

**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 ПРАКТИЧЕСКОЙ РАБОТЕ №1
дисциплины
«Основы кроссплатформенного программирования»
Исследование
основных возможностей Git и GitHub**

Выполнила студентка группы
ИВТ-б-о-21-1
Яковлева Е.А. « » _____ 20__ г.
Подпись студента _____
Работа защищена
« » _____ 20__ г.

Проверил доцент
Кафедры инфокоммуникаций,
старший преподаватель
Воронкин Р.А.

(подпись)

Ставрополь, 2022 г.

Тема: Исследование возможностей Git для работы с локальными репозиториями.

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

Порядок выполнения работы:

1) Необходимо создать новый репозиторий, в файле README.md указать информацию о обучающемся и скопировать репозиторий себе на компьютер.

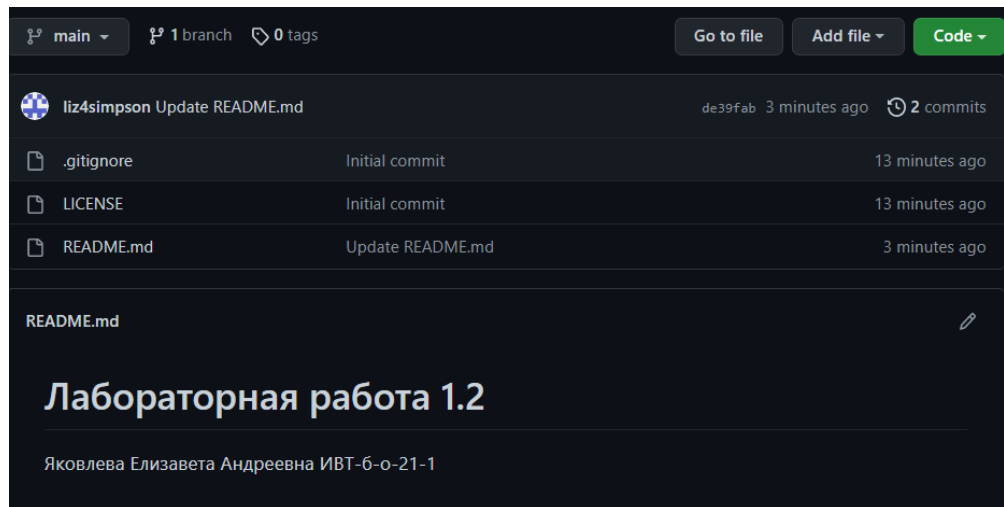


Рисунок 1 – Новый репозиторий

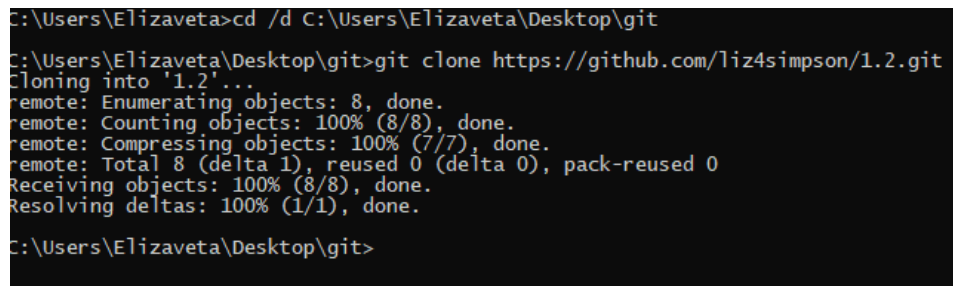


Рисунок 2 – Копирование репозитория на компьютер

2) Проработка примеров в лабораторной работе.

1. Использование команды git log:

```

C:\Users\Elizaveta\Desktop\git\1.2>git log
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore

commit de39fab3c78a96bab6dabca98ce02f30069a1b95
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:05:21 2022 +0300

    Update README.md

commit 1bce381fa16612dbdce5a3f9461bc5b1fdc1f9ec
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Tue May 24 23:55:02 2022 +0300

    Initial commit

C:\Users\Elizaveta\Desktop\git\1.2>

```

Рисунок 3 – Вывод команды git log в консоли

2. Использование команды git log -p -2 (где 2 количество выведенных записей):

```

Git CMD - "C:\Program Files\Git\cmd\git.exe" log -p -2
C:\Users\Elizaveta\Desktop\git\1.2>git log -p -2
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore

diff --git a/.gitignore b/.gitignore
index b6e4761..1d742c3 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,3 +1,14 @@
+# Created by https://www.toptal.com/developers/gitignore/api/python,powershell
+# Edit at https://www.toptal.com/developers/gitignore?templates=python,powershell
+
+### PowerShell ###
+# Exclude packaged modules
+*.zip
+
+# Exclude .NET assemblies from source
+*.dll
+
+### Python ###
+# Byte-compiled / optimized / DLL files
+__pycache__/
+*.py[cod]
@@ -20,7 +31,6 @@ parts/
sdist/
var/
wheels/
...skipping...
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore

diff --git a/.gitignore b/.gitignore
index b6e4761..1d742c3 100644
--- a/.gitignore
+++ b/.gitignore

```

Рисунок 4 – Вывод команды git log -p -2 в консоль

3. Использование команды git log --stat:

```

Git CMD
C:\Users\Elizaveta\Desktop\git\1.2>git log --stat
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore

.gitignore | 50 ++++++++++++++++++++++++++++++++++++++-----
1 file changed, 47 insertions(+), 3 deletions(-)

commit de39fab3c78a96bab6dabca98ce02f30069a1b95
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:05:21 2022 +0300

    Update README.md

README.md | 3 +-
1 file changed, 2 insertions(+), 1 deletion(-)

commit 1bce381fa16612dbdce5a3f9461bc5b1fdc1f9ec
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Tue May 24 23:55:02 2022 +0300

    Initial commit

.gitignore | 129 ++++++++++++++++++++++++++++++++++++++
LICENSE    | 21 +++++
README.md  | 1 +
3 files changed, 151 insertions(+)

C:\Users\Elizaveta\Desktop\git\1.2>

```

Рисунок 5 – Вывод команды git log --stat

4. Использование команды git log --pretty=format:"%h - %an, %ar : %s"

```
C:\Users\Elizaveta\Desktop\git\1.2>git log --pretty=format:"%h - %an, %ar : %s"
e6f50ae - liz4simpson, 13 minutes ago : Update .gitignore
de39fab - liz4simpson, 27 minutes ago : Update README.md
1bce381 - liz4simpson, 37 minutes ago : Initial commit
C:\Users\Elizaveta\Desktop\git\1.2>_
```

Рисунок 6 – Вывод команды git log --pretty в консоль

5. Использование команды git log –s

```
C:\Users\Elizaveta\Desktop\git\1.2>git log -s
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, origin/main, origin/HEAD)
Author: liz4simpson <106166123@users.noreply.github.com>
Date: Wed May 25 00:19:10 2022 +0300

    Update .gitignore

commit de39fab3c78a96bab6dabca98ce02f30069a1b95
Author: liz4simpson <106166123@users.noreply.github.com>
Date: Wed May 25 00:05:21 2022 +0300

    Update README.md

commit 1bce381fa16612dbdccc5a3f9461bc5b1fdc1f9ec
Author: liz4simpson <106166123@users.noreply.github.com>
Date: Tue May 24 23:55:02 2022 +0300

    Initial commit
```

Рисунок 7 – Вывод команды git log –s в консоль

6. Использование команды git remote -v

```
C:\Users\Elizaveta\Desktop\git\1.2>git remote -v
origin https://github.com/liz4simpson/1.2.git (fetch)
origin https://github.com/liz4simpson/1.2.git (push)
C:\Users\Elizaveta\Desktop\git\1.2>_
```

Рисунок 8 – Вывод команды git remote –v в консоль

7. Использование команды git remote show origin

```
C:\Users\Elizaveta\Desktop\git\1.2>git remote show origin
* remote origin
Fetch URL: https://github.com/liz4simpson/1.2.git
Push URL: https://github.com/liz4simpson/1.2.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 (up to date)
```

Рисунок 9 – Вывод команды git remote show origin

2) Написать небольшую программу на выбранном языке программирования(C++). Фиксировать изменения при написании программы в локальном репозитории. Должно быть сделано не менее 7 коммитов, отмеченных не менее 3 тэгами.

1-2. Создала проект в VisualStudio и написала исходную программу.

Рисунок 10 – Написанная программа

3. Создала первый тег

```
main pushed to main (up to date)

C:\Users\Elizaveta\Desktop\git\1.2>git tag -a v1.1 -m "v1.1"

C:\Users\Elizaveta\Desktop\git\1.2>git tag
v1.1

C:\Users\Elizaveta\Desktop\git\1.2>_
```

Рисунок 11 – Созданный тег

```
C:\Users\Elizaveta\Desktop\git\1.2>git show
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, tag: v1.1, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore

diff --git a/.gitignore b/.gitignore
index b6e4761..1d742c3 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,3 +1,14 @@
+# Created by https://www.toptal.com/developers/gitignore/api/python,powershell
+# Edit at https://www.toptal.com/developers/gitignore?templates=python,powershell

### PowerShell ###
# Exclude packaged modules
*.zip
# Exclude .NET assemblies from source
*.dll

### Python ###
# Byte-compiled / optimized / DLL files
__pycache__/
...skipping...
commit e6f50aed11453897442447501b177b85865b86e8 (HEAD -> main, tag: v1.1, origin/main, origin/HEAD)
Author: liz4simpson <106166123+liz4simpson@users.noreply.github.com>
Date:   Wed May 25 00:19:10 2022 +0300

    Update .gitignore
```

Рисунок 12 – Информация о созданном теге

4. Делаю 3 тега и 7 коммитов.

```
C:\Users\Elizaveta\Desktop\git\1.2>
C:\Users\Elizaveta\Desktop\git\1.2>git tag
v1.1
v1.2

C:\Users\Elizaveta\Desktop\git\1.2>git tag -a v1.3 -m
"v1.3"

C:\Users\Elizaveta\Desktop\git\1.2>git tag
v1.1
v1.2
v1.3

C:\Users\Elizaveta\Desktop\git\1.2>git status
On branch main
Your branch is ahead of 'origin/main' by 7 commits.
(use "git push" to publish your local commits)
```

Рисунок 13– Выполненное задание (3 тега и 7 коммитов)

```

C:\Users\Elizaveta\Desktop\git\1.2>git push origin --tags
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 693 bytes | 231.00 KiB/s, done.
Total 7 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 2 local objects.
To https://github.com/liz4simpson/1.2.git
 * [new tag]          v1.1 -> v1.1
 * [new tag]          v1.2 -> v1.2
 * [new tag]          v1.3 -> v1.3

```

Рисунок 14 – Отправка тегов на удаленный репозиторий

5. Просмотреть историю (журнал) хранилища командой `git log`. Например, с помощью команды `git log --graph --pretty=oneline --abbrev-commit`. Добавить скриншот консоли с выводом в отчет по лабораторной работе.

```

C:\Users\Elizaveta\Desktop\git\1.2>git log --graph --pretty=oneline --abbrev-commit
fatal: invalid --pretty format: oneline

C:\Users\Elizaveta\Desktop\git\1.2>git log --graph --pretty=oneline --abbrev-commit
* cfd714e (HEAD -> main, tag: v1.3, tag: v1.2) 7 commit
it
* 5f42b48 6 commit
* 94b4b77 5 commit
* 5ef2fb1 4 commit
* c9b7a84 3 commit
* e64303e 2 commit
* 3084e39 1 commit
* e6f50ae (tag: v1.1, origin/main, origin/HEAD) Update .gitignore
* de39fab Update README.md
* 1bce381 Initial commit

```

Рисунок 15 – Вывод в консоль

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

HEAD: последний коммит;

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

b34a0e: коммит с указанным хэшем.

```

C:\Users\Elizaveta\Desktop\git\1.2>git show Head
commit cfd714eabbc801cc286a6e2efe95b608041d9bd3 (HEAD
-> main, tag: v1.3, tag: v1.2)
Author: eeellliiizzzkkk@yandex.ru <eeellliiizzzkkk@ya
ndex.ru>
Date:   Wed May 25 01:07:38 2022 +0300

    7 commit

diff --git a/programm/test1.2.py b/programm/test1.2.p
y
index 5abc25c..581ac60 100644
--- a/programm/test1.2.py
+++ b/programm/test1.2.py
@@ -9,3 +9,6 @@ print("So,how old are you?")
    old_user=input()
    print("Im " + old_user + "years old")
    print("Ohhh,thats cool!Where you from?")
+country_user=input()
+print("I am from " + country_user)
+print("Wow, me too. Nice to meet you!")

```

Рисунок 16 – Вывод в консоль команды git show Head

```

C:\Users\Elizaveta\Desktop\git\1.2>git show Head~1
commit 5f42b48ec9dcc074956c44a45eaf2527fbffc0ac
Author: eeellliiizzzkkk@yandex.ru <eeellliiizzzkkk@ya
ndex.ru>
Date:   Wed May 25 01:05:49 2022 +0300

    6 commit

diff --git a/programm/test1.2.py b/programm/test1.2.p
y
index e60163e..5abc25c 100644
--- a/programm/test1.2.py
+++ b/programm/test1.2.py
@@ -8,3 +8,4 @@ print("Nice to meet you," + name_user
)
    print("So,how old are you?")
    old_user=input()
    print("Im " + old_user + "years old")
+print("Ohhh,thats cool!Where you from?")

```

Рисунок 17 – Вывод в консоль команды git show Head~1

```

C:\Users\Elizaveta\Desktop\git\1.2>git show cfd714e
commit cfd714eabbc801cc286a6e2efe95b608041d9bd3 (HEAD
-> main, tag: v1.3, tag: v1.2)
Author: eeellliiizzzkkk@yandex.ru <eeellliiizzzkkk@ya
ndex.ru>
Date:   Wed May 25 01:07:38 2022 +0300

    7 commit

diff --git a/programm/test1.2.py b/programm/test1.2.p
y
index 5abc25c..581ac60 100644
--- a/programm/test1.2.py
+++ b/programm/test1.2.py
@@ -9,3 +9,6 @@ print("So,how old are you?")
    old_user=input()
    print("Im " + old_user + "years old")
    print("Ohhh,thats cool!Where you from?")
+country_user=input()
+print("I am from " + country_user)
+print("Wow, me too. Nice to meet you!")

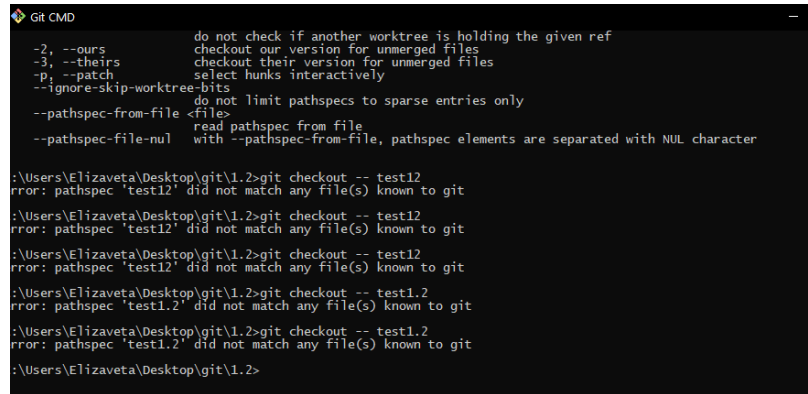
```

Рисунок 18 – Вывод в консоль коммита с указанным хэшем

7. Освоить возможность отката к данной версии.

7.1. Удалила весь код из одного из файлов программы репозитория, например main.cpp, и сохранила этот файл.

7.2. Удалила все несохраненные изменения в файле командой: `git checkout --<имя_файла>` (с ошибкой).



```
Git CMD
do not check if another worktree is holding the given ref
-2, --ours checkout our version for unmerged files
-3, --theirs checkout their version for unmerged files
-p, --patch select hunks interactively
--ignore-skip-worktree-bits do not limit pathspecs to sparse entries only
--pathspec-from-file <file> read pathspec from file
--pathspec-file-nul with --pathspec-from-file, pathspec elements are separated with NUL character

C:\Users\Elizaveta\Desktop\git\1.2>git checkout -- test12
error: pathspec 'test12' did not match any file(s) known to git

C:\Users\Elizaveta\Desktop\git\1.2>git checkout -- test12
error: pathspec 'test12' did not match any file(s) known to git

C:\Users\Elizaveta\Desktop\git\1.2>git checkout -- test12
error: pathspec 'test12' did not match any file(s) known to git

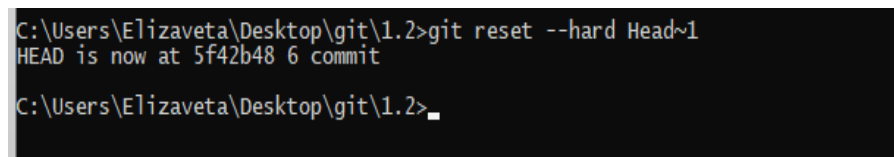
C:\Users\Elizaveta\Desktop\git\1.2>git checkout -- test1.2
error: pathspec 'test1.2' did not match any file(s) known to git

C:\Users\Elizaveta\Desktop\git\1.2>git checkout -- test1.2
error: pathspec 'test1.2' did not match any file(s) known to git

C:\Users\Elizaveta\Desktop\git\1.2>
```

Рисунок 19 – Сделал коммит после внесения изменений

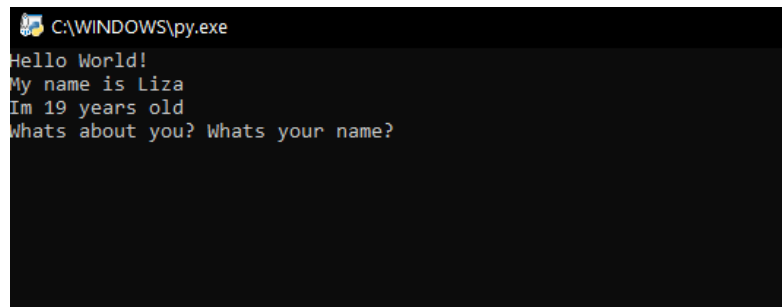
7.4. Откатить состояние хранилища к предыдущей версии командой: `git reset --hard HEAD~1`.



```
C:\Users\Elizaveta\Desktop\git\1.2>git reset --hard Head~1
HEAD is now at 5f42b48 6 commit

C:\Users\Elizaveta\Desktop\git\1.2>
```

Рисунок 20 – Вернулся к предыдущей версии



```
C:\WINDOWS\py.exe
Hello World!
My name is Liza
Im 19 years old
Whats about you? Whats your name?
```

Рисунок 21 – Вернулся к исходной программе

Выводы: освоила возможности отката к конкретным версиям программ. Содержимое программы вернулось к 6 коммиту.

8. Зафиксируйте изменения

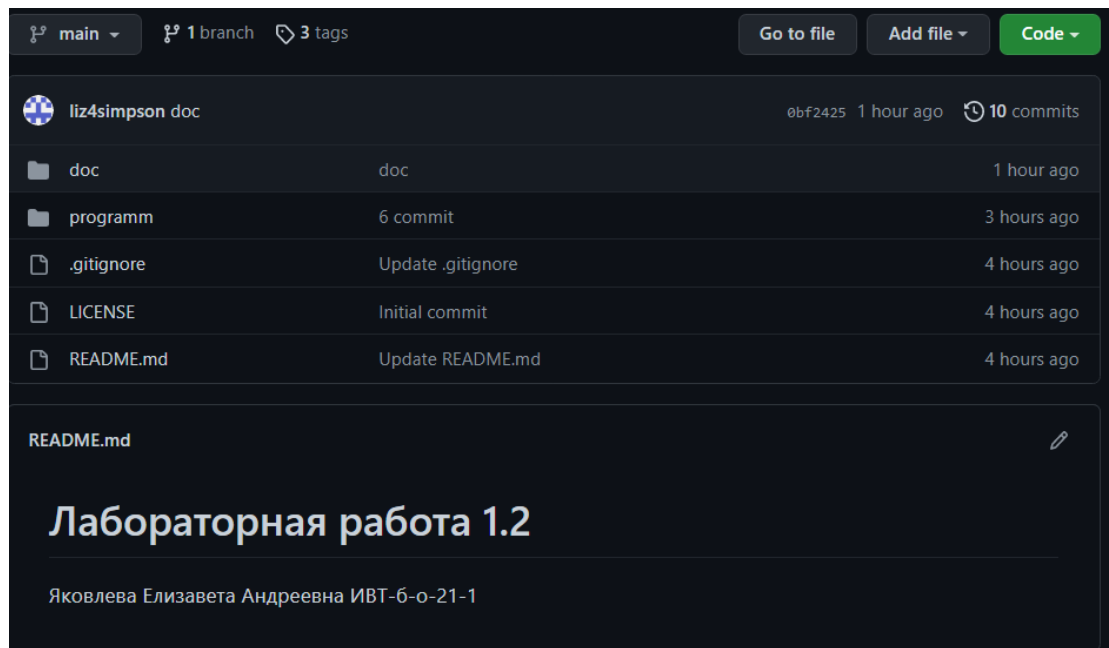


Рисунок 22 – Зафиксировала изменения на удалённом репозитории

Вопросы для защиты работы.

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

Наиболее простой и в то же время мощный инструмент для этого — команда `git log`. По умолчанию, без аргументов, `git log` выводит список коммитов созданных в данном репозитории в обратном хронологическом порядке. То есть самые последние коммиты показываются первыми.

Одна из опций, когда вы хотите увидеть сокращенную статистику для каждого коммита, вы можете использовать опцию `–stat`.

Вторая опция (одна из самых полезных аргументов) является `-p` или `--patch`, который показывает разницу (выводит патч), внесенную в каждый коммит. Так же вы можете ограничить количество записей в выводе команды; используйте параметр `–2` для вывода только двух записей (пример команды `gitlog –p -2`).

Третья действительно полезная опция это `--pretty`. Она меняет формат вывода. Существует несколько встроенных вариантов отображения. Опция `oneline` выводит каждый коммит в одну строку, что может быть очень удобным если вы просматриваете большое количество коммитов. К тому же, опции

short, full и fuller делают вывод приблизительно в том же формате, но с меньшим или большим количеством информации соответственно.

Наиболее интересной опцией является format, которая позволяет указать формат для вывода информации. Особенно это может быть полезным, когда вы хотите сгенерировать вывод для автоматического анализа — так как вы указываете формат явно, он не будет изменен даже после обновления Git. Для опции `git log --pretty=format` существуют различного рода опции для изменения формата отображения.

Опция	Описания вывода
%H	Хеш коммита
%h	Сокращенный хеш коммита
%T	Хеш дерева
%t	Сокращенный хеш дерева
%P	Хеш родителей
%p	Сокращенный хеш родителей
%an	Имя автора
%ae	Электронная почта автора
%ad	Дата автора (формат даты можно задать опцией --date=option)
%ar	Относительная дата автора
%cn	Имя коммитера
%ce	Электронная почта коммитера
%cd	Дата коммитера
%cr	Относительная дата коммитера
%s	Содержание

2. Как ограничить вывод при просмотре истории коммитов?

Для ограничения может использоваться функция `git log <n>`, где `n` число записей.

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

```
git log --since=2.weeks
```

Это команда работает с большим количеством форматов — вы можете указать определенную дату вида 2008-01-15 или же относительную дату, например 2 years 1 day 3 minutes ago.

Также вы можете фильтровать список коммитов по заданным параметрам. Опция --author дает возможность фильтровать по автору коммита, а опция --grep (показывает только коммиты, сообщение которых содержит указанную строку) искать по ключевым словам в сообщении коммита.

Функция -S показывает только коммиты, в которых изменение в коде повлекло за собой добавление или удаление указанной строки.

3. Как внести изменения в уже сделанный коммит?

Внести изменения можно с помощью команды `git commit --amend`.

Эта команда берёт индекс и применяет его к последнему коммиту. Если после последнего коммита не было никаких проиндексированных изменений (например, вы запустили приведённую команду сразу после предыдущего коммита), то состояние проекта будет абсолютно таким же и всё, что мы изменим, это комментарий к коммиту.

Для того, чтобы внести необходимые изменения - нам нужно проиндексировать их и выполнить команду `git commit --amend`.

```
git commit -m 'initial commit'
```

```
git add forgotten_file
```

```
git commit --amend
```

Эффект от выполнения этой команды такой, как будто мы не выполнили предыдущий коммит, а еще раз выполнили команду `git add` и выполнили коммит.

4. Как отменить индексацию файла в Git?

Например, вы изменили два файла и хотите добавить их в разные коммиты, но случайно выполнили команду `git add *` и добавили в индекс оба.

Как исключить из индекса один из них? Команда `git status` напомним вам:

Прямо под текстом «Changes to be committed» говорится: используйте `git reset HEAD <file>` для исключения из индекса.

5. Как отменить изменения в файле?

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

6. Что такое удаленный репозиторий Git?

Удалённый репозиторий это своего рода наше облако, в которое мы сохраняем те или иные изменения в нашей программе/коде/файлах.

7. Как выполнить просмотр удаленных репозитория данного локального репозитория?

Для того, чтобы просмотреть список настроенных удалённых репозитория, необходимо запустить команду `git remote`.

Также можно указать ключ `-v`, чтобы просмотреть адреса для чтения и записи, привязанные к репозиторию. Пример: `git remote -v`

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?

Если необходимо получить изменения, которые есть у Пола, но нету у вас, вы можете выполнить команду `git fetch <Название репозитория>`. Важно отметить, что команда `git fetch` забирает данные в ваш локальный репозиторий, но не сливает их с какими-либо вашими наработками и не модифицирует то, над чем вы работаете в данный момент. Вам необходимо вручную слить эти данные с вашими, когда вы будете готовы.

Если ветка настроена на отслеживание удалённой ветки, то вы можете использовать команду `git pull` чтобы автоматически получить изменения из удалённой ветки и слить их со своей текущей. Выполнение `git pull`, как правило, извлекает (fetch) данные с сервера, с которого вы изначально клонировали, и автоматически пытается слить (merge) их с кодом, над которым вы в данный момент работаете.

Чтобы отправить изменения на удалённый репозиторий необходимо отправить их в удалённый репозиторий. Команда для этого действия простая: `git push <remote-name> <branch-name>`.

10. Как выполнить просмотр удаленного репозитория?

Для просмотра удалённого репозитория, можно использовать команду `git remote show <remote>`.

11. Каково назначение тэгов Git?

Теги - это ссылки указывающие на определённые версии кода/написанной программы. Они удобны чтобы в случае чего вернуться к нужному моменту. Также при помощи тегов можно помечать важные моменты.

12. Как осуществляется работа с тэгами Git?

Просмотреть наличие тегов можно с помощью команды: `git tag`.

А назначить (указать, добавить тег) можно с помощью команды `git tag -a v1.4(версия изначальная) -m "Название"`.

С помощью команды `git show` вы можете посмотреть данные тега вместе с коммитом: `git show v1.4`.

Отправка тегов, 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 отправки всех тегов можно использовать команду `git push origin tags`.

Для удаления тега в локальном репозитории достаточно выполнить команду `git tag -d <tagname>`. Например, удалить созданный ранее легко весный тег можно следующим образом: `git tag -d v1.4-lw`

Для удаления тега из внешнего репозитория используется команда `git push origin --delete <tagname>`.

Если вы хотите получить версии файлов, на которые указывает тег, то вы можете сделать `git checkout` для тега пример: `git checkout -b version2 v2.0.0`.

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

Git fetch --prune команда получения всех изменений с репозитория GitHub.

В команде git push --prune удаляет удаленные ветки, у которых нет локального аналога.

Вывод: исследовала базовые возможности системы контроля версий git для работы с локальными репозиториями. Также, благодаря созданию тегов и после изменения файлов освоила возможность отката к заданной версии