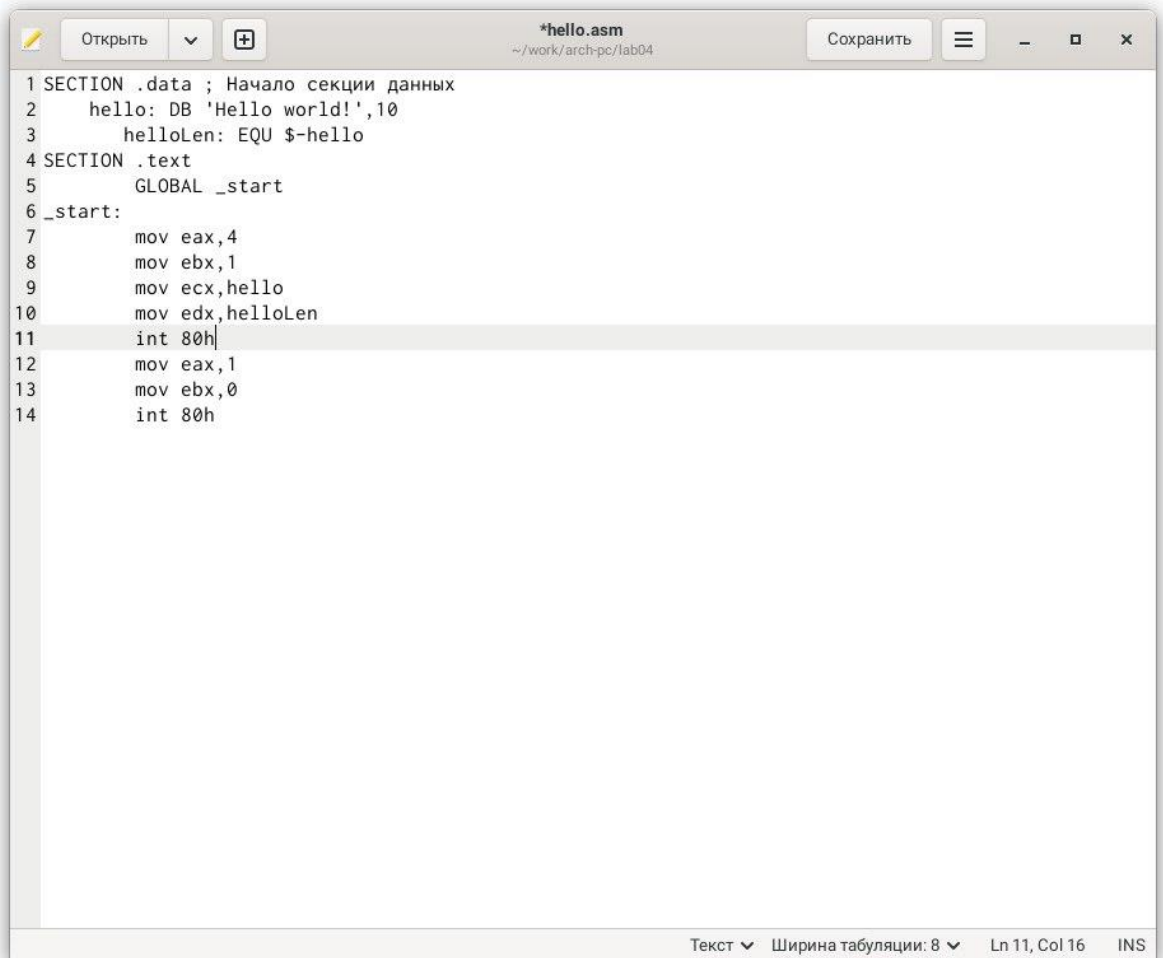
Создаю в нем файл `hello.asm`, в котором буду писать программу на языке ассем- блера. (Рис 4. 2)

A screenshot of a terminal window titled "lab04 : gedit - Konsole". The window has a dark theme. At the top, there is a menu bar with options: "Новая вкладка", "Разделить окно", "Копировать", "Вставить", "Поиск...", and a hamburger menu icon. Below the menu bar, there are two tabs: "lab04 : bash" and "lab04 : gedit". The "lab04 : bash" tab is active, showing a terminal prompt. The terminal text shows the user "kpisupova@dk7n03" in the directory "~/work/arch-pc/lab04" executing the command "touch hello.asm" and then "gedit hello.asm". The "lab04 : gedit" tab is also visible, showing an empty editor window.

```
lab04 : bash × lab04 : gedit ×
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
[]
```

Рис 4. 2 Создание .asm файла

С помощью редактора пишу программу в созданном файле. (Рис 4. 3)



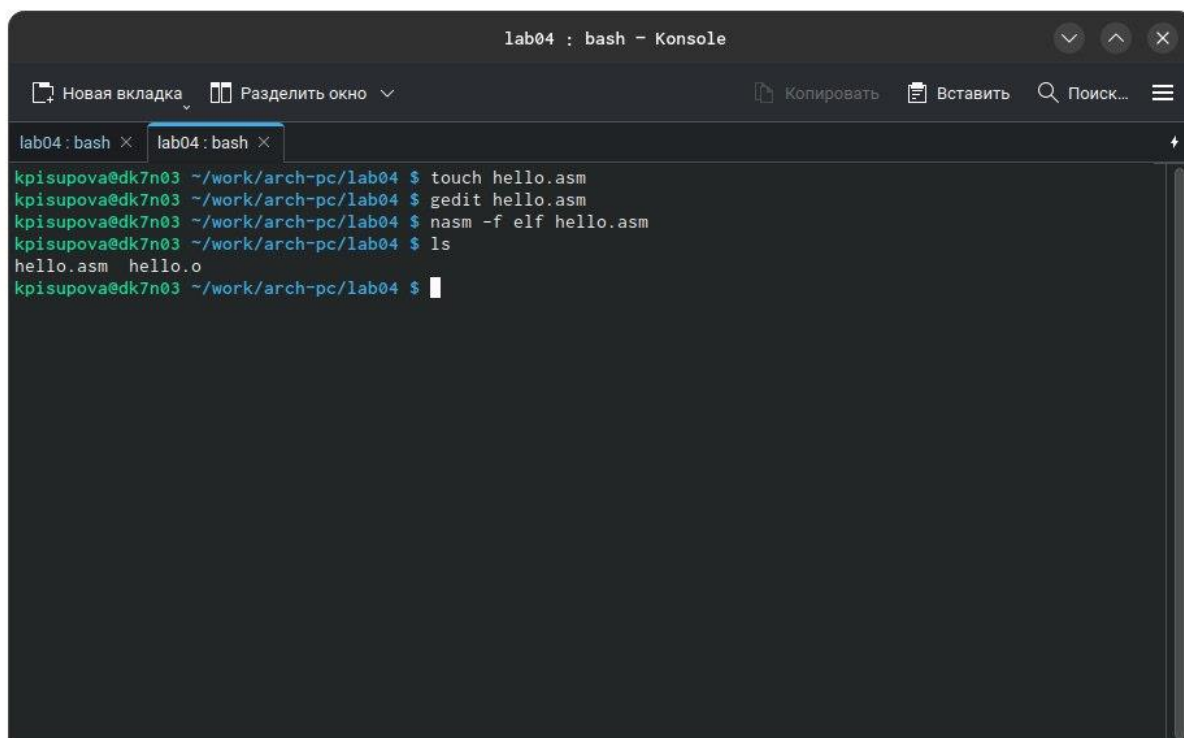
```
1 SECTION .data ; Начало секции данных
2     hello: DB 'Hello world!',10
3     helloLen: EQU $-hello
4 SECTION .text
5     GLOBAL _start
6 _start:
7     mov eax,4
8     mov ebx,1
9     mov ecx,hello
10    mov edx,helloLen
11    int 80h
12    mov eax,1
13    mov ebx,0
14    int 80h
```

Текст ▾ Ширина табуляции: 8 ▾ Ln 11, Col 16 INS

Рис 4. 3 Редактирование файла

4.2 Транслятор NASM

Компилирую с помощью NASM свою программу. (Рис 4. 4)

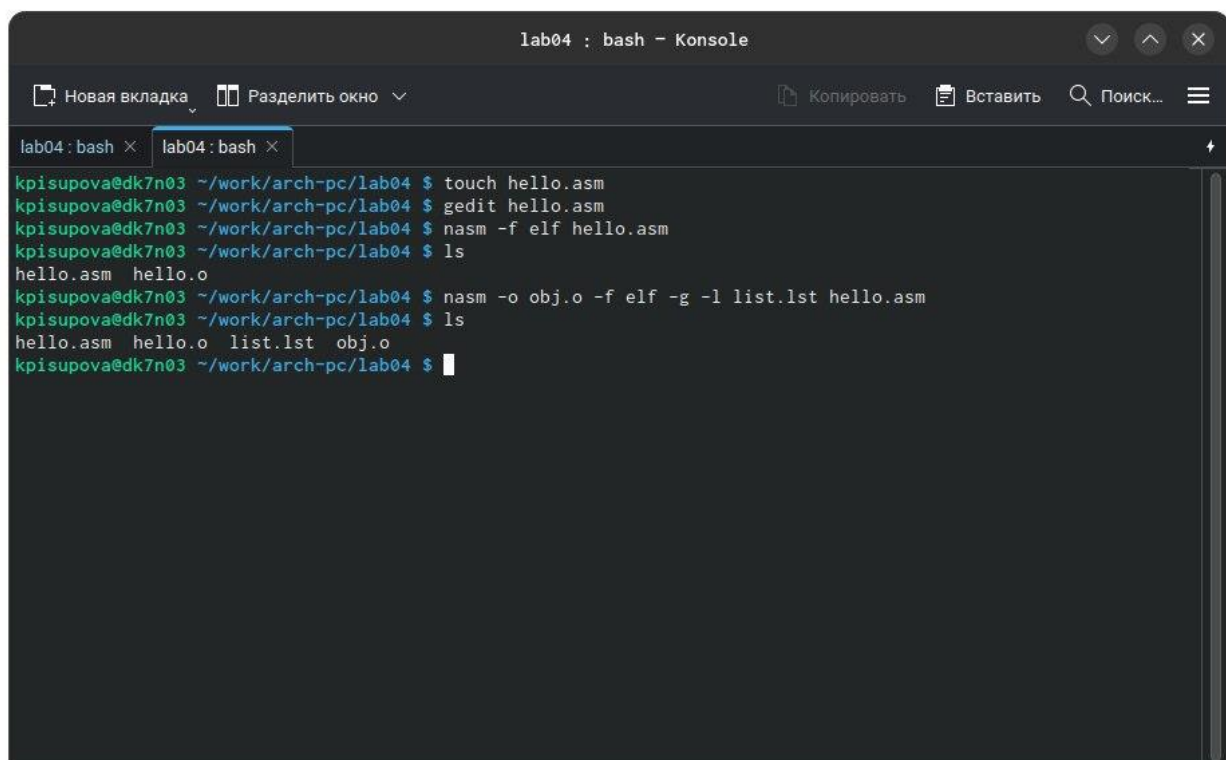


```
lab04 : bash - Konsole
lab04 : bash x lab04 : bash x
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
kpisupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 4 Компиляция программы

4.3 Расширенный синтаксис командной строки NASM

Выполняя команду, указанную на (Рис 4. 5) она скомпилировала исходный файл hello.asm в obj.o, расширение .o говорит о том, что файл - объектный, помимо него флаги -g -l подготовят файл отладки и листинга соответственно.



```
lab04 : bash - Konsole
Новая вкладка  Разделить окно  Копировать  Вставить  Поиск...
lab04 : bash x lab04 : bash x
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
kpisupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o
kpisupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 5 Возможности синтаксиса NASM

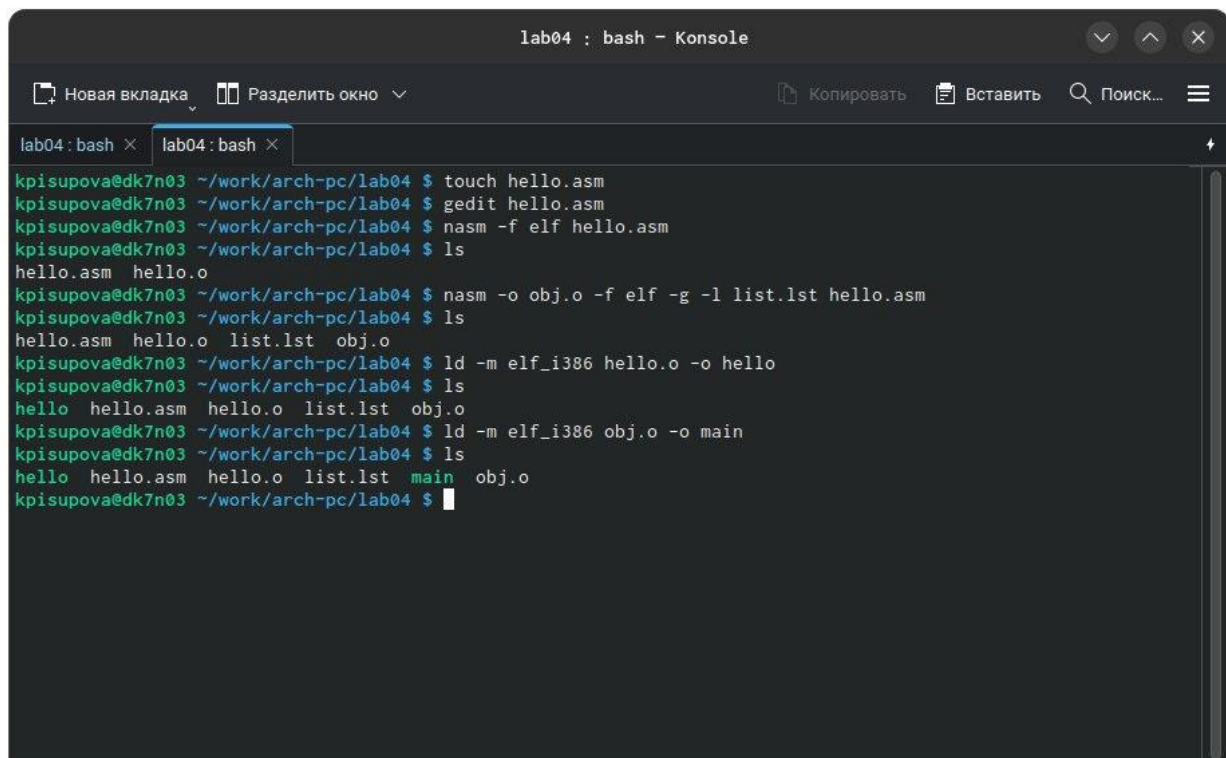
4.4 Компоновщик LD

Затем мне необходимо передать объектный файл компоновщику, делаю это с помощью команды `ld`. (Рис 4. 6)

```
lab04 : bash - Konsole
Новая вкладка Разделить окно
Копировать Вставить Поиск...
lab04 : bash x lab04 : bash x
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 6 Отправка файла компоновщику

Выполняю следующую команду ..., результатом исполнения команды будет созданный файл main, скомпонованный из объектного файла obj.o. (Рис 4. 7)

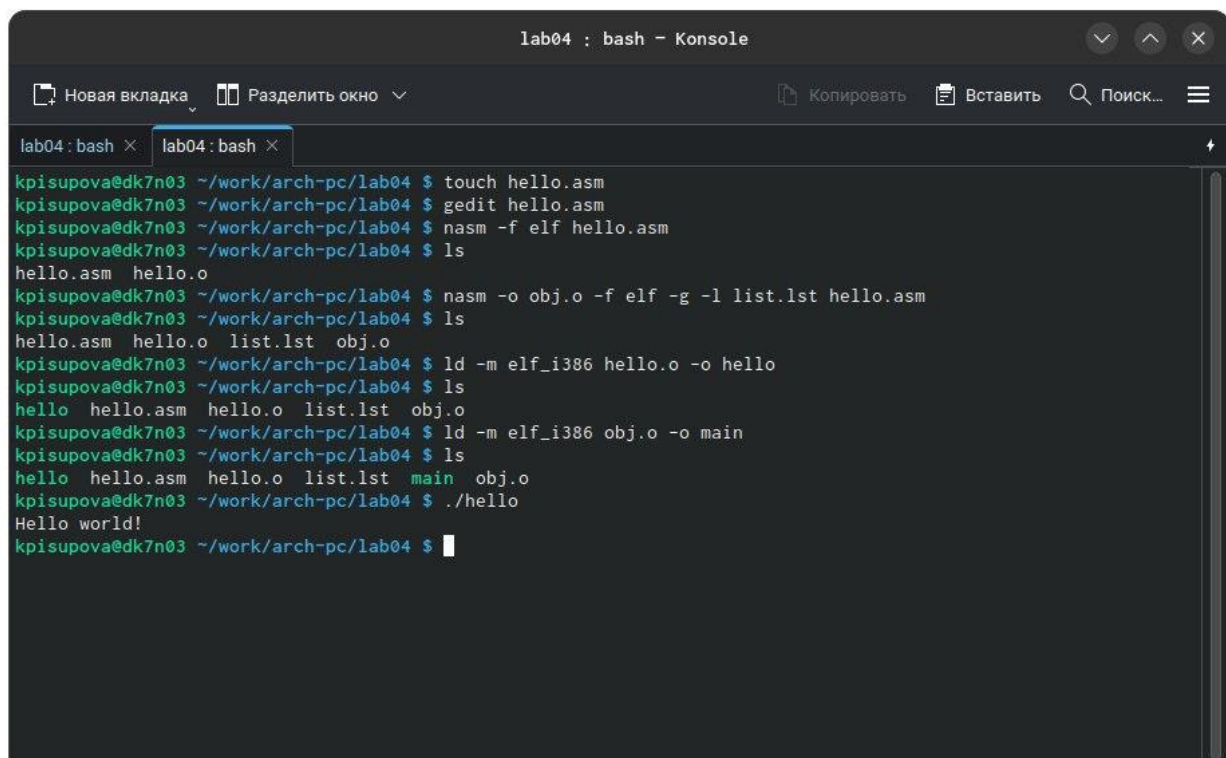


```
lab04 : bash - Konsole
Новая вкладка Разделить окно
Копировать Вставить Поиск...
lab04: bash x lab04: bash x
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst main obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 7 Создание исполняемого файла

4.5 Запуск исполняемого файла

Запускаю исполняемый файл из текущего каталога. (Рис 4. 8)

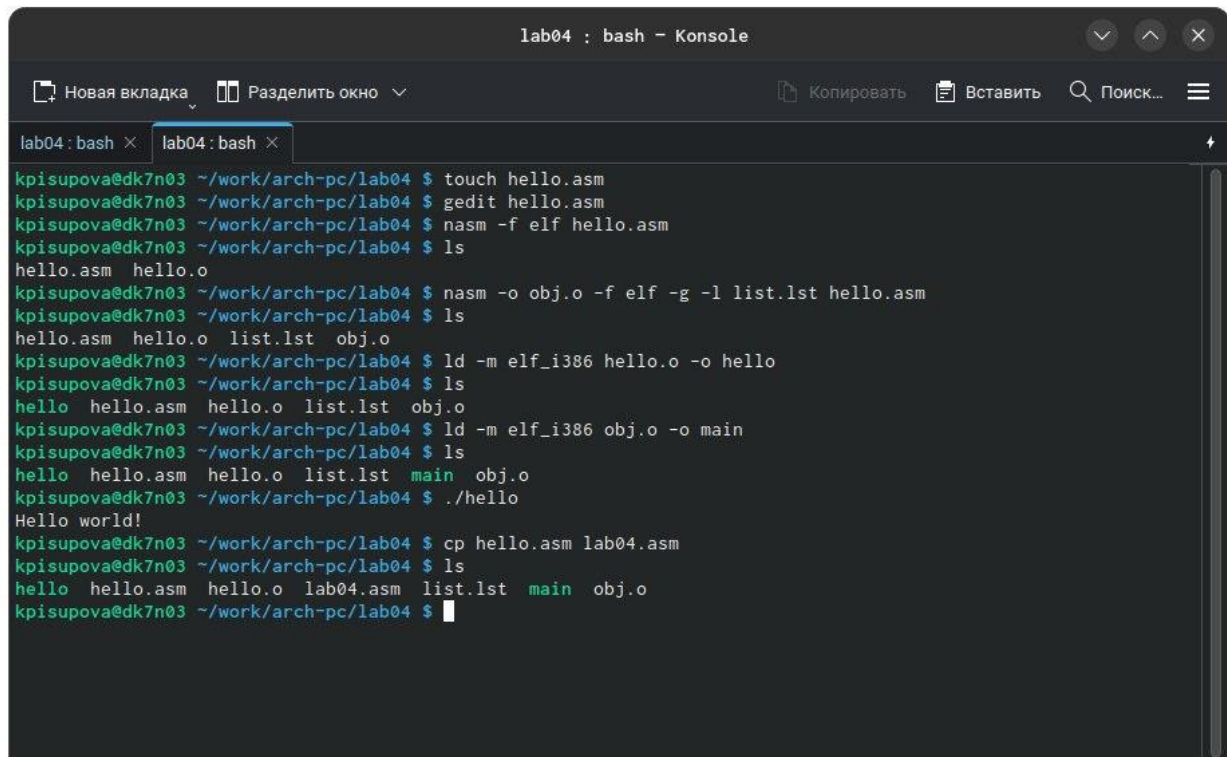


```
lab04 : bash - Konsole
Новая вкладка Разделить окно
Копировать Вставить Поиск...
lab04: bash x lab04: bash x
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst main obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ./hello
Hello world!
kpiupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 8 Запуск программы

4.6 Задания для самостоятельной работы

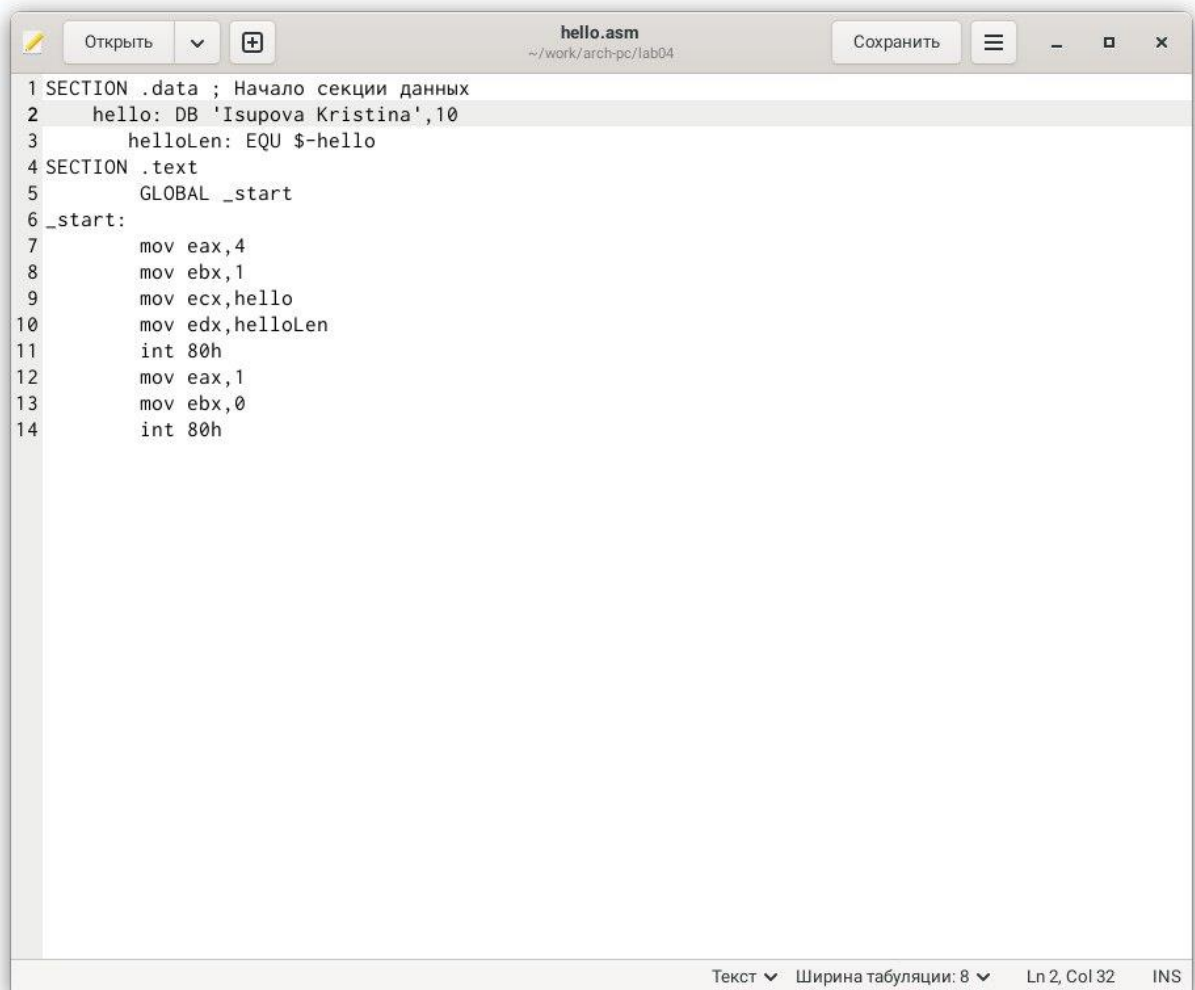
Создаю копию файла для последующей работы с ней. (Рис 4. 9)



```
lab04 : bash - Konsole
[New tab] [Split window] [Copy] [Paste] [Search] [Menu]
lab04: bash x lab04: bash x
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ touch hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ gedit hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -g -l list.lst hello.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o list.lst main obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ./hello
Hello world!
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ cp hello.asm lab04.asm
kpiupova@dk7n03 ~/work/arch-pc/lab04 $ ls
hello hello.asm hello.o lab04.asm list.lst main obj.o
kpiupova@dk7n03 ~/work/arch-pc/lab04 $
```

Рис 4. 9 Создание копии

Редактирую копию файла, заменив текст на свое имя и фамилию. (Рис 4. 10)

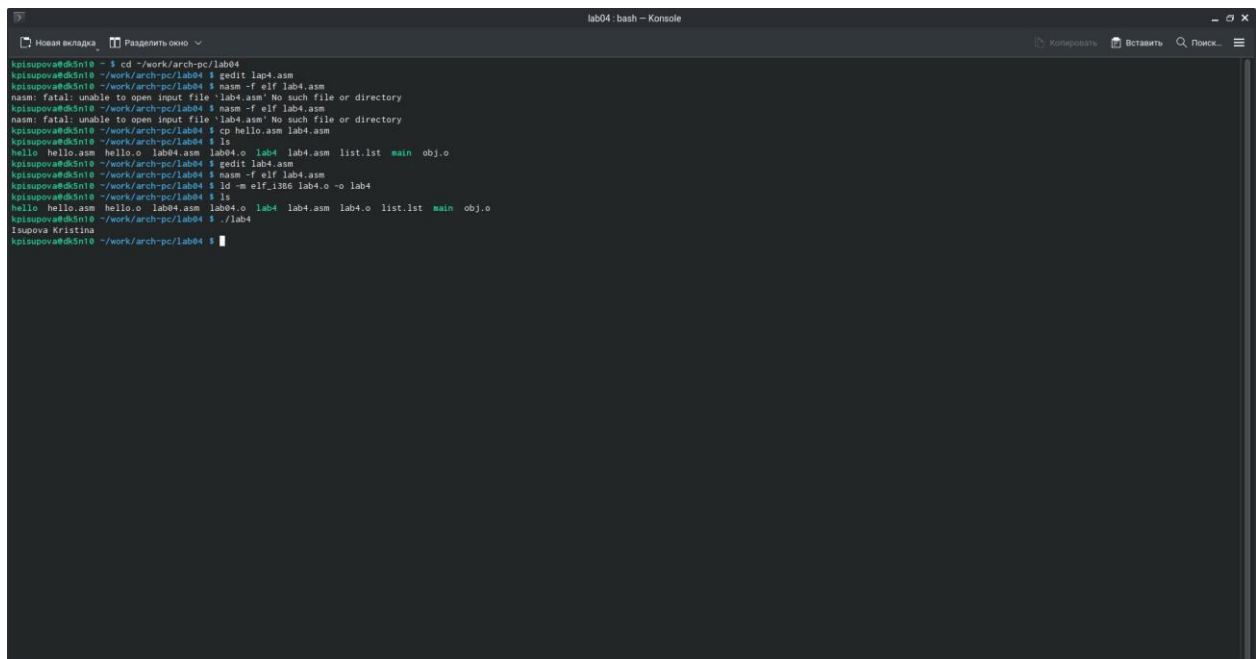


```
1 SECTION .data ; Начало секции данных
2     hello: DB 'Isupova Kristina',10
3         helloLen: EQU $-hello
4 SECTION .text
5     GLOBAL _start
6 _start:
7     mov eax,4
8     mov ebx,1
9     mov ecx,hello
10    mov edx,helloLen
11    int 80h
12    mov eax,1
13    mov ebx,0
14    int 80h
```

Текст ▾ Ширина табуляции: 8 ▾ Ln 2, Col 32 INS

Рис 4. 10 Редактирование копии

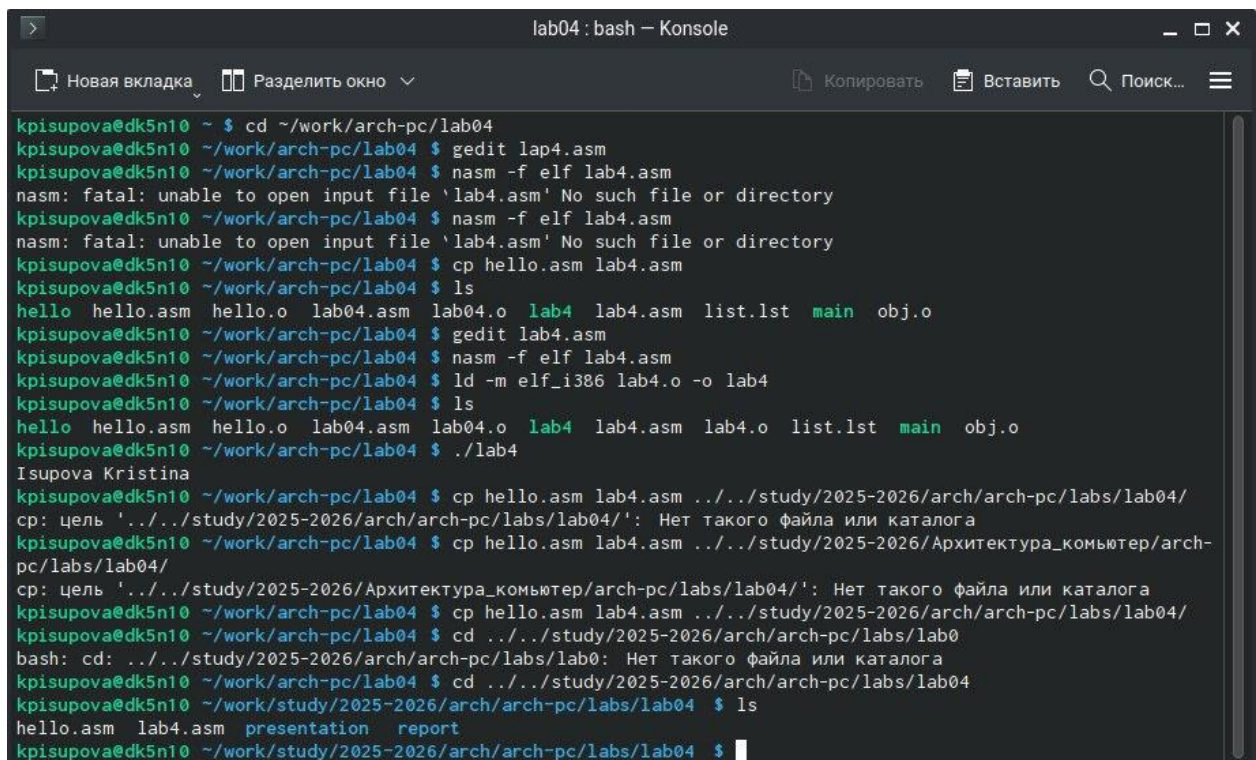
Транслирую копию файла в объектный файл, компоную и запускаю. (Рис 4. 11)



```
lab04: bash - Konsole
krisupova@dk5n10 ~ $ cd ~/work/arch-pc/lab04
krisupova@dk5n10 ~/work/arch-pc/lab04 $ gedit lap4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
nasm: fatal: unable to open input file 'lab4.asm' No such file or directory
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
nasm: fatal: unable to open input file 'lab4.asm' No such file or directory
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab04.asm  lab04.o  lab4  lab4.asm  list.lst  main  obj.o
krisupova@dk5n10 ~/work/arch-pc/lab04 $ gedit lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab04.asm  lab04.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ./lab4
Isupova Kristina
krisupova@dk5n10 ~/work/arch-pc/lab04 $
```

Рис 4. 11 Проверка работоспособности скомпонованной программы

Убедившись в корректности работы программы, копирую рабочие файлы в свой локальный репозиторий.(Рис 4. 12)



```
lab04: bash - Konsole
krisupova@dk5n10 ~ $ cd ~/work/arch-pc/lab04
krisupova@dk5n10 ~/work/arch-pc/lab04 $ gedit lap4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
nasm: fatal: unable to open input file 'lab4.asm' No such file or directory
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
nasm: fatal: unable to open input file 'lab4.asm' No such file or directory
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab04.asm  lab04.o  lab4  lab4.asm  list.lst  main  obj.o
krisupova@dk5n10 ~/work/arch-pc/lab04 $ gedit lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ nasm -f elf lab4.asm
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ld -m elf_i386 lab4.o -o lab4
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  lab04.asm  lab04.o  lab4  lab4.asm  lab4.o  list.lst  main  obj.o
krisupova@dk5n10 ~/work/arch-pc/lab04 $ ./lab4
Isupova Kristina
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm ../../study/2025-2026/arch/arch-pc/labs/lab04/
cp: цель '../../study/2025-2026/arch/arch-pc/labs/lab04/': Нет такого файла или каталога
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm ../../study/2025-2026/Архитектура_комьютер/arch-
pc/labs/lab04/
cp: цель '../../study/2025-2026/Архитектура_комьютер/arch-pc/labs/lab04/': Нет такого файла или каталога
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cp hello.asm lab4.asm ../../study/2025-2026/arch/arch-pc/labs/lab04/
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cd ../../study/2025-2026/arch/arch-pc/labs/lab0
bash: cd: ../../study/2025-2026/arch/arch-pc/labs/lab0: Нет такого файла или каталога
krisupova@dk5n10 ~/work/arch-pc/lab04 $ cd ../../study/2025-2026/arch/arch-pc/labs/lab04
krisupova@dk5n10 ~/work/study/2025-2026/arch/arch-pc/labs/lab04 $ ls
hello.asm  lab4.asm  presentation  report
krisupova@dk5n10 ~/work/study/2025-2026/arch/arch-pc/labs/lab04 $
```

Рис 4. 12 Отправка файлов в локальный репозиторий

Загрузка изменений на свой удаленный репозиторий на GitHub.
(Рис 4. 13)


```
lab04: bash — Konsole
Новая вкладка Разделить окно
Копировать Вставить Поиск...
Уже актуально.
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git push origin master
Перечисление объектов: 8, готово.
Подсчет объектов: 100% (8/8), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (5/5), готово.
Запись объектов: 100% (5/5), 654 байта | 654.00 КиБ/с, готово.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:krisupova/study_2025-2026_arh-pc.git
 c099b88..3080894 master -> master
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git add .
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git status
Текущая ветка: master
Эта ветка соответствует «origin/master».

нечего коммитить, нет изменений в рабочем каталоге
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git commit -m "feat(main):
upload 4 lab work"
Текущая ветка: master
Эта ветка соответствует «origin/master».

нечего коммитить, нет изменений в рабочем каталоге
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git push
Everything up-to-date
krisupova@dk5n10 ~/work/study/2025-2026/ Архитектура компьютера/arch-pc/labs/lab04 $ git commit -m "feat(main):
add files lab04"
>
```

Рис 4. 13 Загрузка изменений

5 Выводы

При выполнении данной лабораторной работы я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

6 Список литературы

Телекоммуникационная учебно-информационная система

<https://esystem.rudn.ru/mod/page/view.php?id=1030505>

<https://esystem.rudn.ru/mod/resource/view.php?id=1030552>

<https://esystem.rudn.ru/course/view.php?id=112#section-13>