

**Оглавление**

[**Часть 1. Основные команды git** 2](#_Toc129217647)

[**Часть 2. Системы управления репозиториями** 10](#_Toc129217648)

[**Часть 3. Работа с ветвлением и оформление кода** 16](#_Toc129217649)

[**Контрольные вопросы:** 22](#_Toc129217650)

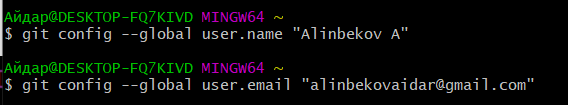
[**Вывод:** 24](#_Toc129217651)

# **Часть 1. Основные команды git**

Пройдите курс на сайте https://githowto.com/ (до раздела 22).

Обязательно должны быть выполнены следующие действия:

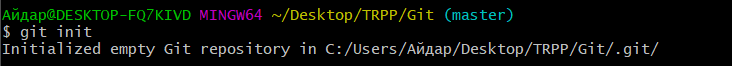
1. Установите и настройте клиент git на своей рабочей станции.



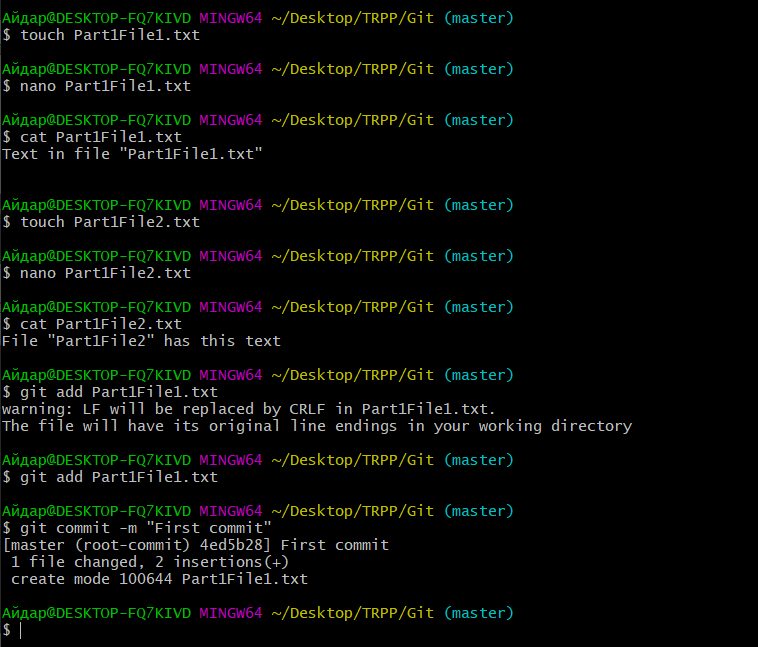
1. Создайте локальный репозиторий и добавьте в него несколько

файлов.

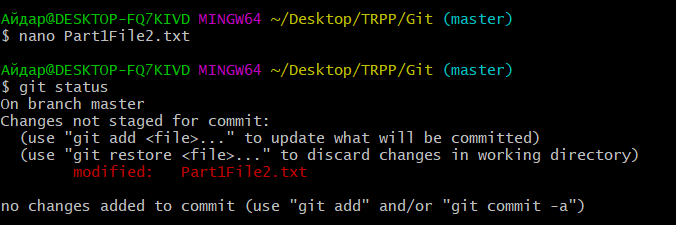
Создал локальный репозиторий



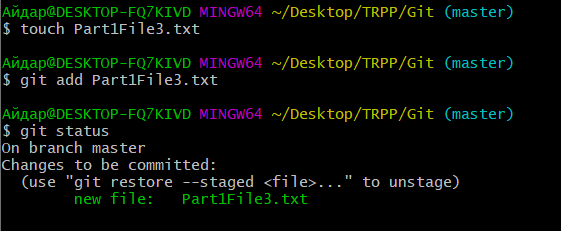
Создал файлы и добавил в репозиторий



1. Внесите изменения в один из файлов.

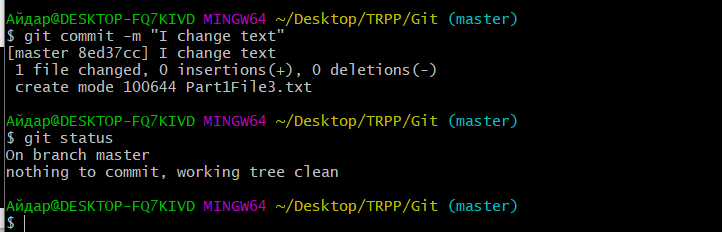


1. Проиндексируйте изменения и проверьте состояние.



1. Сделайте коммит того, что было проиндексировано в

репозиторий. Добавьте к коммиту комментарий.



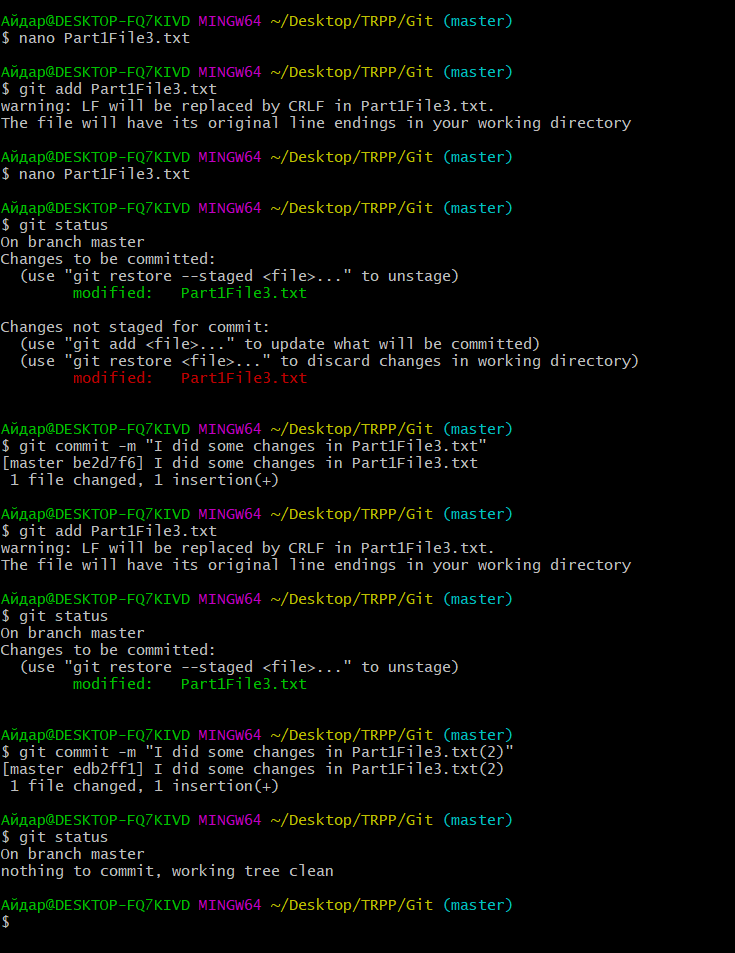
1. Измените еще один файл. Добавьте это изменение в индекс git.

Измените файл еще раз. Проверьте состояние и произведите коммит

проиндексированного изменения. Теперь добавьте второе изменение в

индекс, а затем проверьте состояние с помощью команды git status.

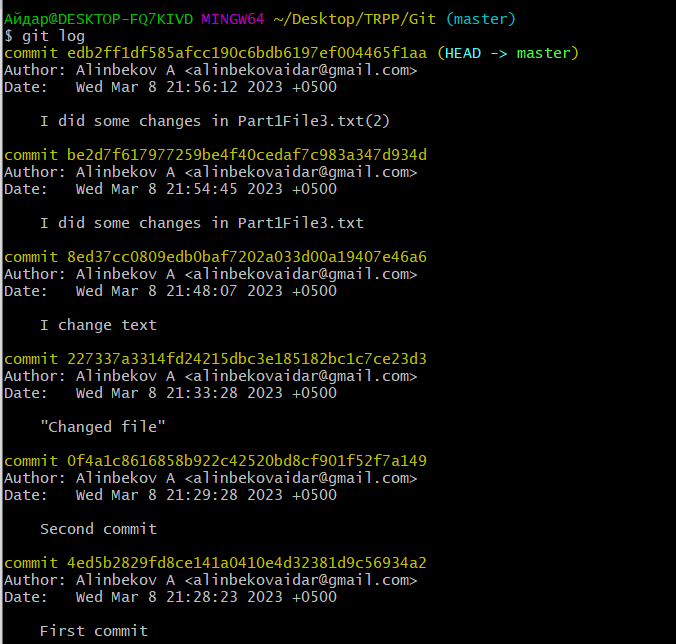
Сделайте коммит второго изменения.

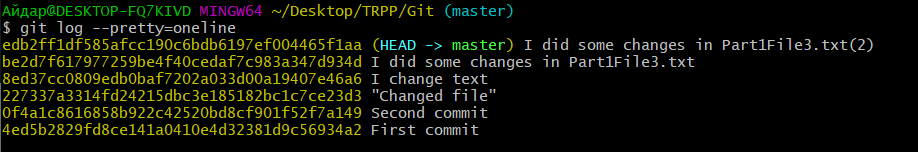


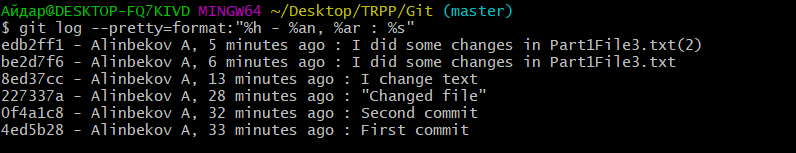
1. Просмотрите историю коммитов с помощью команды git log.

Ознакомьтесь с параметрами команды и используйте некоторые из них

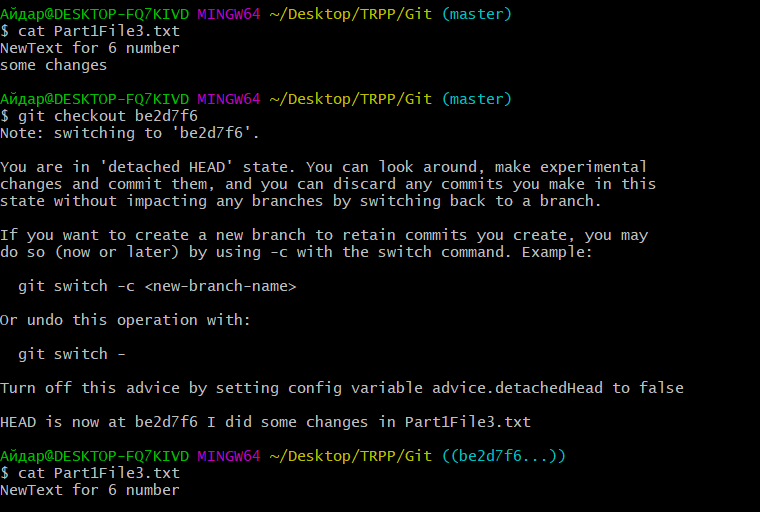
для различного формата отображения истории коммитов.





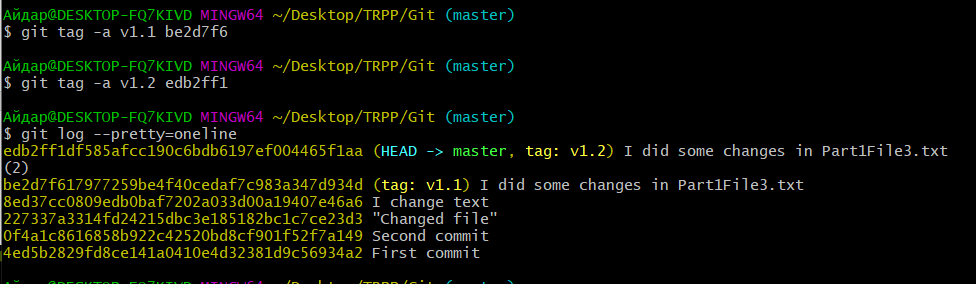


1. Верните рабочий каталог к одному из предыдущих состояний.



1. Изучите, как создавать теги для коммитов для использования в

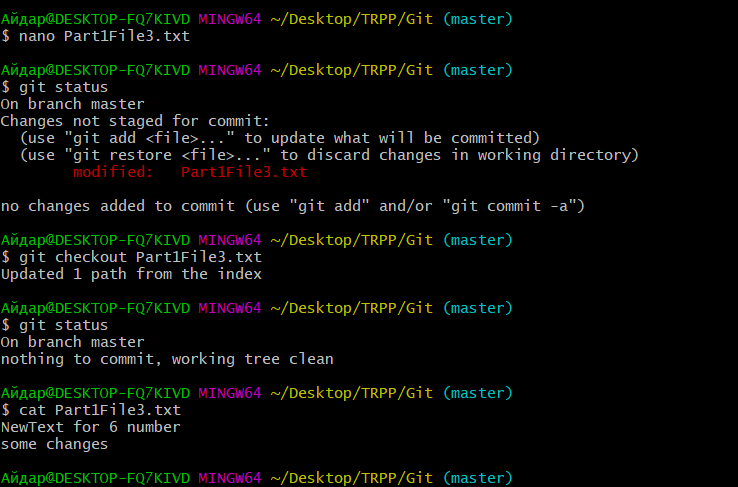
будущем.



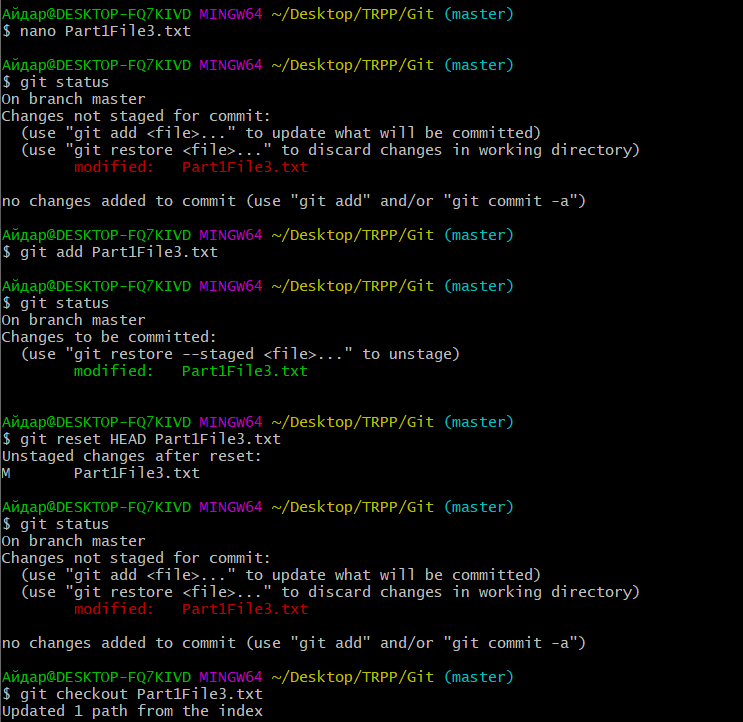
1. Отмените некоторые изменения в рабочем каталоге (до и после

индексирования).

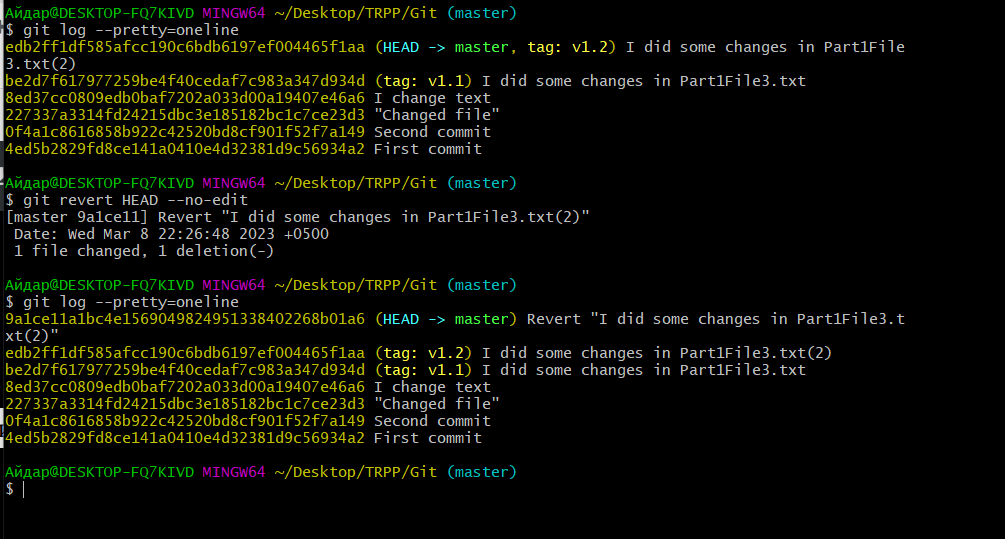
До индексирования:



После индексирования:

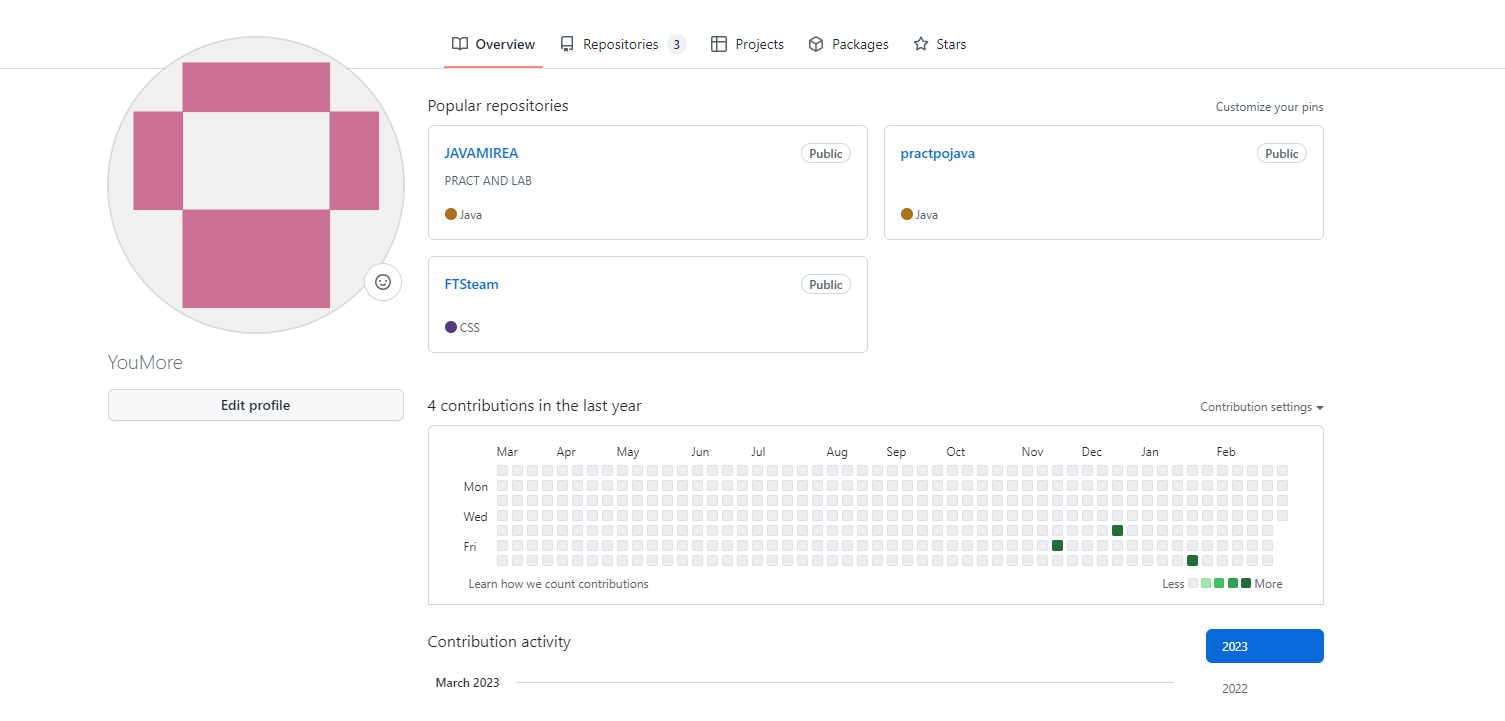


1. Отмените один из коммитов в локальном репозитории.

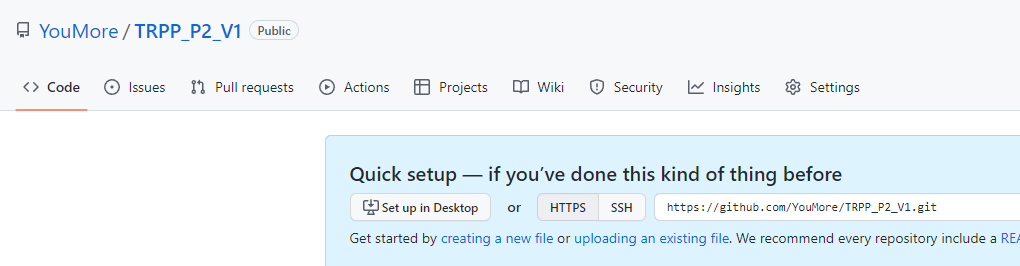


# **Часть 2. Системы управления репозиториями**

1. Создайте аккаунты на GitHub.

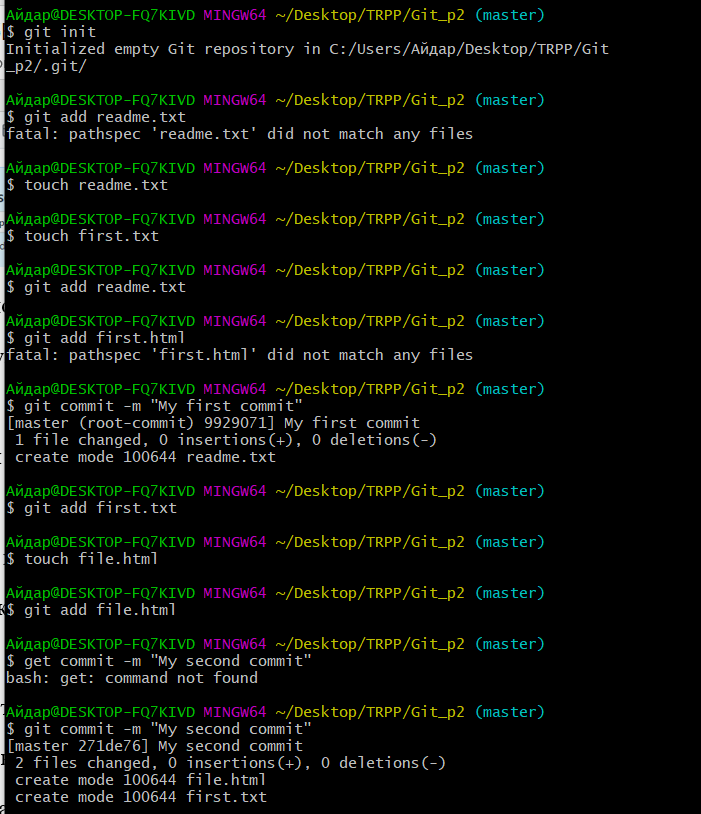


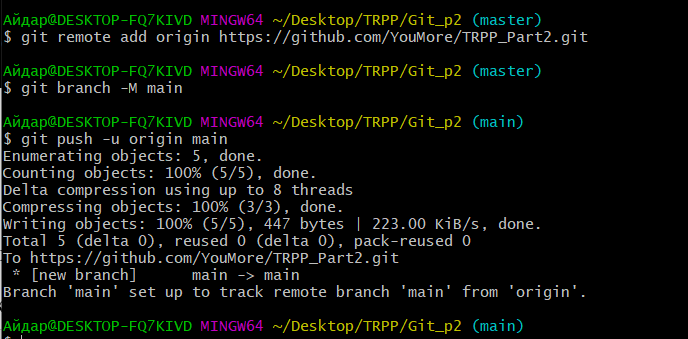
1. Создайте репозиторий согласно варианту.

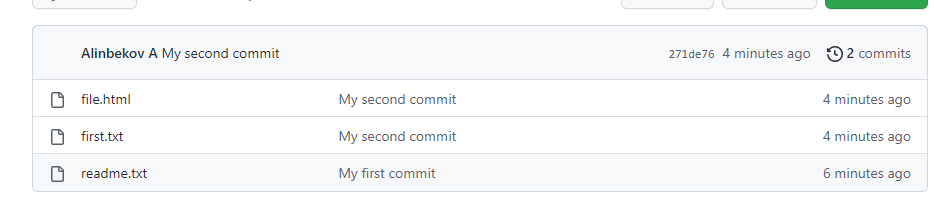


1. Создайте новый локальный репозиторий с несколькими

файлами на рабочей станции и загрузите его содержимое на GitHub.

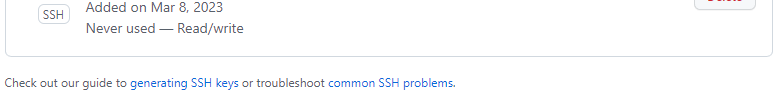






1. Чтобы избежать ввода логина и пароля, создайте SSH-ключ

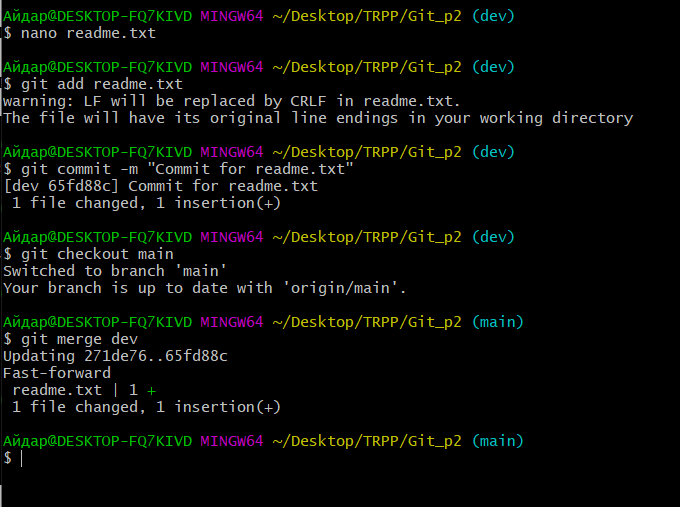
для авторизации.



1. Создайте в репозитории новую ветку. Произведите в ней

несколько изменений и слейте с веткой master.

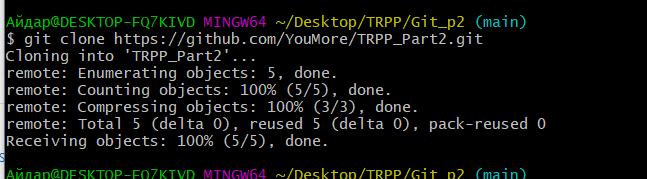




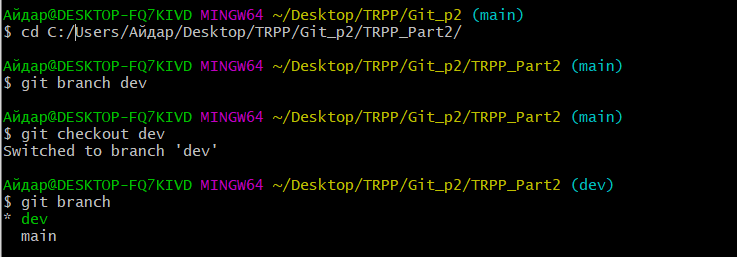
1. Выполните цепочку действий в репозитории, согласно варианту.

**Вариант 1:**

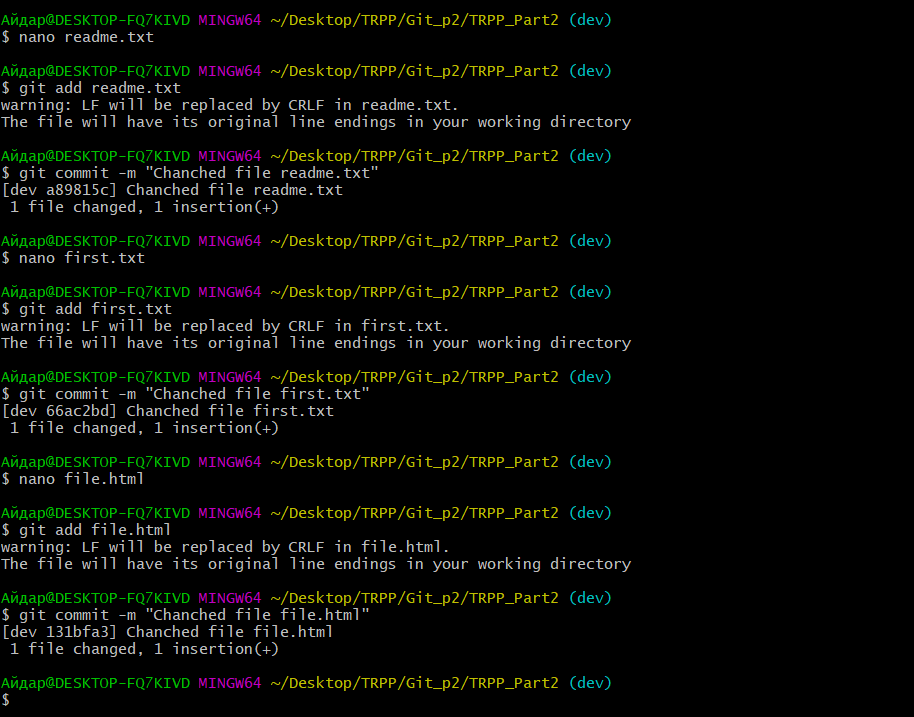
1. Клонируйте непустой удаленный репозиторий на локальную машину



1. Создайте новую ветку и выведите список всех веток

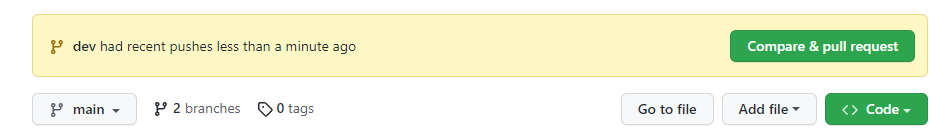


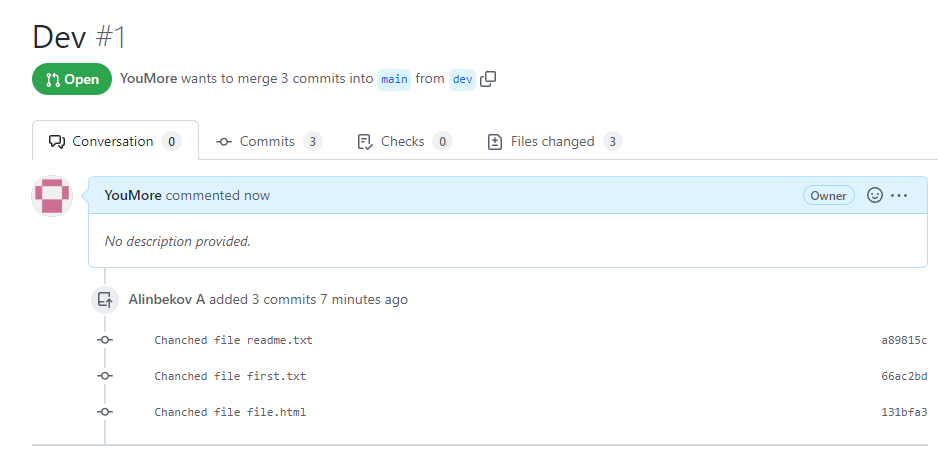
1. Произведите 3 коммита в новой ветке



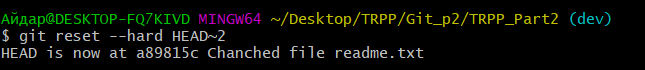
1. Выгрузите изменения в удаленный репозиторий

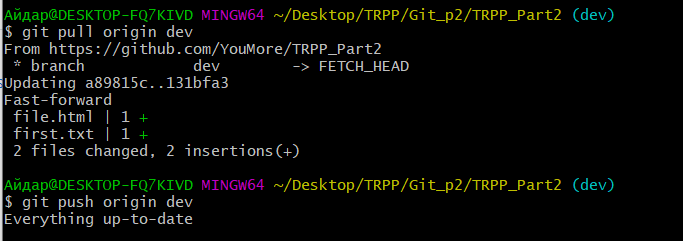




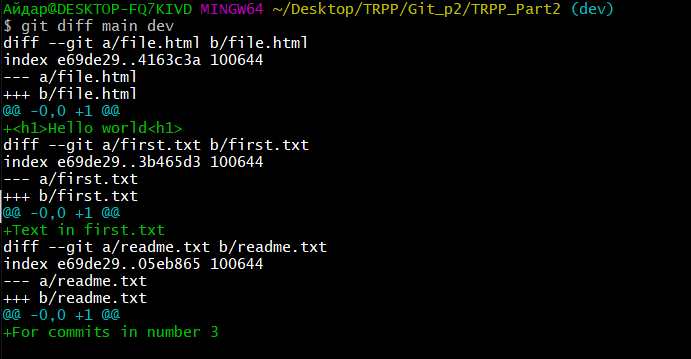


1. Откатите ветку обратно на 2 коммита

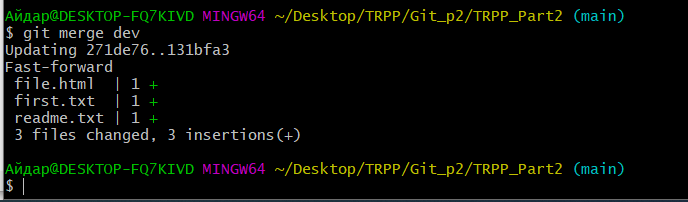




1. Выведите в консоли различия между веткой master и новой веткой



1. Слейте новую ветку с master при помощи merge



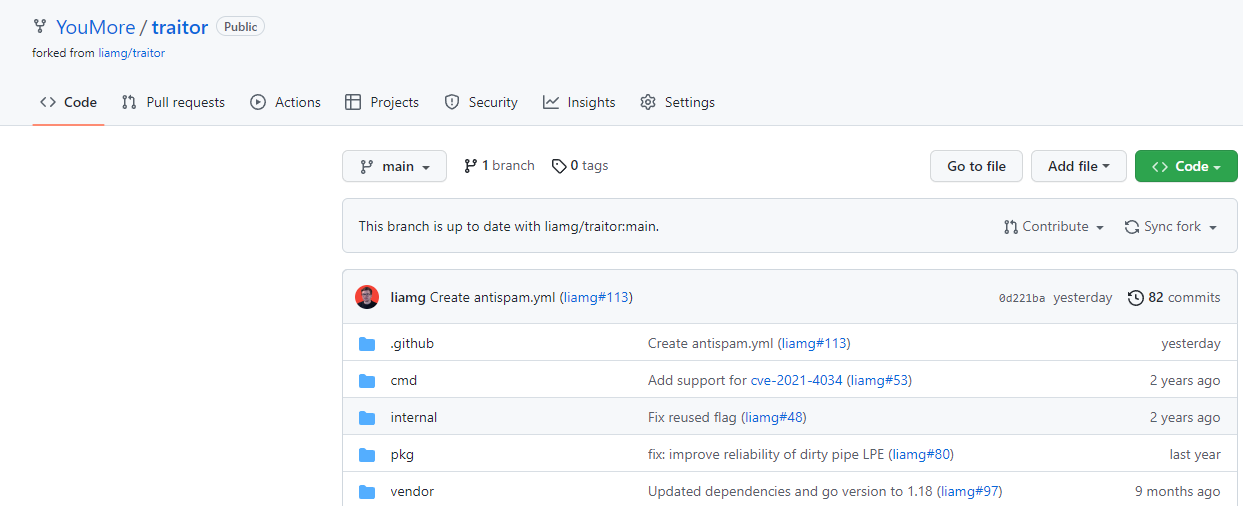
# **Часть 3. Работа с ветвлением и оформление кода**

Продолжите проходить курс на сайте https://githowto.com/ (начиная с

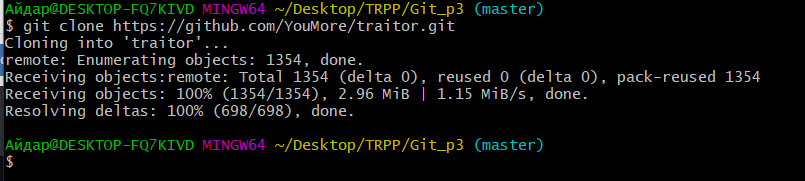
раздела 22).

Создайте и разрешите конфликты в репозитории, согласно варианту:

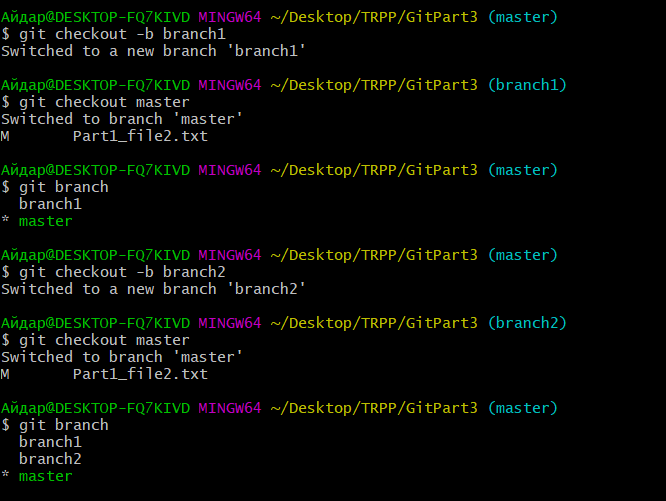
1. Сделайте форк репозитория в соответствии с Вашим вариантом.



1. Склонируйте его (форк репозитория) на локальную машину.



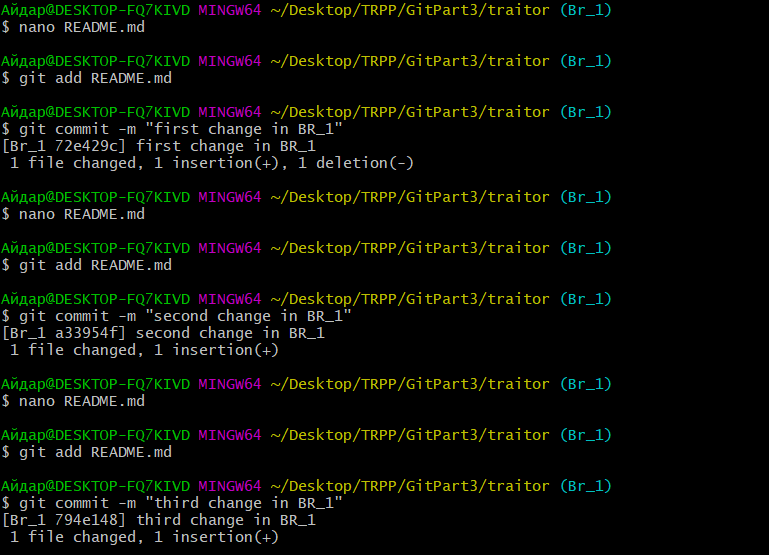
1. Создайте две ветки branch1 и branch2 от последнего коммита в master'е.



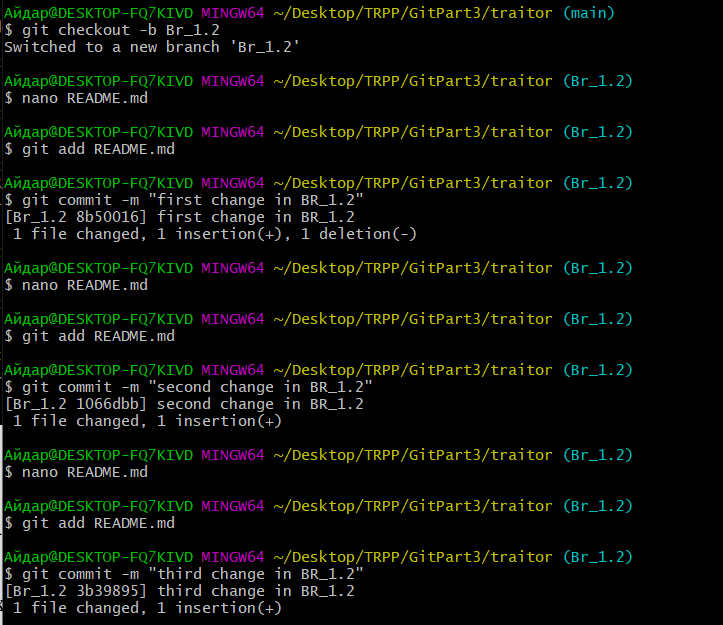
1. Проведите по 3 коммита в каждую из веток, которые меняют один и тот

же кусочек файла.

Branch1:

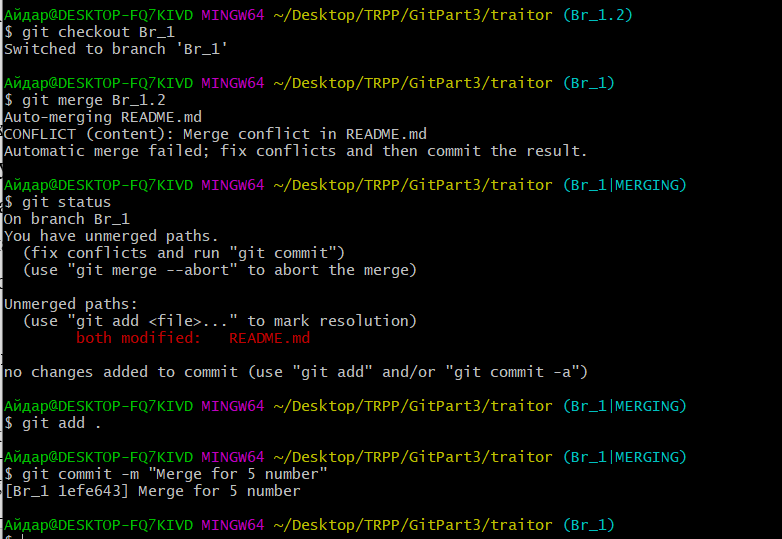


Branch2:

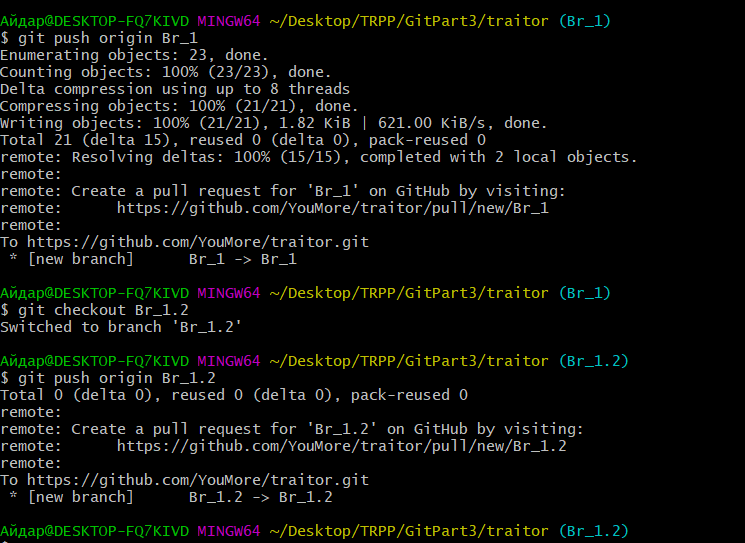


1. Выполните слияние ветки branch1 в ветку branch2, разрешив при этом

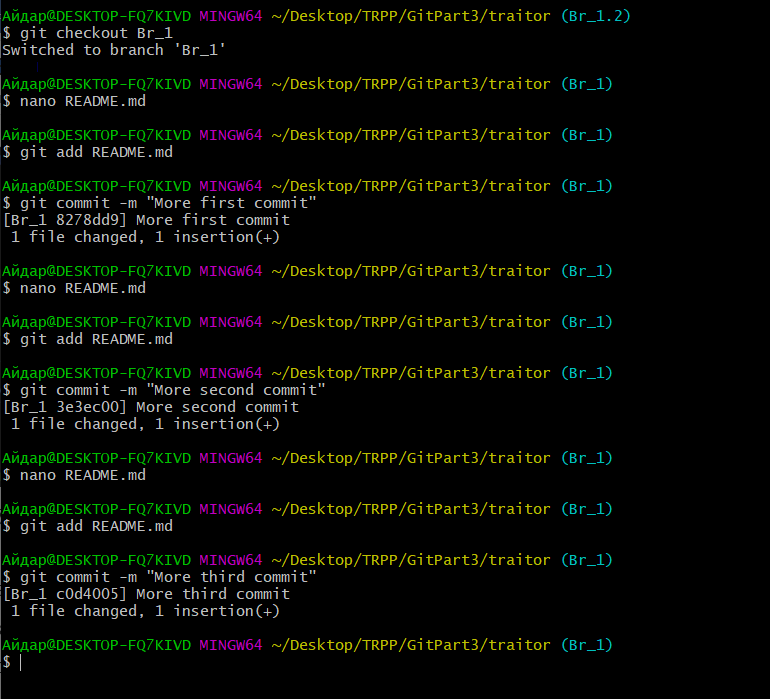
конфликты.



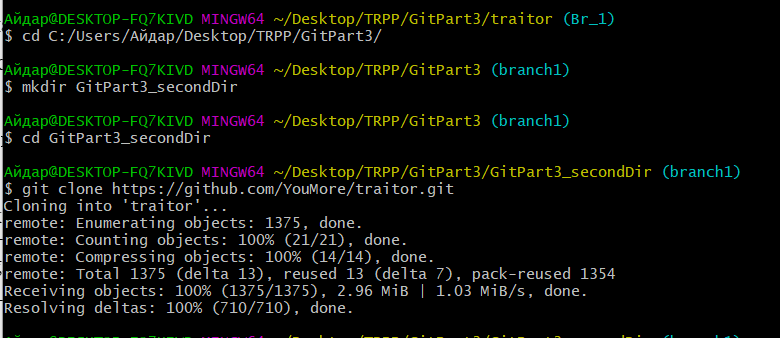
1. Выгрузите все изменения во всех ветках в удаленный репозиторий.



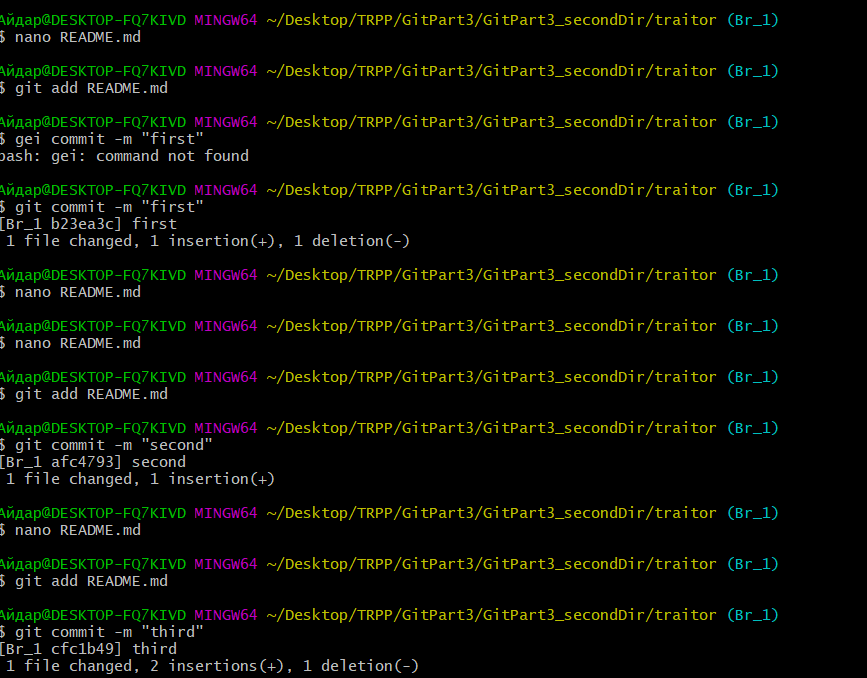
1. Проведите еще 3 коммита в ветку branch1.



1. Склонируйте репозиторий еще раз в другую директорию.



1. В новом клоне репозитории сделайте 3 коммита в ветку branch1.



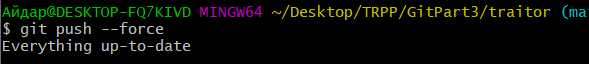
1. Выгрузите все изменения из нового репозитория в удаленный

репозиторий.

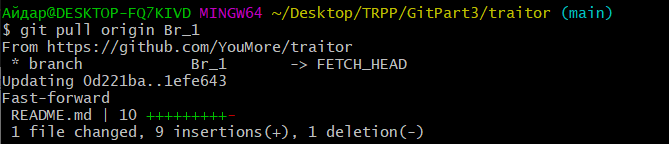


1. Вернитесь в старый клон с репозиторием, выгрузите изменения с

опцией –force



1. Получите все изменения в новом репозитории



|  |  |
| --- | --- |
| 1 | https://github.com/liamg/traitor |

# **Контрольные вопросы:**

2. К какому типу систем контроля версий относится Git?

Git — это распределённая система версий.

3. Что такое репозиторий Git? Репозиторий Git или репозиторий — это папка, в которую Git отслеживает изменения. На компьютере может быть любое количество репозиториев, каждое из которых хранится в собственной папке. Каждый репозиторий Git в системе является независимым, поэтому изменения, сохраненные в одном репозитории Git, не влияют на содержимое другого.

4*.* Что такое коммит?Контрольные точки называются коммитами. Один коммит — это пакет изменений, хранящий информацию с добавленными, отредактированными или удалёнными файлами кода. В один коммит принято добавлять не более десяти изменений — так получается длинная история версий, которая позволяет в случае ошибки откатиться с минимальной потерей работоспособного кода.

5. Что такое ветка в репозитории Git?

Под веткой принято понимать независимую последовательность коммитов в хронологическом порядке. Однако конкретно в Git реализация ветки выполнена как указатель на последний коммит в рассматриваемой ветке. После создания ветки уже новый указатель ссылается на текущий коммит. Имя основной ветки Git-проекта по умолчанию — master (однако зачастую бывает main, например, в GitHub), она появляется сразу при инициализации репозитория. Эта ветка ничем не отличается от остальных и также ее можно переименовать, но по договоренности master принято считать главной веткой в проекте.

6. Что такое тег в репозитории Git? Теги — это ссылки, указывающие на определенные точки в истории Git. Команда git tag обычно используется для захвата некой точки в истории, которая используется для релиза нумерованной версии (например, v1.0.1). Теги похожи на неизменяемые ветки, но они, в отличие от веток, не имеют истории коммитов после создания.

7. Что такое слияние двух веток?

Слияние веток – это перенос изменений с одной ветки на другую. При этом слияние не затрагивает сливаемую ветку, то есть она остается в том же состоянии, что позволяет нам потом продолжить работу с ней.

11. Что делает команда git status?

Команда git status отображает состояние рабочего каталога и раздела проиндексированных файлов. С ее помощью можно проверить индексацию изменений и увидеть файлы, которые не отслеживаются Git. Информация об истории коммитов проекта не отображается при выводе данных о состоянии.

13. Что делает команда git log? Команда git log выводит историю комитов.

# **Вывод:**

В ходе выполнения практического задания были достигнуты следующие цели:

* Научились работать с системой контроля версий Git;
* Прошли курс на сайте <https://githowto.com/>.