

Отчет по выполнению лабораторной работы №4

Дисциплина: Архитектура компьютеров

Воронов Александр Валерьевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Задание для самостоятельной работы	14
6	Выводы	17
	Список литературы	18

Список иллюстраций

4.1	Новый каталог	9
4.2	Новый файл	10
4.3	Открытие файла	10
4.4	Компиляция текста программы	11
4.5	Компиляция в obj.o	11
4.6	Передача файла	12
4.7	Выполнение команды	12
4.8	Выполнение команды	13
5.1	Переход	14
5.2	Копирование	15
5.3	Вывод собственной команды	15
5.4	Копирование файлов в локальный репозиторий	16
5.5	Загрузка на GitHub	16

Список таблиц

1 Цель работы

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

2 Задание

1. Знакомство с теоретической информацией (работа с языком “Ассемблер”)
2. Написание команды ‘Hello world!’
3. Научиться работать с шаблоном, чтобы создавать собственные команды.

3 Теоретическое введение

Ассемблер (англ. «Assembler») — это низкоуровневый язык программирования, который представляет собой промежуточное звено между машинным кодом и высокоуровневыми языками программирования. Он используется для написания программ, которые управляют компьютером или другими устройствами на более низком уровне, непосредственно взаимодействуя с аппаратным обеспечением. Код, написанный на этом языке, обычно сохраняется с помощью расширения ASM.

Директивы

В языке ассемблера директивы — это специальные инструкции. Они используются для предоставления дополнительной информации ассемблеру или компоновщику, а не выполняются как часть программы. Директивы обычно обозначают специальным символом, например точкой или решеткой.

SECTION: эта директива нужна для определения разделов программы, которые используют для группировки связанного кода и данных вместе.

ORG: чтобы установить исходный или начальный адрес программы или раздела.

EQU: чтобы определить константы или символы, которые используют во всей программе.

DB, DW, DD: для определения значений данных байтов, слов или двойных слов в памяти.

ALIGN: для выравнивания ячейки памяти следующей инструкции или значения данных с указанной границей.

EXTERN, GLOBAL: чтобы указать, определяется ли символ внешне или глобально. Эту информацию использует компоновщик для разрешения ссылок на символы в разных объектных файлах.

INCLUDE: для включения файла кода на языке ассемблера в текущую программу.

Директивы помогают управлять структурой и организацией программы на языке ассемблера, указывать дополнительную информацию для создания конечной исполняемой программы.

Команды

Команды языка ассемблера — основные строительные блоки программ. Эти инструкции используют, чтобы сообщить процессору, какие операции следует выполнять. В одних архитектурах сотни или тысячи различных инструкций, в других может быть всего несколько десятков.

Основные:

Команды перемещения данных: Перемещают данные между регистрами или ячейками памяти: MOV, PUSH и POP.

Арифметические команды: Выполняют арифметические операции с данными в регистрах или ячейках памяти: ADD, SUB и MUL.

Логические команды: Выполняют логические операции с данными в регистрах или ячейках памяти: AND, OR и XOR.

Команды ветвления: Управляют путем перехода к другому разделу кода: JMP, JZ и JE.

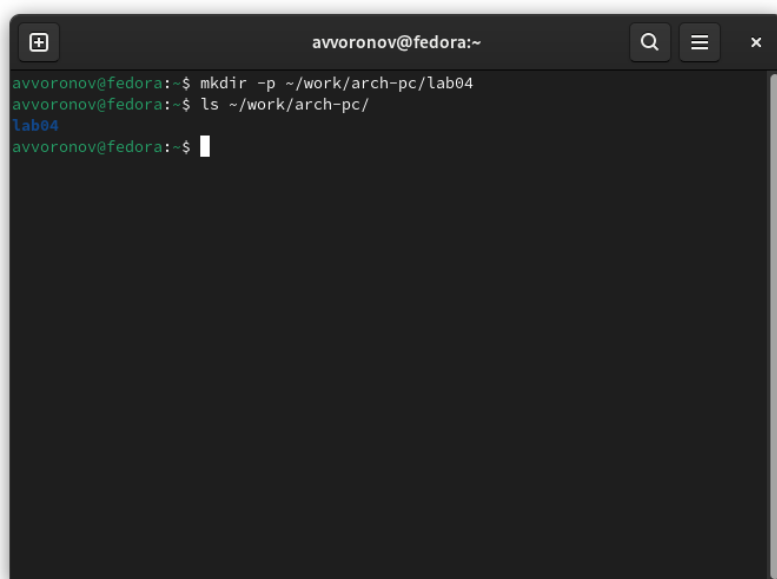
Команды стека: Управляют стеком — областью памяти для хранения данных — и управляющей информацией во время вызовов функций и возвратов: PUSH и POP.

Системные вызовы: Позволяют программам на ассемблере взаимодействовать с операционной системой или другими системными функциями, такими как INT, которые запускают программное прерывание.

4 Выполнение лабораторной работы

Устанавливаю необходимое ПО

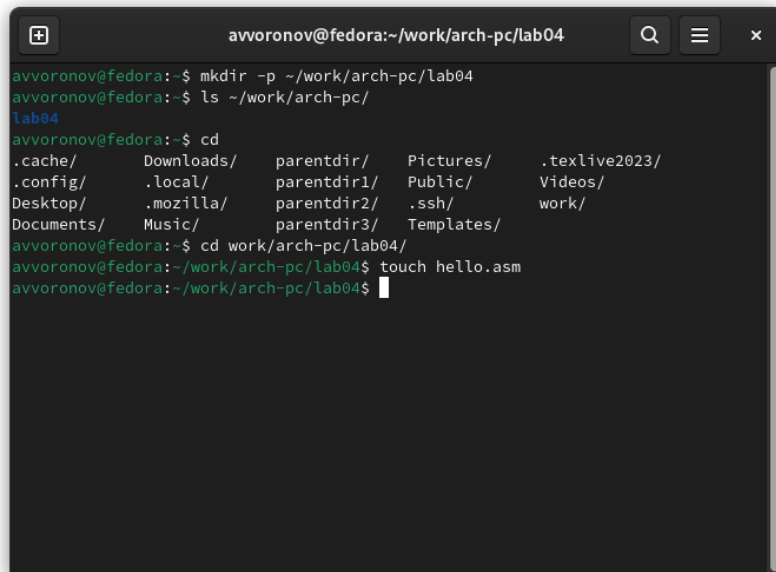
Создаю каталог для работы с программами на языке ассемблера(рис. 4.1)

A terminal window titled 'avvoronov@fedora:~' with search, menu, and close icons. It shows the execution of two commands: 'mkdir -p ~/work/arch-pc/lab04' and 'ls ~/work/arch-pc/'. The output of the second command is 'lab04'.

```
avvoronov@fedora:~  
avvoronov@fedora:~$ mkdir -p ~/work/arch-pc/lab04  
avvoronov@fedora:~$ ls ~/work/arch-pc/  
lab04  
avvoronov@fedora:~$
```

Рис. 4.1: Новый каталог

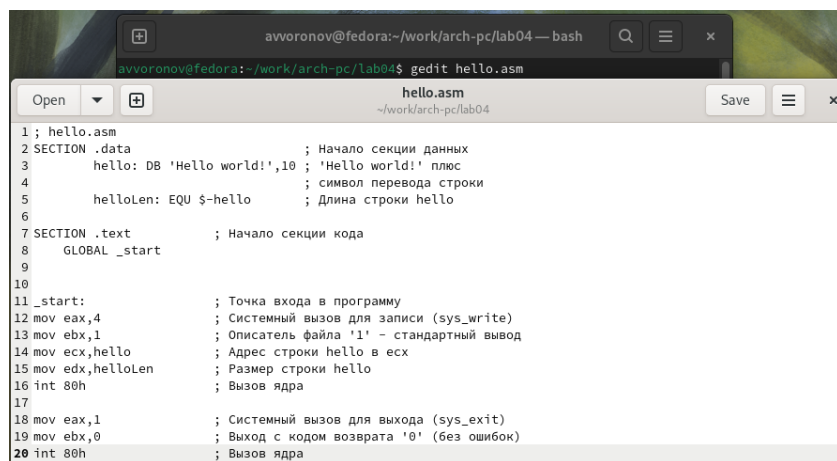
Перехожу в созданный каталог и создаю текстовый файл с именем hello.asm (рис. 4.2)



```
avvoronov@fedora:~/work/arch-pc/lab04
avvoronov@fedora:~$ mkdir -p ~/work/arch-pc/lab04
avvoronov@fedora:~$ ls ~/work/arch-pc/
lab04
avvoronov@fedora:~$ cd
.cache/      Downloads/  parentdir/  Pictures/    .texlive2023/
.config/     .local/    parentdir1/  Public/      Videos/
Desktop/     .mozilla/  parentdir2/  .ssh/        work/
Documents/   Music/     parentdir3/  Templates/
avvoronov@fedora:~$ cd work/arch-pc/lab04/
avvoronov@fedora:~/work/arch-pc/lab04$ touch hello.asm
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 4.2: Новый файл

Открываю созданный файл (рис. 4.3)

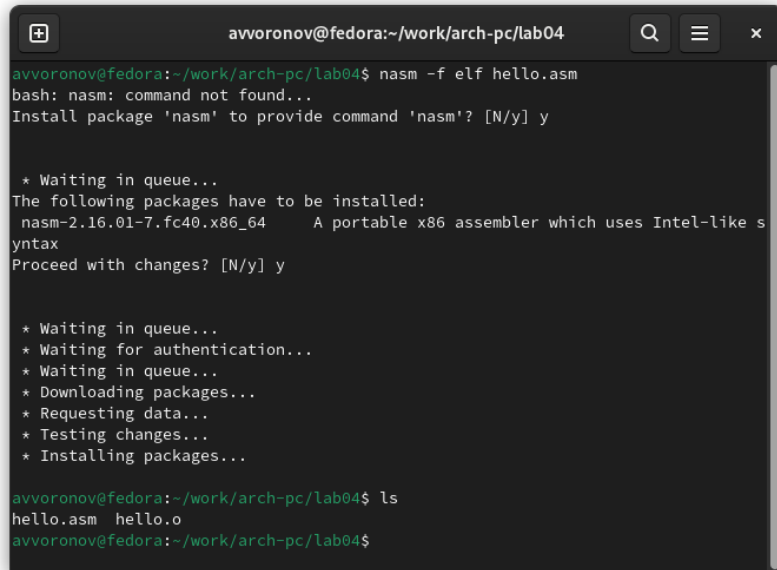


```
avvoronov@fedora:~/work/arch-pc/lab04$ gedit hello.asm
hello.asm
~/work/arch-pc/lab04
Save

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
7 SECTION .text          ; Начало секции кода
8     GLOBAL _start
9
10
11 _start:                ; Точка входа в программу
12 mov eax,4              ; Системный вызов для записи (sys_write)
13 mov ebx,1              ; Описатель файла '1' - стандартный вывод
14 mov ecx,hello           ; Адрес строки hello в ехх
15 mov edx,helloLen        ; Размер строки hello
16 int 80h                ; Вызов ядра
17
18 mov eax,1              ; Системный вызов для выхода (sys_exit)
19 mov ebx,0              ; Выход с кодом возврата '0' (без ошибок)
20 int 80h                ; Вызов ядра
```

Рис. 4.3: Открытие файла

Для компиляции приведённого выше текста программы «Hello World» использую команду (рис. 4.4)



```
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -f elf hello.asm
bash: nasm: command not found...
Install package 'nasm' to provide command 'nasm'? [N/y] y

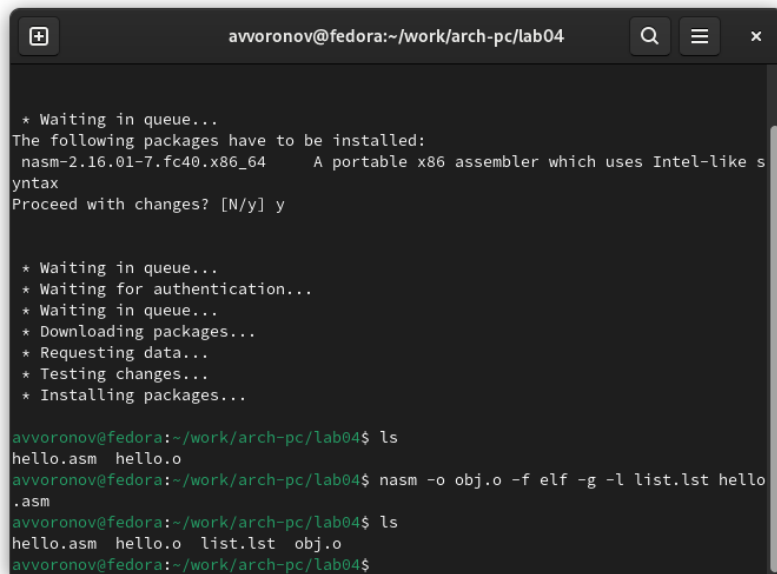
* Waiting in queue...
The following packages have to be installed:
nasm-2.16.01-7.fc40.x86_64    A portable x86 assembler which uses Intel-like s
yntax
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

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

Рис. 4.4: Компиляция текста программы

Компилирую исходный файл hello.asm в obj.o (рис. 4.5)



```
* Waiting in queue...
The following packages have to be installed:
nasm-2.16.01-7.fc40.x86_64    A portable x86 assembler which uses Intel-like s
yntax
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

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

Рис. 4.5: Компиляция в obj.o

Передача объектного файла на работу компановщику (рис. 4.6)

```
avvoronov@fedora:~/work/arch-pc/lab04
The following packages have to be installed:
nasm-2.16.01-7.fc40.x86_64    A portable x86 assembler which uses Intel-like s
yntax
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello
.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 4.6: Передача файла

Выполняю еще одну команду (рис. 4.7)

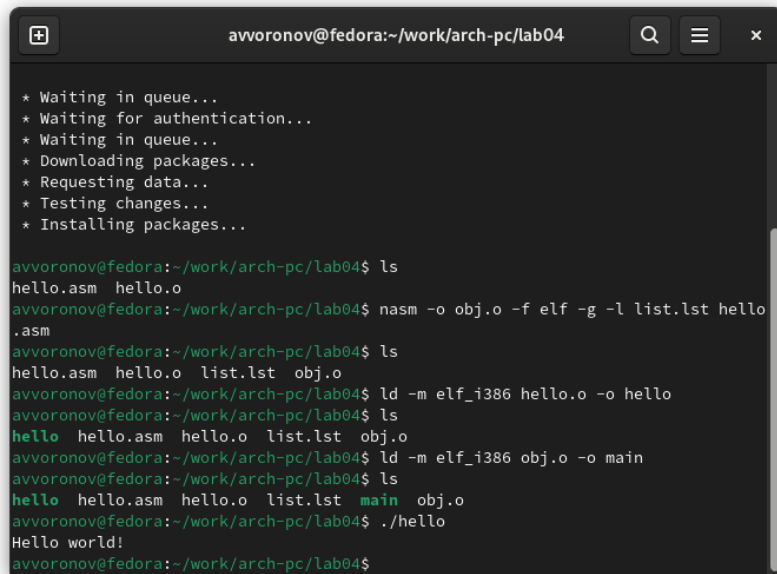
```
avvoronov@fedora:~/work/arch-pc/lab04
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello
.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 4.7: Выполнение команды

Запускаю на выполнение созданный файл (рис. 4.8)



```
avvoronov@fedora:~/work/arch-pc/lab04

* Waiting in queue...
* Waiting for authentication...
* Waiting in queue...
* Downloading packages...
* Requesting data...
* Testing changes...
* Installing packages...

avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -o obj.o -f elf -g -l list.lst hello
.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 hello.o -o hello
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 obj.o -o main
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello  hello.asm  hello.o  list.lst  main  obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ./hello
Hello world!
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 4.8: Выполнение команды

5 Задание для самостоятельной работы

Перехожу в нужный каталог (рис. 5.1)

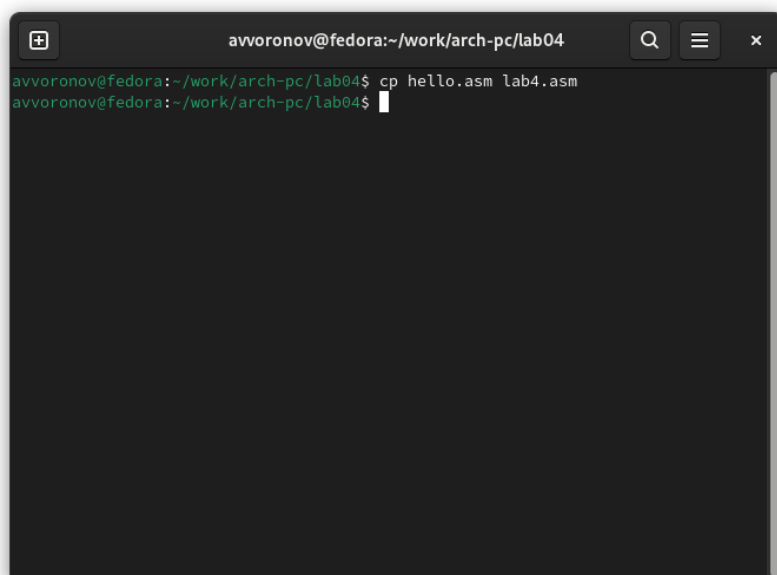
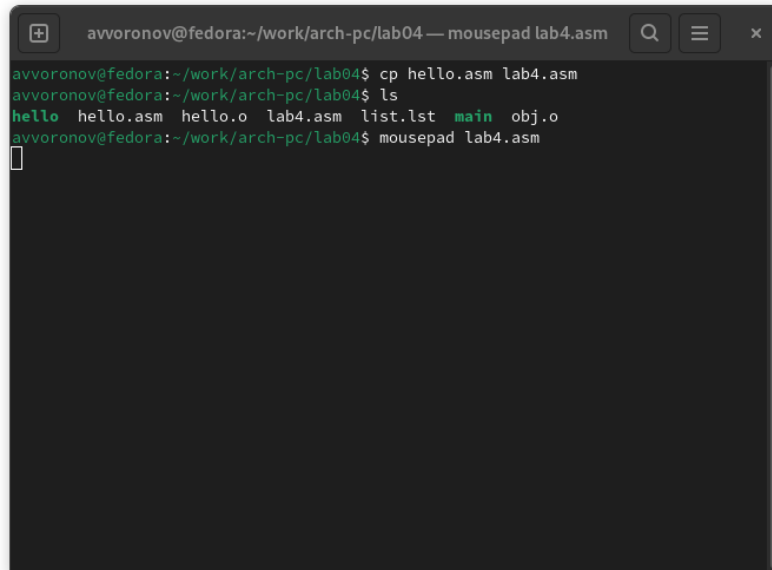


Рис. 5.1: Переход

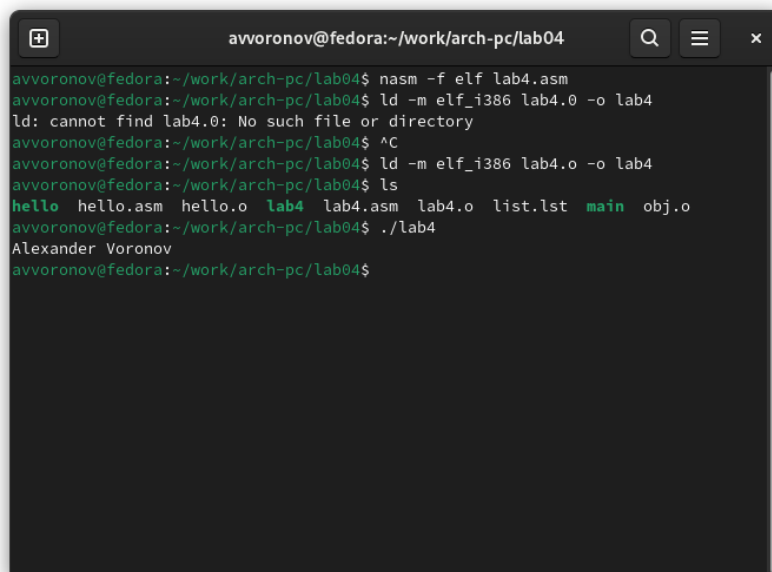
Создаю копию файла hello.asm с именем lab4.asm и начинаю его редактировать (рис. 5.2)

A terminal window titled 'avvoronov@fedora:~/work/arch-pc/lab04 — mousepad lab4.asm'. The terminal shows the following commands and output:

```
avvoronov@fedora:~/work/arch-pc/lab04$ cp hello.asm lab4.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4.asm list.lst main obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ mousepad lab4.asm
```

Рис. 5.2: Копирование

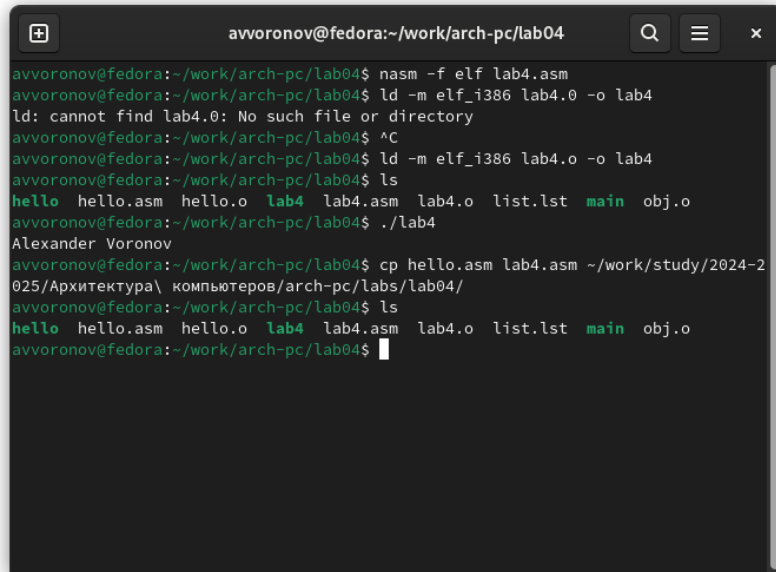
С помощью любого текстового редактора вношу изменения в текст программы в файле lab4 и вывожу полученный результат (рис. 5.3)

A terminal window titled 'avvoronov@fedora:~/work/arch-pc/lab04'. The terminal shows the following commands and output:

```
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
ld: cannot find lab4.o: No such file or directory
avvoronov@fedora:~/work/arch-pc/lab04$ ^C
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ./lab4
Alexander Voronov
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 5.3: Вывод собственной команды

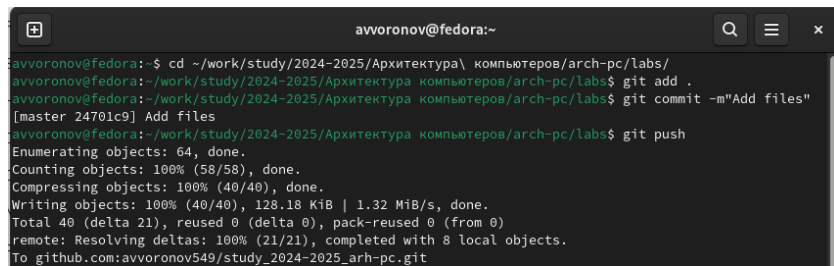
Копирую файлы в свой локальный репозиторий (рис. 5.4)

A terminal window titled 'avvoronov@fedora:~/work/arch-pc/lab04'. The user runs 'nasm -f elf lab4.asm', then 'ld -m elf_i386 lab4.o -o lab4', which fails with 'ld: cannot find lab4.o: No such file or directory'. After pressing '^C', they run 'ld -m elf_i386 lab4.o -o lab4' successfully. Then they run 'ls' showing 'hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o'. Finally, they run './lab4' and the output 'Alexander Voronov' is displayed.

```
avvoronov@fedora:~/work/arch-pc/lab04$ nasm -f elf lab4.asm
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
ld: cannot find lab4.o: No such file or directory
avvoronov@fedora:~/work/arch-pc/lab04$ ^C
avvoronov@fedora:~/work/arch-pc/lab04$ ld -m elf_i386 lab4.o -o lab4
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
avvoronov@fedora:~/work/arch-pc/lab04$ ./lab4
Alexander Voronov
avvoronov@fedora:~/work/arch-pc/lab04$ cp hello.asm lab4.asm ~/work/study/2024-2
025/Архитектура\ компьютеров/arch-pc/labs/lab04/
avvoronov@fedora:~/work/arch-pc/lab04$ ls
hello hello.asm hello.o lab4 lab4.asm lab4.o list.lst main obj.o
avvoronov@fedora:~/work/arch-pc/lab04$
```

Рис. 5.4: Копирование файлов в локальный репозиторий

Провожу загрузку на GitHub (рис. 5.5)

A terminal window titled 'avvoronov@fedora:~'. The user navigates to the directory '~/work/study/2024-2025/Архитектура\ компьютеров/arch-pc/labs/' and runs 'git add .' and 'git commit -m "Add files"'. Then they run 'git push', which shows progress for enumerating, counting, compressing, and writing objects, and finally pushing to the remote repository 'github.com:avvoronov549/study_2024-2025_arh-pc.git'.

```
avvoronov@fedora:~$ cd ~/work/study/2024-2025/Архитектура\ компьютеров/arch-pc/labs/
avvoronov@fedora:~/work/study/2024-2025/Архитектура\ компьютеров/arch-pc/labs$ git add .
avvoronov@fedora:~/work/study/2024-2025/Архитектура\ компьютеров/arch-pc/labs$ git commit -m "Add files"
[master 24701c9] Add files
avvoronov@fedora:~/work/study/2024-2025/Архитектура\ компьютеров/arch-pc/labs$ git push
Enumerating objects: 64, done.
Counting objects: 100% (58/58), done.
Compressing objects: 100% (40/40), done.
Writing objects: 100% (40/40), 128.18 KiB | 1.32 MiB/s, done.
Total 40 (delta 21), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (21/21), completed with 8 local objects.
To github.com:avvoronov549/study_2024-2025_arh-pc.git
```

Рис. 5.5: Загрузка на GitHub

6 Выводы

Данная лабораторная работа позволила мне познакомиться с таким языком как Ассемблер. Теперь я знаю как писать простейшие команды на данном языке через консоль. В заключение хочется отметить, что приведенный язык и не считается популярным, я все же убеждаюсь в его важности.

Список литературы

1. Ассемблер: что это за язык программирования, для чего нужен, пример кода
2. Архитектура ЭВМ
3. Что такое язык ассемблера и кому его нужно изучать/Skillbox Media