

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

**«Исследование возможностей Git для работы с локальными
репозиториями»**

**Отчет по лабораторной работе № 1.2
по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-22-1
Душин Александр Владимирович.
Подпись студента _____
Работа защищена « » _____ 20__ г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.

Ход работы:

1. Создадим общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный язык программирования:

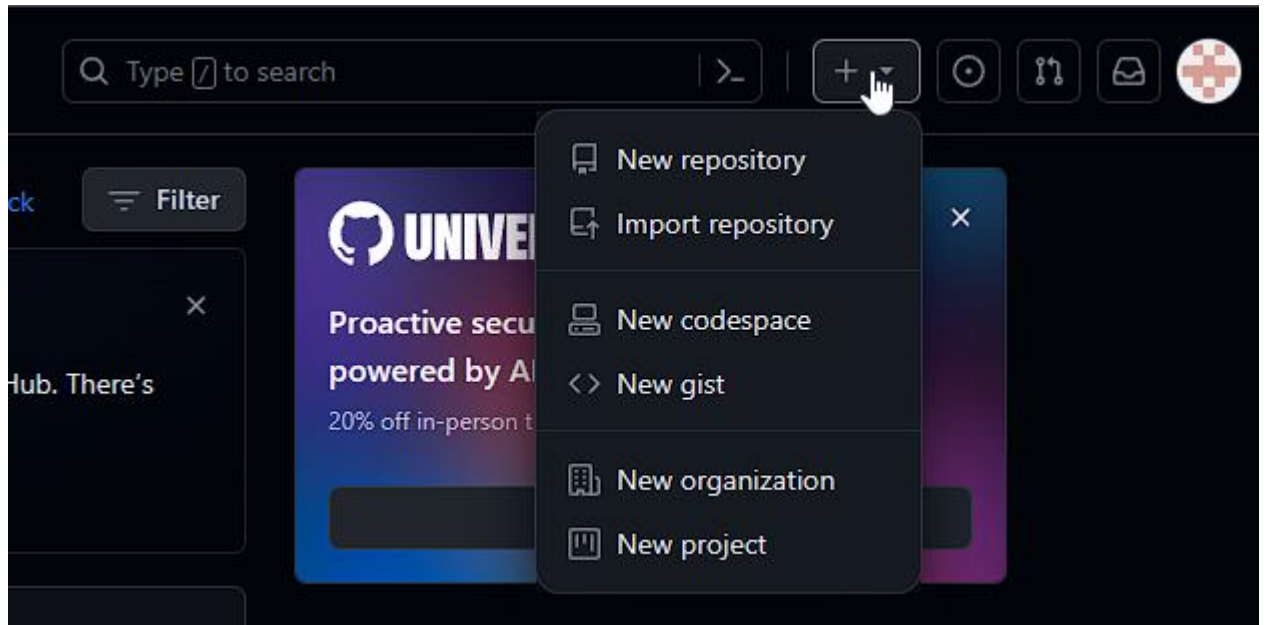



Рисунок 1 – Создание нового репозитория

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 *

 MrPlatinum ▾

Repository name *

ProgrammEngineering2

✔ ProgrammEngineering2 is available.

Great repository names are short and memorable. Need inspiration? How about [jubilant-umbrella](#) ?

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: **None** ▾

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

Choose a license

License: **None** ▾

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 – Страница создания нового репозитория

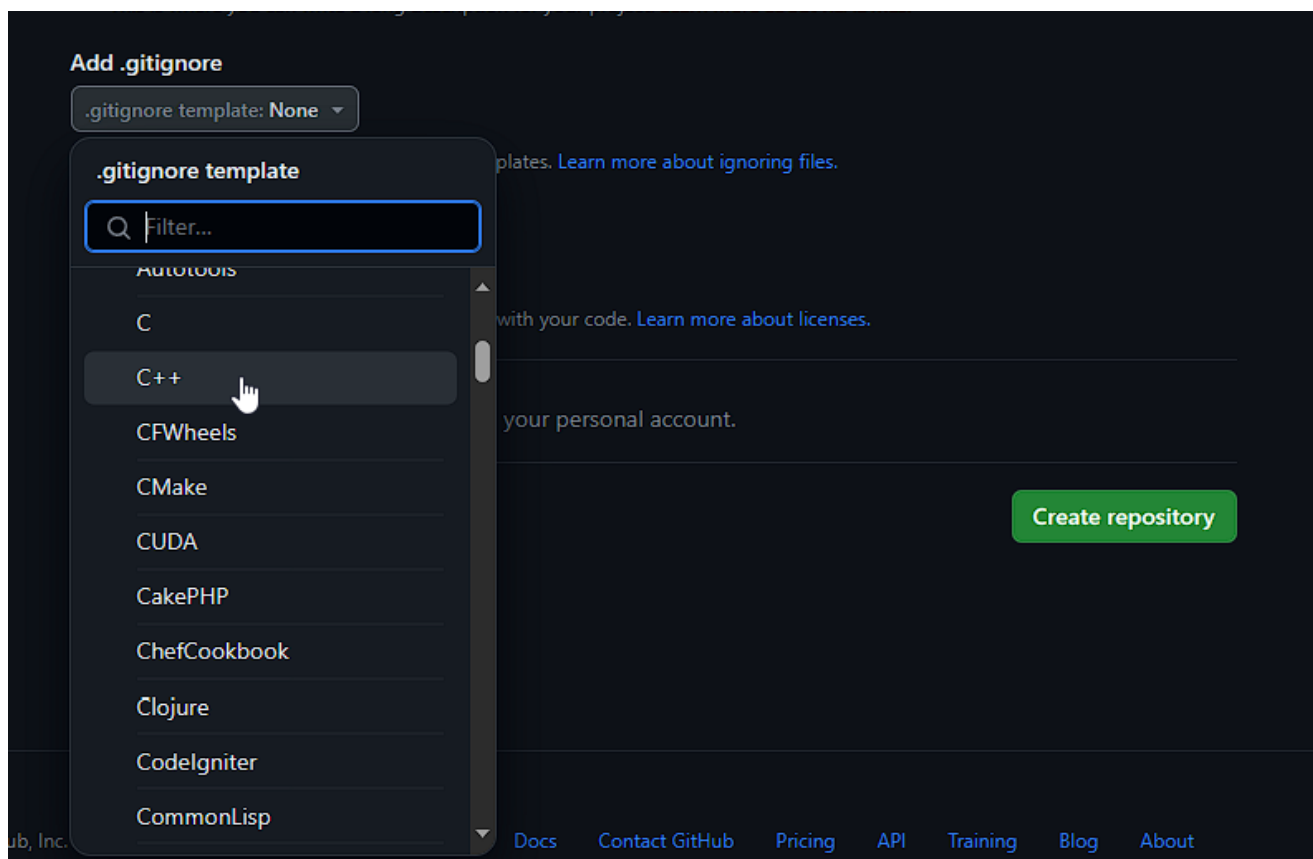


Рисунок 3 – Выбор языка программирования

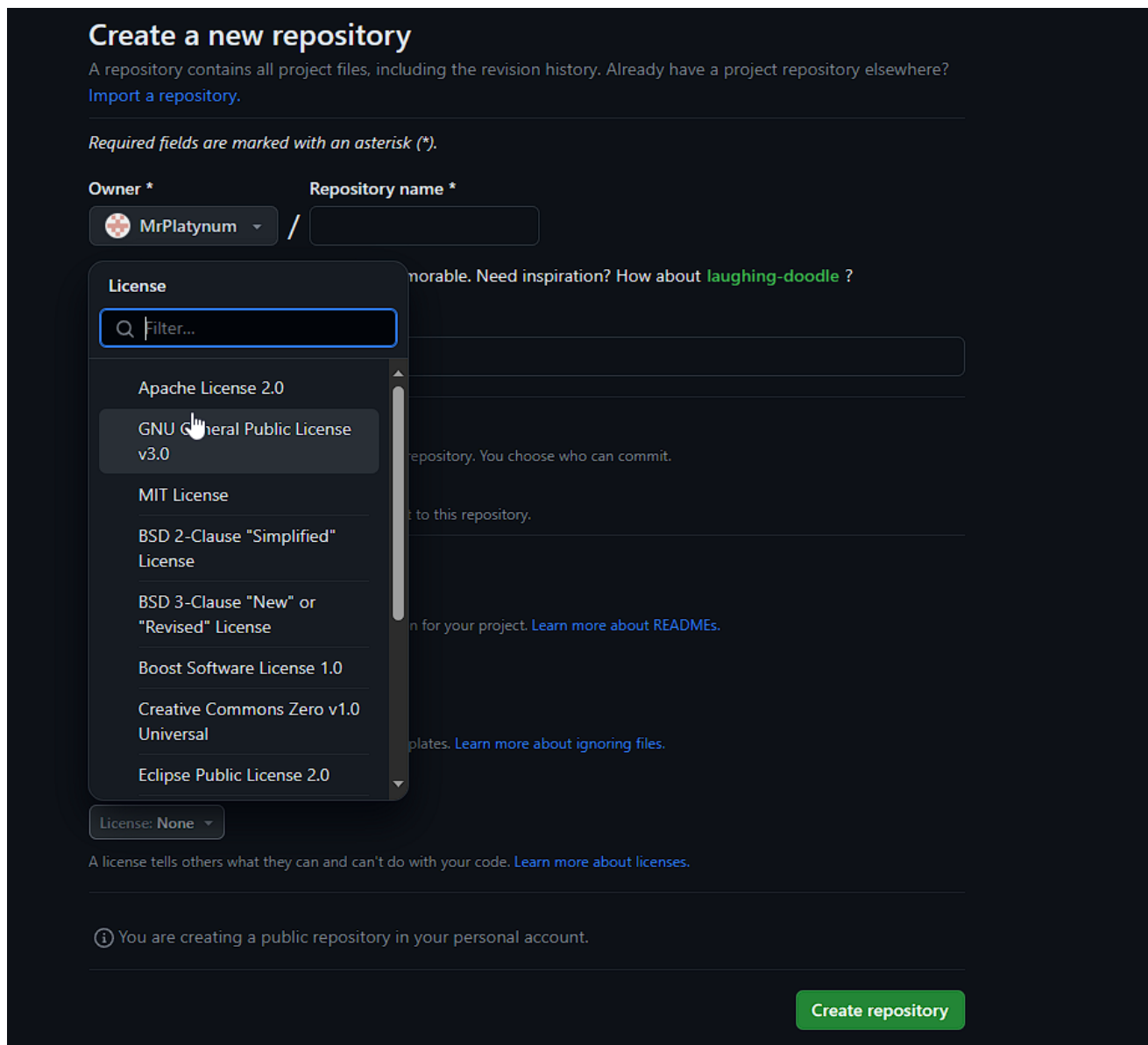


Рисунок 4 – Выбор лицензии

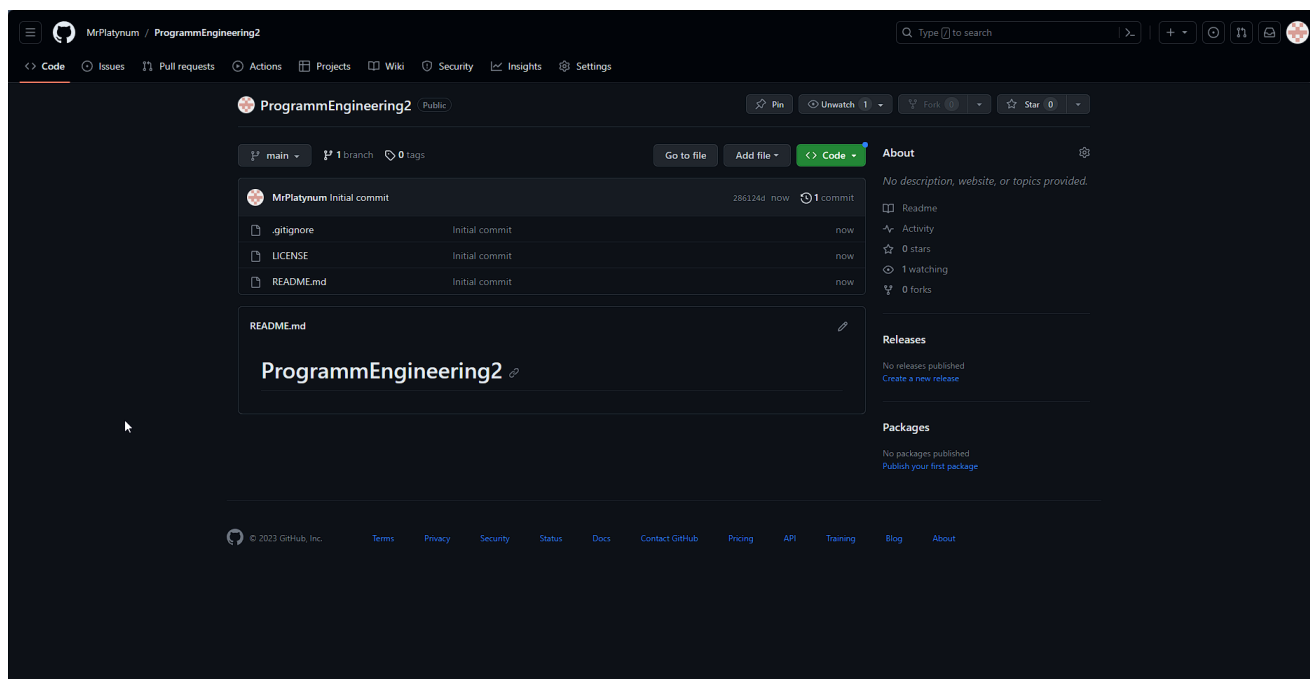
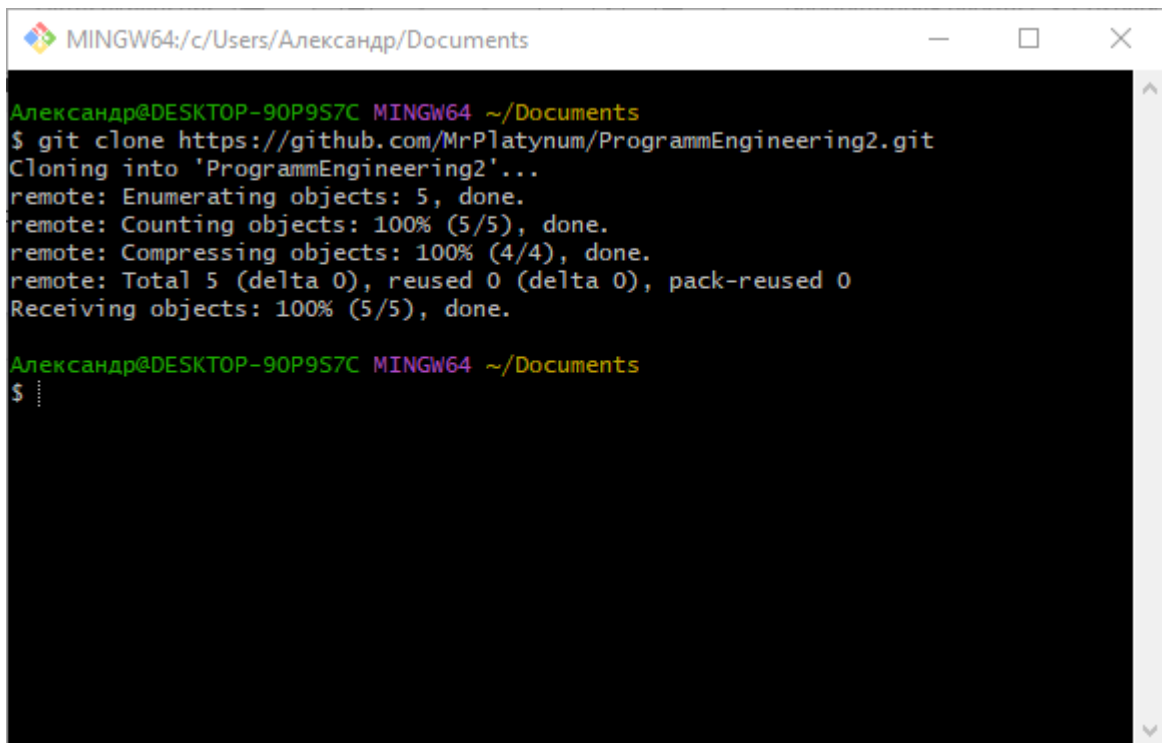


Рисунок 5 – Созданный репозиторий

2. Выполним клонирование созданного репозитория на рабочий компьютер:

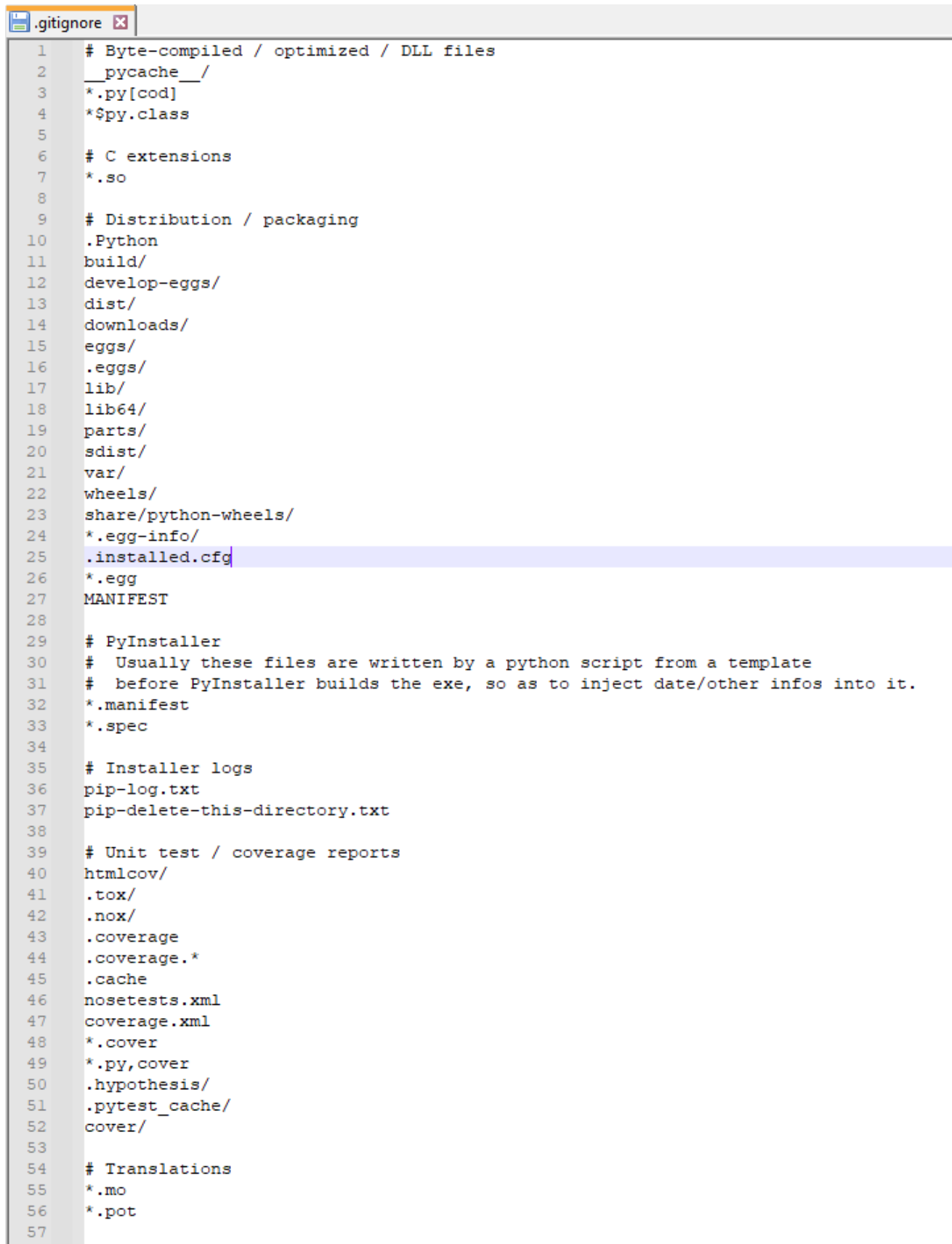


```
MINGW64:/c/Users/Александр/Documents
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents
$ git clone https://github.com/MrPlatynum/ProgrammEngineering2.git
Cloning into 'ProgrammEngineering2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents
$ .....
```

Рисунок 6 – Клонирование репозитория с помощью команды «git clone»

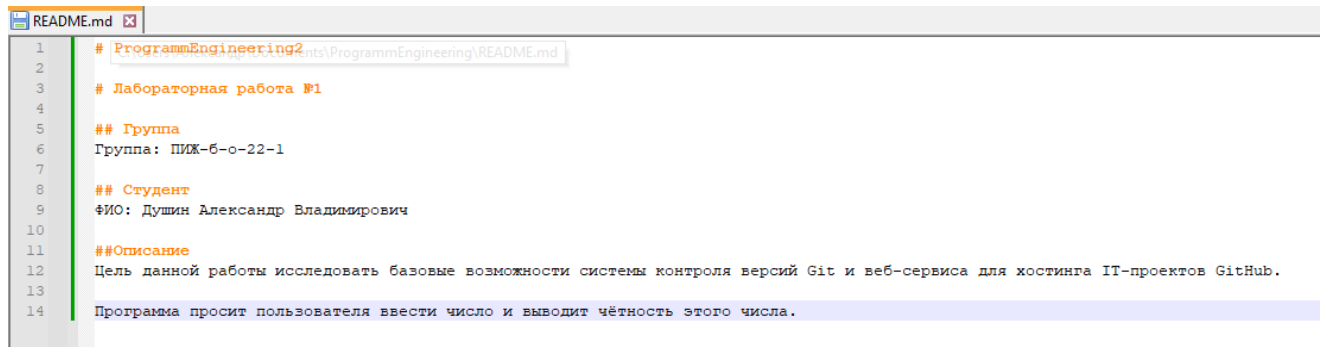
3. Покажем содержимое .gitignore:



```
1 # Byte-compiled / optimized / DLL files
2 __pycache__ /
3 *.py[cod]
4 *$py.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 .eggs/
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/
41 .tox/
42 .nox/
43 .coverage
44 .coverage.*
45 .cache
46 nosetests.xml
47 coverage.xml
48 *.cover
49 *.py,cover
50 .hypothesis/
51 .pytest_cache/
52 cover/
53
54 # Translations
55 *.mo
56 *.pot
57
```

Рисунок 7 – Содержимое файла «.gitignore»

4. Добавим в файл README.md информацию о группе и ФИО студента, выполняющего лабораторную работу:



```
1 # ProgrammEngineering2nts\ProgrammEngineering\README.md
2
3 # Лабораторная работа №1
4
5 ## Группа
6 Группа: ПИЖ-б-о-22-1
7
8 ## Студент
9 ФИО: Душин Александр Владимирович
10
11 ## Описание
12 Цель данной работы исследовать базовые возможности системы контроля версий Git и веб-сервиса для хостинга IT-проектов GitHub.
13
14 Программа просит пользователя ввести число и выводит чётность этого числа.
```

Рисунок 8 – Добавление информации в файл README.md

5. Напишем небольшую программу на выбранном языке программирования. Зафиксируем изменения в локальном репозитории:


```
MINGW64:/c:/Users/Александр/Documents/ProgrammEngineering2
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git log
commit bf222cc224bb290b490ee1f8399fe86238578d53 (HEAD -> main)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:16:43 2023 +0300

    Добавление комментариев

commit d99ec8dc58df06ce69dd4689e77f434dfa67da68 (tag: v1.2)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:15:25 2023 +0300

    Вывод результата

commit 9ce182087c8e6229695e8851435333221736a381 (tag: v1.1)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:10:03 2023 +0300

    Проверка, что число не отрицательное

commit c8443a1ae6cad16e51eeef538ff2e0d42ecd770f
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:08:51 2023 +0300

    Запрос числа, для вычисления факториала
    ....skipping...
commit bf222cc224bb290b490ee1f8399fe86238578d53 (HEAD -> main)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:16:43 2023 +0300

    Добавление комментариев

commit d99ec8dc58df06ce69dd4689e77f434dfa67da68 (tag: v1.2)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:15:25 2023 +0300

    Вывод результата

commit 9ce182087c8e6229695e8851435333221736a381 (tag: v1.1)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:10:03 2023 +0300

    Проверка, что число не отрицательное

commit c8443a1ae6cad16e51eeef538ff2e0d42ecd770f
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:08:51 2023 +0300

    Запрос числа, для вычисления факториала

commit 4986b4337e87a38dae2e985e53572d5e92515e31 (tag: v1.0)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:03:37 2023 +0300

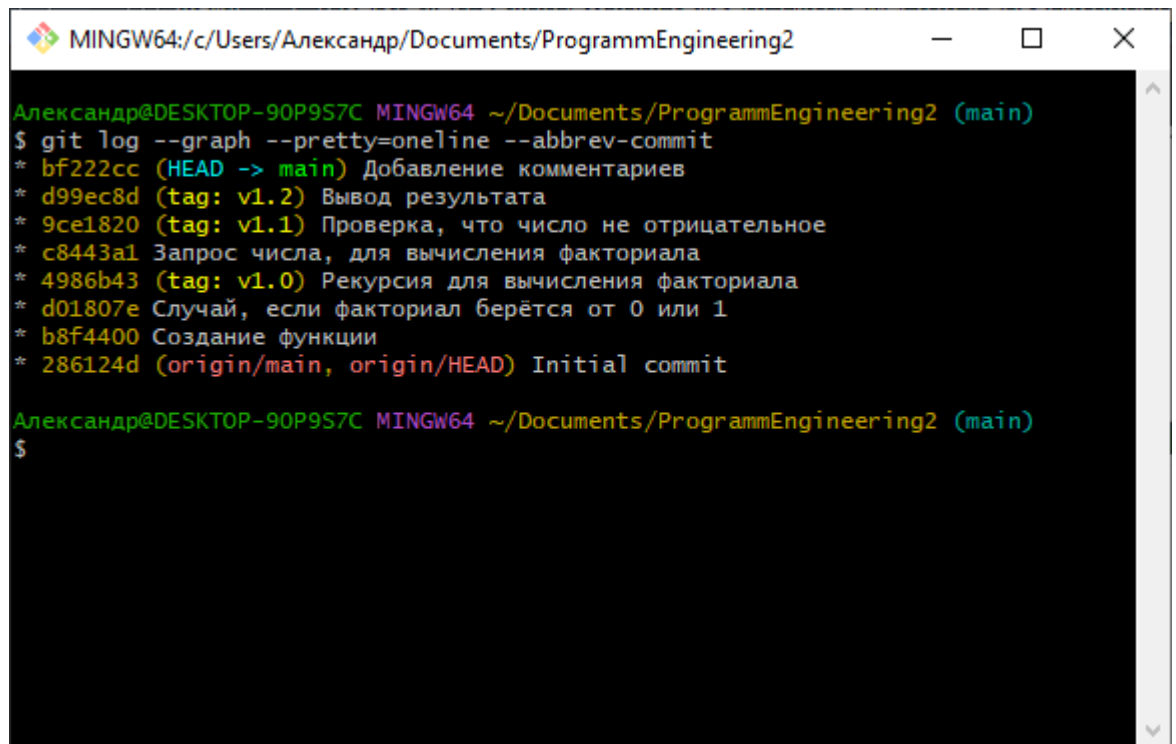
    Рекурсия для вычисления факториала

commit d01807e6ee5b153e7c1aedcef44989a5a2d18777
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:02:19 2023 +0300

    Случай, если факториал берётся от 0 или 1

commit b8f4400b44321d560c27f260f125871632fdd212
```

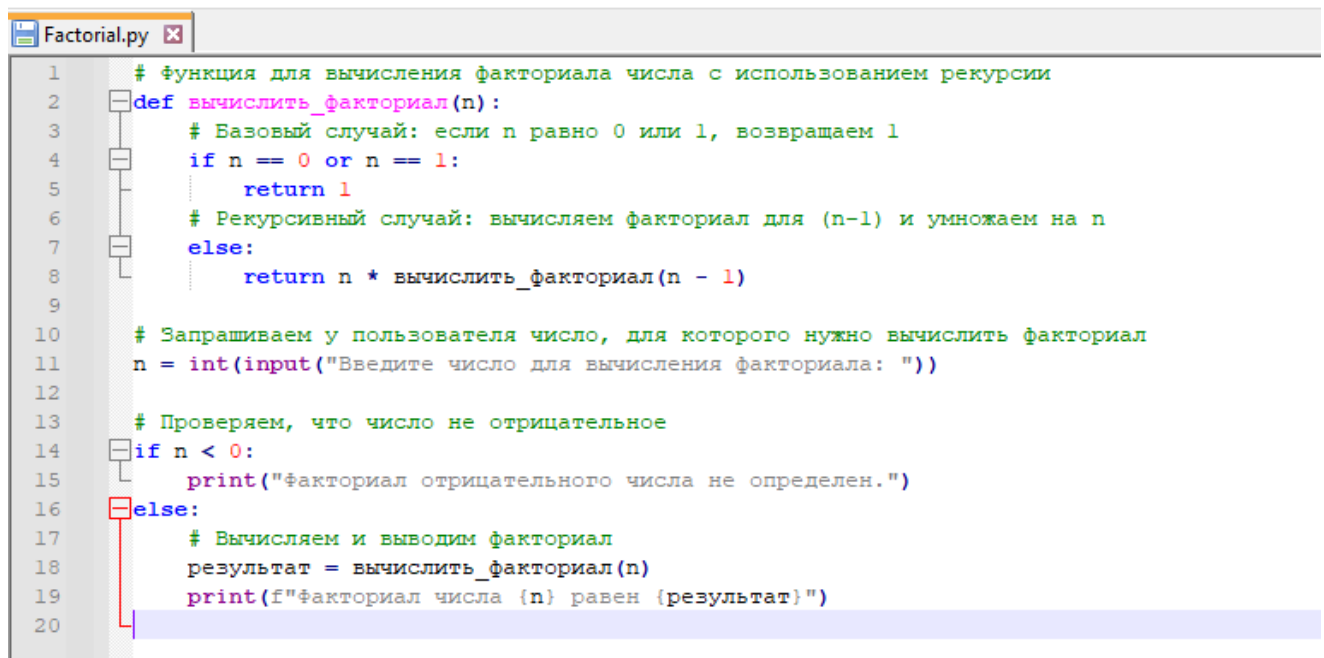
Рисунок 9 – История коммитов



```
MINGW64:/c/Users/Александр/Documents/ProgrammEngineering2
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git log --graph --pretty=oneline --abbrev-commit
* bf222cc (HEAD -> main) Добавление комментариев
* d99ec8d (tag: v1.2) Вывод результата
* 9ce1820 (tag: v1.1) Проверка, что число не отрицательное
* c8443a1 Запрос числа, для вычисления факториала
* 4986b43 (tag: v1.0) Рекурсия для вычисления факториала
* d01807e Случай, если факториал берётся от 0 или 1
* b8f4400 Создание функции
* 286124d (origin/main, origin/HEAD) Initial commit

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$
```

Рисунок 10 – История коммитов



```
Factorial.py
1  # функция для вычисления факториала числа с использованием рекурсии
2  def вычислить_факториал(n):
3      # Базовый случай: если n равно 0 или 1, возвращаем 1
4      if n == 0 or n == 1:
5          return 1
6      # Рекурсивный случай: вычисляем факториал для (n-1) и умножаем на n
7      else:
8          return n * вычислить_факториал(n - 1)
9
10 # Запрашиваем у пользователя число, для которого нужно вычислить факториал
11 n = int(input("Введите число для вычисления факториала: "))
12
13 # Проверяем, что число не отрицательное
14 if n < 0:
15     print("факториал отрицательного числа не определен.")
16 else:
17     # Вычисляем и выводим факториал
18     результат = вычислить_факториал(n)
19     print(f"факториал числа {n} равен {результат}")
20
```

Рисунок 11 – Готовая программа

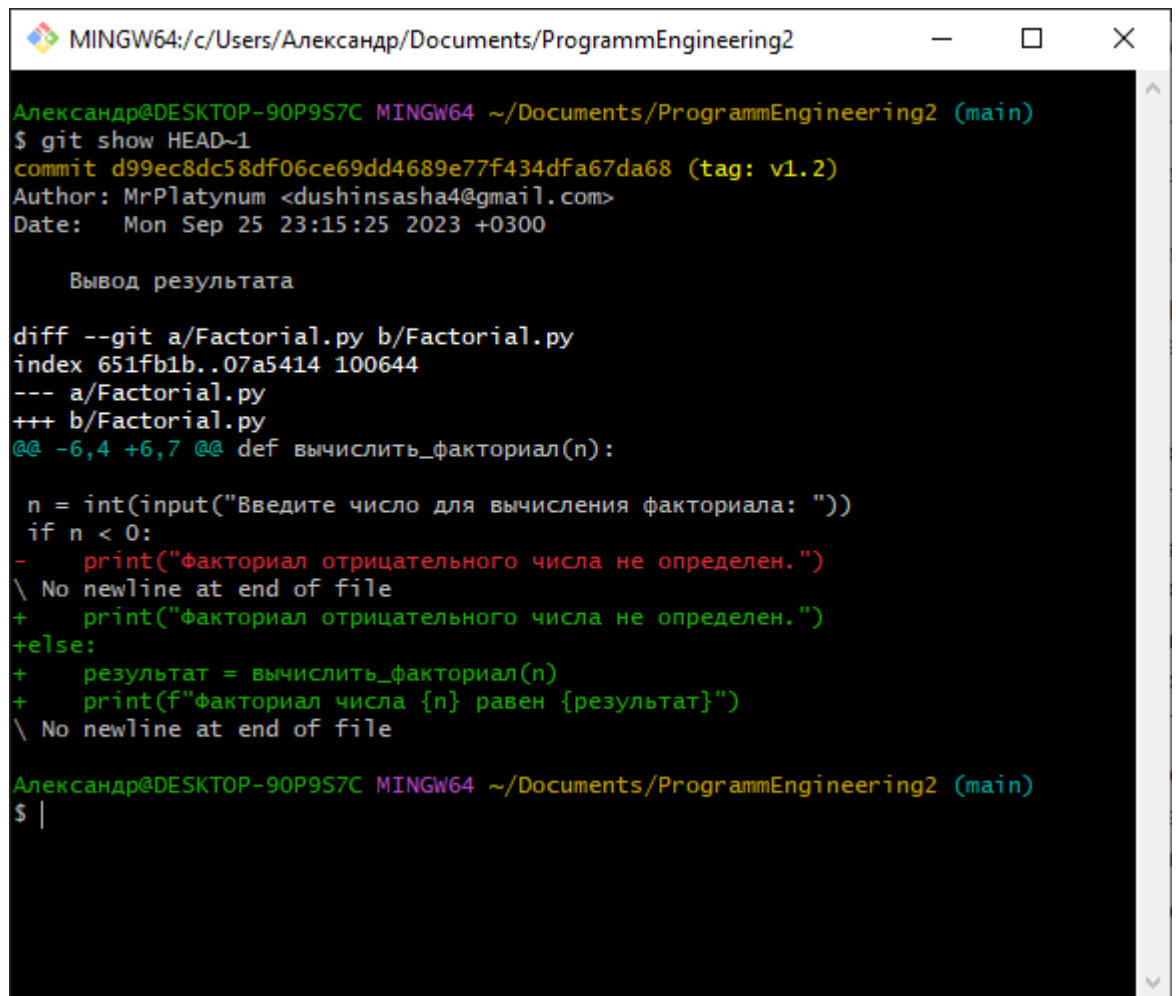
6. Просмотреть содержимое коммитов командой `git show <ref>`, где `<ref>`:

```
MINGW64/c:/Users/Александр/Documents/ProgrammEngineering2
commit bf222cc224bb290b490ee1f8399fe86238578d53 (HEAD -> main)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:16:43 2023 +0300

    Добавление комментариев

diff --git a/Factorial.py b/Factorial.py
index 07a5414..90b48f2 100644
--- a/Factorial.py
+++ b/Factorial.py
@@ -1,12 +1,19 @@
+# функция для вычисления факториала числа с использованием рекурсии
def вычислить_факториал(n):
+   # Базовый случай: если n равно 0 или 1, возвращаем 1
    if n == 0 or n == 1:
        return 1
-   else:
+   # Рекурсивный случай: вычисляем факториал для (n-1) и умножаем на n
+   else:
        return n * вычислить_факториал(n - 1)
-
+
+# Запрашиваем у пользователя число, для которого нужно вычислить факториал
n = int(input("Введите число для вычисления факториала: "))
+
+# Проверяем, что число не отрицательное
if n < 0:
    print("Факториал отрицательного числа не определен.")
else:
+   # Вычисляем и выводим факториал
    результат = вычислить_факториал(n)
-   print(f"Факториал числа {n} равен {результат}")
\ No newline at end of file
+   print(f"Факториал числа {n} равен {результат}")
~
~
~
(END)
```

Рисунок 11 – git show HEAD



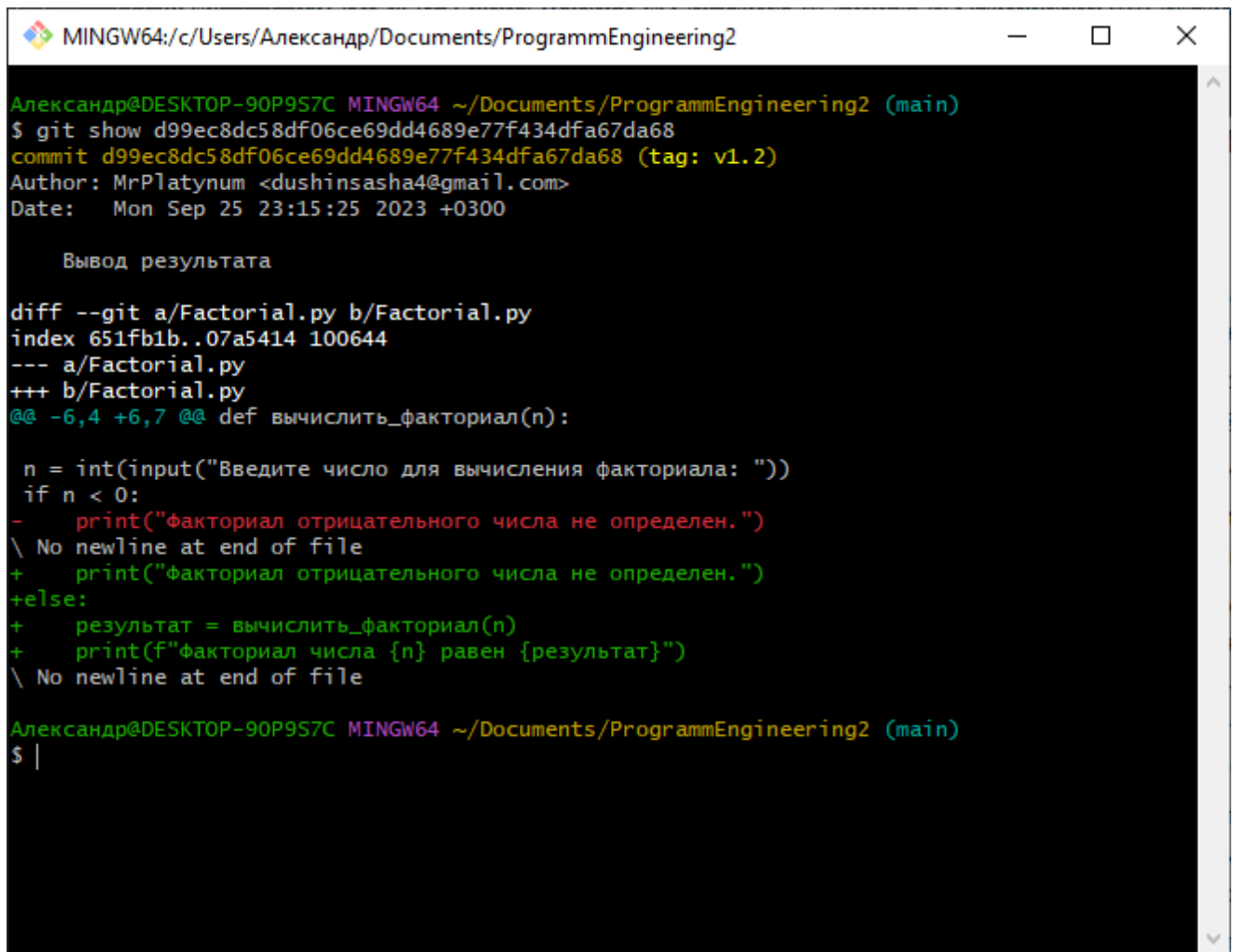
```
MINGW64:/c/Users/Александр/Documents/ProgrammEngineering2
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git show HEAD~1
commit d99ec8dc58df06ce69dd4689e77f434dfa67da68 (tag: v1.2)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date:   Mon Sep 25 23:15:25 2023 +0300

    Вывод результата

diff --git a/Factorial.py b/Factorial.py
index 651fb1b..07a5414 100644
--- a/Factorial.py
+++ b/Factorial.py
@@ -6,4 +6,7 @@ def вычислить_факториал(n):
    n = int(input("Введите число для вычисления факториала: "))
    if n < 0:
-       print("факториал отрицательного числа не определен.")
\ No newline at end of file
+       print("факториал отрицательного числа не определен.")
+else:
+       результат = вычислить_факториал(n)
+       print(f"факториал числа {n} равен {результат}")
\ No newline at end of file

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ |
```

Рисунок 12 – git show HEAD~1



```
MINGW64:/c/Users/Александр/Documents/ProgrammEngineering2

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git show d99ec8dc58df06ce69dd4689e77f434dfa67da68
commit d99ec8dc58df06ce69dd4689e77f434dfa67da68 (tag: v1.2)
Author: MrPlatynum <dushinsasha4@gmail.com>
Date: Mon Sep 25 23:15:25 2023 +0300

    Вывод результата

diff --git a/Factorial.py b/Factorial.py
index 651fb1b..07a5414 100644
--- a/Factorial.py
+++ b/Factorial.py
@@ -6,4 +6,7 @@ def вычислить_факториал(n):
    n = int(input("Введите число для вычисления факториала: "))
    if n < 0:
-       print("Факториал отрицательного числа не определен.")
\ No newline at end of file
+       print("Факториал отрицательного числа не определен.")
+else:
+   результат = вычислить_факториал(n)
+   print(f"Факториал числа {n} равен {результат}")
\ No newline at end of file

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ |
```

Рисунок 13 – git show <хеш>

- Освойте возможность отката к заданной версии:

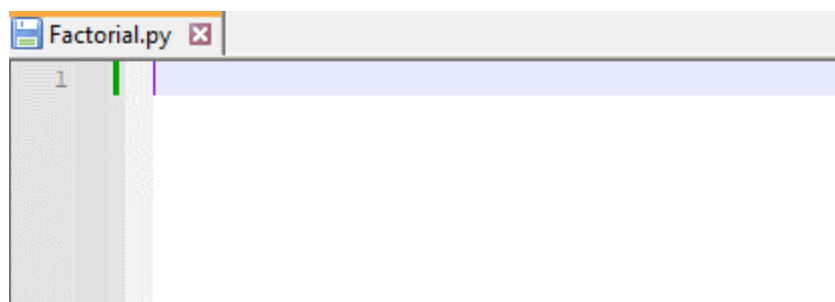
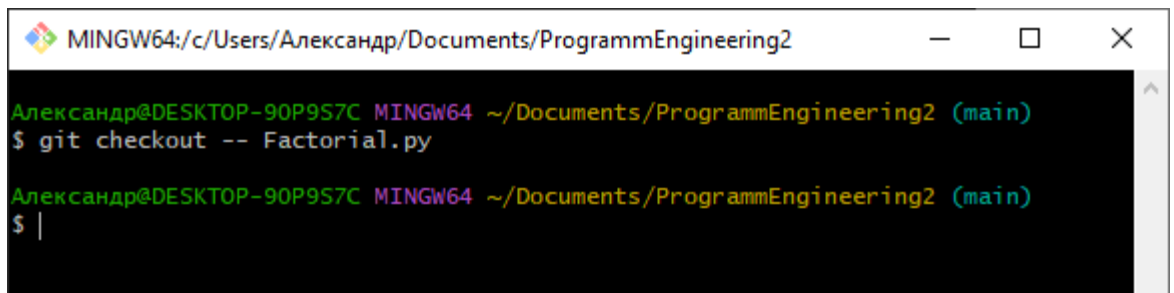


Рисунок 14 – Удаление кода

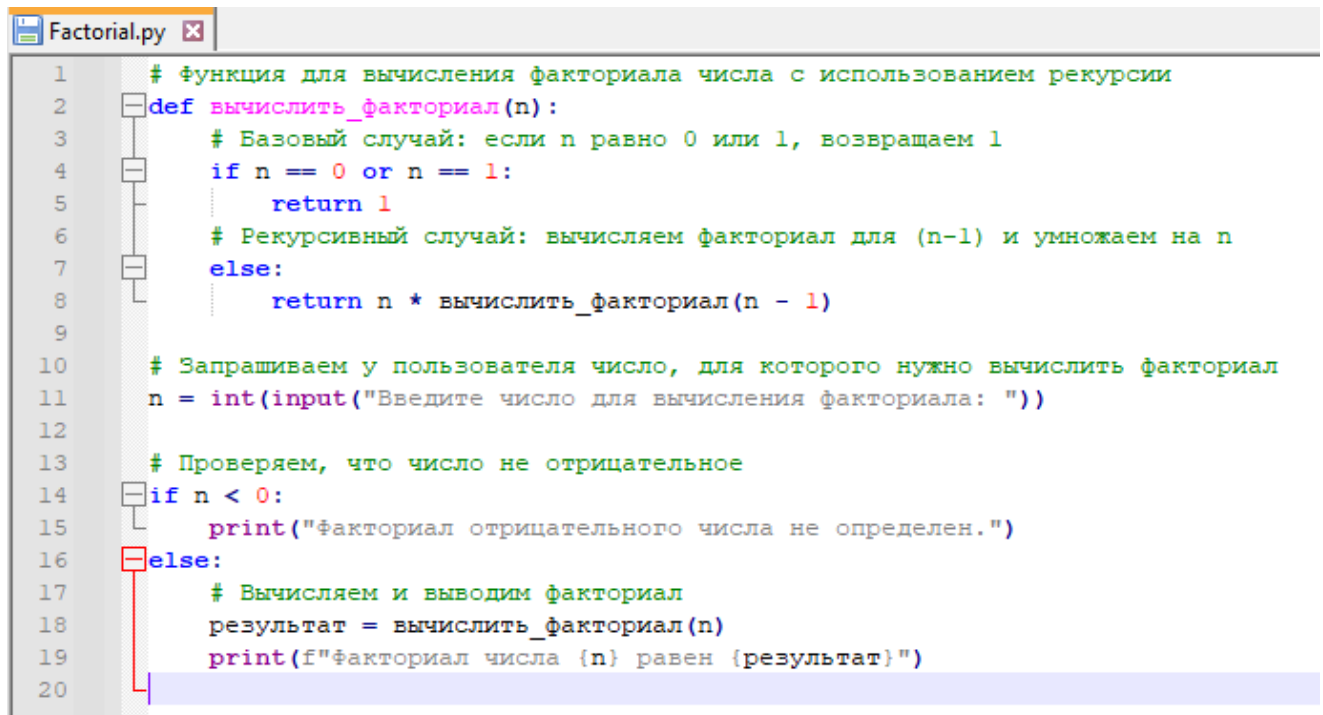


```
MINGW64:/c/Users/Александр/Documents/ProgrammEngineering2

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git checkout -- Factorial.py

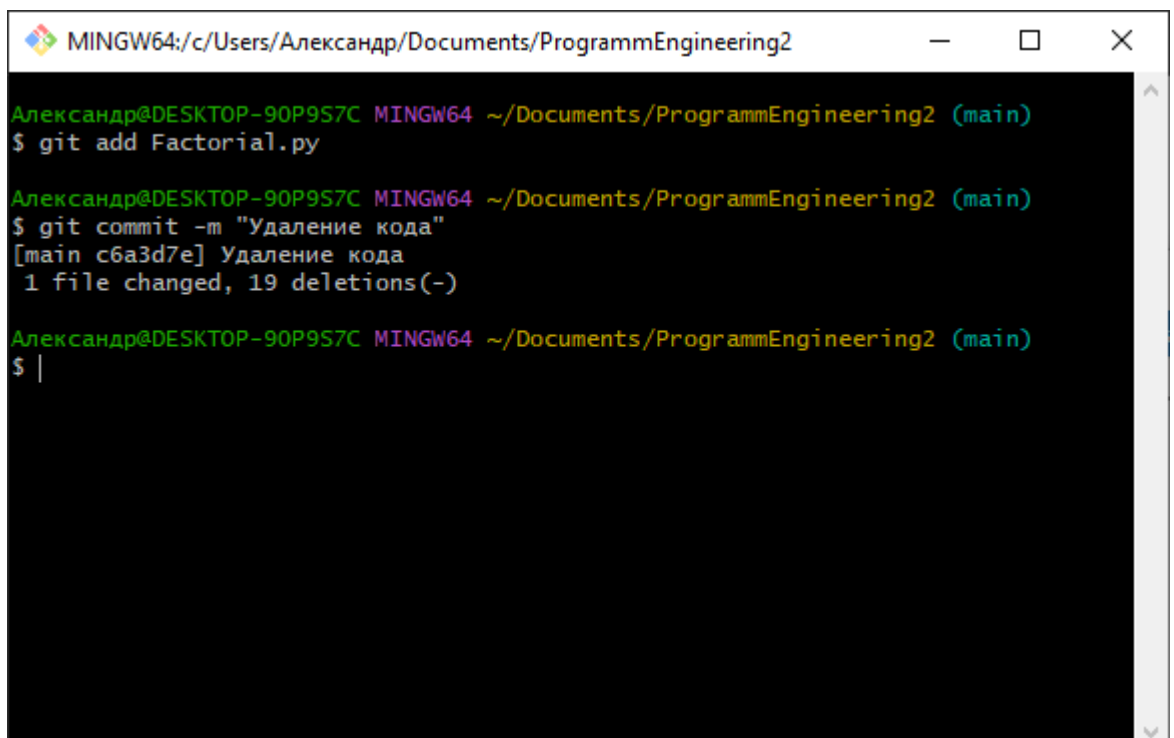
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ |
```

Рисунок 15 – Удаление всех несохраненных изменений



```
Factorial.py x
1  # функция для вычисления факториала числа с использованием рекурсии
2  def вычислить_факториал(n):
3      # Базовый случай: если n равно 0 или 1, возвращаем 1
4      if n == 0 or n == 1:
5          return 1
6      # Рекурсивный случай: вычисляем факториал для (n-1) и умножаем на n
7      else:
8          return n * вычислить_факториал(n - 1)
9
10 # Запрашиваем у пользователя число, для которого нужно вычислить факториал
11 n = int(input("Введите число для вычисления факториала: "))
12
13 # Проверяем, что число не отрицательное
14 if n < 0:
15     print("факториал отрицательного числа не определен.")
16 else:
17     # Вычисляем и выводим факториал
18     результат = вычислить_факториал(n)
19     print(f"факториал числа {n} равен {результат}")
20
```

Рисунок 16 – Результат выполнения команды



```
MINGW64:/c/Users/Александр/Documents/ProgrammEngineering2

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git add Factorial.py

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git commit -m "Удаление кода"
[main c6a3d7e] Удаление кода
1 file changed, 19 deletions(-)

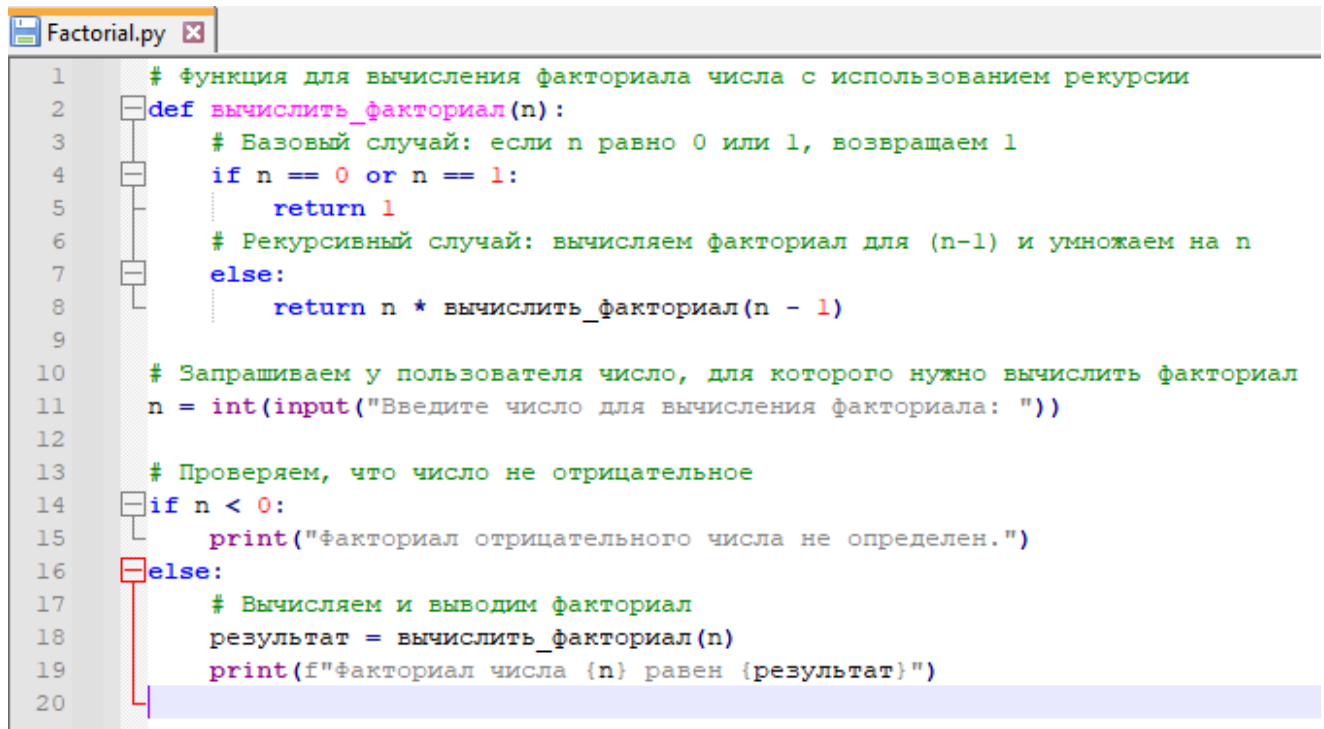
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ |
```

Рисунок 17 – Коммит пустого файла

```
Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ git reset --hard HEAD~1
HEAD is now at bf222cc Добавление комментариев

Александр@DESKTOP-90P9S7C MINGW64 ~/Documents/ProgrammEngineering2 (main)
$ |
```

Рисунок 18 – Откат к предыдущей версии



```
Factorial.py
1  # функция для вычисления факториала числа с использованием рекурсии
2  def вычислить_факториал(n):
3      # Базовый случай: если n равно 0 или 1, возвращаем 1
4      if n == 0 or n == 1:
5          return 1
6      # Рекурсивный случай: вычисляем факториал для (n-1) и умножаем на n
7      else:
8          return n * вычислить_факториал(n - 1)
9
10 # Запрашиваем у пользователя число, для которого нужно вычислить факториал
11 n = int(input("Введите число для вычисления факториала: "))
12
13 # Проверяем, что число не отрицательное
14 if n < 0:
15     print("факториал отрицательного числа не определен.")
16 else:
17     # Вычисляем и выводим факториал
18     результат = вычислить_факториал(n)
19     print(f"факториал числа {n} равен {результат}")
20
```

Рисунок 19 – Результат выполнения команд

Ответы на контрольные вопросы:

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

Для просмотра истории коммитов в Git используется команда `git log`. Она позволяет просматривать список коммитов с различными опциями. Например, вы можете использовать `git log --oneline` для просмотра краткой информации о коммитах или `git log --graph` для отображения графа ветвления и слияния. Существует множество других опций для настройки вывода истории коммитов.

2. Как ограничить вывод при просмотре истории коммитов?

Вы можете ограничить вывод при просмотре истории коммитов, используя различные фильтры, например:

`git log -n <количество>` - ограничивает вывод указанным количеством последних коммитов.

`git log --since=<дата>` и `git log --until=<дата>` - фильтруют коммиты по дате.

`git log <путь_к_файлу>` - выводит историю коммитов только для указанного файла.

3. Как внести изменения в уже сделанный коммит?

Вы не можете изменить коммит напрямую, так как коммиты в Git неизменяемы. Однако, вы можете создать новый коммит, который отменяет изменения предыдущего коммита с помощью команды `git revert`.

4. Как отменить индексацию файла в Git?

Используйте команду `git reset <файл>` для снятия файла с индексации (unstage). Это уберет файл из подготовки к следующему коммиту, но оставит его изменения в рабочей директории.

5. Как отменить изменения в файле?

Используйте команду `git checkout -- <файл>` для отмены изменений в файле и восстановления его к состоянию последнего коммита.

6. Что такое удаленный репозиторий Git?

Удаленный репозиторий Git (remote repository) - это репозиторий, который находится на удаленном сервере. Он используется для совместной работы и обмена изменениями между разными разработчиками. Примеры популярных удаленных репозиторий включают GitHub и GitLab.

7. Как выполнить просмотр удаленных репозиторий данного локального репозитория?

Вы можете выполнить команду `git remote -v`, чтобы просмотреть список удаленных репозиторий, связанных с вашим локальным репозиторием.

8. Как добавить удаленный репозиторий для данного локального репозитория?

Вы можете добавить удаленный репозиторий с помощью команды `git remote add <имя> <URL>`, где <имя> - это имя удаленного репозитория, а <URL> - URL до него.

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

Для отправки изменений в удаленный репозиторий используйте `git push <удаленный_репо> <ветка>`.

Для получения изменений из удаленного репозитория используйте `git pull <удаленный_репо> <ветка>` или `git fetch <удаленный_репо>` для загрузки изменений без слияния.

10. Как выполнить просмотр удаленного репозитория?

Вы можете просмотреть удаленный репозиторий, перейдя к нему в браузере и просматривая его содержимое на веб-сайте хостинга, таком как GitHub или GitLab.

11. Каково назначение тэгов Git?

Тэги Git используются для пометки определенных коммитов как важных моментов в истории проекта. Они облегчают идентификацию и доступ к конкретным версиям программного продукта.

12. Как осуществляется работа с тэгами Git?

Создание тэга: `git tag <название_тэга>`.

Просмотр тэгов: `git tag`.

Удаление тэга: `git tag -d <название_тэга>`.

Отправка тэга на удаленный репозиторий: `git push origin <название_тэга>`.

13. Как осуществляется работа с тэгами Git?

Создание тэга: `git tag <название_тэга>`.

Просмотр тэгов: `git tag`.

Удаление тэга: `git tag -d <название_тэга>`.

Отправка тэга на удаленный репозиторий: `git push origin <название_тэга>`.