**ОГЛАВЛЕНИЕ**

[Часть 1. Основные команды git 2](#_bookmark0)

[Часть 2. Системы управления репозиториями 12](#_bookmark1)

[Часть 3. Работа с ветвлением и оформление кода 19](#_bookmark2)

[Ответы на вопросы 26](#_bookmark3)

[Выводы 28](#_bookmark4)

# Часть 1. Основные команды git

# Установите и настройте клиент git на своей рабочей станции

На рисунке 1 показан процесс настройки конфигурации git.

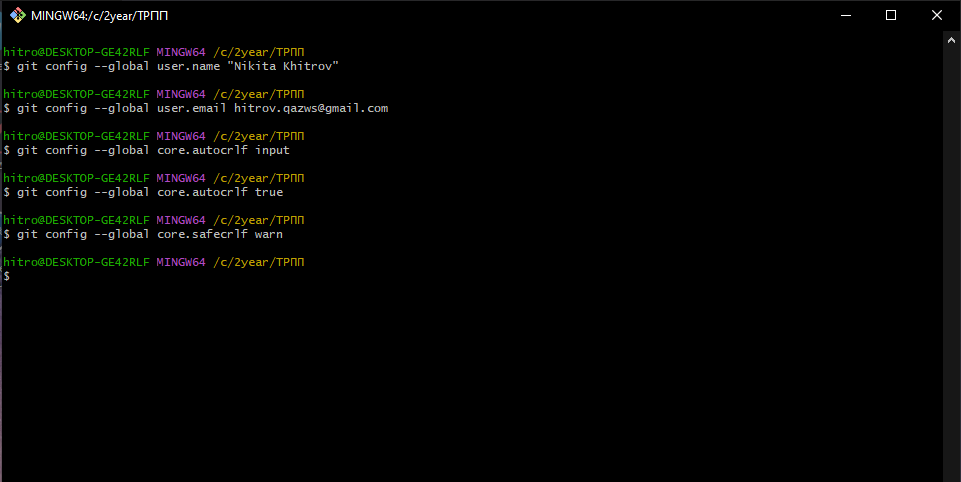


Рисунок 1 – Настройка конфигурации git

# Создайте локальный репозиторий и добавьте в него несколько файлов.

На рисунке 2 показан процесс инициализации пустого репозитория и добавления файлов в папку проекта.



Рисунок 2 – Инициализация репозитория git

# Внесите изменения в один из файлов.

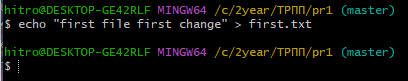


Рисунок 3 – Внесение изменений в файл

# Проиндексируйте изменения и проверьте состояние.

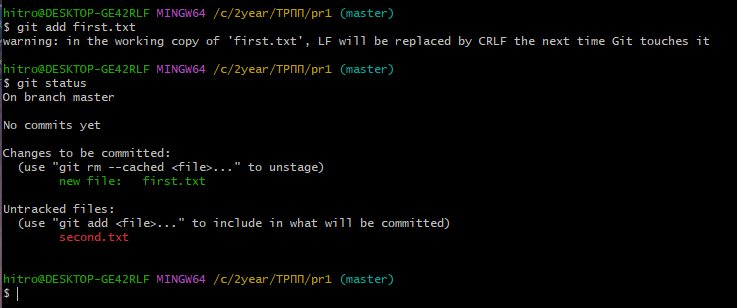


Рисунок 4 – Индексация изменений

# Сделайте коммит того, что было проиндексировано в репозиторий. Добавьте к коммиту комментарий.

На рисунке 5 показан процесс изменения файла index.html, индексация изменений и создание коммита.

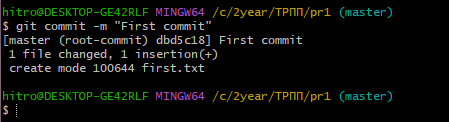


Рисунок 5 – изменение файла и создание коммита

# 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.

На рисунке 6 показан процесс создания коммитов с разными изменениями одного файла.

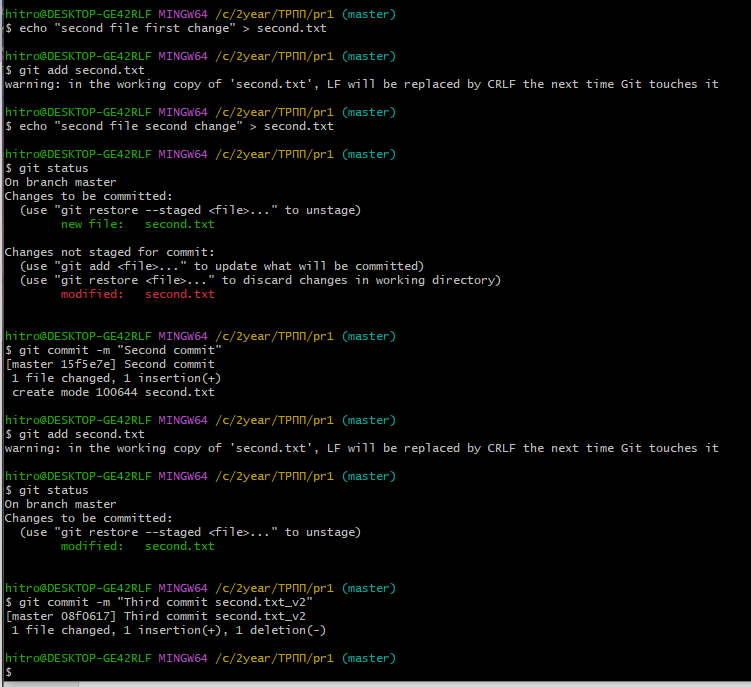


Рисунок 6 – Изменение другого файла, индесация, повторное изменение и тд.

# Просмотрите историю коммитов с помощью команды git log. Ознакомьтесь с параметрами команды и используйте некоторые из них для различного формата отображения истории коммитов.

На рисунке 7 показана пользовательская конфигурация команд.

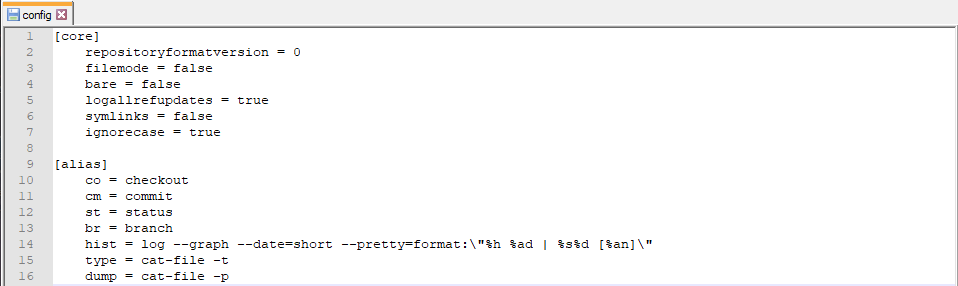


Рисунок 7 – пользовательская конфигурация команд

На рисунке 8 показана пользовательская конфигурация команд.

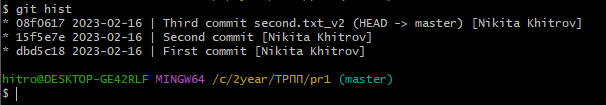


Рисунок 8 – История коммитов

# Верните рабочий каталог к одному из предыдущих состояний.

На рисунке 9 показан процесс возврата репозитория к одному из предыдущих состояний.

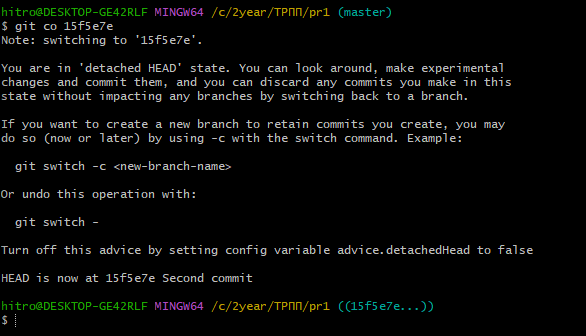


Рисунок 9 – Возврат репозитория к предыдущему состоянию

# Изучите, как создавать теги для коммитов для использования в будущем.

На рисунке 10 показан процесс создания тегов для коммита.

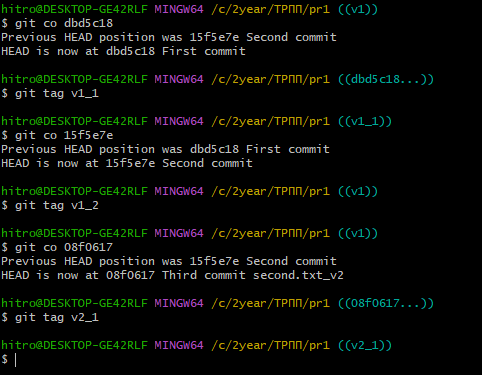


Рисунок 10 – Создание тегов для коммита

# Отмените некоторые изменения в рабочем каталоге (до и после индексирования).

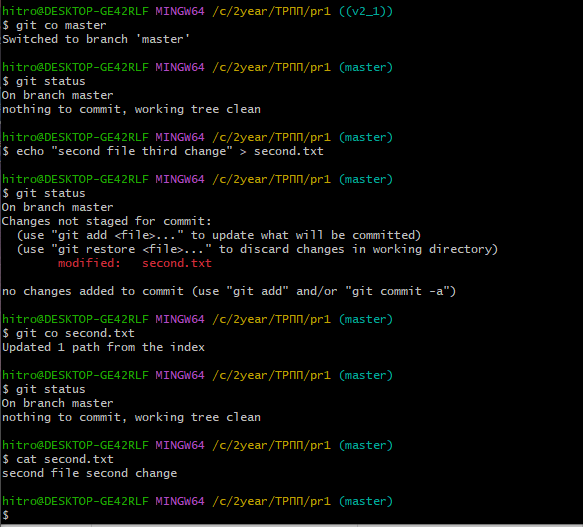
На рисунке 11 показан процесс отмены изменений до индексации.

Рисунок 11 – Отмена изменений до индексации

На рисунке 12 показан процесс отмены изменений после индексации.

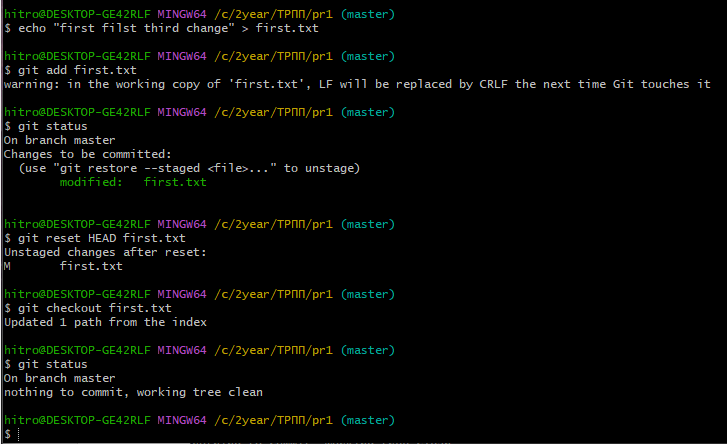
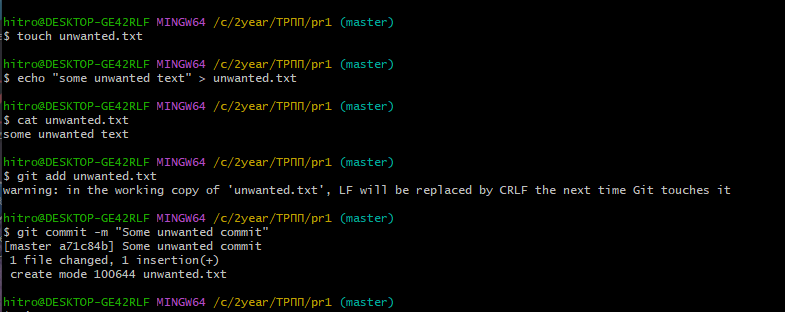


Рисунок 12 – Отмена проиндексированных изменений

# Отмените один из коммитов в локальном репозитории.

На рисунке 14 показан процесс создания нежелаемого коммита.



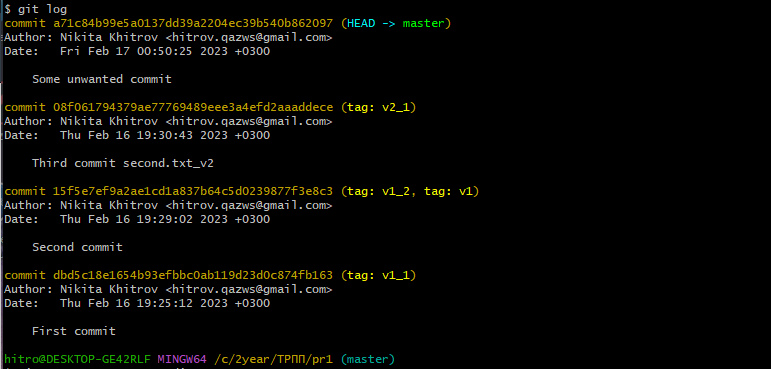


Рисунок 14 – создание нежелаемого коммита

На рисунке 15 показан процесс удаления последнего коммита.

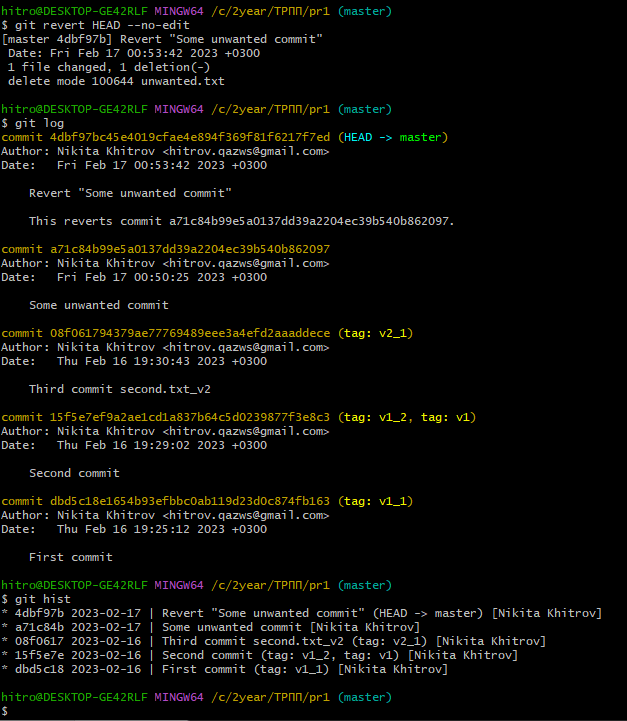


Рисунок 15 – Удаление последнего коммита

# Часть 2. Системы управления репозиториями Вариант 3

1. **Создайте аккаунт на GitHub**

На рисунке 16 показан созданный аккаунт на GitHub.

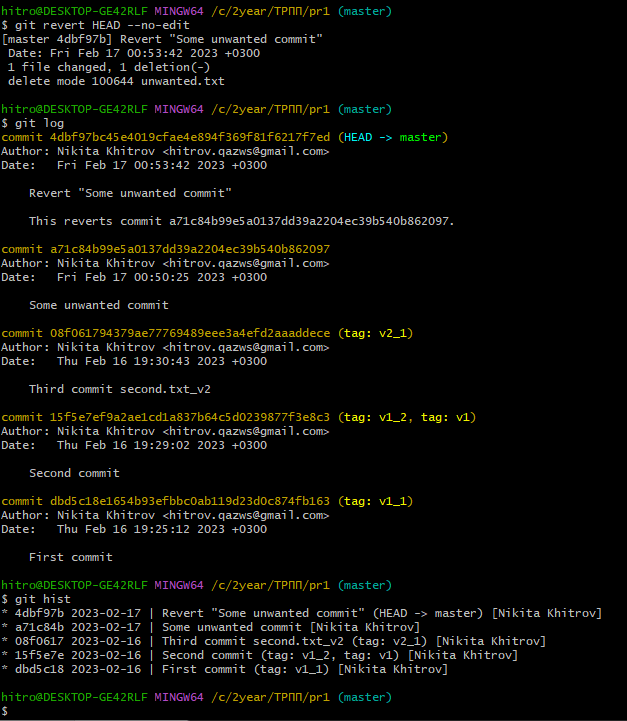


Рисунок 16 – Аккаунт GitHub

# Создайте репозиторий согласно варианту

На рисунке 17 показан процесс создания удалённого репозитория.

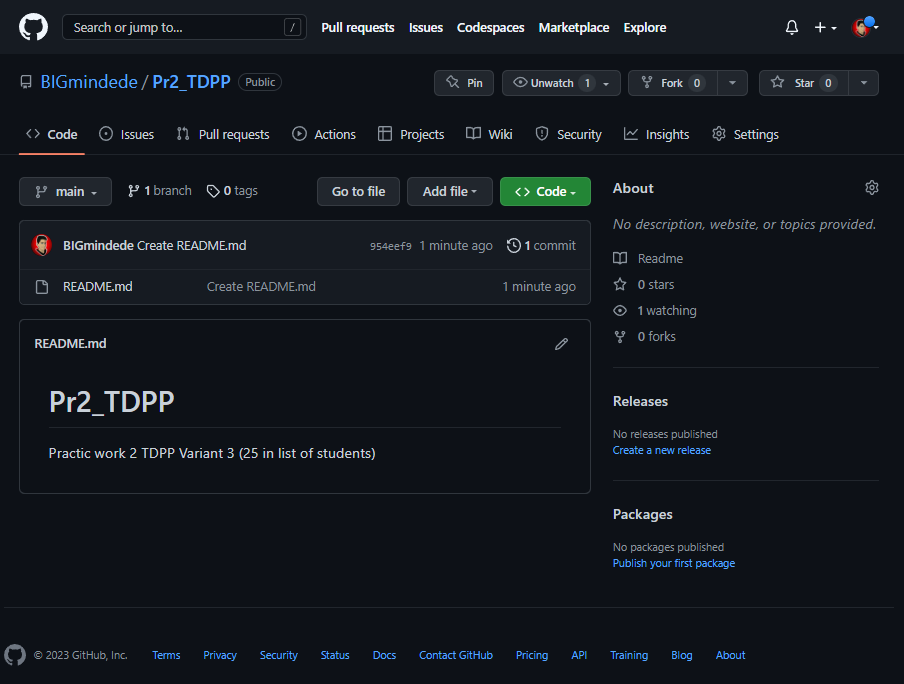


Рисунок 17 – Создание удалённого репозитория

# Создайте новый локальный репозиторий с несколькими файлами на рабочей станции и загрузите его содержимое на GitHub

На рисунке 17 показан процесс загрузки существующего репозитория на GitHub.

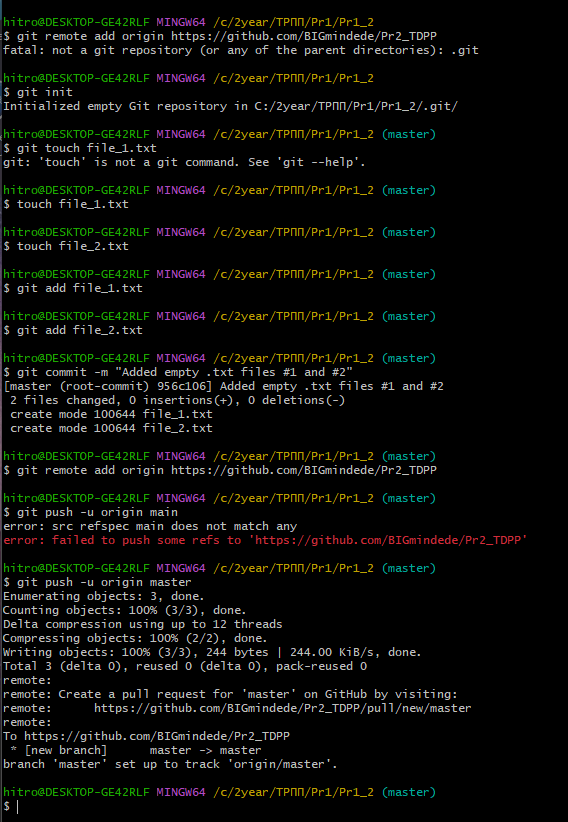


Рисунок 17 – Загрузка репозитория на GitHub

# Чтобы избежать ввода логина и пароля, создайте SSH-ключ для авторизации

На рисунке 18 показан процесс генерации SSH ключа.

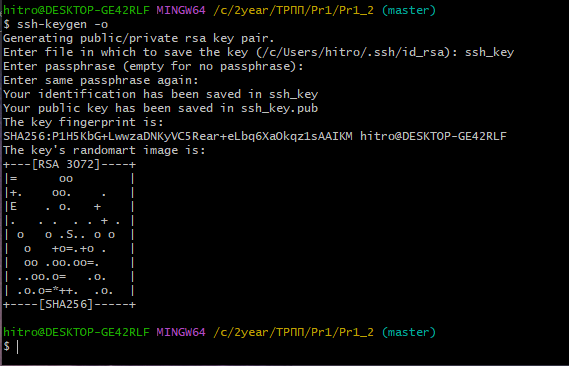


Рисунок 18 – Генерация SSH ключа

# Создайте в репозитории новую ветку. Произведите в ней несколько изменений и слейте с веткой master

На рисунках 19-20 показан процесс создания новой ветки и слияние её с веткой master.

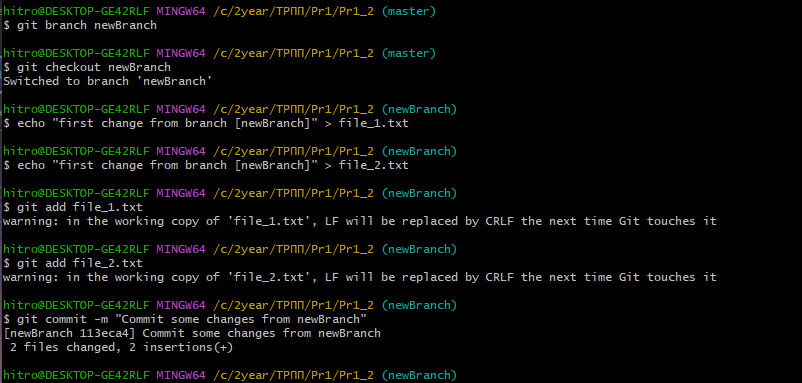


Рисунок 19 – Создание новой ветки и первый коммит в нее

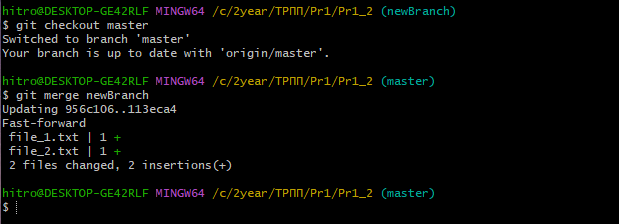


Рисунок 19 – Слияние новой ветки с веткой master

# Клонируйте непустой удаленный репозиторий на локальную машину

На рисунке 21 показан процесс клонирования удалённого репозитория.

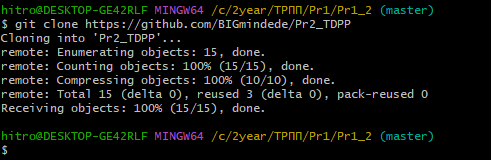


Рисунок 21 – Клонирование репозитория

# Создайте тег указывающий на последний коммит в ветке

На рисунке 22 показан процесс создания тега, указывающего на последний коммит в ветке.

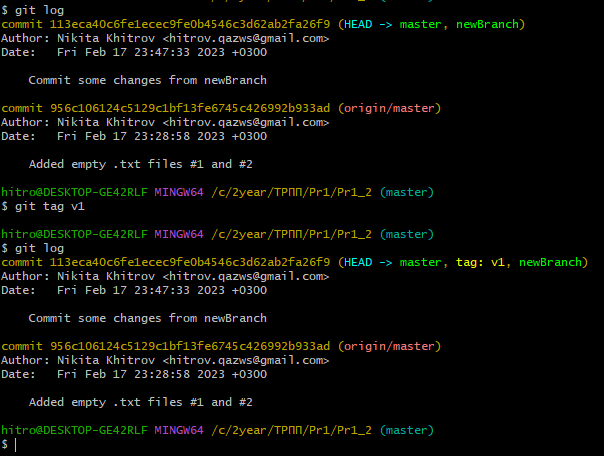


Рисунок 22 – Создание тега, указывающего на последний коммит в ветке

# Создайте новую ветку и выведите список всех веток

На рисунке 23 показан процесс создания новой ветки и вывода списка всех веток.

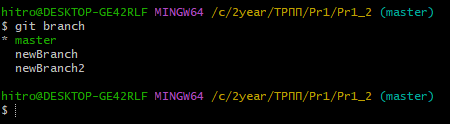
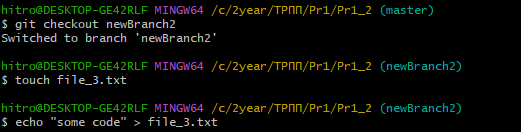
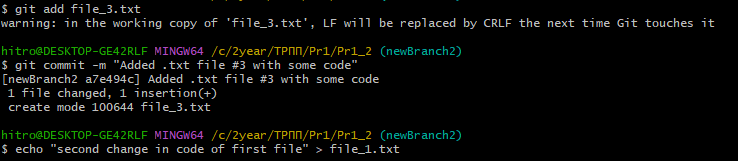


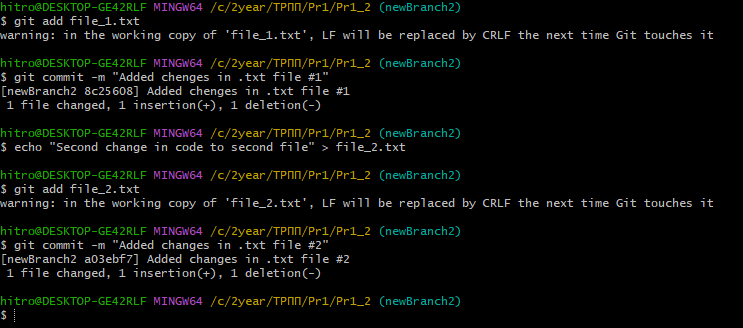
Рисунок 23 – Создание новой ветки и вывод списка всех веток

# Произведите 3 коммита в новой ветке

На рисунках 24-26 показан процесс произведения трех коммитов в новой ветке.







Рисунки 24-26 – Произведение трех коммитов в новой ветке

# Выгрузите все изменения в удаленный репозиторий

На рисунке 27 показана выгрузка изменений в удаленный репозиторий.

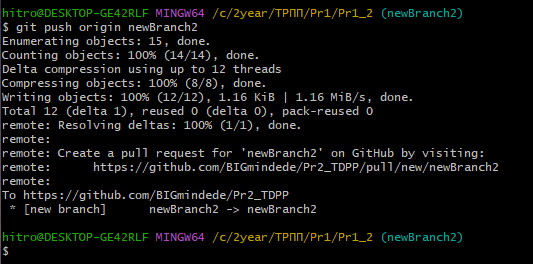


Рисунок 27 – Выгрузка изменений в удаленный репозиторий

# Откатите ветку к созданному тегу (в том числе в удаленном репозитории)

На рисунке 28 показан процесс отката ветки к созданному тегу, в том числе в удаленном репозитории.

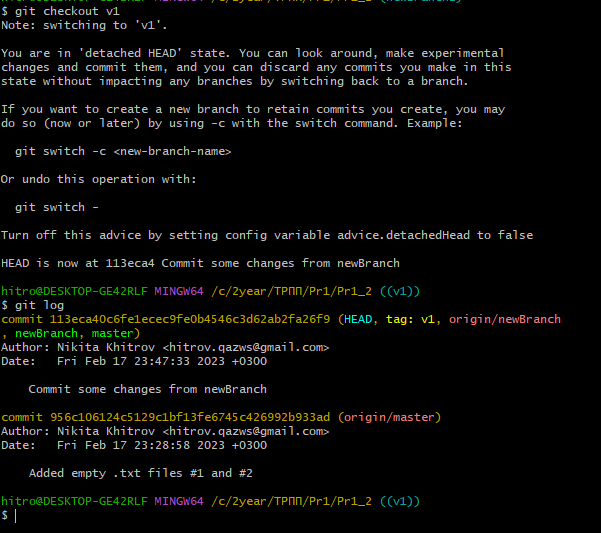


Рисунок 26 – откат ветки к созданному тэгу

# Выведите в консоли различия между веткой master и новой веткой

На рисунке 29 показаны различия между веткой newBranch и master.

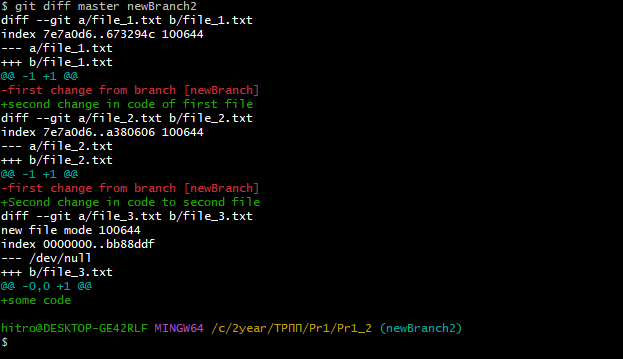


Рисунок 29 – Сравнение веток branch2 и master

# Часть 3. Работа с ветвлением и оформление кода Вариант 5

1. **Сделайте форк репозитория в соответствии с Вашим вариантом**

На рисунке 30 показан процесс создания форка репозитория.

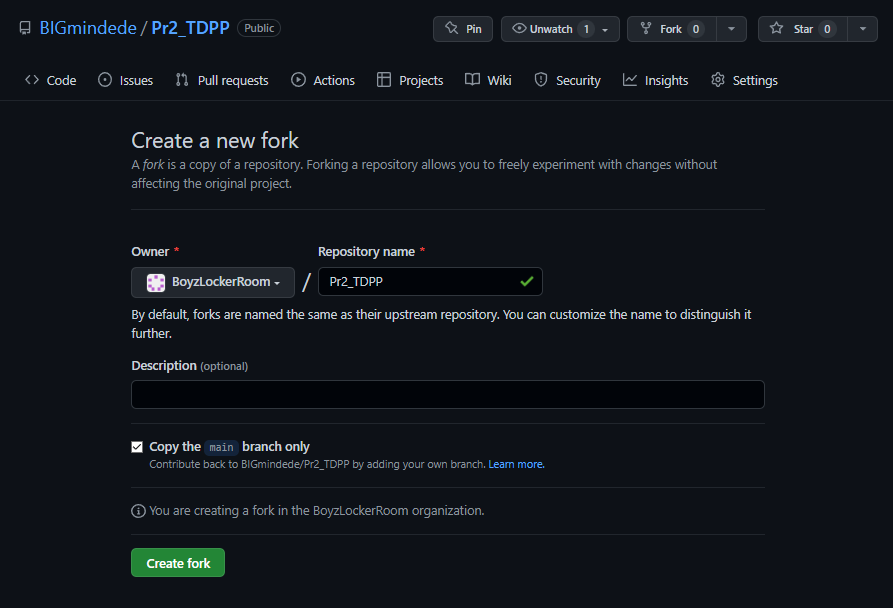


Рисунок 30 – Форк репозитория

# Склонируйте его (форк репозитория) на локальную машину.

На рисунке 31 показан процесс клонирования репозитория на локальную машину.

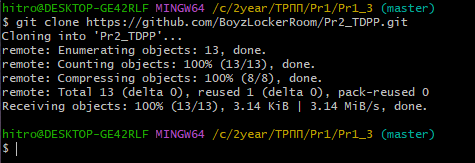


Рисунок 31 – Клонирование репозитория на локальную машину

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

На рисунке 32 показан процесс создания двух веток branch1 и branch2 от последнего коммита в master.

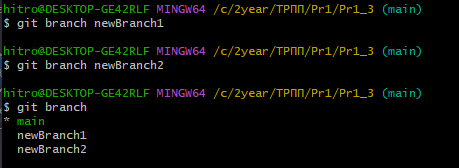
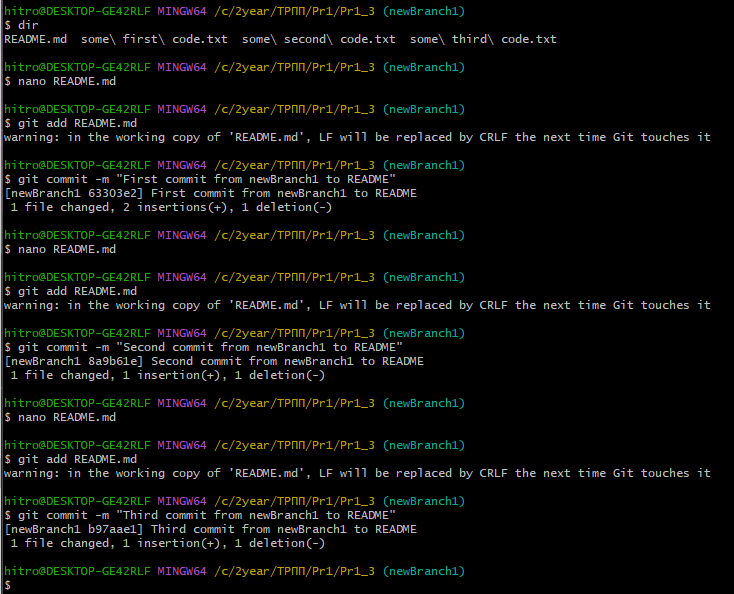
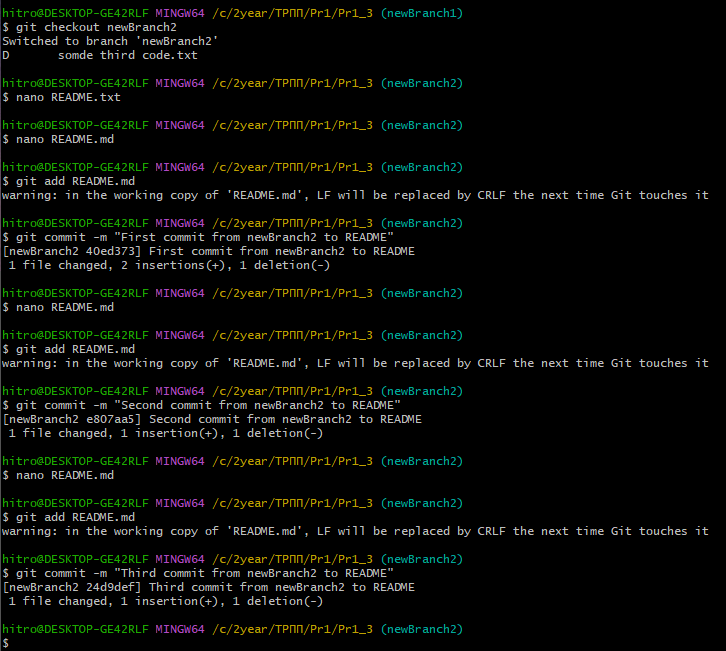


Рисунок 32 – Создание двух веток от последнего коммита

# Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла.

На рисунках 33 - 34 показаны 3 коммита на каждой из веток.





Рисунки 33-34 – Создание 3-ёх коммитов на ветках

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

На рисунке 35 показан процесс слияния ветки branch и branch2 и разрешения при этом конфликтов.

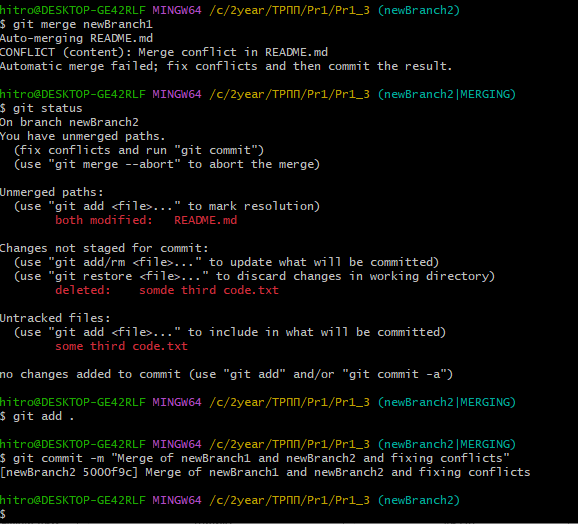


Рисунок 35 – Слияние двух веток

# Выгрузите все изменения во всех ветках в удаленный репозиторий.

На рисунке 36 показан процесс выгрузки всех изменений в удалённый репозиторий.

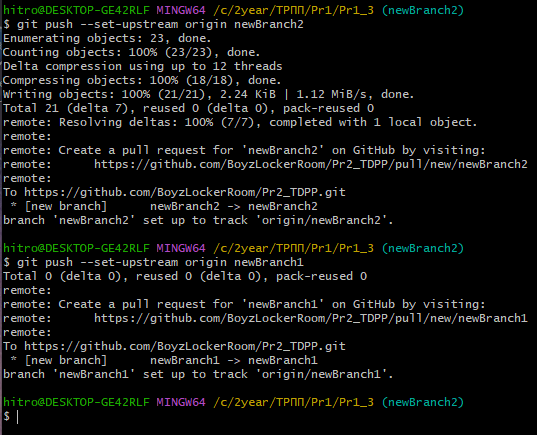


Рисунок 36 – Выгрузка изменений в удалённый репозиторий

# Проведите еще 3 коммита в ветку branch1

На рисунке 37 показан процесс создания 3-ёх коммитов на ветке branch1.

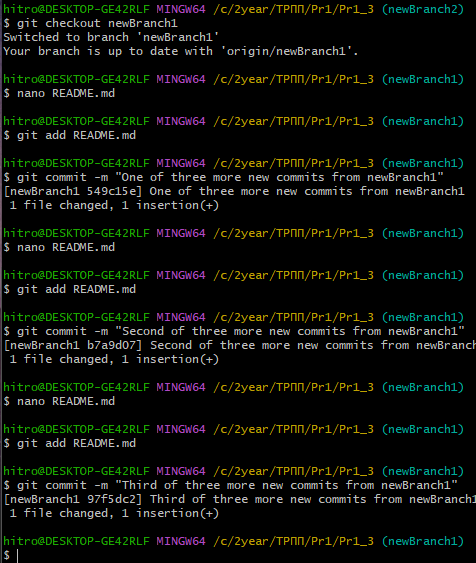


Рисунок 37 – 3 коммита на ветке branch1

# Склонируйте репозиторий еще раз в другую директорию

На рисунке 38 показан процесс клонирования репозитория в другую директорию.

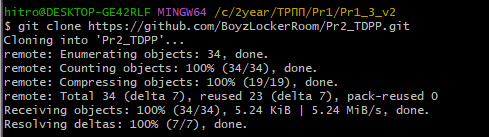


Рисунок 38 – Клонирование репозитория

# В новом клоне репозитории сделайте 3 коммита в ветку branch1

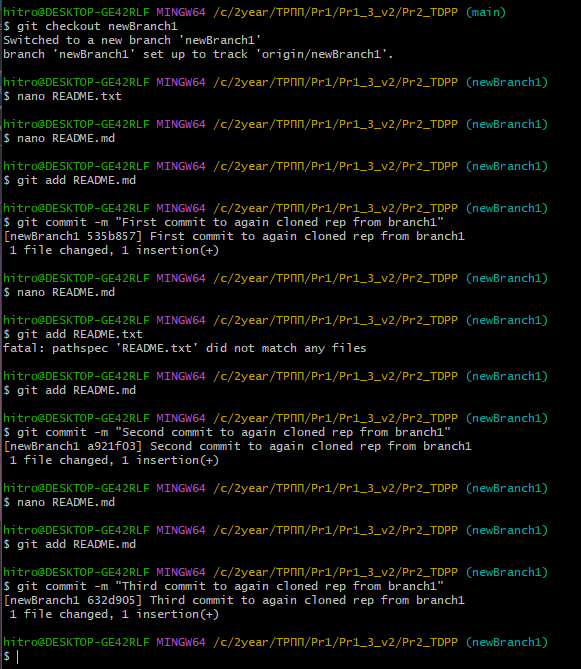
На рисунке 39 показан процесс создания 3-ёх коммитов на ветке branch1 в новом репозитории.

Рисунок 39 – Создание коммитов в новом репозитории

# Выгрузите все изменения из нового репозитория в удаленный репозиторий

На рисунке 40 показан процесс выгрузки изменений в удалённые репозиторий.

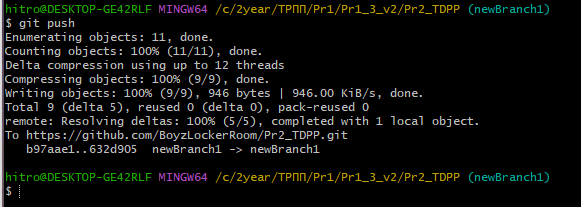


Рисунок 40 – Выгрузка изменений в удалённый репозиторий

# Вернитесь в старый клон с репозиторием, выгрузите изменения с опцией –force

На рисунке 41 показан процесс выгрузки изменений с тегом –force.

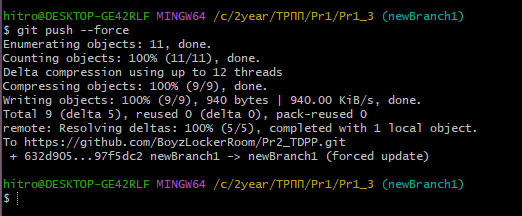


Рисунок 41 – Выгрузка в удалённый репозиторий с тегом –force

# Получите все изменения в новом репозитории

На рисунке 42 показан процесс получения изменений из удалённого репозитория.

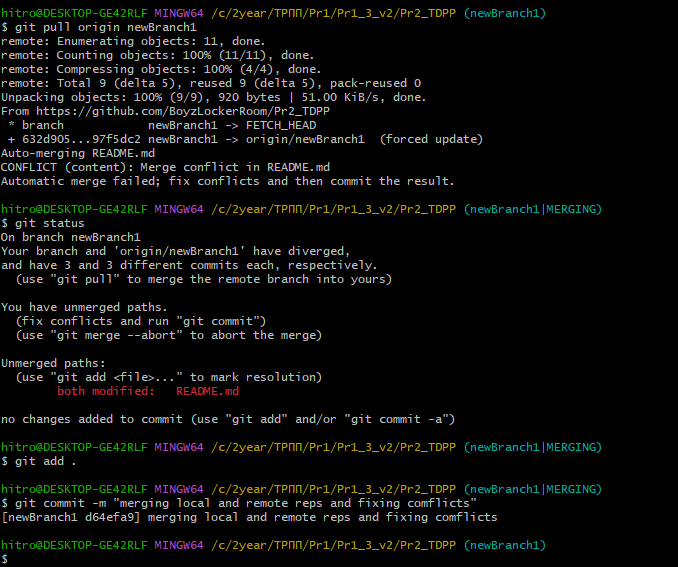


Рисунок 42 – Получение изменений из удалённого репозитория

# Ответы на вопросы

1. **К какому типу систем контроля версий относится Git?**

Git относится к распределённой системе контроле версий.

# Что такое репозиторий Git?

Репозиторий – место, где система контроля версий хранит свои метаданные и базу данных объектов проекта.

Локальный репозиторий — репозиторий, расположенный на локальном компьютере разработчика в каталоге. Именно в нём происходит разработка и фиксация изменений, которые отправляются на удалённый репозиторий.

Удалённый репозиторий — репозиторий, находящийся на удалённом сервере. Это общий репозиторий, в который приходят все изменения и из которого забираются все обновления.

# Что такое коммит?

Коммиты – основные конструктивные элементы временной шкалы Git. Их можно рассматривать как снимки состояния или контрольные точки на временной шкале проекта Git. Коммиты создаются с помощью команды git commit, которая делает снимок состояния проекта на текущий момент времени.

# Что такое ветка в репозитории Git?

Ветка (Branch) — это параллельная версия репозитория. Она включена в этот репозиторий, но не влияет на главную версию, тем самым позволяя свободно работать в параллельной. Когда вы внесли нужные изменения, то вы можете объединить их с главной версией.

# Что делает команда git status?

Команда git status отображает состояние рабочего каталога и раздела проиндексированных файлов. С её помощью можно проверить индексацию изменений и увидеть файлы, которые не отслеживаются Git.

# Что делает команда git add?

Команда git add добавляет изменение из рабочего каталога в раздел проиндексированных файлов. Она сообщает Git, что вы хотите включить изменения в конкретном файле в следующий коммит.

# 18. Что сделает команда "git branch" без какого-либо параметра?

Команда git branch без параметров выведет список всех веток локального репозитория. Также она выделит зелёным цветом ветку, на которой находится указатель HEAD.

# 27. Как сделать ветку с названием my\_branch?

Для создания ветки с именем my\_branch необходимо написать команду git branch my\_branch в консоль, либо можно написать команду git checkout -b my\_branch, которая создаст новую ветку и переключит указатель HEAD на неё.

# Выводы

В ходе данной работы был изучен функционал и возможности системы контроля версий git.