|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

Институт Информационных технологий (ИТ)

Кафедра Математического обеспечения и стандартизации информационных технологий (МОСИТ)

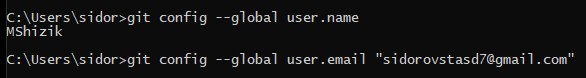
|  |
| --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1** |
| **по дисциплине** |
| **«Технология разработки программных приложений»**  **Тема: «Система контроля версий»** |

|  |  |
| --- | --- |
| Выполнил студент группы ИКБО-20-21 | Сидоров С.Д. |
| Принял | Петренко А.А. |

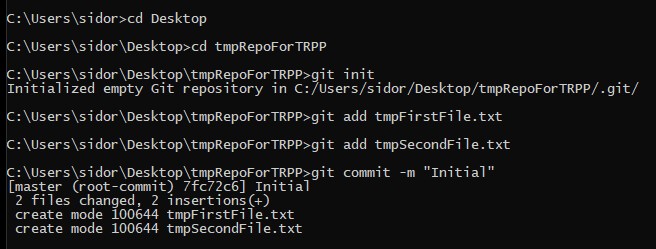
|  |  |  |
| --- | --- | --- |
| Практическая работа выполнена | «\_\_»\_\_\_\_\_\_\_2023г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2023 г. | *(подпись руководителя)* |

ОТЧЁТ

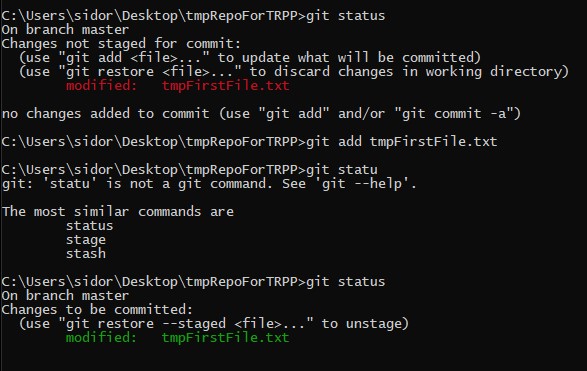
1. Основные команды Git
   1. Установите и настройте клиент git на своей рабочей станции.



* 1. Создайте локальный репозиторий и добавьте в него несколько файлов.

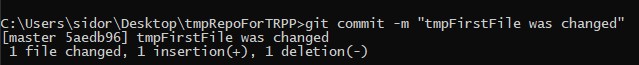


* 1. Внесите изменения в один из файлов
  2. Проиндксируйте изменения и проверьте



1.5 Сделайте коммит того, что было проиндексировано в

репозиторий. Добавьте к коммиту комментарий.



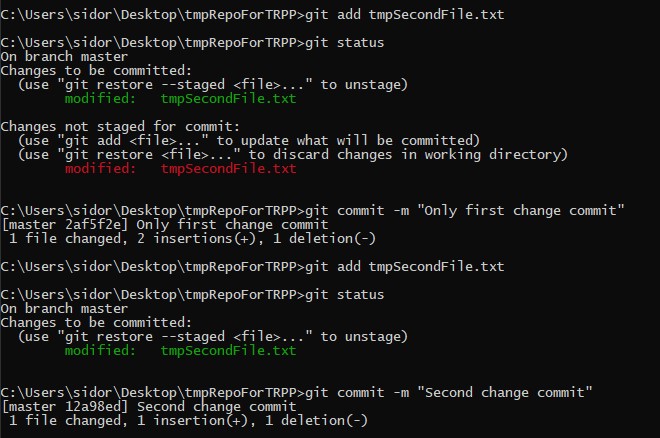
1.6 Измените еще один файл. Добавьте это изменение в индекс git.

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.

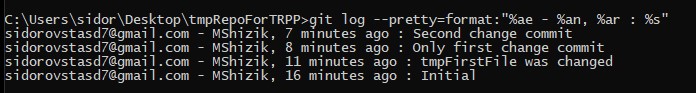
Сделайте коммит второго изменения.



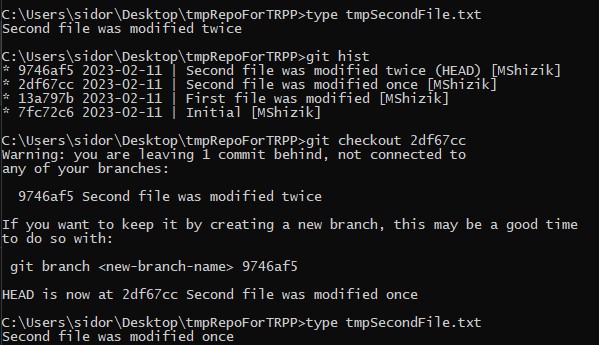
1.7 Просмотрите историю коммитов с помощью команды git log.

Ознакомьтесь с параметрами команды и используйте некоторые из них

для различного формата отображения истории коммитов.

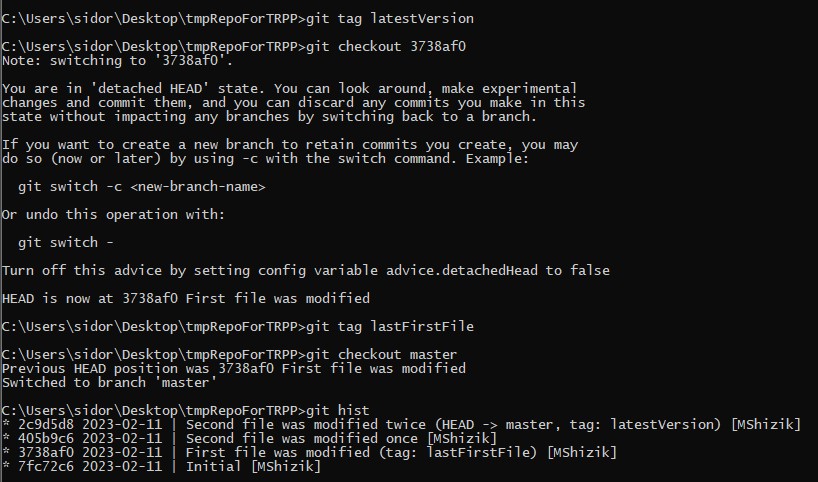


1.8 Верните рабочий каталог к одному из предыдущих состояний.



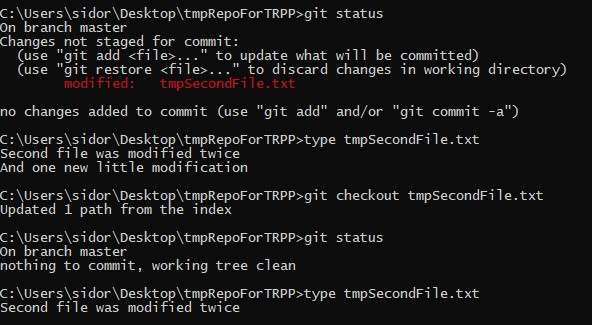
1.9 Изучите, как создавать теги для коммитов для использования в

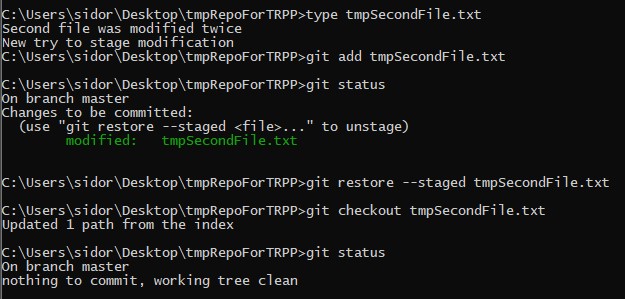
будущем.



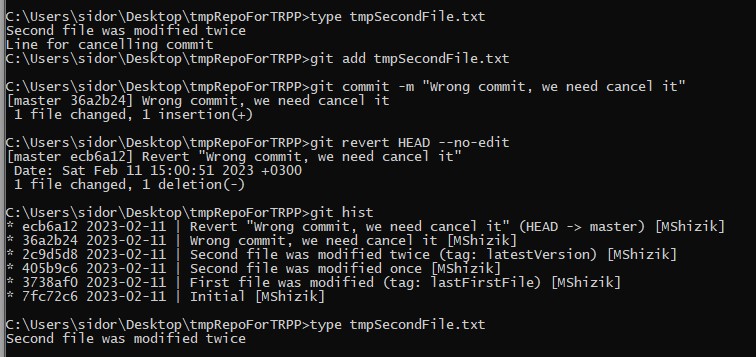
1.10 Отмените некоторые изменения в рабочем каталоге (до и после

индексирования).



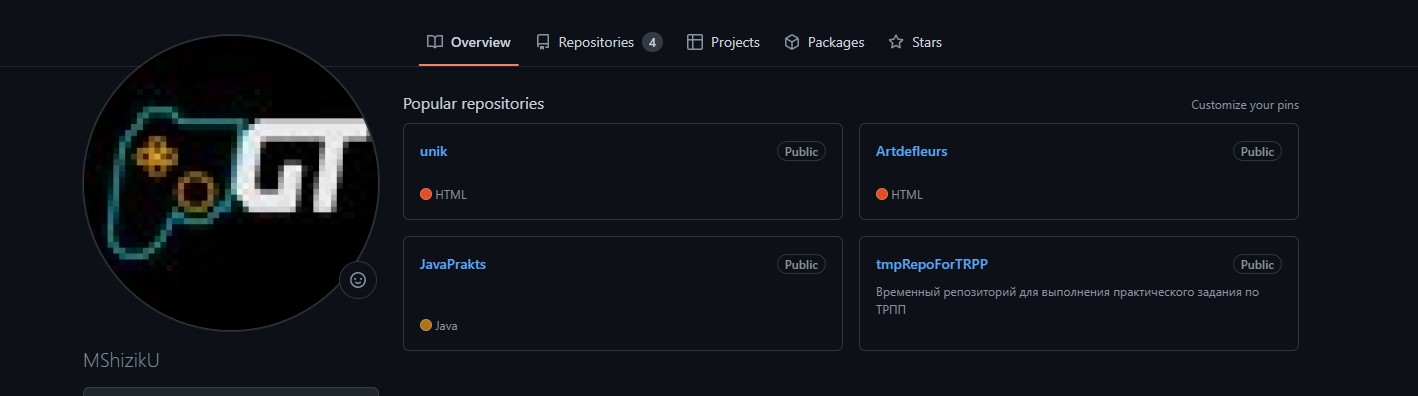


1.11 Отмените один из коммитов в локальном репозитории.

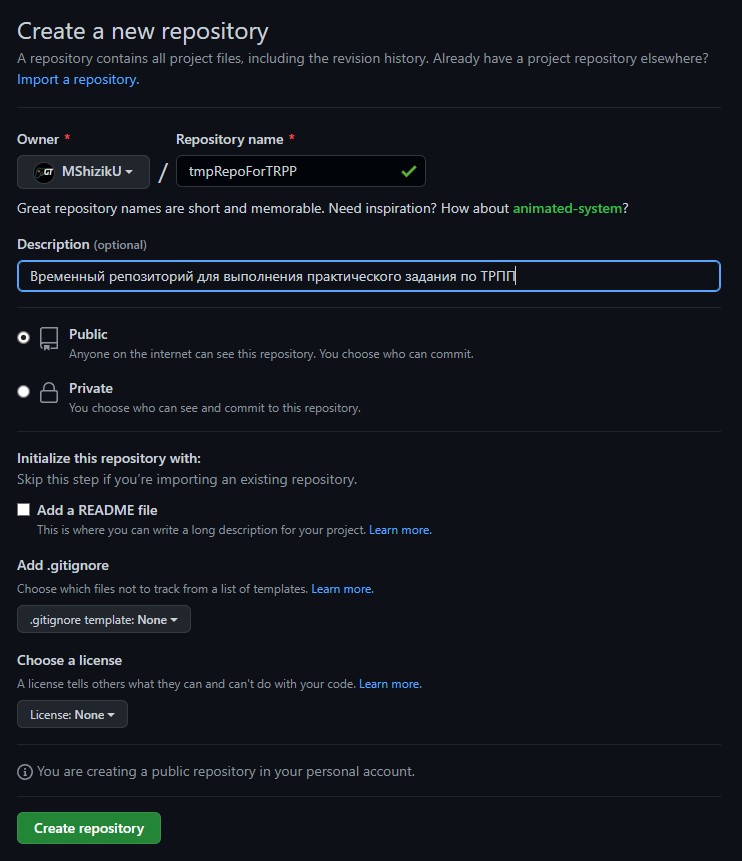


1. Системы управления репозиториями (Вариант 10)

2.1 Создайте аккаунты на GitHub.

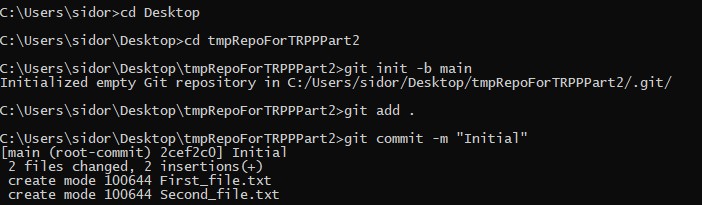


2.2 Создайте репозиторий согласно варианту.



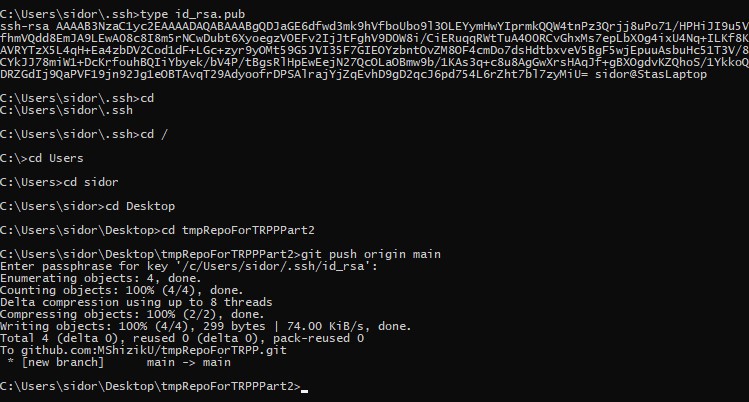
2.3 Создайте новый локальный репозиторий с несколькими

файлами на рабочей станции и загрузите его содержимое на GitHub.



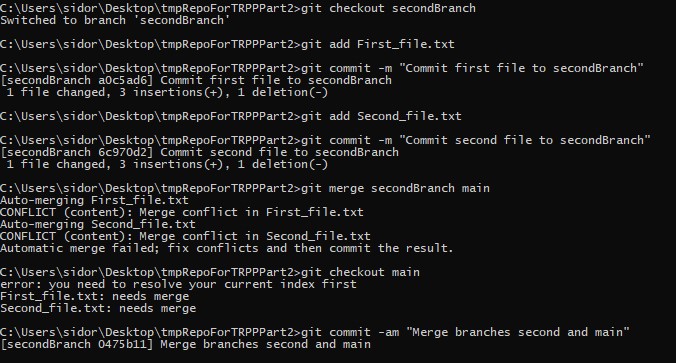
2.4 Чтобы избежать ввода логина и пароля, создайте SSH-ключ

для авторизации.

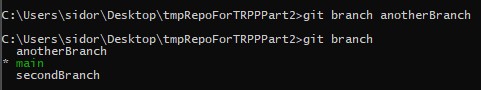


2.5 Создайте в репозитории новую ветку. Произведите в ней

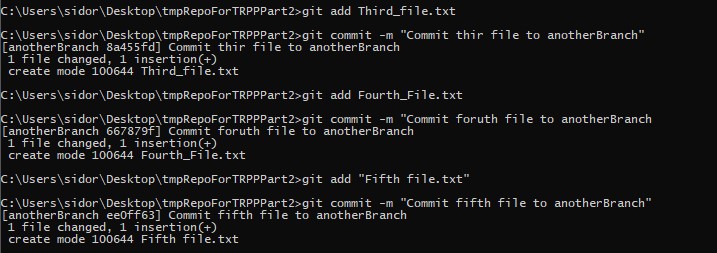
несколько изменений и слейте с веткой master.



2.6 Создайте новую ветку и выведите список всех веток.



2.7 Произведите 3 коммита в новой ветке в разные файлы.



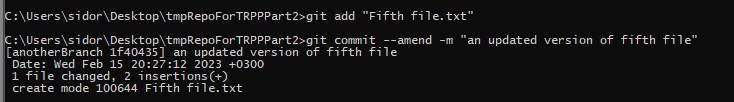
2.8 Выгрузите изменения в удаленный репозиторий.



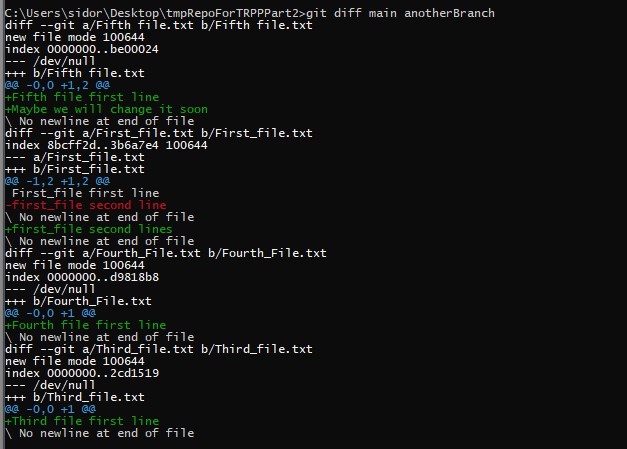
2.9 Произведите какие-нибудь изменения в файле который

существует в ветке, но не коммитьте их.

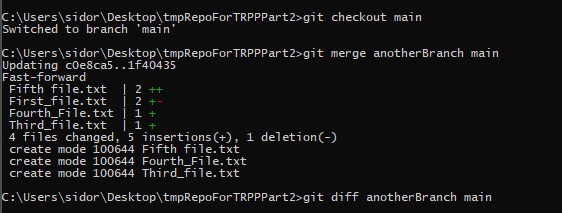
2.10 Внесите эти изменения в последний коммит (amend)



2.11 Выведите в консоли различия между веткой master и новой веткой

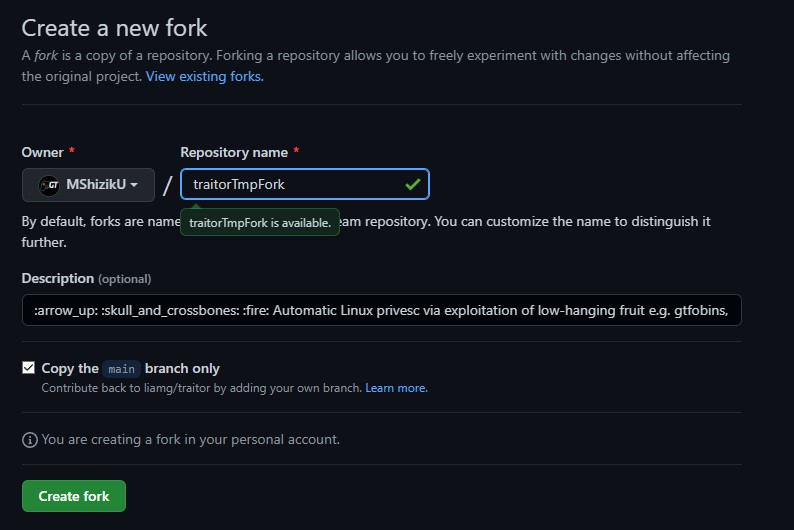


2.12 Слейте новую ветку с master при помощи merge



1. Работа с ветвлением и форомлением кода (Вариант 1)

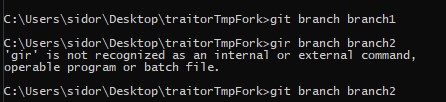
3.1 Сделайте форк репозитория.



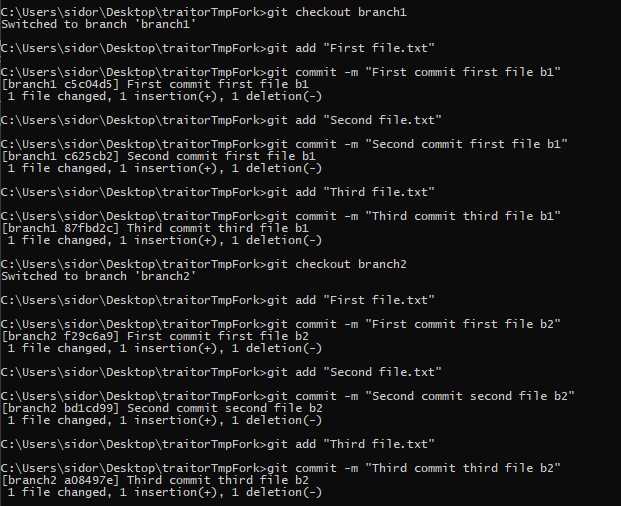
3.2 Склонируйте его (форк репозитория) на локальную машину



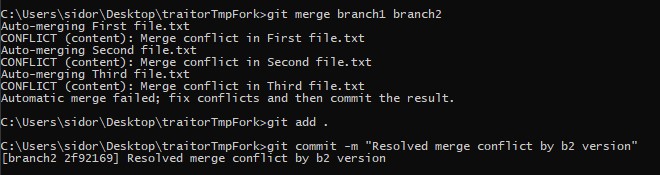
3.3 Создайте две ветки branch1 и branch2 от последнего коммита в master'е.



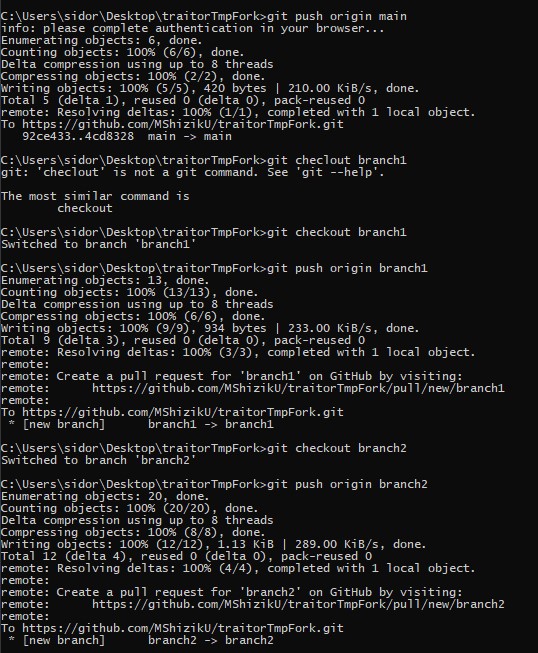
3.4 Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла.



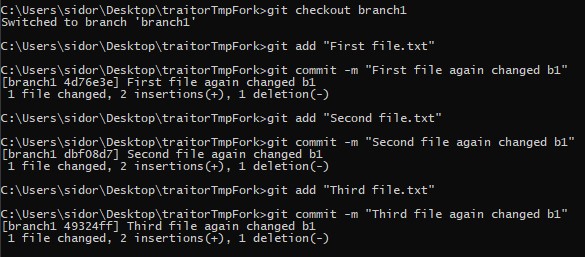
3.5 Выполните слияние ветки branch1 в ветку branch2, разрешив при этом конфликты.



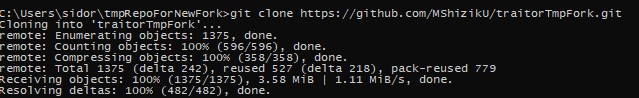
3.6 Выгрузите все изменения во всех ветках в удаленный репозиторий.



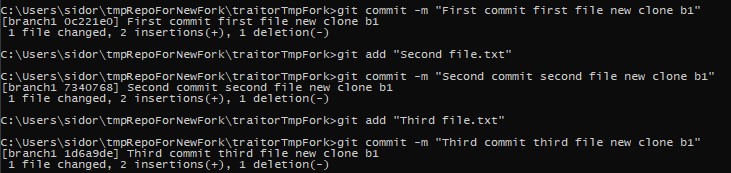
3.7 Проведите еще 3 коммита в ветку branch1.



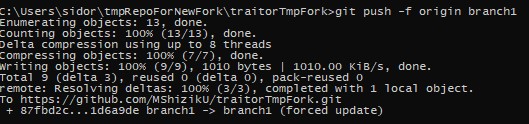
3.8 Склонируйте репозиторий еще раз в другую директорию.



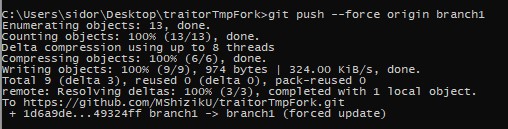
3.9 В новом клоне репозитории сделайте 3 коммита в ветку branch1.



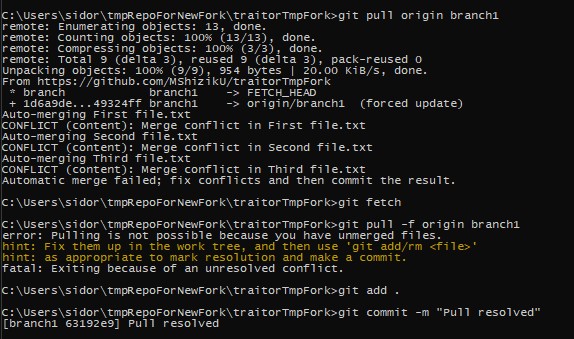
3.10 Выгрузите все изменения из нового репозитория в удаленный репозиторий.



3.11 Вернитесь в старый клон с репозиторием, выгрузите изменения с опцией --force.



3.12 Получите все изменения в новом репозитории.



1. Ответы на контрольные вопросы

Вопрос 4: Что такое коммит?

Ответ: коммит - это пакет изменений, хранящий информацию с добавленными, отредактированными или удалёнными файлами кода.

Вопрос 6: Что такое тег в репозитории Git?

Ответ: 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. Теги похожи на неизменяемые ветки, но они, в отличие от веток, не имеют истории коммитов после создания.

Вопрос 10: Для чего нужен .gitignore?

Ответ: Файл .gitignore предназначен для исключения из индексации Git файлов и папок проекта. Размещаться этот файл может в любой папке проекта и количество этих файлов в разных папках не ограничено. Зона действия файла распространяется от папки в которой он лежит и на все вложенные файлы и папки.

Вопрос 11: Что делает команда git status?

Ответ: выводит информацию обо всех изменениях, внесенных в дерево директорий проекта по сравнению с последним коммитом рабочей ветки.

Вопрос 13: Что делает команда git log?

Ответ: отображает отправленные снимки состояния и позволяет просматривать и фильтровать историю проекта, а также искать в ней конкретные изменения.

Вопрос 24: Как узнать, кто автор строчки в файле, используя систему Git?

Ответ: git blame filename позволяет определить автора послежних изменений для каждой строки в файле.

Вопрос 27: Как сделать ветку с названием my\_branch?

Ответ: git branch my\_branch

Вопрос 33: Что такое форк репозитория?

Ответ: копия оригинального репозитория во владении другого разработчика, который хочет внести изменения в оригинальный.

Выводы

В ходе выполнения данной практической работы были получены знания по работе с системой контроля версий git, а также по системе работы с репозиториями и работы с ветвлением и оформлением кода.