

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 №3
дисциплины «Основы программной инженерии»

Выполнил:
Яблоновский Дмитрий Николаевич
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:
Богданов С.С., ассистент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Основы ветвления Git.

Цель работы: исследование базовых возможностей по работе с локальными и удаленными ветками Git.

Порядок выполнения работы

1. Создал репозиторий GitHub.

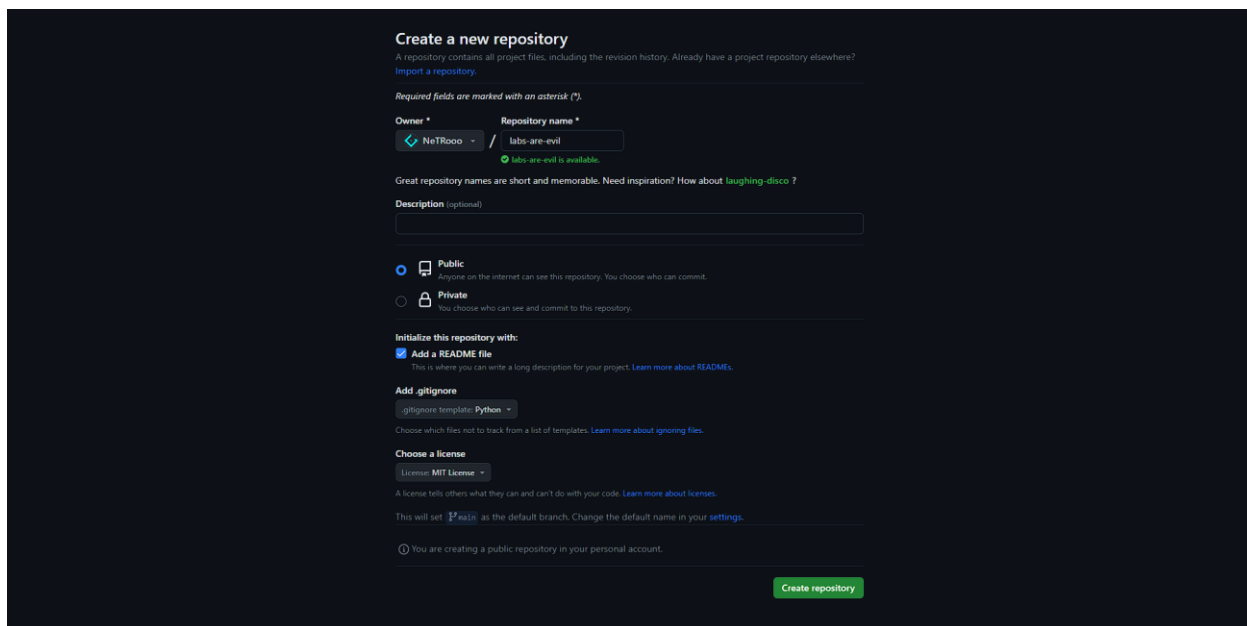


Рисунок 1- Создание репозитория

2. Проработал примеры из лабораторной работы.

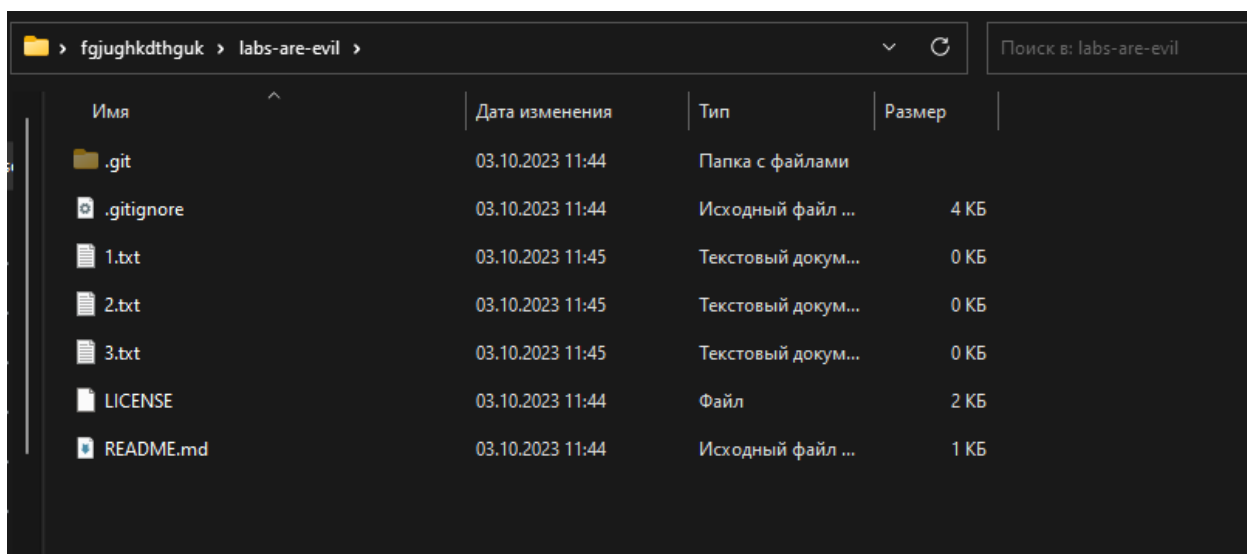


Рисунок 2 – Создал файлы «1.txt», «2.txt», «3.txt»

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\fgjughkdthguk\labs-are-evil> git add 1.txt
PS C:\Users\maxik\OneDrive\Desktop\fgjughkdthguk\labs-are-evil> |
```

Рисунок 3 – Проиндексировал файл «1.txt»

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\fgjughkdthguk\labs-are-evil> git commit -m "add 1.txt file"
[main e5e4f33] add 1.txt file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
PS C:\Users\maxik\OneDrive\Desktop\fgjughkdthguk\labs-are-evil> |
```

Рисунок 4 – Сделал первый коммит

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git add 2.txt 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit --amend -m "add 2.txt and 3.txt"
git: 'commit' is not a git command. See 'git --help'.

The most similar command is
    commit
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit --amend -m "add 2.txt and 3.txt"
[main 1c7a866] add 2.txt and 3.txt
Date: Tue Oct 3 12:07:22 2023 +0300
3 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
create mode 100644 2.txt
create mode 100644 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 5 – Проиндексировал файлы «2.txt», «3.txt» и создал коммит

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch my_first_branch
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 6 – Создал новую ветку «my_first_branch»

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout my_first_branch
Already on 'my_first_branch'
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git add in_branch.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit -m "add in_branch.txt"
[my_first_branch 5602f9c] add in_branch.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 7 – Перешёл в ветку «my_first_branch», создал новый файл и сделал КОММИТ ИЗМЕНЕНИЙ

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 8 – Вернулся в ветку «main»

```
(use "git push" to publish your local commits)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout -b new_branch
Switched to a new branch 'new_branch'
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 9 – Создал и перешёл в ветку «new_branch»

3. Выполнил клонирование репозитория на рабочий компьютер.

```
Switched to a new branch 'new_branch'
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> echo "new row in the 1.txt file" >> 1.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git add 1.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit -m "add a new row to 1.txt"
[new_branch 77620ce] add a new row to 1.txt
1 file changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 10 – Сделал изменения в файле «1.txt», сделал коммит

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git merge my_first_branch
Updating 1c7a866..5602f9c
Fast-forward
 in_branch.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git merge new_branch
Merge made by the 'ort' strategy.
 1.txt | Bin 0 -> 56 bytes
 1 file changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 11 – Перешёл в ветку «main», выполнил слияние всех веток

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch -d my_first_branch
Deleted branch my_first_branch (was 5602f9c).
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch -d new_branch
Deleted branch new_branch (was 77620ce).
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 12 – Удалил ветки

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch branch_1
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch branch_2
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 13 – Создал новые ветки

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout branch_1
Switched to branch 'branch_1'
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> echo "fix in the 1.txt" > 1.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> echo "fix in the 3.txt" > 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git add 1.txt 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit -m "Fix in 1.txt and 3.txt"
[branch_1 569a46f] Fix in 1.txt and 3.txt
 2 files changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 14 – Перешёл в ветку, изменил файлы и сделал коммит

```
Windows PowerShell
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git checkout branch_2
Already on 'branch_2'
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> echo "My fix in the 1.txt" > 1.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> echo "My fix in the 3.txt" > 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git add 1.txt 3.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git commit -m "Fix in 1.txt and 3.txt"
[branch_2 fa8b5f2] Fix in 1.txt and 3.txt
 2 files changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 15 – Перешёл в другую ветку, изменил файлы и сделал коммит

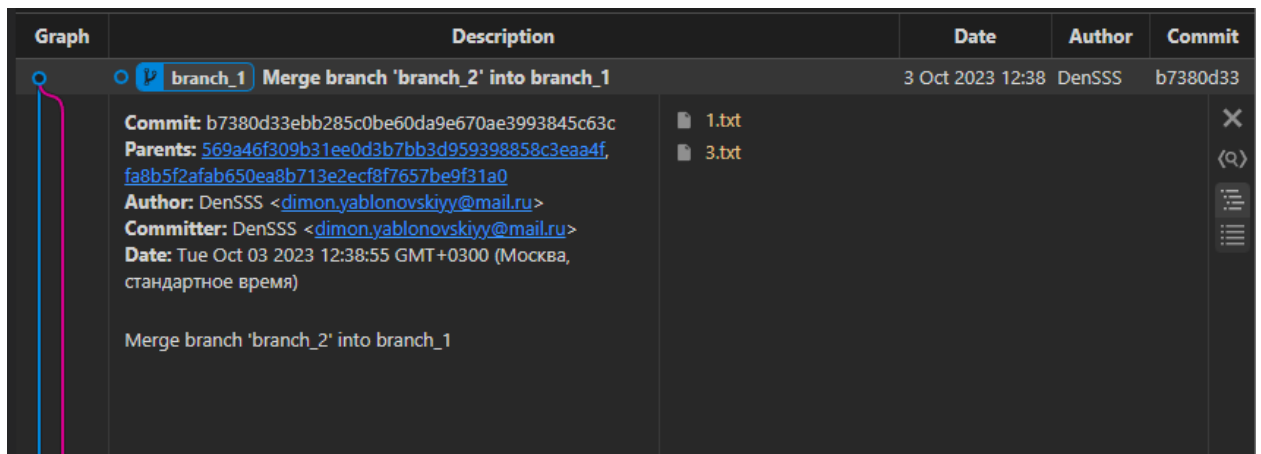


Рисунок 16 – Решил конфликт «1.txt» в ручном режиме

```
Windows PowerShell
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git mergetool -t meld
No files need merging
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 17 – Решил конфликт «3.txt» используя mergetool

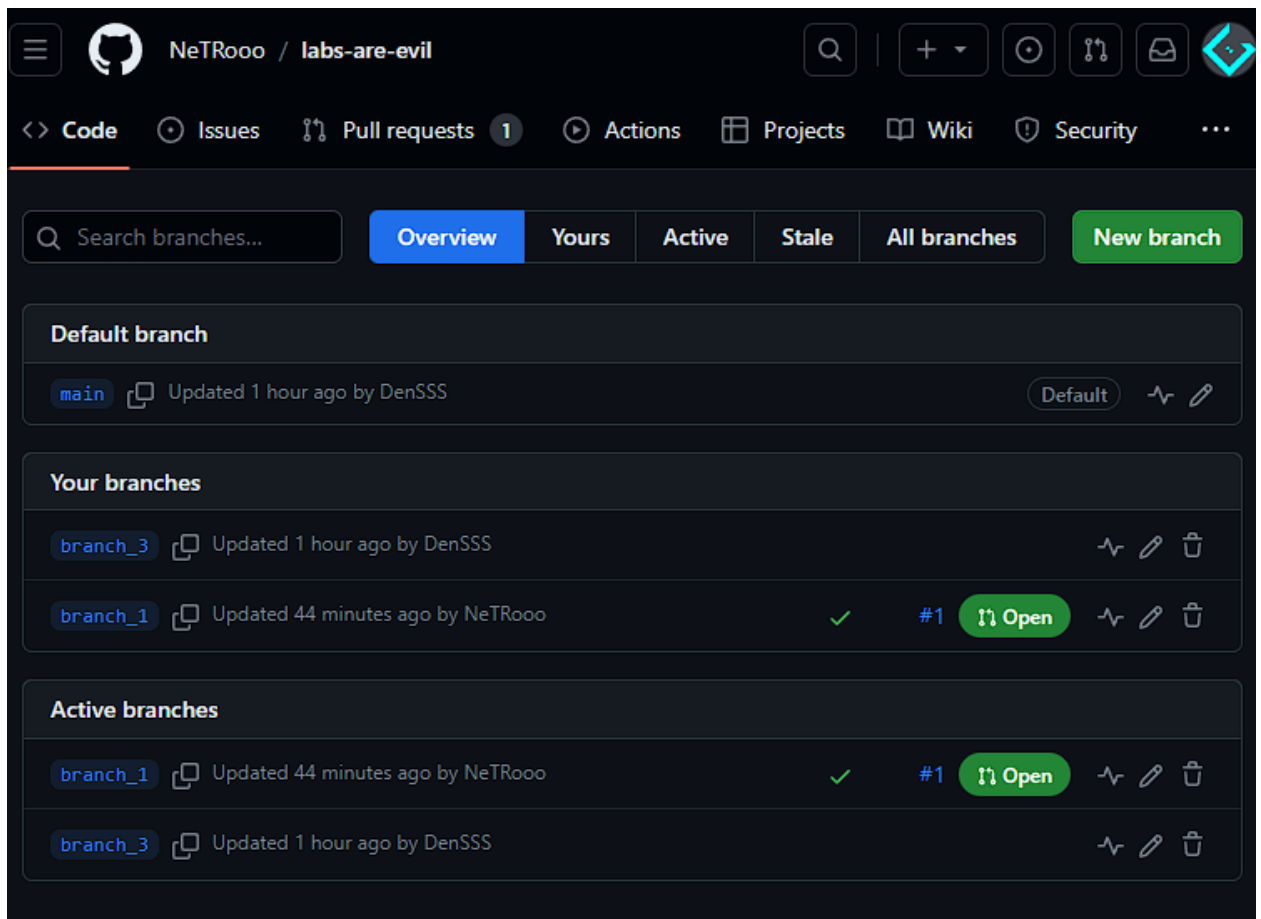


Рисунок 20 – Создал ветку branch_3 на сайте github

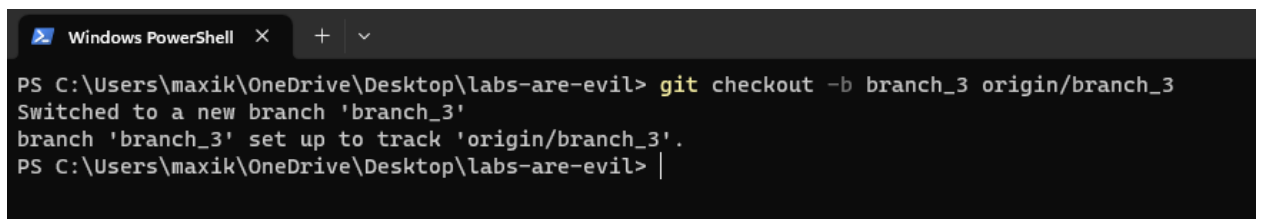


Рисунок 21 – Создал в локальном репозитории ветку отслеживания удалённой ветки branch_3

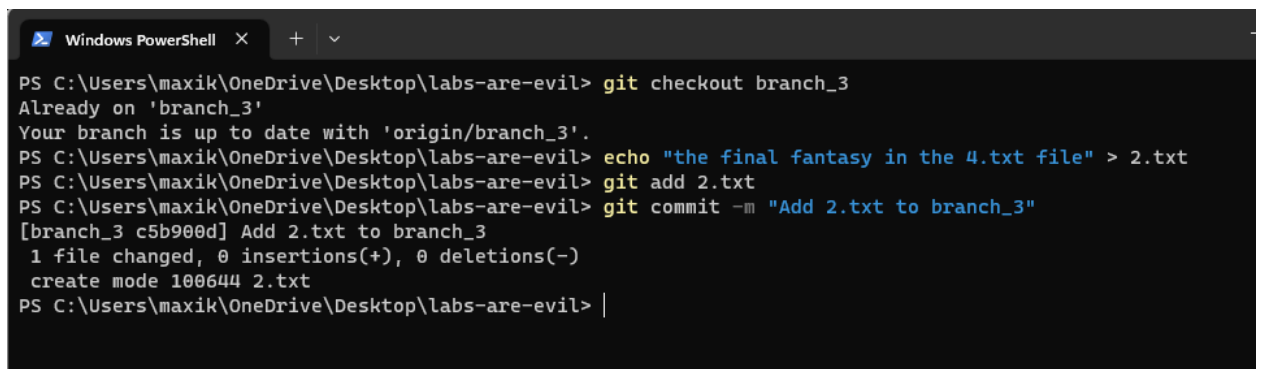


Рисунок 22 – Сменил ветку, изменил файл и сделал коммит


```
Create mode 100644 2.txt
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git branch -f master branch_2
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 23 – Выполнил перемещение ветки main на ветку branch_2

```
Windows PowerShell X + v
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git push origin main branch_2
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/NeTRooo/labs-are-evil.git
 b4127c5..eff2e59  main -> main
 * [new branch]      branch_2 -> branch_2
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> |
```

Рисунок 24 – Отправил изменения веток

Ответы на контрольные вопросы

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

С помощью команды `git log`. Существуют множество дополнительных опций вот некоторые из них:

- `-p` или `-patch` – показывает разницу, внесенную в каждый коммит;
- `--stat` – позволяет увидеть сокращенную статистику;
- `--pretty` – эта опция меняет формат вывода;
- `--since` и `--until` – опции для ограничения вывода по времени

2. Как ограничить вывод при просмотре истории коммитов?

С помощью команды `git log` и его аргументов, например `--since`.

3. Как внести изменения в уже сделанный коммит?

С помощью команды `git commit --amend`.

4. Как отменить индексацию файла в Git?

С помощью команды `git reset HEAD`.

5. Как отменить изменения в файле?

С помощью команды `git checkout -- <file>`.

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?

С помощью команды `git remote -v`.

8. Как добавить удаленный репозиторий для данного локального репозитория?

С помощью команды `git remote add <shortname> <url>`.

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

Получение изменение – `git fetch [remote-name]`, отправка изменений `git push <remote-name> <branch-name>`.

10. Как выполнить просмотр удаленного репозитория?
С помощью команды `git remote -v`.

11. Каково назначение тэгов Git?

Тэги Git - это ссылки на определенные коммиты в истории разработки. Они используются для пометки определенных версий или моментов в вашем проекте. Тэги обычно используются для обозначения релизов или важных этапов в разработке.

12. Как осуществляется работа с тэгами Git?

Для просмотра тэгов – `git tag`, для создания аннотированного тэга – `git tag -a v1.4 -m “сообщение”`, для отправки тэга на удаленный сервер – `git push origin <tagname>`, для удаления тэгов – `git tag -d <tagname>`.

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

Флаг `--prune` в командах `git fetch` и `git push` используется для удаления удаленных веток или тэгов, которые больше не существуют на удаленном репозитории. Если выполнить команду `git fetch --prune`, Git удалит локальные ссылки на удаленные ветки и тэги, которые были удалены на удаленном репозитории. Если выполнить команду `git push --prune`, Git удалит удаленные

ветки и тэги на удаленном репозитории, которых больше нет в вашем локальном репозитории.