

# **Лабораторная работа №4**

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

Ротару Валерия Игоревна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

## **Список иллюстраций**

# Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . . .	7
-----	---	---

# 1 Цель работы

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

## 2 Задание

1. Программа Hello world!
2. Транслятор NASM и его расширенный синтаксис
3. Компоновщик LD и запуск исполняемого файла
4. Самостоятельная работа

## 3 Теоретическое введение

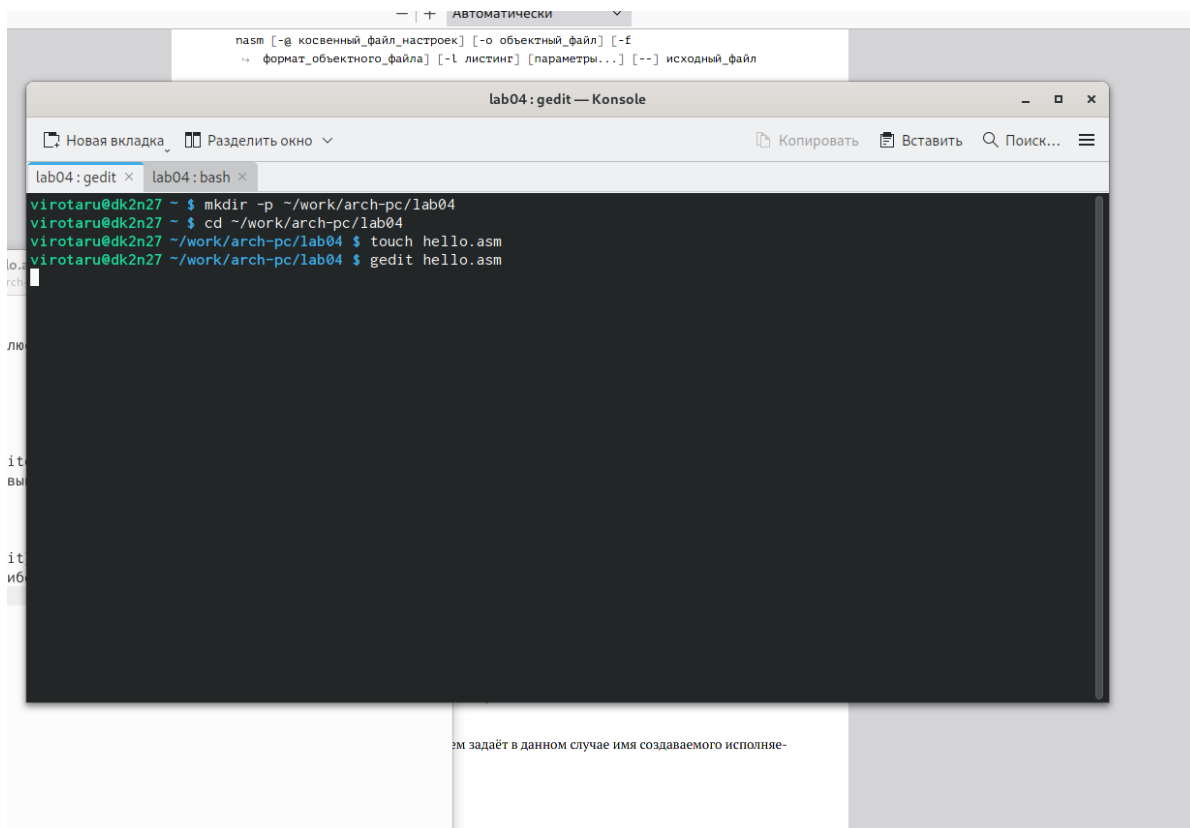
Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux	
Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

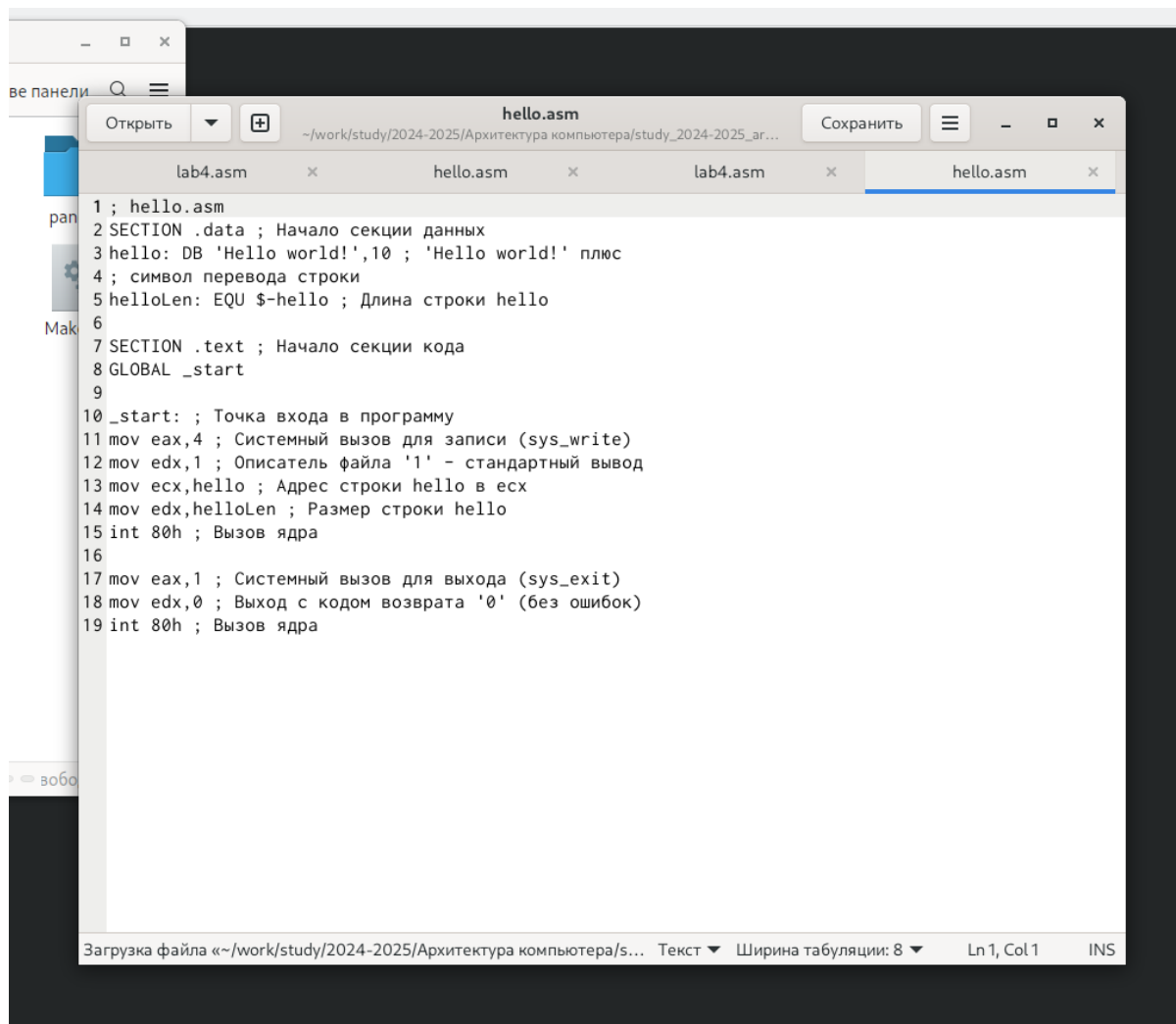
## 4 Выполнение лабораторной работы

### 1. Программа Hello world!(См рис\_1 и рис\_2)



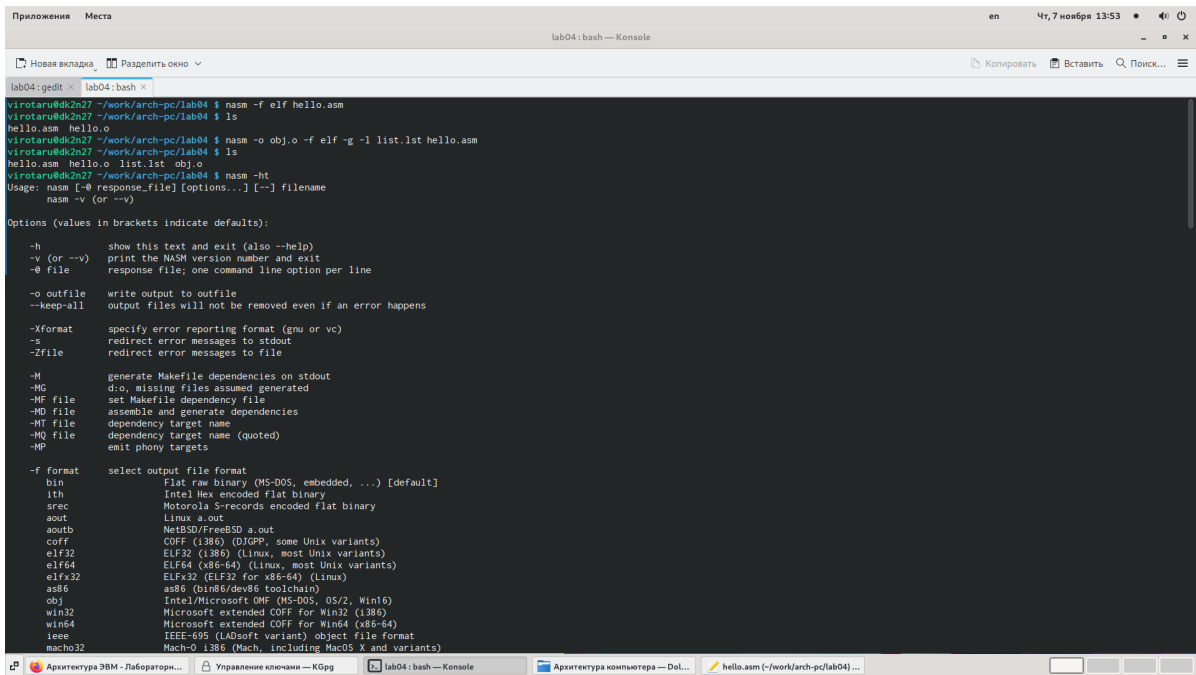
Создание текстового редактора(рис\_1)





Вводим текст(рис\_2)

2. Транслятор NASM и его расширенный синтаксис(См рис\_3)



```
lab04:gedit x lab04:bash x
lab04:bash — Konsole
lab04:gedit x lab04:bash x
virotaru@dk2n27 ~/work/arch-pc/lab04 $ nasm -f elf hello.asm
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o
virotaru@dk2n27 ~/work/arch-pc/lab04 $ nasm -o obj.o -f elf -l list.lst hello.asm
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ls
hello.asm  hello.o  list.lst  obj.o
virotaru@dk2n27 ~/work/arch-pc/lab04 $ nasm -ht
Usage: nasm [-@ response_file] [options...] [-] filename
nasm -v (or --v)

Options (values in brackets indicate defaults):
  -h          show this text and exit (also --help)
  -v (or --v) print the NASM version number and exit
  -@ file     response file; one command line option per line
  -o outfile  write output to outfile
  --keep-all  output files will not be removed even if an error happens
  -Xformat    specify error reporting format (gnu or vc)
  -s          redirect error messages to stdout
  -Zfile      redirect error messages to file
  -M          generate Makefile dependencies on stdout
  -MG         d.o, missing files assumed generated
  -MF file    set Makefile dependency file
  -MD file    assemble and generate dependencies
  -MT file    dependency target name
  -MQ file    dependency target name (quoted)
  -MP         emitphony targets
  -f format   select output file format
    bin      Flat raw binary (MS-DOS, embedded, ...) [default]
    ith      Intel Hex encoded flat binary
    srec     Motorola Srecords encoded flat binary
    aout     Linux a.out
    aoutb    NetBSD/FreeBSD a.out
    coff     COFF (i386) (DJGPP, some Unix variants)
    elf32    ELF32 (i386) (Linux, most Unix variants)
    elf64    ELF64 (x86-64) (Linux, most Unix variants)
    elfx32   ELFx32 (ELF32 for x86-64) (Linux)
    as86     as86 (Gnu86/dev48 toolchain)
    obj      Intel/Microsoft OMF (MS-DOS, OS/2, Win16)
    win32    Microsoft extended COFF for Win32 (i386)
    win64    Microsoft extended COFF for Win64 (x86-64)
    ieee695  IEEE695 (IA64soft variant) object file format
    tee      Mach-O i386 (Mach, including MacOS X and variants)
    macho32  Mach-O i386 (Mach, including MacOS X and variants)
```

Выполнение компиляции в объектный код и исходного файла

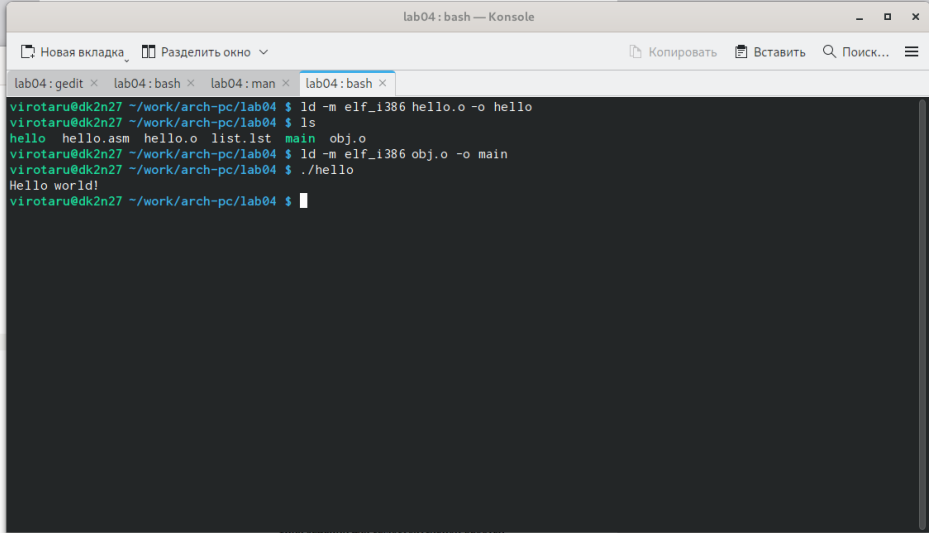
### 3. Компоновщик LD и запуск исполняемого файла(См рис\_4)

#### 4.4.1. Запуск исполняемого файла

Запустить на выполнение созданный исполняемый файл, находящийся в текущем каталоге, можно, набрав в командной строке:

```
./hello
```

#### 4.5. Задание для самостоятельной работы

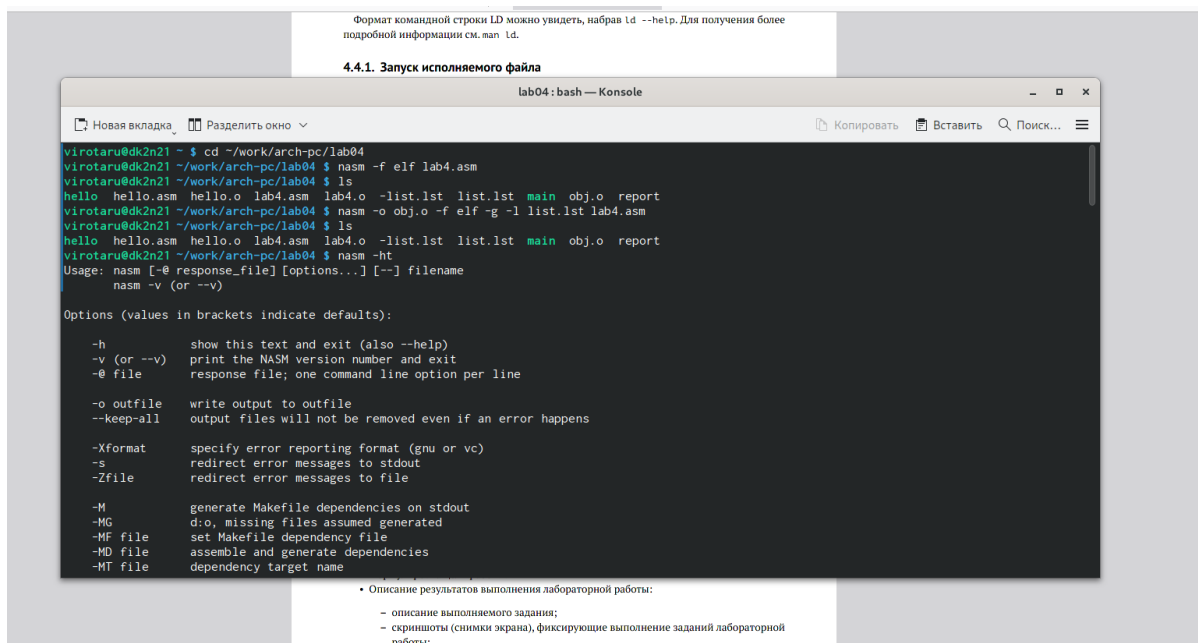


```
lab04:gedit x lab04:bash x lab04:man x lab04:bash x
lab04:bash — Konsole
lab04:gedit x lab04:bash x lab04:man x lab04:bash x
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ld -m elf_i386 hello.o -o hello
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ld -m elf_i386 obj.o -o main
virotaru@dk2n27 ~/work/arch-pc/lab04 $ ./hello
Hello world!
virotaru@dk2n27 ~/work/arch-pc/lab04 $
```

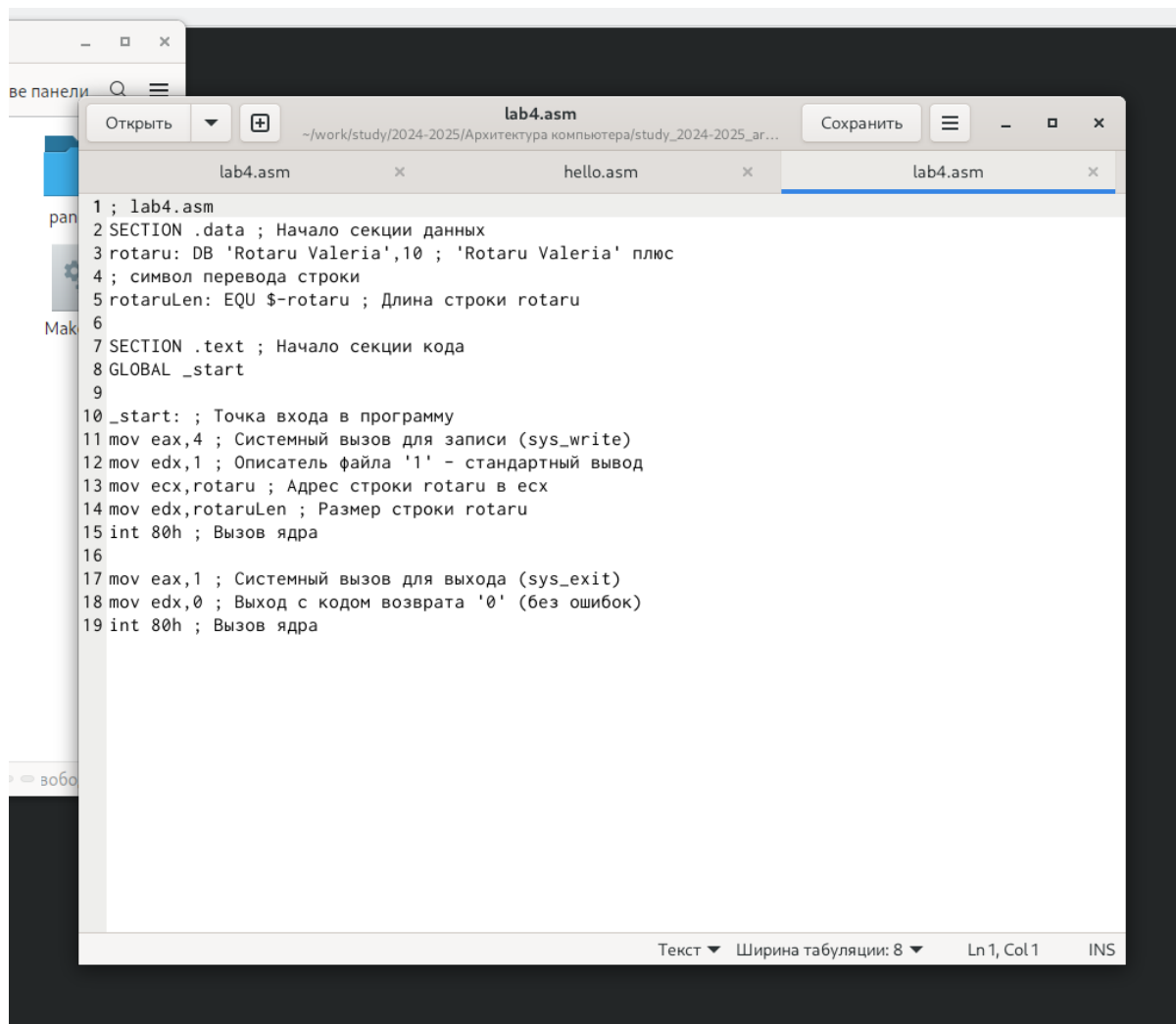
Передача на обработку компоновщику и запуск файла

## 4. Самостоятельная работа(См рис\_5, рис\_6, рис\_7 и рис\_8)

### 4.1. Копия файла с помощью команды ср

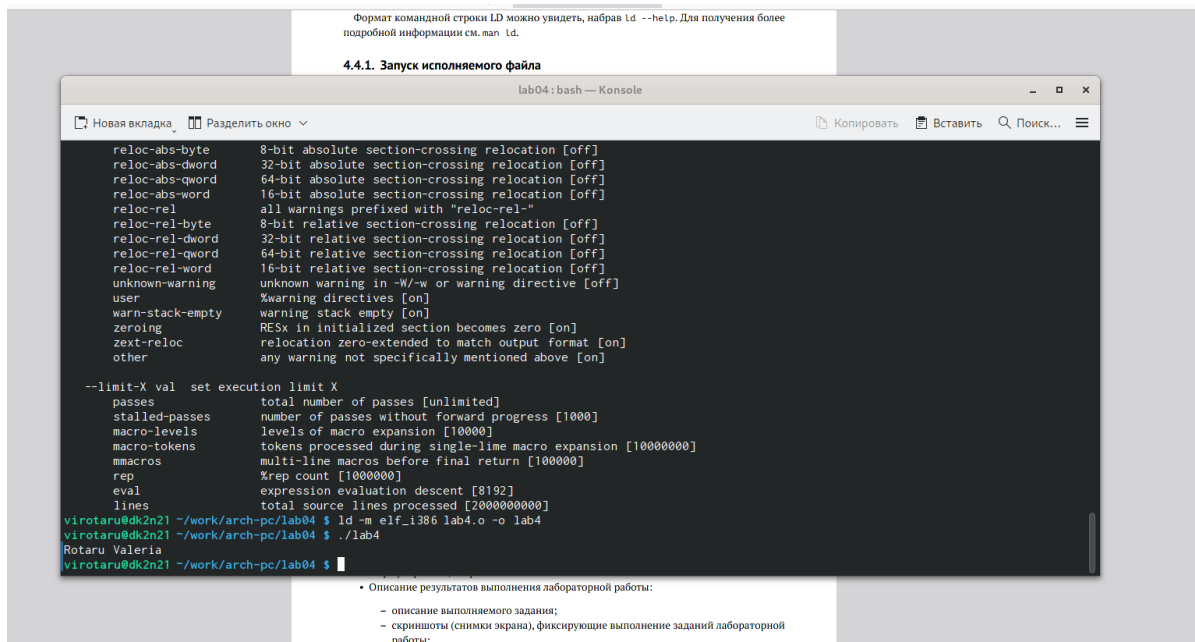


рис\_5 4.2. Внесение изменений в текст программы



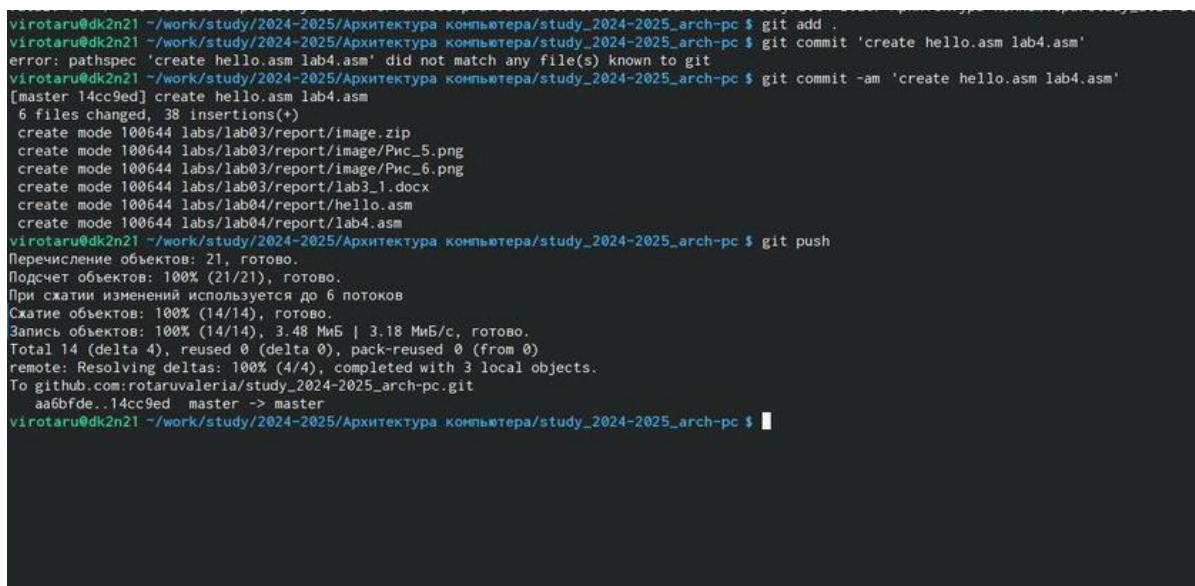
рис\_6

#### 4.3. Запуск получившегося исполняемого файла



рис\_7

#### 4.4. Загрузка на GitHub



рис\_8

## **5 Выводы**

Я освоила процедуры компиляции и сборки программ, написанных на ассемблере NASM.

## Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.
2. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.