

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 №1.2
дисциплины «Основы кроссплатформенного программирования»

Выполнил:

Худяков Максим Дмитриевич

1 курс, группа ИТС-б-о-22-1,

11.03.02 «Инфокоммуникационные

технологии и системы связи»,

направленность (профиль)

«Инфокоммуникационные системы и

сети», очная форма обучения

(подпись)

Руководитель практики:

Воронкин Р.А., канд. тех. наук, доцент,

доцент кафедры инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ GIT ДЛЯ РАБОТЫ С ЛОКАЛЬНЫМИ РЕПОЗИТОРИЯМИ

Цель работы: исследовать базовые возможности системы контроля версий Git для работы с локальными репозиториями.

Порядок выполнения работы:

Задание 1.

Создал новый репозиторий и клонировал его на свой компьютер. Добавил некоторое правило в файл *gitignore*, чтобы Git игнорировал файлы в формате .jpeg.

```
C:\GIT>git clone https://github.com/maxim-hoodyakov/lab1.2.git
Cloning into 'lab1.2'...
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.
```

Рисунок 1. Новый репозиторий.

Задание 2. Проработать примеры лабораторной работы. Отразить вывод на консоли при выполнении команд git в отчете для лабораторной работы.

Пример 1. Просмотр истории коммитов

```
C:\GIT\lab1.2>git log
commit d1339ebc7285ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD)
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 12:00:36 2023 +0300

    Удалил ненужные файлы

commit 0ee4ec64b4c36a9f60cca243f1b98c68fe70388e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:59:31 2023 +0300

    Добавил папку с файлом для отчета

commit 3f922709d3a70b9a983ed7f494a270866a71630d
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:57:25 2023 +0300

    Переименовал снимок

commit c37eefd50c56c605f1137f6ef4bba52da6b068
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:54:30 2023 +0300

    Добавил снимок

commit 6983aa50c4eba9b0bec9f9d974c606b8ab6a761fc
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:53:23 2023 +0300

    Добавил информации в .txt

commit c4ab7706290b1e015f4bdc1c7efb2f204d0176e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:50:22 2023 +0300

    Добавил текстовый файл

commit 7c543384cae6f07e87be5ab31788e67b34766d0d
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:49:31 2023 +0300

    Удалил ненужные файлы

commit d1339ebc7285ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD)
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 12:00:36 2023 +0300

    Удалил ненужные файлы

commit 0ee4ec64b4c36a9f60cca243f1b98c68fe70388e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:59:31 2023 +0300
```

Рисунок 2. Команда “git log”

Вывод только двух записей. Команда - `git log -p -2`.

```
C:\GIT\lab1.2>git log -p -2
commit d1339ebc7205ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD)
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 12:00:36 2023 +0300

    Удалил ненужные файлы

diff --git a/T.txt b/T.txt
deleted file mode 100644
index 0c65b2f..0000000
--- a/T.txt
+++ /dev/null
@@ -1,0,0 @@
-Мой профиль в GIT - https://github.com/maxim-hoodyakov
diff --git "a/doc/\320\236\321\202\321\207\320\265\321\202.txt" "b/doc/\320\236\321\202\321\207\320\265\321\202.txt"
deleted file mode 100644
index e69de29..0000000

commit 0ee4ec64b4c36a9f60cca243f1b98c68fe70388e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:59:31 2023 +0300

    Добавил папку с файлом для отчета

diff --git "a/doc/\320\236\321\202\321\207\320\265\321\202.txt" "b/doc/\320\236\321\202\321\207\320\265\321\202.txt"
new file mode 100644
index 0000000..e69de29

C:\GIT\lab1.2>
```

Рисунок 3. Команда “`git log -p -2`”

Чтобы увидеть сокращенную статистику для каждого коммита, можно использовать опцию `--stat` :

```
C:\GIT\lab1.2>git log --stat
commit d1339ebc7205ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD)
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 12:00:36 2023 +0300

    Удалил ненужные файлы

T.txt | 1 -
"doc/\320\236\321\202\321\207\320\265\321\202.txt" | 0
2 files changed, 1 deletion(-)

commit 0ee4ec64b4c36a9f60cca243f1b98c68fe70388e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:59:31 2023 +0300

    Добавил папку с файлом для отчета

"doc/\320\236\321\202\321\207\320\265\321\202.txt" | 0
1 file changed, 0 insertions(+), 0 deletions(-)

commit 3f929709d3a78be9a03ed2f494a870066a71630d
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:57:25 2023 +0300

    Переименовал снимок

...277\321\200\320\276\321\204\320\270\320\273\321\214.PNG" | Bin
1 file changed, 0 insertions(+), 0 deletions(-)

commit c37eefd550c566c605f1137f6ef4bba52da6b868
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:54:30 2023 +0300

    Добавил снимок

"\320\241\320\275\320\270\320\274\320\276\320\272.PNG" | Bin 0 -> 13672 bytes
1 file changed, 0 insertions(+), 0 deletions(-)
```

Рисунок 4. Команда “`git log --stat`”

Опция oneline выводит каждый коммит в одну строку, что может быть очень удобным если вы просматриваете большое количество коммитов.

```
C:\GIT\lab1.2>git log --pretty=oneline
d1339ebc7205ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD) Удалил ненужные файлы
0ee4ec64b4c36a9f60cca243f1b98c68fe70388e Добавил папку с файлом для отчета
3f929709d3a78be9a03ed2f494a870066a71630d Переименовал снимок
c37eefd550c566c605f1137f6ef4bba52da6b868 Добавил снимок
6983aa50c4eba90bec9f9d974c606b48b6a761fc Добавил информацию в .txt
c4abb7706290b1e415f4bdc1c7efb2f204d01766 Добавил текстовый файл
7c543384cae6f07e87be5ab31788e67b34766d0d Указал свои данные
b95f883d225f0e7ada3bd1e9641d1925152c27a8 Initial commit
```

Рисунок 5. Команда “git log --pretty=oneline”

Наиболее интересной опцией является format, которая позволяет указать формат для вывода информации.

```
C:\GIT\lab1.2>git log --pretty=format:"%h - %an, %ar : %s"
d1339eb - Максим, 13 minutes ago : Удалил ненужные файлы
0ee4ec6 - Максим, 14 minutes ago : Добавил папку с файлом для отчета
3f92970 - Максим, 16 minutes ago : Переименовал снимок
c37eefd - Максим, 19 minutes ago : Добавил снимок
6983aa5 - Максим, 20 minutes ago : Добавил информацию в .txt
c4abb77 - Максим, 23 minutes ago : Добавил текстовый файл
7c54338 - Максим, 25 minutes ago : Указал свои данные
b95f883 - maxim-hoodyakov, 32 minutes ago : Initial commit
```

Рисунок 6. Команда “git log --pretty=format:"%h - %an, %ar : %s"”

Опции oneline и format являются особенно полезными с опцией --graph команды log .

```
C:\GIT\lab1.2>git log --pretty=format:"%h %s" --graph
* d1339eb Удалил ненужные файлы
* 0ee4ec6 Добавил папку с файлом для отчета
* 3f92970 Переименовал снимок
* c37eefd Добавил снимок
* 6983aa5 Добавил информацию в .txt
* c4abb77 Добавил текстовый файл
* 7c54338 Указал свои данные
* b95f883 Initial commit
```

Рисунок 7. Команда “git log --pretty=format:"%h %s" --graph”

Ограничение вывода

Опции для ограничения вывода по времени, такие как --since и --until, являются очень удобными. Например, следующая команда покажет список коммитов, сделанных за последние две недели:

```

C:\GIT\lab1.2>git log --since=1.hour
commit d1339ebc7205ea513c4372d15aa6ae2a7c917726 (HEAD -> main, origin/main, origin/HEAD)
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 12:00:36 2023 +0300

    Удалил ненужные файлы

commit 0ee4ec64b4c36a9f60cca243f1b98c68fe70388e
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:59:31 2023 +0300

    Добавил папку с файлом для отчета

commit 3f929709d3a78be9a03ed2f494a870066a71630d
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:57:25 2023 +0300

    Переименовал снимок

commit c37eefd550c566c605f1137f6ef4bba52da6b868
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:54:30 2023 +0300

    Добавил снимок

commit 6983aa50c4eba90becf9d974c606b48b6a761fc
Author: Максим <max.hoodyakov@gmail.com>
Date:   Wed May 10 11:53:23 2023 +0300

    Добавил информацию в .txt

```

Рисунок 8. Команда “git log --since=1.hour”

Следующим действительно полезным фильтром является опция -S, которая принимает аргумент в виде строки и показывает только те коммиты, в которых изменение в коде повлекло за собой добавление или удаление этой строки.

```

C:\GIT\lab1.2>git log -S int
commit b95f883d225f0e7ada3bd1e9641d1925152c27a8
Author: maxim-hoodyakov <126660051+maxim-hoodyakov@users.noreply.github.com>
Date:   Wed May 10 11:41:29 2023 +0300

    Initial commit

```

Рисунок 9. Команда “git log -S int”

Операции отмены

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

```
C:\GIT\lab1.2>git add .

C:\GIT\lab1.2>git commit -m "Изменил файл README"
[main 52dcad6] Изменил файл README
1 file changed, 1 insertion(+), 1 deletion(-)

C:\GIT\lab1.2>git commit -m "Изменил файл README" --amend
[main bd9136d] Изменил файл README
Date: Wed May 10 12:22:06 2023 +0300
1 file changed, 1 insertion(+), 1 deletion(-)
```

Рисунок 10. Команда “git --amend”

Просмотр удалённых репозиторийев

Для того, чтобы просмотреть список настроенных удалённых репозиторийев, вы можете запустить команду git remote .

```
C:\GIT\lab1.2>git remote -v
origin https://github.com/maxim-hoodyakov/lab1.2.git (fetch)
origin https://github.com/maxim-hoodyakov/lab1.2.git (push)
```

Рисунок 11. Команда “git remote -v”

Просмотр удаленного репозитория

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

```
C:\GIT\lab1.2>git remote show origin
* remote origin
Fetch URL: https://github.com/maxim-hoodyakov/lab1.2.git
Push URL: https://github.com/maxim-hoodyakov/lab1.2.git
HEAD branch: main
Remote branch:
  main tracked
Local branch configured for 'git pull':
  main merges with remote main
Local ref configured for 'git push':
  main pushes to main (fast-forwardable)
```

Рисунок 12. Команда “git show origin”

Работа с тегами

Просмотр списка тегов

Просмотреть список имеющихся тегов в Git можно очень просто. Достаточно набрать команду `git tag` (параметры `-l` и `--list` опциональны):

```
C:\GIT\lab1.2>git tag  
v1
```

Рисунок 14. Команда “git tag”

Создание аннотированного тега в Git выполняется легко. Самый простой способ — это указать `-a` при выполнении команды `tag`

```
C:\GIT\lab1.2>git tag -a v1 -m "v1"
```

Рисунок 15. Создание тега.

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>`

```
C:\GIT\lab1.2>git push origin --tags  
Enumerating objects: 6, done.  
Counting objects: 100% (6/6), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (4/4), done.  
Writing objects: 100% (4/4), 471 bytes | 471.00 KiB/s, done.  
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0  
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.  
To https://github.com/maxim-hoodyakov/lab1.2.git  
* [new tag]          v1 -> v1
```

Рисунок 16. Команда “git push origin --tags”

Задание 3.

Написал небольшую программу в новом файле `programm.cpp`, сделал не менее 7-ми коммитов с 3-мя тегами. Использование команды “`git log --graph --pretty=oneline --abbrev-commit`”.

```
C:\GIT\lab1.2>git log --graph --pretty=oneline --abbrev-commit
* 979a1cd (HEAD -> main, tag: v3) Изменил .py
* 299613c (tag: v2, origin/main, origin/HEAD) Добавил .py
* bd9136d (tag: v1) Изменил файл README
* d1339eb Удалил ненужные файлы
* 0ee4ec6 Добавил папку с файлом для отчета
* 3f92970 Переименовал снимок
* c37eefd Добавил снимок
* 6983aa5 Добавил информацию в .txt
* c4abb77 Добавил текстовый файл
* 7c54338 Указал свои данные
* b95f883 Initial commit
```

Рисунок 17. История хранилища.

Задание 5.

Посмотрел содержимое коммитов командой `git show <ref>`, где <ref>:

1) HEAD: последний коммит;

```
C:\GIT\lab1.2>git show HEAD
commit 979a1cd3b2a9d0bd23ee7762ddf2185d54ed5316 (HEAD -> main, tag: v3)
Author: Максим <max.hoodyakov@gmail.com>
Date: Wed May 10 12:37:39 2023 +0300

    Изменил .py

diff --git a/Cod.py b/Cod.py
index e69de29..2f9a147 100644
--- a/Cod.py
+++ b/Cod.py
@@ -0,0 +1 @@
+print("Hello")
```

Рисунок 18. Последний коммит.

2) HEAD~1 : предпоследний коммит (и т. д.);

```
C:\GIT\lab1.2>git show HEAD~1
commit 299613cf82b0df15769fc5d4690a7a7da7f45763 (tag: v2)
Author: Максим <max.hoodyakov@gmail.com>
Date: Wed May 10 12:35:19 2023 +0300

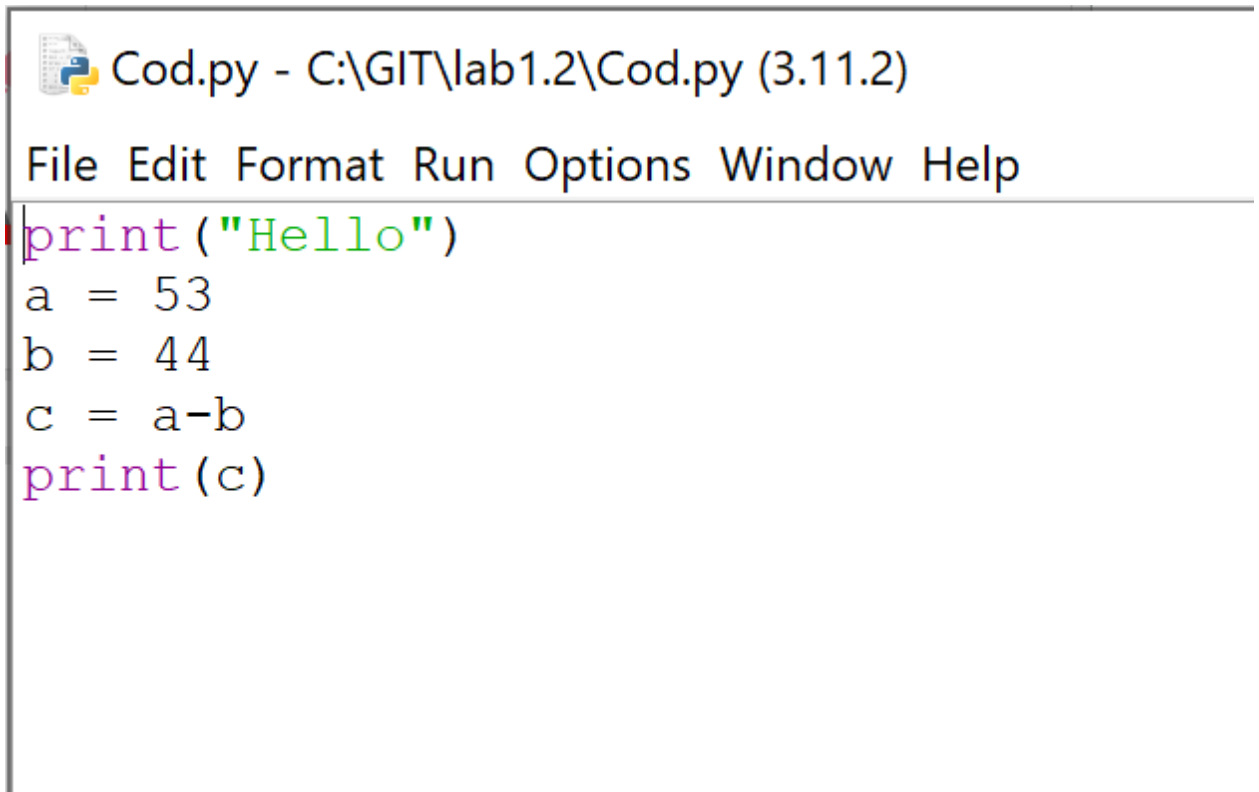
    Добавил .py

diff --git a/Cod.py b/Cod.py
new file mode 100644
index 0000000..e69de29
```

Рисунок 19. Предпоследний коммит.

3) 299613c : коммит с указанным хэшем.

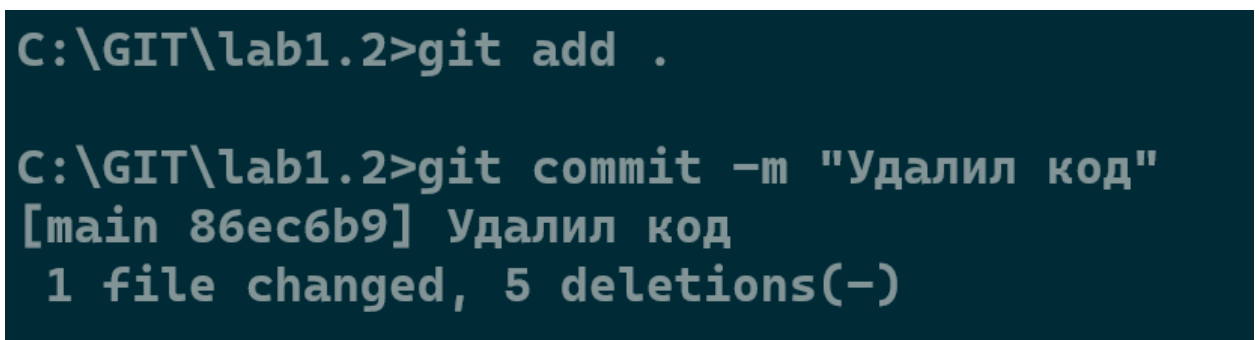
Код вернулся к прежнему состоянию.



```
Cod.py - C:\GIT\lab1.2\Cod.py (3.11.2)
File Edit Format Run Options Window Help
print("Hello")
a = 53
b = 44
c = a-b
print(c)
```

Рисунок 23. Восстановление программы.

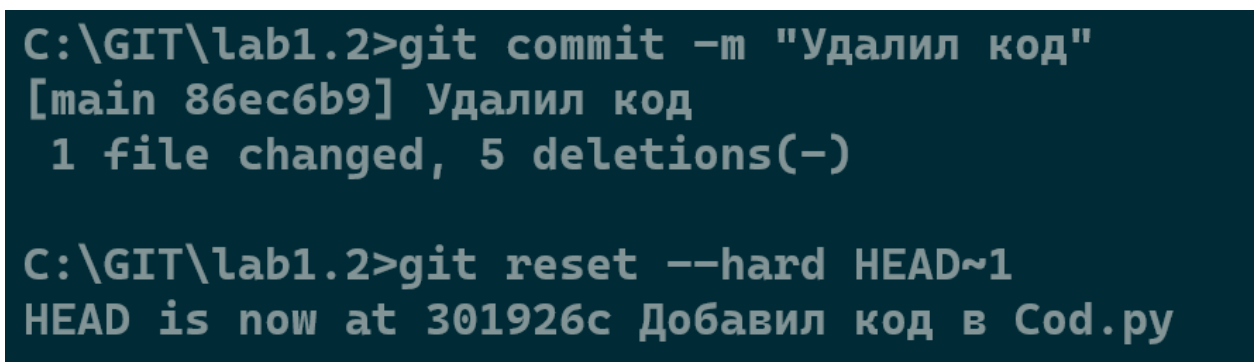
1.3. Вновь повторил пункт 1.1. и сделал коммит.



```
C:\GIT\lab1.2>git add .
C:\GIT\lab1.2>git commit -m "Удалил код"
[main 86ec6b9] Удалил код
1 file changed, 5 deletions(-)
```

Рисунок 24. Коммит.

1.4. Откатить состояние хранилища к предыдущей версии командой:
git reset --hard HEAD~1 .



```
C:\GIT\lab1.2>git commit -m "Удалил код"
[main 86ec6b9] Удалил код
1 file changed, 5 deletions(-)

C:\GIT\lab1.2>git reset --hard HEAD~1
HEAD is now at 301926c Добавил код в Cod.py
```

Рисунок 25. Возвращение к предпоследней версии коммита.

После проделанных пунктов можно сделать вывод, что можно отменять ненужные изменения, в случае если коммит не произошёл и изменения не нужны. Также, если коммит был уже сделан, то можно вернуться к предпоследней версии коммита (где было сохранение до сохранения с ненужными изменениями).

Ответы на контрольные вопросы:

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

Историю коммитов можно выполнить с помощью команды `git log`.

Дополнительные опции для просмотра истории:

`%H, %h, %T, %t, %P, %p` тд.

`-p, --stat, --shortstat, --name-only, --name-status` и тд.

2) Как ограничить вывод при просмотре истории коммитов?

Ограничить вывод при просмотре истории коммитов можно с помощью команды `git log -n`, где `n` – число последних коммитов.

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` : `git commit --amend`.

4) Как отменить индексацию файла в Git?

Отменить индексацию файла можно с помощью команды: `git reset HEAD <file>`.

5) Как отменить изменения в файле?

Отменить изменения в файле можно 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?

Выполнить просмотр удаленных репозиторий данного локального репозитория можно с помощью команды: *git remote*.

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?

Для получения данных из удалённых проектов, следует выполнить: *git fetch [remote-name]*.

Для отправки изменений в удаленный репозиторий используется команда: *git push <remote-name> <branch-name>*

10) Как выполнить просмотр удаленного репозитория?

Если хотите получить побольше информации об одном из удалённых репозиторий, вы можете использовать команду: *git remote show <remote>*.

11) Каково назначение тэгов Git?

Git имеет возможность пометить определённые моменты в истории как важные. Для таких случаев были придуманы тэги.

12) Как осуществляется работа с тэгами Git?

Просмотреть список имеющихся тегов в Git можно очень просто. Достаточно набрать команду *git tag*.

Создание аннотированного тега в Git выполняется легко. Самый простой способ — это указать *-a* при выполнении команды *tag*.

С помощью команды *git show* вы можете посмотреть данные тега вместе с коммитом.

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>*.

Для удаления тега в локальной репозитории достаточно выполнить команду `git tag -d <tagname>` .

Если вы хотите получить версии файлов, на которые указывает тег, то вы можете сделать `git checkout <tagname>` для тега.

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

Git prune-это команда, которая удаляет все файлы, недоступные из текущей ветки. Команда prune полезна, когда в вашем рабочем каталоге много файлов, которые вы не хотите хранить.

`git fetch --prune` делает то же самое: удалит ссылки на ветки, которые не существуют на удаленном компьютере.

Опция `--prune` в команде `git push` удалит ветку из удаленного репозитория, если в локальной репозитории не существует ветки с таким именем.

Вывод: исследовал базовые возможности системы контроля версий Git для работы с локальными репозиториями.