

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2
дисциплины «Основы программной инженерии»

Выполнила:
Панюкова Ксения Юрьевна
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Ход работы

1. Я изучила теоретический материал работы

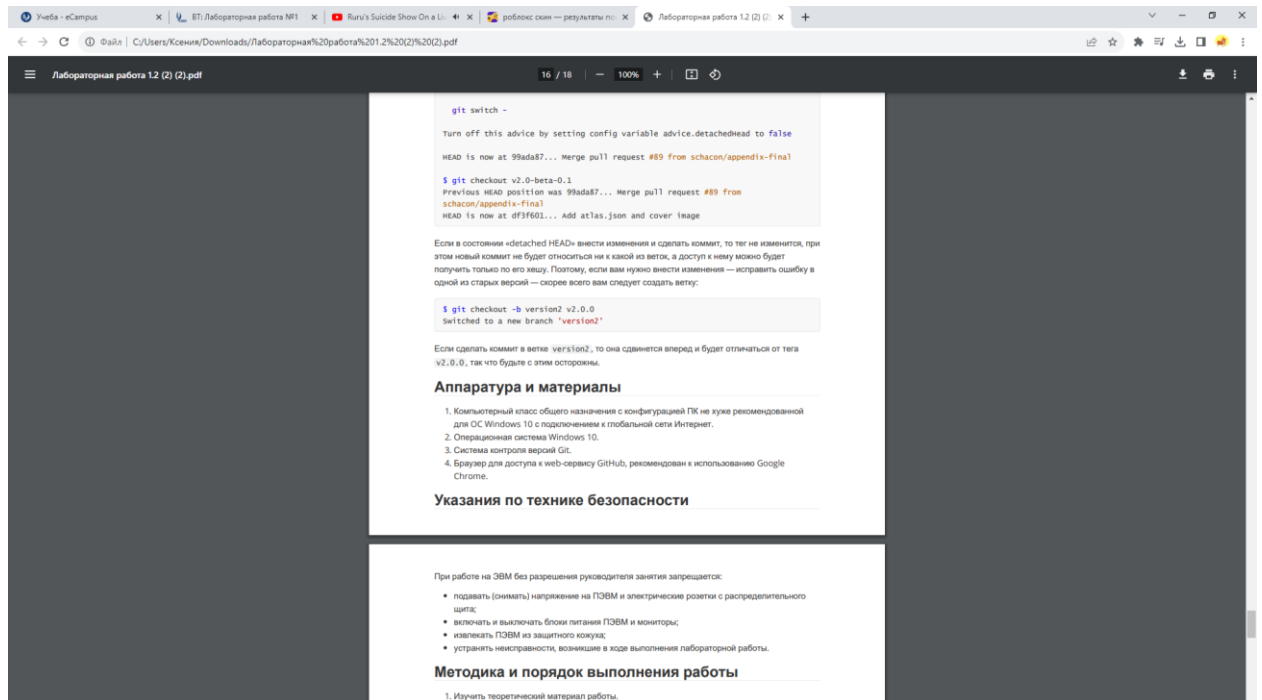


Рисунок 1.1 – Изучение материала для лабораторной работы

2. Создала общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный мною язык программирования

Create a new repository



A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *	Repository name *
<div>MrFinicheck ▾</div>	<div>Whatiam</div>
✔ Whatiam is available.	

Great repository names are short and memorable. Need inspiration? How about **stunning-sniffle** ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository

Рисунок 2.1 – Настройка репозитория

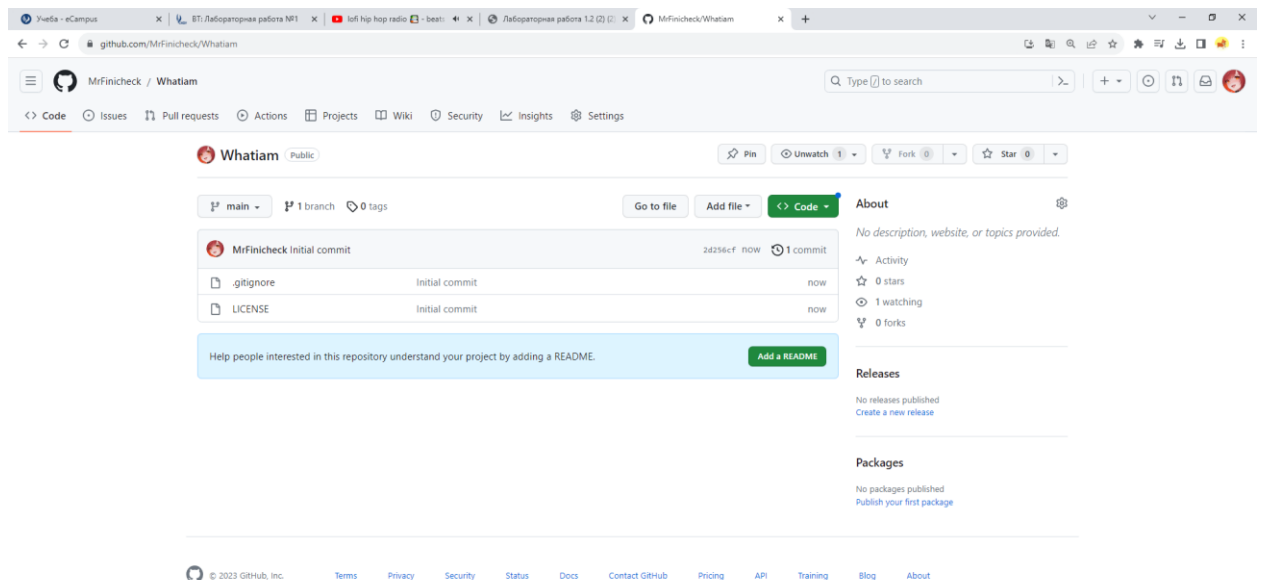


Рисунок 2.2 – Готовый репозиторий

3. Проработала примеры лабораторной работы

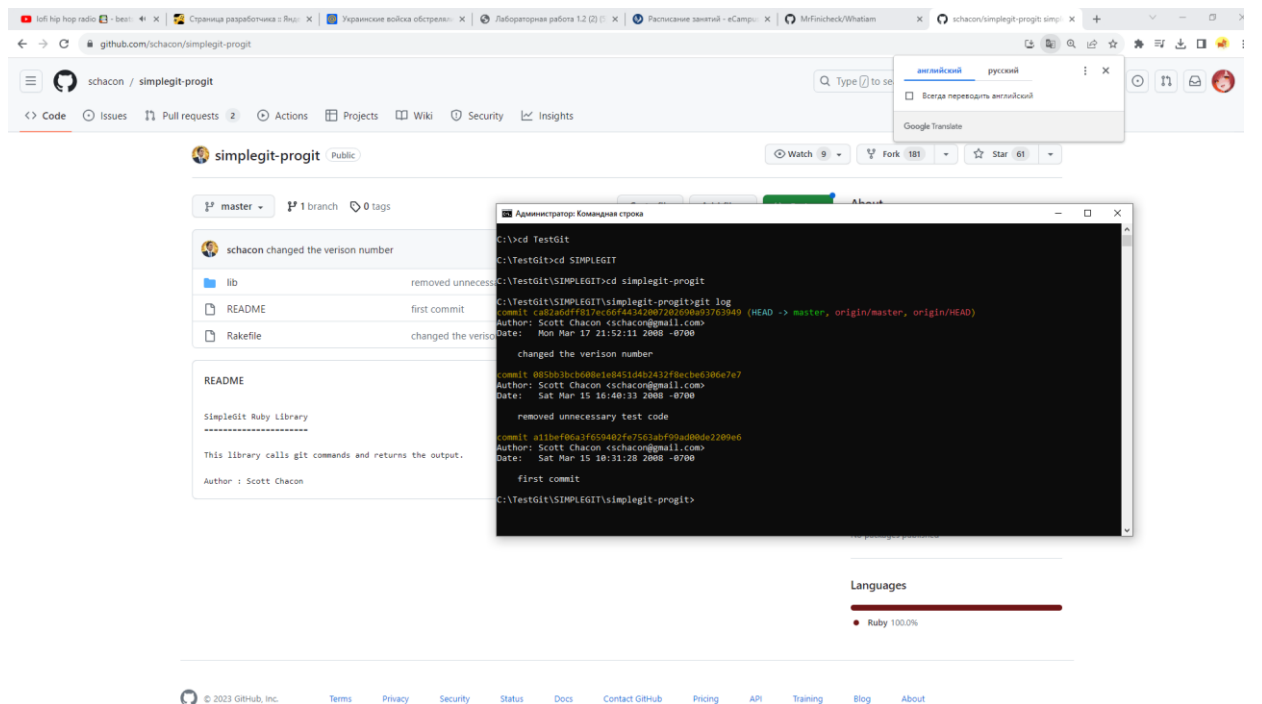


Рисунок 3.1 – Проработка примеров

4. Выполнила клонирование созданного репозитория на рабочий компьютер

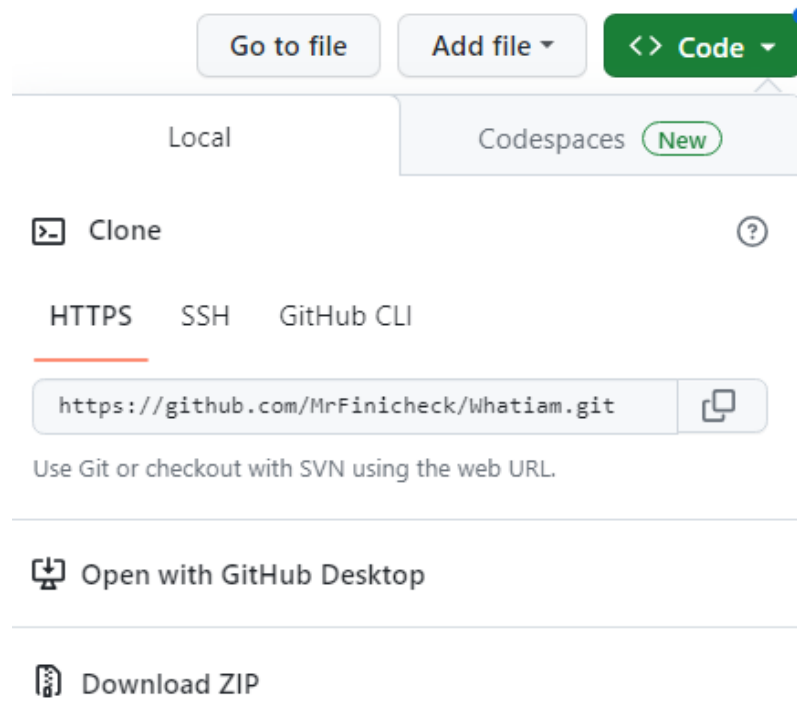


Рисунок 4.1 – Копирование ссылки репозитория

```
commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gmail.com>
Date: Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gmail.com>
Date: Sat Mar 15 10:31:28 2008 -0700

    first commit

C:\TestGit\SIMPLEGIT\simplegit-progit>cd ..
C:\TestGit\SIMPLEGIT>cd ..
C:\TestGit>cd WHATIAM
C:\TestGit\WHATIAM>git clone https://github.com/MrFinicheck/Whatiam.git
Cloning into 'Whatiam'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
C:\TestGit\WHATIAM>cd Whatiam
C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 4.2 – Копирование репозитория на рабочий компьютер с помощью командной строки

5. Дополнила файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки

```

1  # Byte-compiled / optimized / DLL files
2  __pycache__
3  *.py[co]
4  *.egg-class
5
6  # C extensions
7  *.so
8
9  # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 *.egg-info
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 share/python-wheels/
24 *.egg-info
25 .installed.cfg
26 *.egg
27 MANIFEST
28
29 # PyInstaller
30 # Usually these files are written by a python script from a template
31 # before PyInstaller builds the exe, so as to inject date/other infos into it.
32 *.manifest
33 *.spec
34
35 # Installer logs
36 pip-log.txt
37 pip-delete-this-directory.txt
38
39 # Unit test / coverage reports
40 htmlcov/

```

Рисунок 5.1 – Файл с необходимыми правилами для языка Python

6. Добавила в файл README.md информацию о своей группе и моём ФИО



Рисунок 6.1 – Локально создаю файл README.md

```
Администратор: Командная строка

On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   README.md

C:\TestGit\WHATIAM\Whatiam>git commit -m"add README.md file"
[main 381dc8b] add README.md file
1 file changed, 2 insertions(+)
create mode 100644 README.md

C:\TestGit\WHATIAM\Whatiam>git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 404 bytes | 404.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MrFinichek/Whatiam.git
2d256cf..381dc8b  main -> main

C:\TestGit\WHATIAM\Whatiam>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 6.2 – С помощью командной строки пересылаю обновленный репозиторий на GitHub

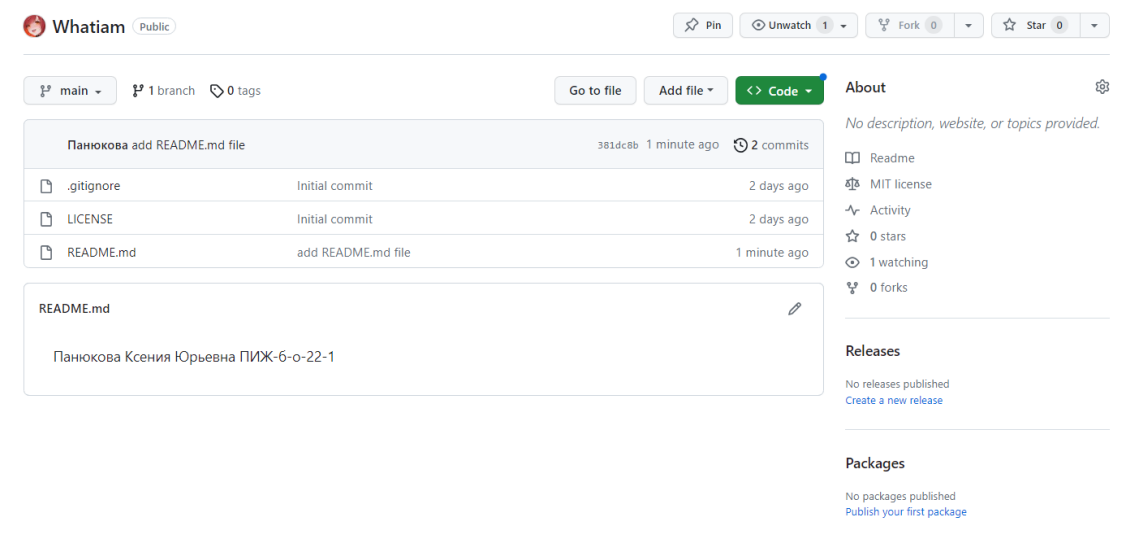


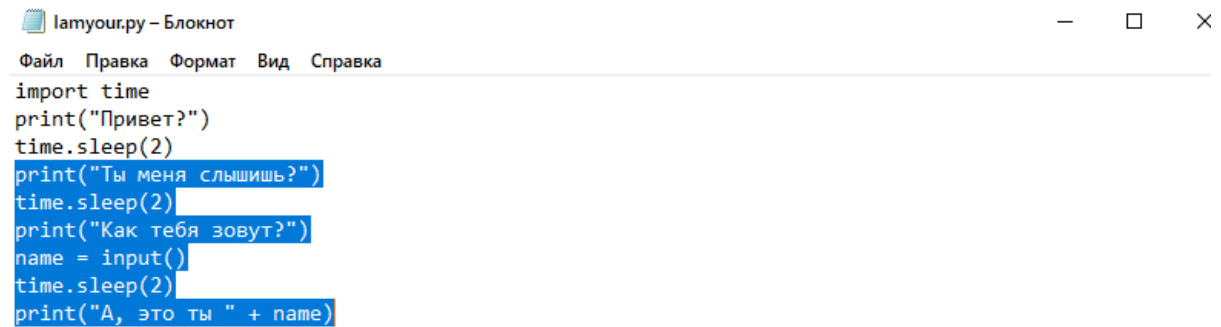
Рисунок 6.3 – Мой файл README.md на GitHub

7. Написала небольшую программу на выбранном мною языке программирования. Зафиксировала изменения при написании программы в локальном репозитории. Сделала не менее 7 коммитов, отмеченных не менее 3 тэгами



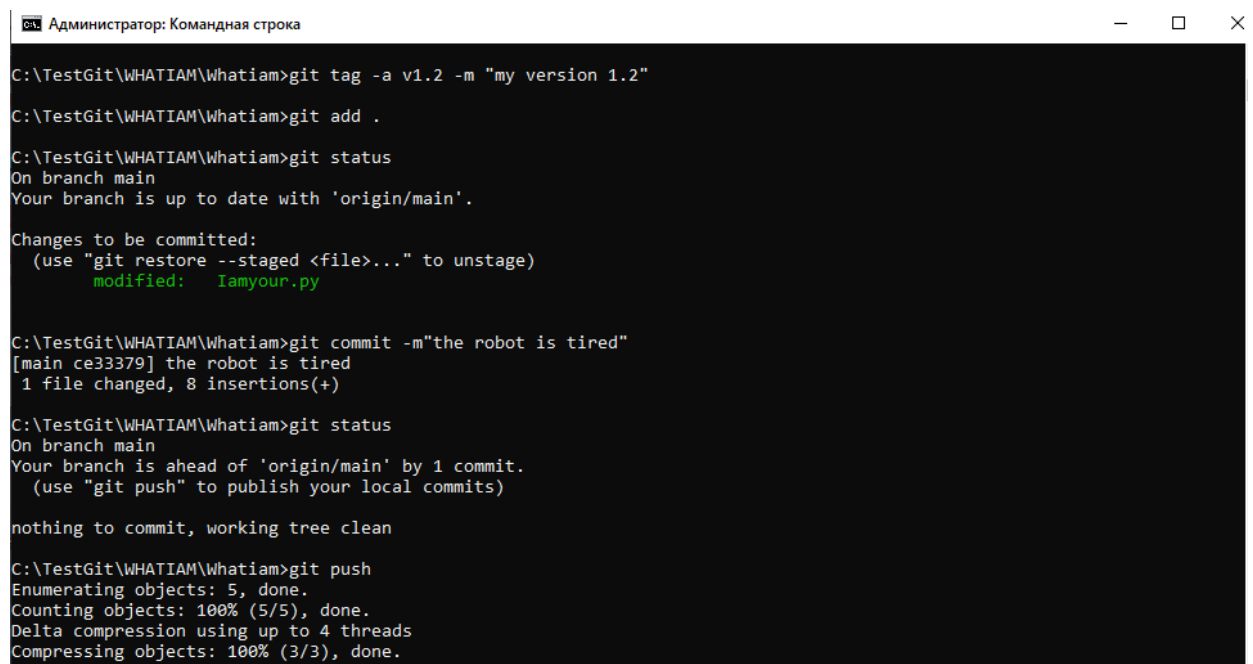
```
import time
print("Привет?")
time.sleep(2)
```

Рисунок 7.1 – Моя программа на Python



```
import time
print("Привет?")
time.sleep(2)
print("Ты меня слышишь?")
time.sleep(2)
print("Как тебя зовут?")
name = input()
time.sleep(2)
print("А, это ты " + name)
```

Рисунок 7.2 – Изменяю программу



```
C:\TestGit\WHATIAM\Whatiam>git tag -a v1.2 -m "my version 1.2"
C:\TestGit\WHATIAM\Whatiam>git add .
C:\TestGit\WHATIAM\Whatiam>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   Iamyour.py

C:\TestGit\WHATIAM\Whatiam>git commit -m "the robot is tired"
[main ce33379] the robot is tired
1 file changed, 8 insertions(+)

C:\TestGit\WHATIAM\Whatiam>git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

C:\TestGit\WHATIAM\Whatiam>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
```

Рисунок 7.3 – Делаю не менее 7 коммитов


```
Администратор: Командная строка
Writing objects: 100% (3/3), 319 bytes | 319.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/MrFinicheck/Whatiam.git
   cc0aba1..ce33379  main -> main

C:\TestGit\WHATIAM\Whatiam>git tag
v1.0
v1.1
v1.2

C:\TestGit\WHATIAM\Whatiam>git tag -a v1.3 -m"my version 1.3"

C:\TestGit\WHATIAM\Whatiam>git push origin --tags
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 629 bytes | 629.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MrFinicheck/Whatiam.git
 * [new tag]          v1.0 -> v1.0
 * [new tag]          v1.1 -> v1.1
 * [new tag]          v1.2 -> v1.2
 * [new tag]          v1.3 -> v1.3

C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 7.4 – Отмечаю не менее 3 тэгов

Commits

main

Commits on Sep 18, 2023

the robot is tired Панюкова committed 14 minutes ago	ce33379	<>
the robot doesn't want to forget you Панюкова committed 16 minutes ago	cc0aba1	<>
a robot is a robot Панюкова committed 17 minutes ago	57967fd	<>
the robot thought a lot Панюкова committed 18 minutes ago	ccc1183	<>
the robot missed you Панюкова committed 20 minutes ago	8bd76a6	<>
the robot remembered Панюкова committed 21 minutes ago	e88c688	<>
it asks you to enter a name Панюкова committed 24 minutes ago	7a1b071	<>
who is this Панюкова committed 27 minutes ago	ff29458	<>
add README.md file Панюкова committed 1 hour ago	381dc8b	<>

Commits on Sep 16, 2023

Initial commit MrFinicheck committed 2 days ago	Verified 2d256cf	<>
--	------------------	----

Newer Older

Рисунок 7.5 – Изменения, отображённые на GitHub

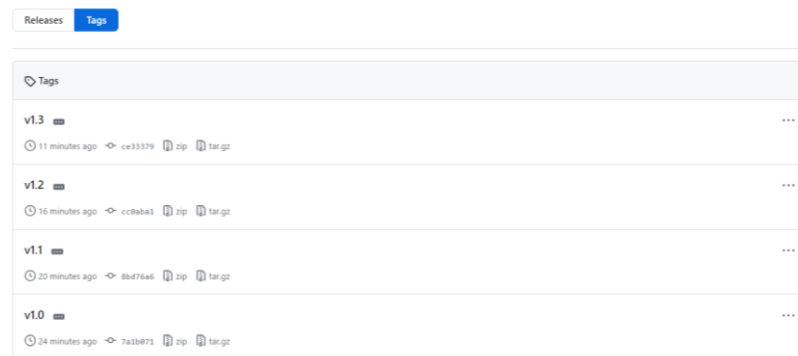


Рисунок 7.6 – Отмеченные тэги, отображённые на GitHub

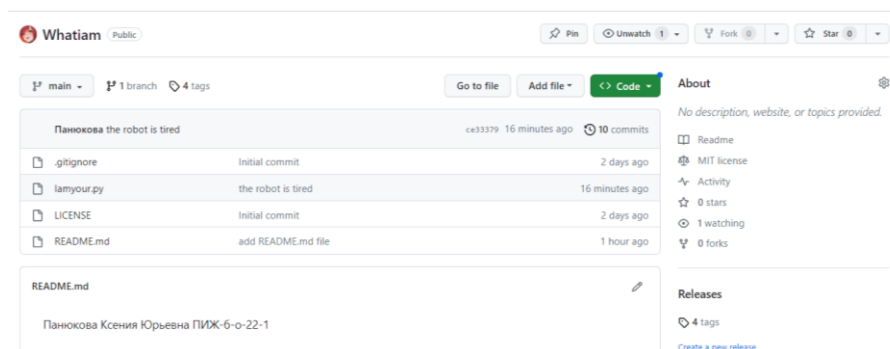


Рисунок 7.8 – Мой репозиторий

8. Просмотрела историю (журнал) хранилища командой `git log` с помощью команды `git log --graph --pretty=oneline --abbrev-commit`

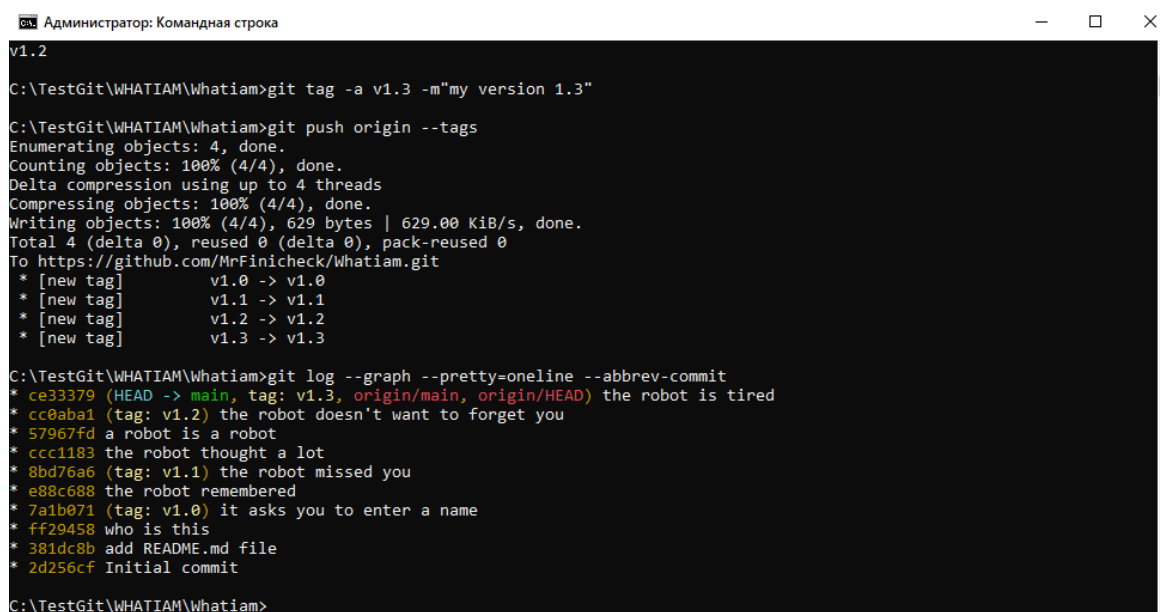


Рисунок 8.1 – История хранилища

9. Просмотреть содержимое коммитов командой `git show HEAD`, `git show HEAD~1`, `git show HEAD`

```
C:\TestGit\WHATIAM\Whatiam>git show HEAD
commit ce33379d7d06d51bedc53e60c770da67cdb9388a (HEAD -> main, tag: v1.3, origin/main, origin/HEAD)
Author: Панюкова <dima.laskov.ru@mail.ru>
Date: Mon Sep 18 18:28:31 2023 +0300

    the robot is tired

diff --git a/Iamyour.py b/Iamyour.py
index a17d49e..0698d33 100644
--- a/Iamyour.py
+++ b/Iamyour.py
@@ -28,4 +28,12 @@ time.sleep(2)
    print("Но я машина")
    time.sleep(4)
    print("И могу делать обыденные вещи гораздо быстрее чем ты думаешь, "+ name)
+time.sleep(5)
+print("...")
+time.sleep(5)
+print("...")
+time.sleep(5)
+print("...")
+time.sleep(5)
+print("...")
+time.sleep(5)
\ No newline at end of file
C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 9.1 – Вывод последнего коммита с помощью команды `git show HEAD`

```
C:\TestGit\WHATIAM\Whatiam>git show HEAD~1
commit cc0aba12db7ca227b96cd8c02653e45a3a1381e7 (tag: v1.2)
Author: Панюкова <dima.laskov.ru@mail.ru>
Date: Mon Sep 18 18:26:57 2023 +0300

    the robot doesn't want to forget you

diff --git a/Iamyour.py b/Iamyour.py
index 2948e4a..a17d49e 100644
--- a/Iamyour.py
+++ b/Iamyour.py
@@ -25,4 +25,7 @@ print("Возможно тебе покажется, что за такой ко
    time.sleep(5)
    print("...")
    time.sleep(2)
-    print("Но я машина")
\ No newline at end of file
+print("Но я машина")
+time.sleep(4)
+print("И могу делать обыденные вещи гораздо быстрее чем ты думаешь, "+ name)
+time.sleep(5)
\ No newline at end of file
C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 9.2 – Вывод предпоследнего коммита с помощью команды `git show HEAD~1`

```

C:\TestGit\WHATIAM\Whatiam>git show 7a1b07
commit 7a1b071496c2c5d11d6e703f22cab4c631492a7e (tag: v1.0)
Author: Панюкова <dima.laskov.ru@mail.ru>
Date:   Mon Sep 18 18:18:04 2023 +0300

    it asks you to enter a name

diff --git a/Iamyour.py b/Iamyour.py
index d722da7..08c76d3 100644
--- a/Iamyour.py
+++ b/Iamyour.py
@@ -4,3 +4,5 @@ time.sleep(2)
 print("Ты меня слышишь?")
 time.sleep(2)
 print("Как тебя зовут?")
+name = input()
+time.sleep(2)
C:\TestGit\WHATIAM\Whatiam>

```

Рисунок 9.3 – Вывод второго коммита с помощью команды `git show 7a1b07`

10. Я освоила возможность отката к заданной версии

10.1. Удалила весь код из одного из файлов программы репозитория, в моём случае `Iamyour.py`, и сохранила этот файл.

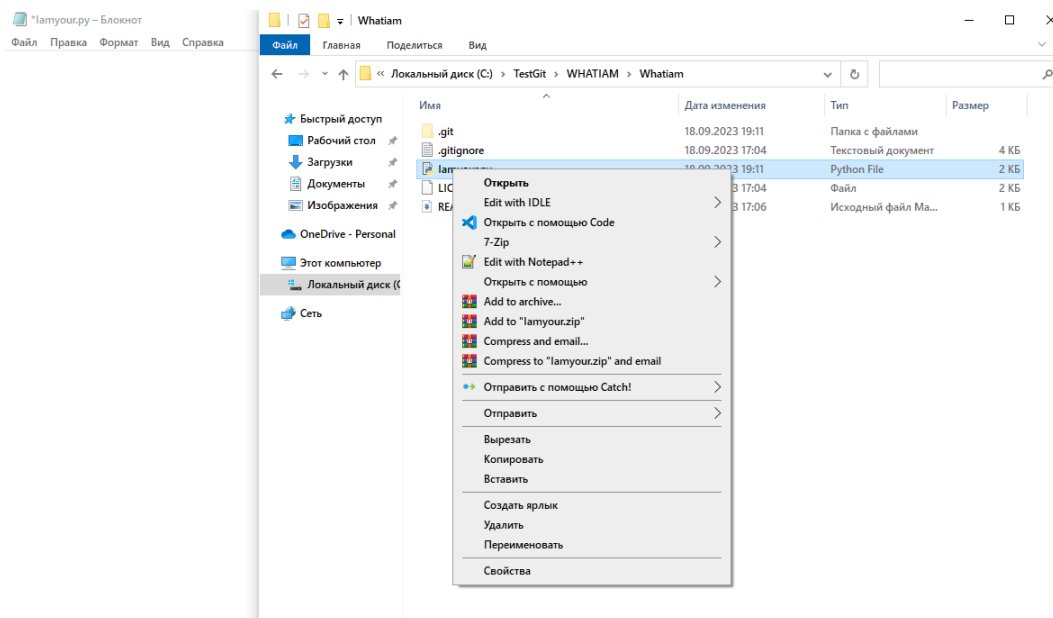


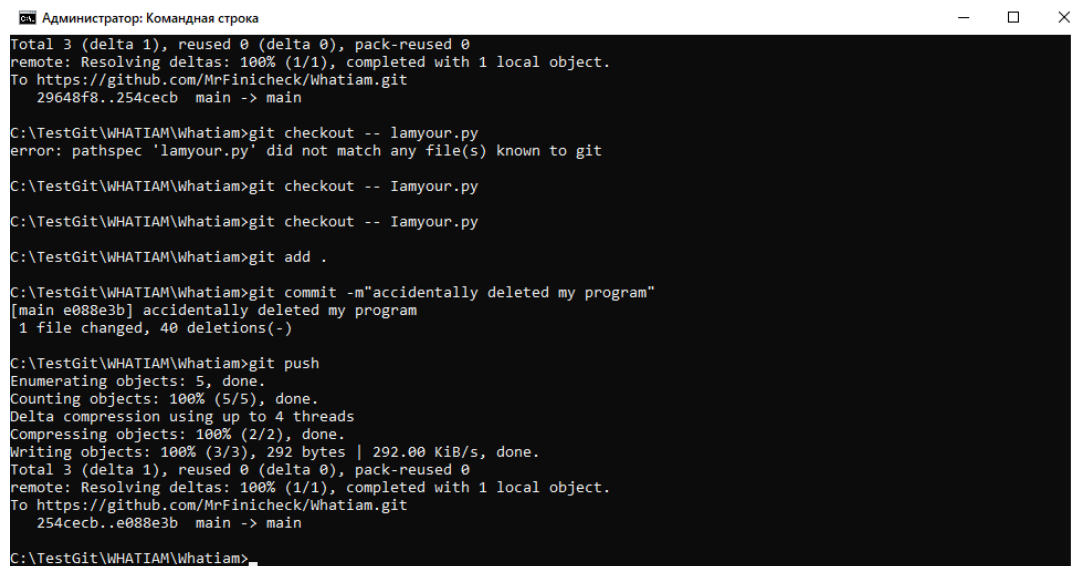
Рисунок 10.1.1 – Удаление кода и сохранение файла

10.2. Удалила все несохраненные изменения в файле командой: `git checkout – Iamyour.py`

```
C:\TestGit\WHATIAM\Whatiam>git checkout -- Iamyour.py
C:\TestGit\WHATIAM\Whatiam>git checkout -- Iamyour.py
C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 10.2.1 – Восстановление последнего коммита с помощью команды `git checkout -- Iamyour.py`

10.3. Повторяю пункт 10.1 и делаю коммит



```
Администратор: Командная строка
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/MrFinicheck/Whatiam.git
   29648f8..254cecb  main -> main

C:\TestGit\WHATIAM\Whatiam>git checkout -- lamyour.py
error: pathspec 'lamyour.py' did not match any file(s) known to git

C:\TestGit\WHATIAM\Whatiam>git checkout -- Iamyour.py

C:\TestGit\WHATIAM\Whatiam>git checkout -- Iamyour.py

C:\TestGit\WHATIAM\Whatiam>git add .

C:\TestGit\WHATIAM\Whatiam>git commit -m"accidentally deleted my program"
[main e088e3b] accidentally deleted my program
   1 file changed, 40 deletions(-)

C:\TestGit\WHATIAM\Whatiam>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 292 bytes | 292.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/MrFinicheck/Whatiam.git
   254cecb..e088e3b  main -> main

C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 10.3.1 – Отправляю удалённый файл на GitHub

10.4. Откатываю состояние хранилища к предыдущей версии командой: `git reset --hard HEAD~1`

```
C:\TestGit\WHATIAM\Whatiam>git reset --hard HEAD~1
HEAD is now at 254cecb I'm restoring
C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 10.4.1 – Возвращаем предпоследний коммит

Вывод: Git предоставляет возможность отката к предыдущей версии. Это позволяет исправить ошибки или проблемы, возникшие после обновления кода, а также сохранять историю изменений, что облегчает восстановление предыдущих версий проекта при необходимости. С помощью команды `git checkout -- <имя файла>` можно вернуть файл к последнему фиксированному состоянию, а с помощью команды `git reset --hard HEAD~1` откатить состояние

хранилища к предыдущей версии. То есть даже после выполнения коммита можно вернуть предыдущую версию программы.

11. Зафиксировала сделанные изменения

```
Администратор: Командная строка
Указано недопустимое время.
Введите новое время:

C:\TestGit\WHATIAM\Whatiam>git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
    add

C:\TestGit\WHATIAM\Whatiam>git add .

C:\TestGit\WHATIAM\Whatiam>git commit -m"I return the deleted file"
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

C:\TestGit\WHATIAM\Whatiam>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 755 bytes | 755.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/MrFinichcheck/Whatiam.git
   e088e3b..4ec4253  main -> main

C:\TestGit\WHATIAM\Whatiam>
```

Рисунок 11.1 – Отправляю изменённый файл на GitHub

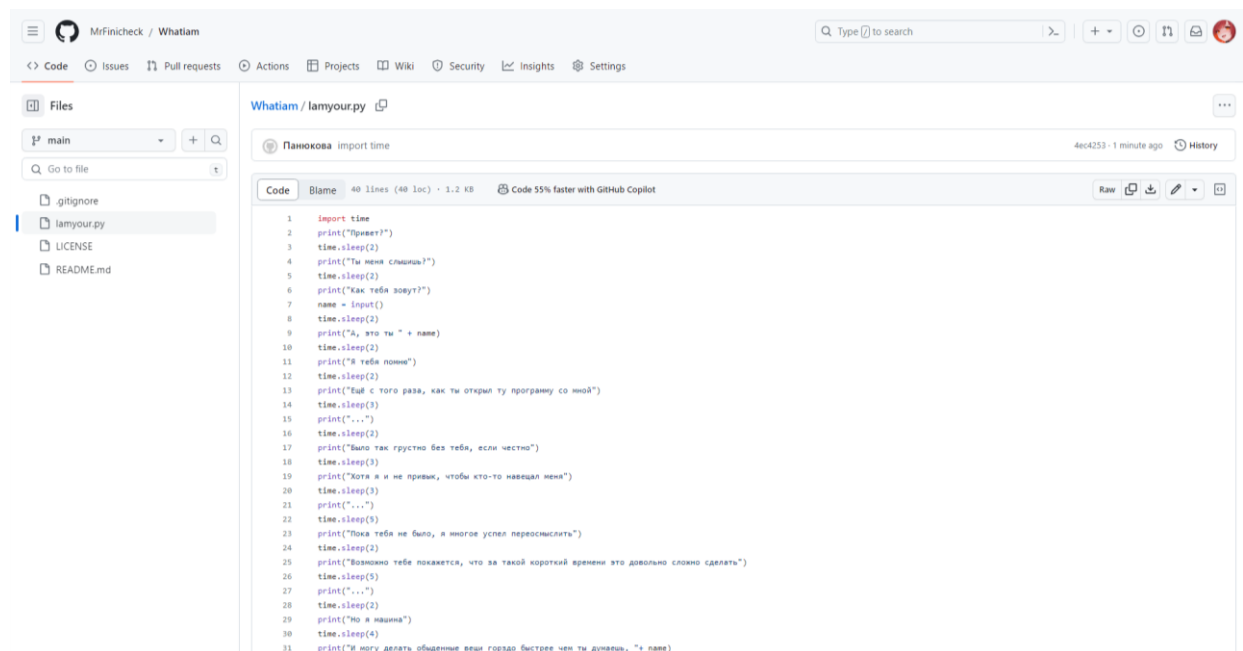


Рисунок 11.2 – Моя программа на GitHub

Контрольные вопросы

1. Как выполнить историю коммитов в Git? Какие существуют дополнительные опции для просмотра истории коммитов?

Одним из основных и наиболее мощных инструментов для просмотра истории коммитов является команда `git log`. Команда `git log` имеет очень большое количество опций для поиска коммитов по разным критериям. Одним из самых полезных аргументов является `-p` или `--patch`, который показывает разницу, внесенную в каждый коммит. Если вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `--stat`. Наиболее интересной опцией является `format`, которая позволяет указать формат для вывода информации

2. Как ограничить вывод при просмотре истории коммитов?

В дополнение к опциям форматирования вывода команда `git log` принимает несколько опций для ограничения вывода — опций, с помощью которых можно увидеть определенное подмножество коммитов. Вы можете использовать `--<n>`, где `n` — это любое натуральное число и представляет собой `n` последних коммитов. Опции для ограничения вывода по времени, такие как `--since` и `--until`, являются очень удобными. Опция `--author` дает возможность фильтровать по автору коммита, а опция `--grep` искать по ключевым словам в сообщении коммита.

3. Как внести изменения в уже сделанный коммит?

Если вы хотите переделать коммит — внесите необходимые изменения, добавьте их в индекс и сделайте коммит ещё раз, указав параметр `--amend`

4. Как отменить индексацию файла в Git?

Используйте `git reset HEAD <имя_файла>...` для отмены индексации файла

5. Как отменить изменения в файле?

С помощью команды `git checkout -- <имя_файла>`. Она выполнит откат изменений

6. Что такое удаленный репозиторий Git?

Удалённые репозитории представляют собой версии вашего проекта, сохранённые в интернете или ещё где-то в сети

7. Как выполнить просмотр удаленных репозиториях данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозиториях, вы можете запустить команду `git remote`. Она выведет названия доступных удалённых репозиториях.

8. Как добавить удаленный репозиторий для данного локального репозитория?

Для того, чтобы добавить удалённый репозиторий и присвоить ему имя (shortname), просто выполните команду `git remote add <shortname> <url>`

9. Как выполнить отправку/получение изменений с удаленного репозитория?

Отправка изменений с удалённого репозитория осуществляется с помощью команды `git push`, а получение изменений с удаленного репозитория – с помощью команды `git fetch` или `git pull`

10. Как выполнить просмотр удаленного репозитория?

Если хотите получить побольше информации об одном из удалённых репозиториях, вы можете использовать команду `git remote show <remote>`

11. Каково назначение тэгов Git?

Эта функциональность используется для отметки моментов выпуска версий

12. Как осуществляется работа с тэгами Git?

Git использует два основных типа тегов: легковесные и аннотированные. Легковесный тег — это что-то очень похожее на ветку, которая не изменяется — просто указатель на определённый коммит. А вот аннотированные теги хранятся в базе данных Git как полноценные объекты. Создание аннотированного тега в Git выполняется легко. Самый простой способ — это указать - а при выполнении команды `tag`. Опция `-m` задаёт сообщение, которое будет храниться вместе с тегом. По умолчанию, команда

`git push` не отправляет теги на удалённые сервера. После создания теги нужно отправлять явно на удалённый сервер. Процесс аналогичен отправке веток — достаточно выполнить команду `git push origin <tagname>`. Если у вас много тегов, и вам хотелось бы отправить все за один раз, то можно использовать опцию `--tags` для команды `git push`. Для удаления тега в локальном репозитории достаточно выполнить команду `git tag -d`.

13. Самостоятельно изучите назначение флага `--prune` в командах `git fetch` и `git push`. Каково назначение этого флага?

Команда `--prune` вводится для очистки любых устаревших или «зависших» ссылок в локальном репозитории. Сюда входят объекты, которые больше недоступны из какой-либо ветви или тега, а также ветви удалённого отслеживания, которые больше не существуют на удалённом компьютере.