**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнёва»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Институт информатики и телекоммуникаций\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(институт, факультет, подразделение)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Кафедра информационно-управляющих систем\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(кафедра, цикловая комиссия)

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_Знакомство с системой GIT\_\_\_\_\_\_\_\_\_\_\_

(тема)

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Роза М.П.

подпись, дата инициалы, фамилия

Обучающийся БИА20-02 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сычев Д.О.

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2022

**Ход выполнения работы:**

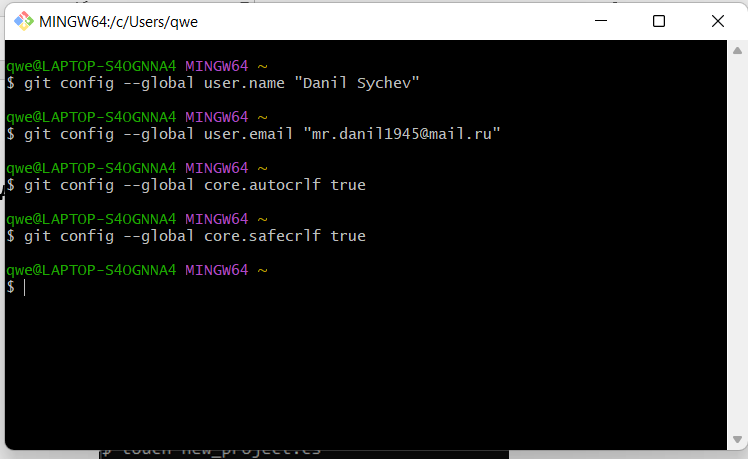


Рисунок 1 - Настройка GIT

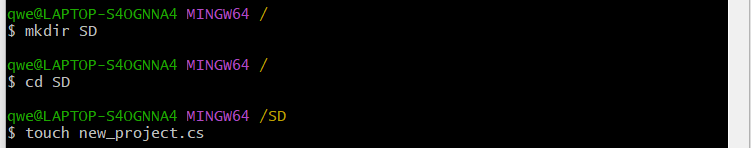


Рисунок 2 - Создание проекта

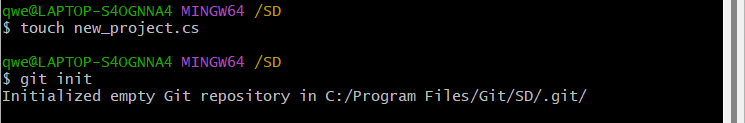


Рисунок 3 - Создание репозитория

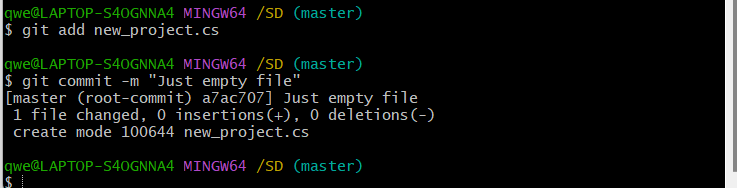


Рисунок 4 - Добавление файла в репозиторий и коммита

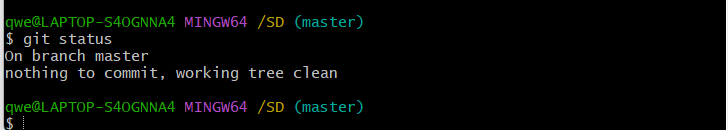


Рисунок 5 - Проверка состояния репозитория

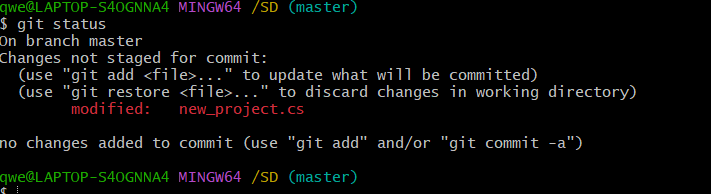


Рисунок 6 - Результат проверки после внесения изменений



Рисунок 7 - Индексация файла и добавление коммита

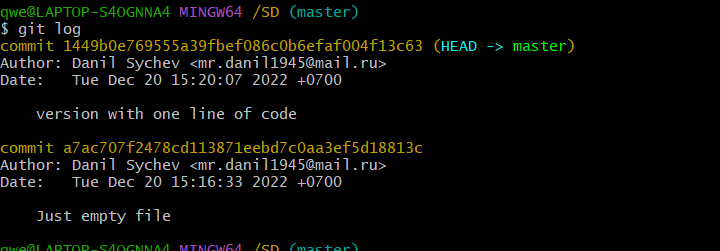


Рисунок 8 - История проекта

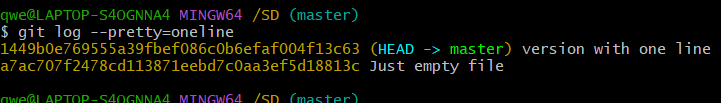


Рисунок 9 - История проекта в режиме «одной строки»

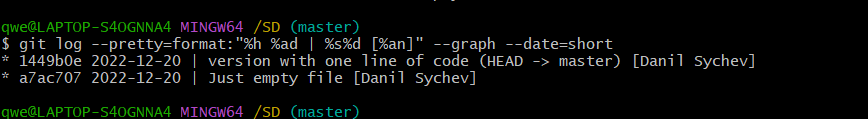


Рисунок 10 -Форматированная история

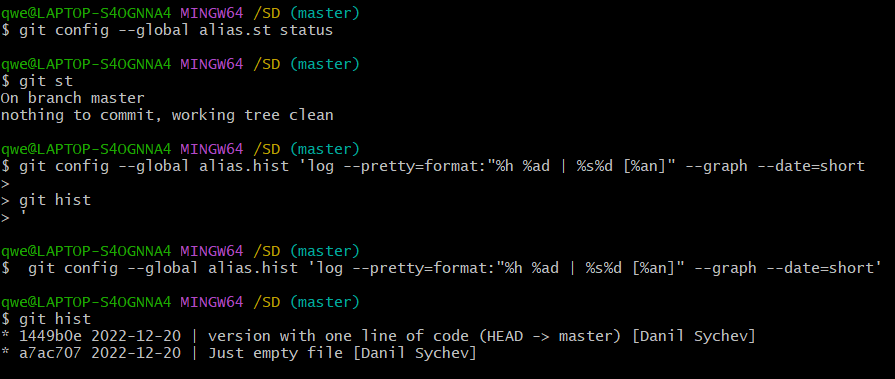


Рисунок 11 - Добавление алиаса

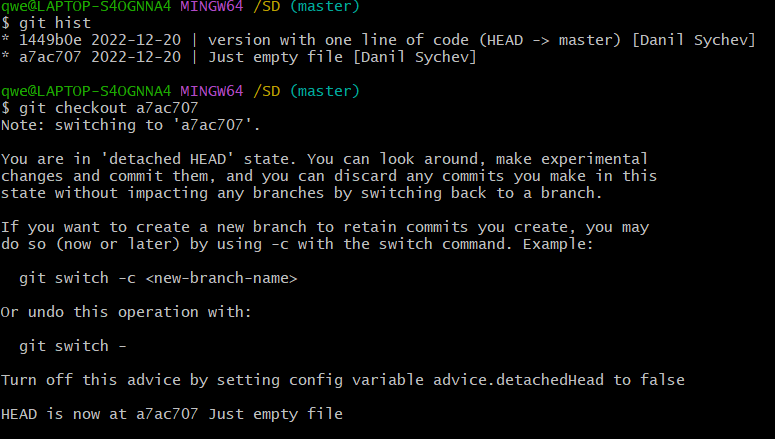


Рисунок 12 - Возврат к прошлым версиям

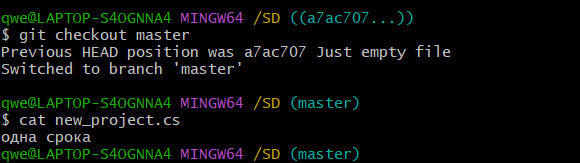


Рисунок 13 - Возврат к последней версии

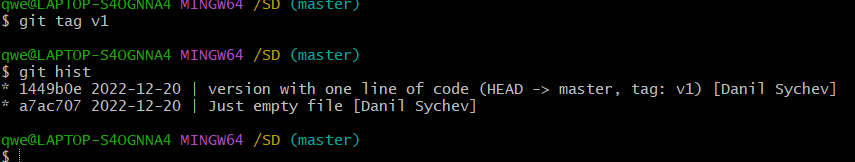


Рисунок 14 - Тегирование

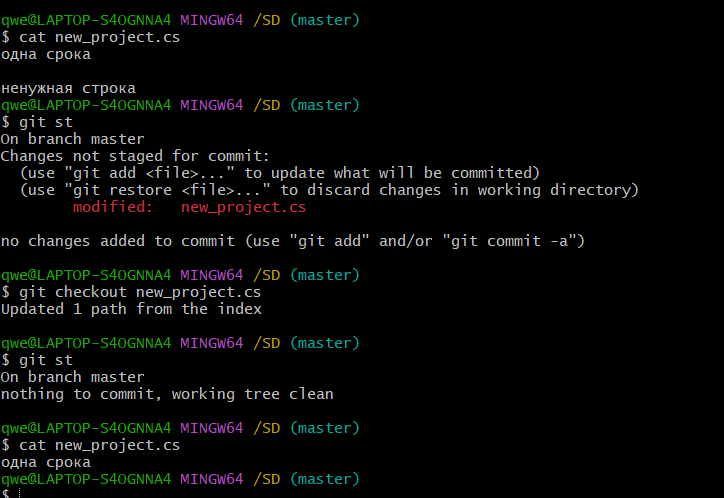


Рисунок 15 - Возврат к предыдущей версии

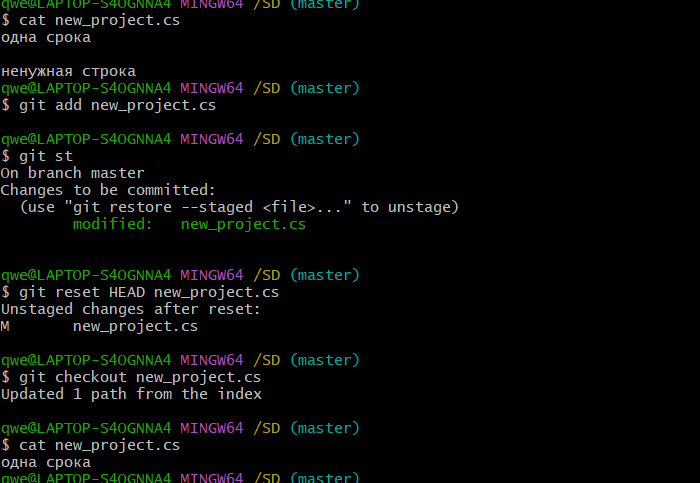


Рисунок 16 - Отмена проиндексированных изменений

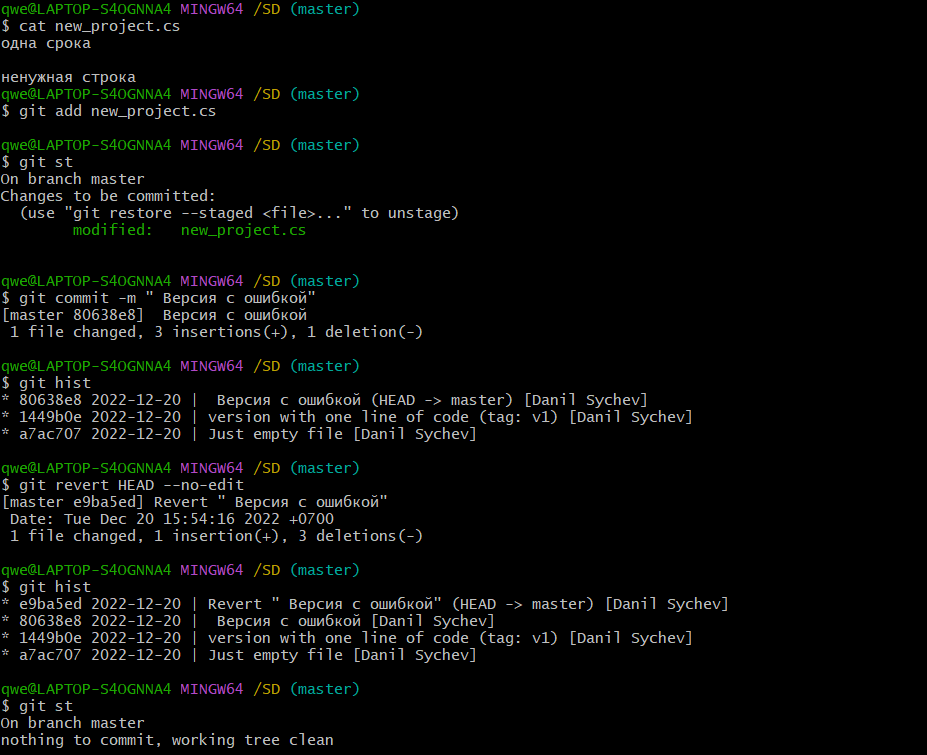


Рисунок 17 - Пример отмены коммита

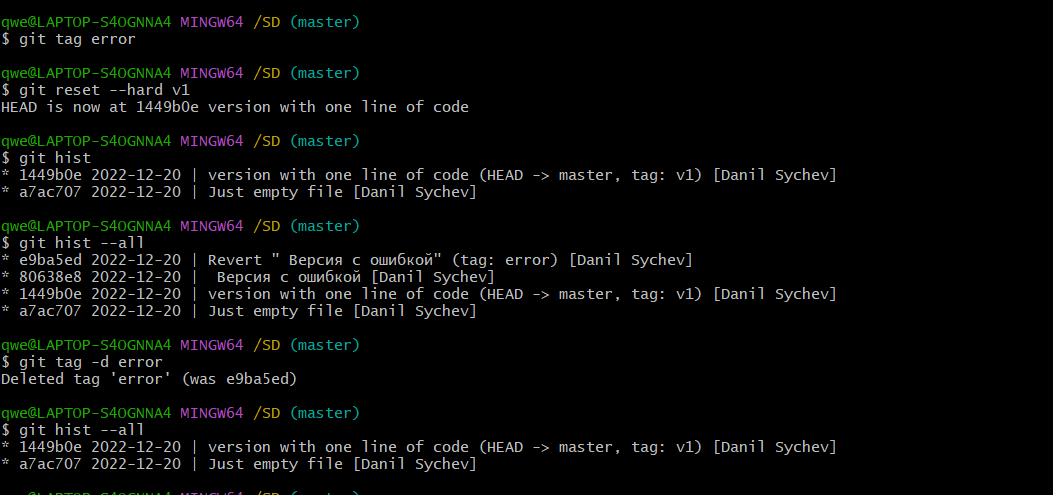


Рисунок 18 - Удаление коммита из истории

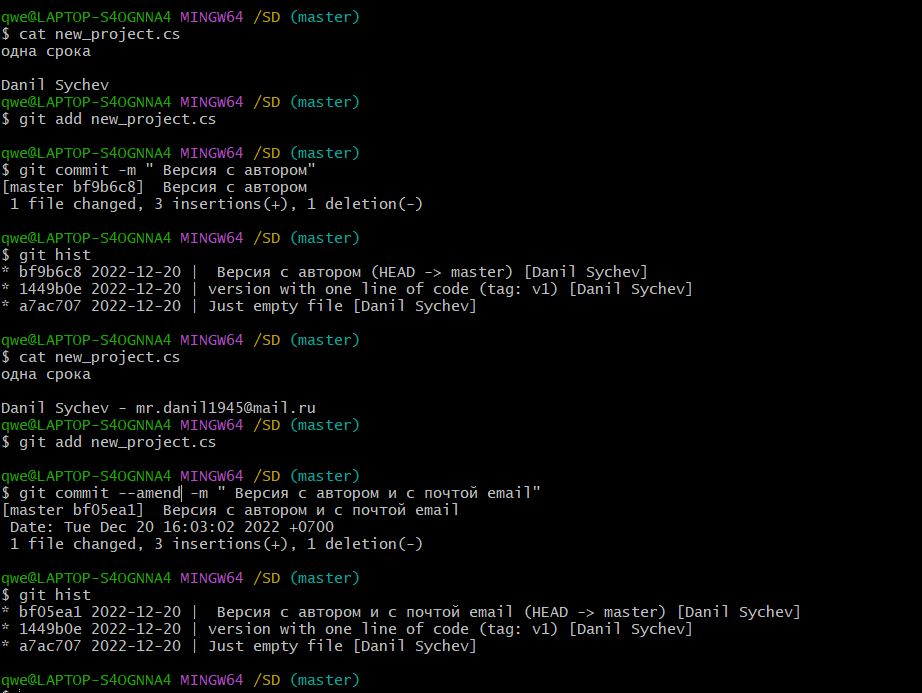


Рисунок 19 - Исправление коммита

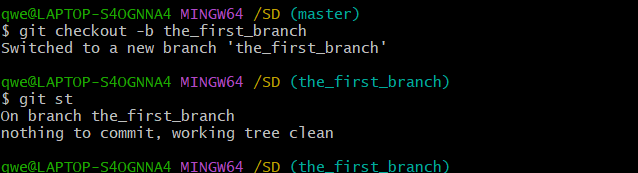


Рисунок 20 - Создание ветки

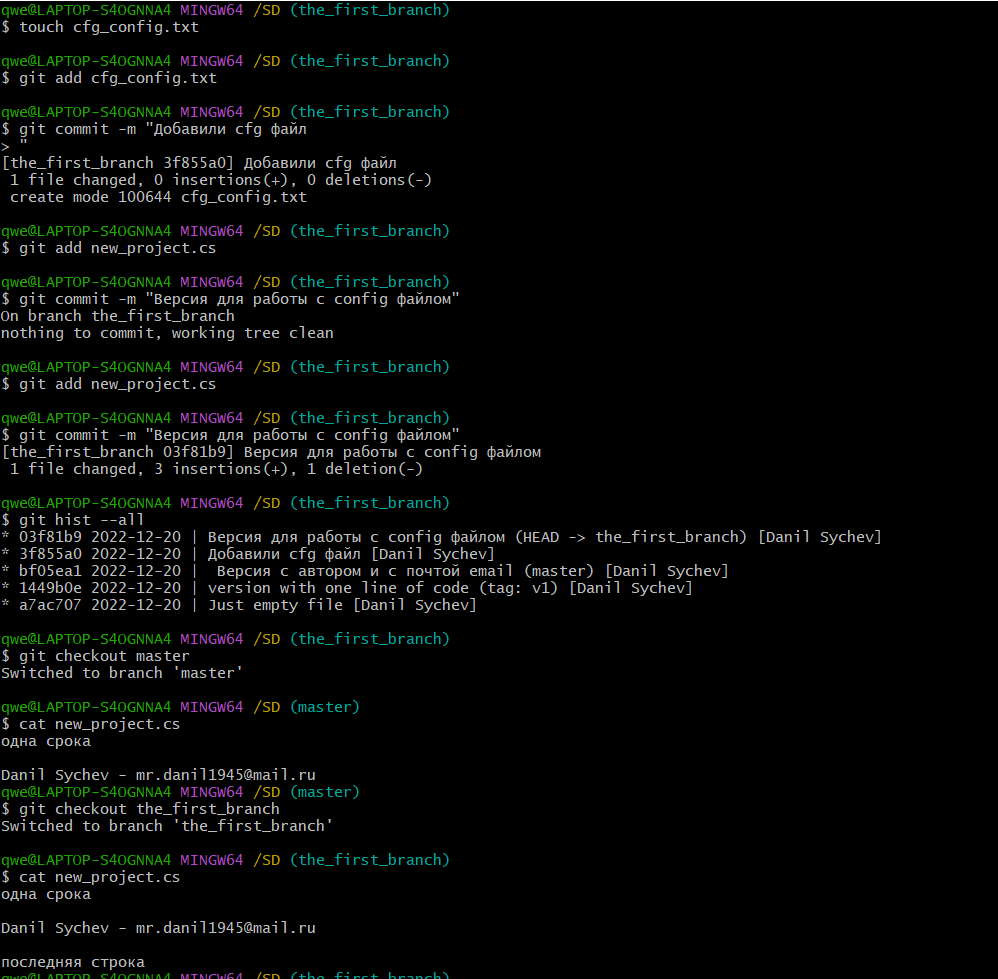
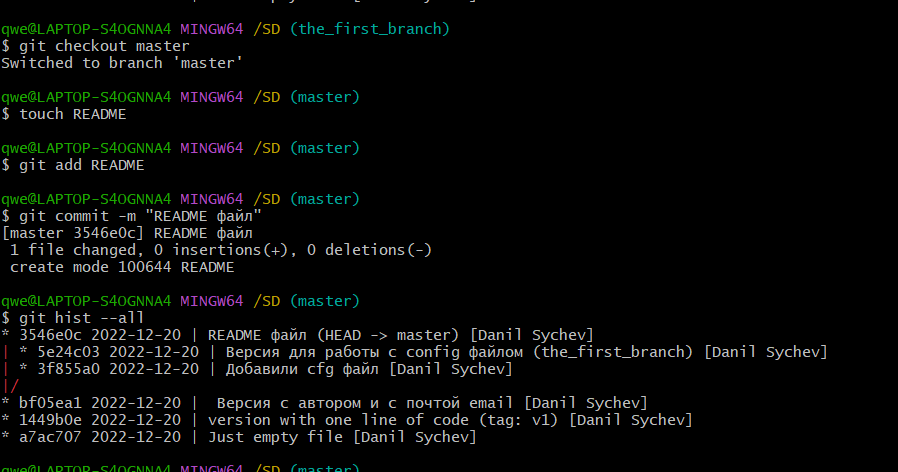


Рисунок 21 - Работа с ветками

  
Рисунок 22 - Работа с ветками

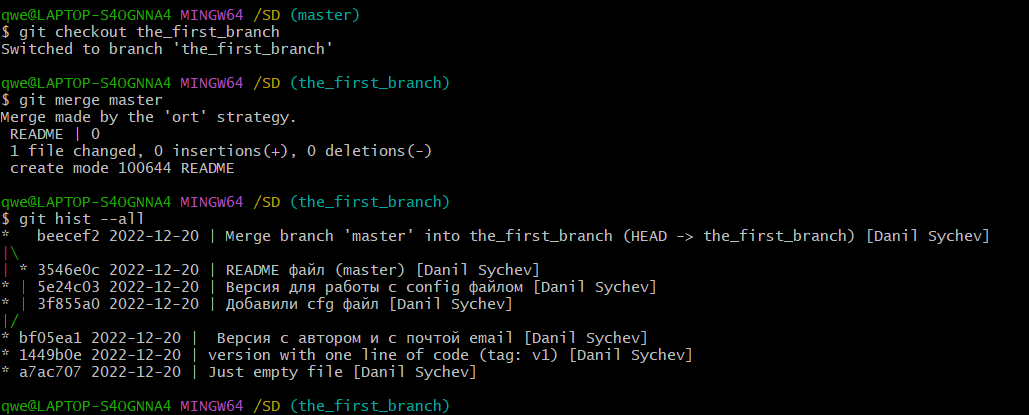


Рисунок 23 - Слияние веток

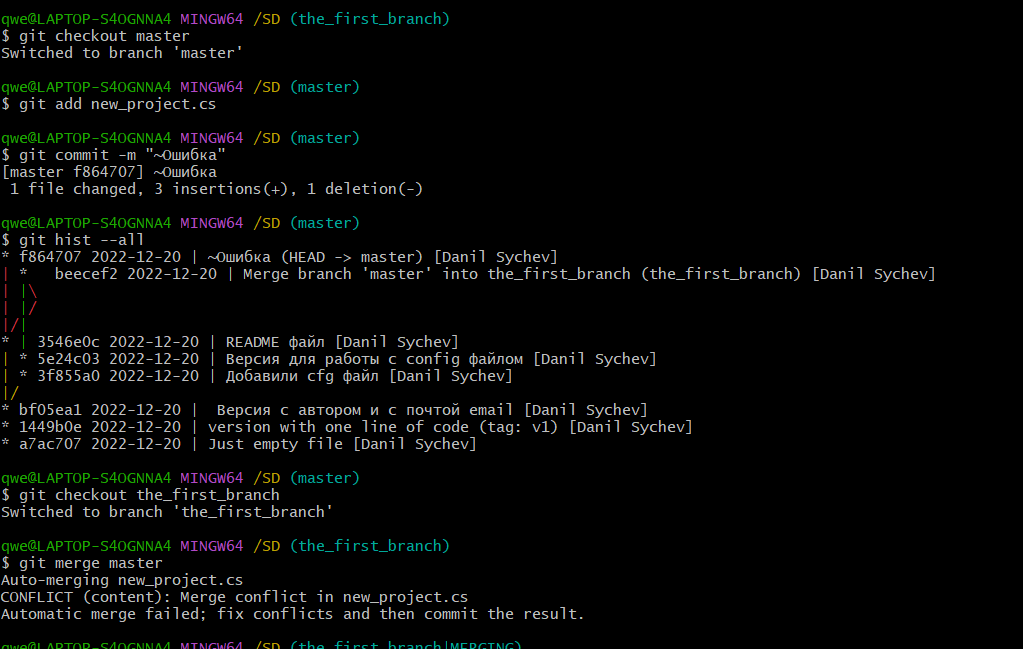
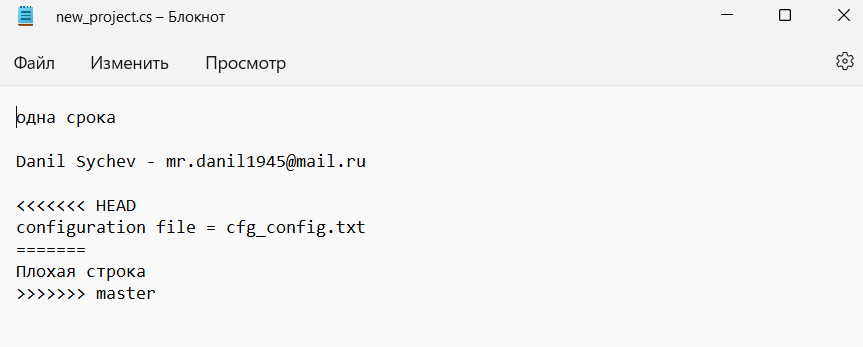


Рисунок 24 - Конфликт веток



Комментарии Git относительно места конфликта.

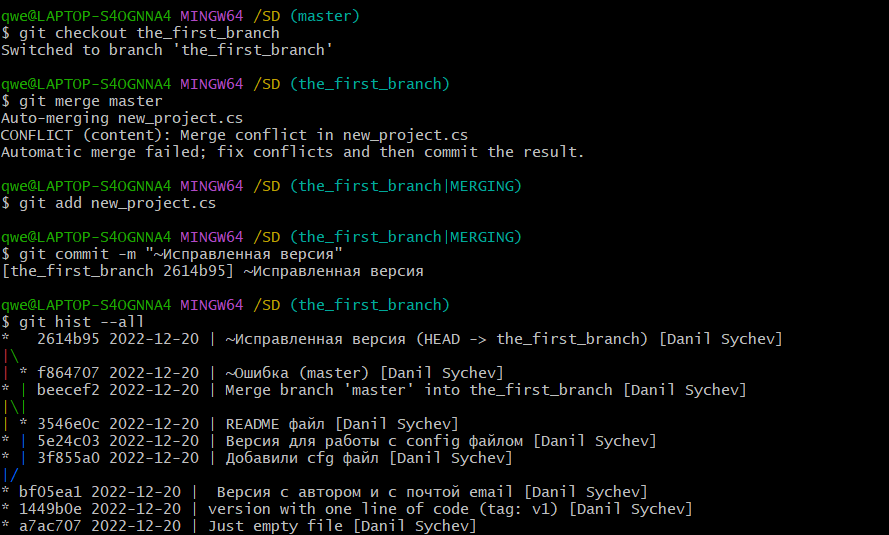
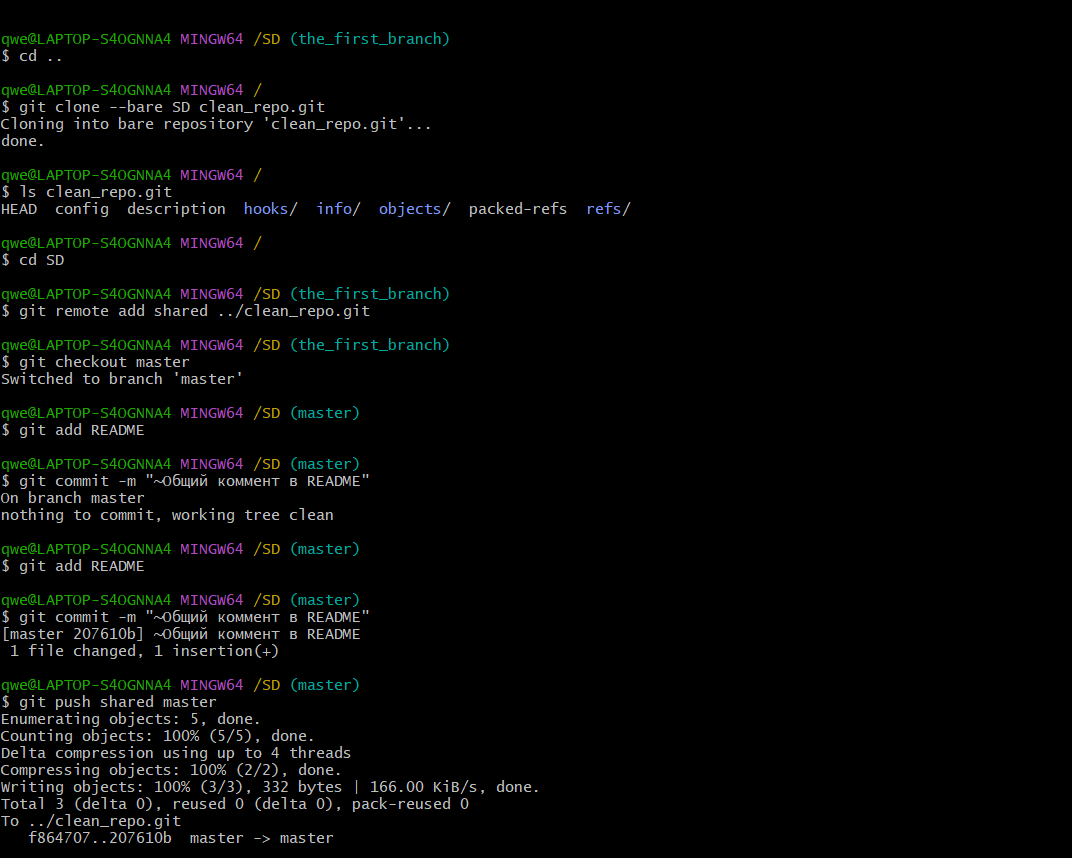


Рисунок 25 - Разрешенный конфликт

  
Рисунок 26 - Создание репозитория clean\_repo.

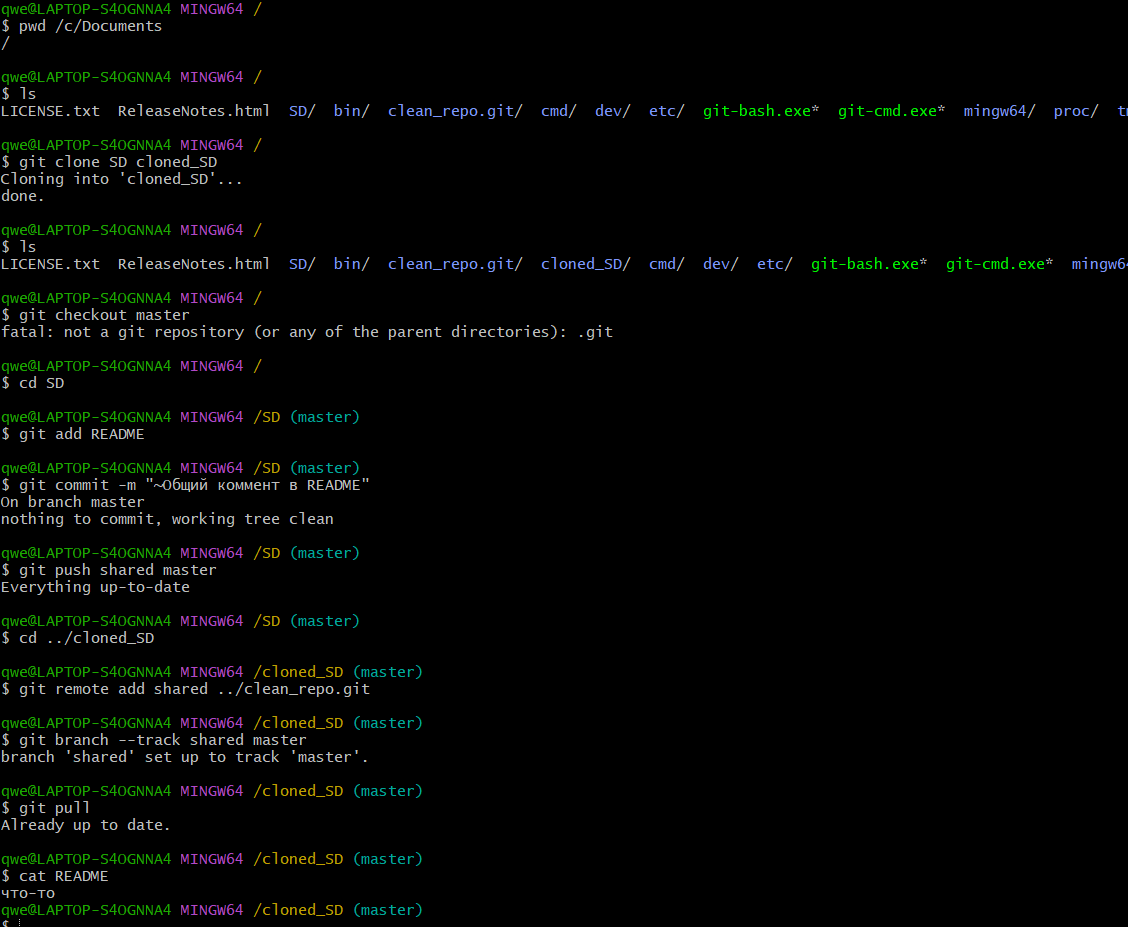


Рисунок 27 - Работа с удаленным репозиторием.

**Индивидуальное задание:** Использовать в работе 2 разных файла, для одного из них использовать 3 различные ветки.

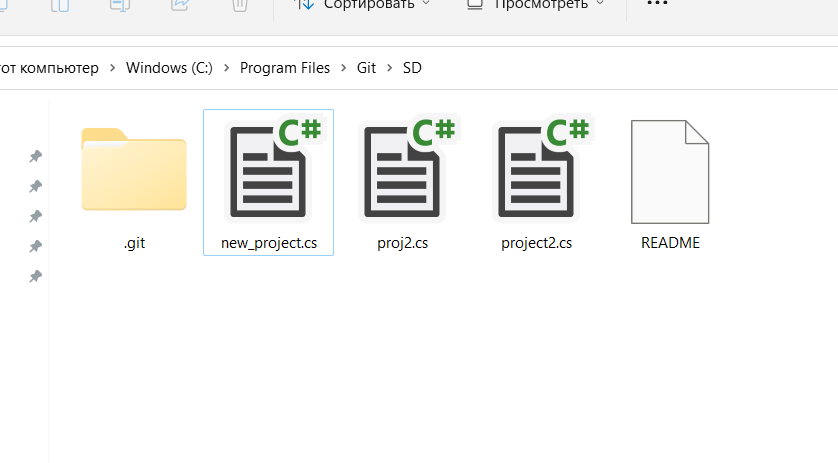


Рисунок 1 – Список всех файлов

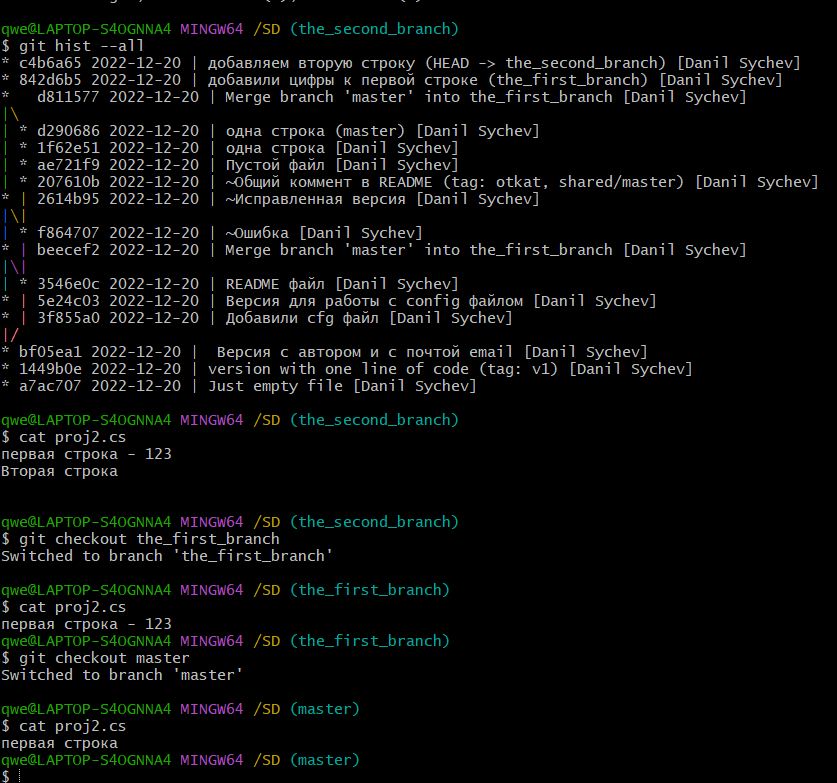


Рисунок 2 – Содержимое файла proj2.cs в трех ветках

**Ответы на контрольные вопросы:**

1. **В чем заключается удобство использования системы контроля версий Git?**   
   Она позволяет вернуть файлы к состоянию, в котором они были до изменений, вернуть проект к исходному состоянию, увидеть изменения, увидеть, кто последний менял что-то и вызвал проблему, кто поставил задачу и когда и многое другое.
2. **Как изменить формат выводимой истории версий?**Добавить необходимые параметры
3. **Как сократить время набора команд в системе Git?**Написать часть команды или имени так, чтобы написанная часть была уникальная и подходила только для одной команды или имени и нажать TAB.
4. **Если необходимо выделить в истории группы версий, с определенными изменениями, как удобнее всего это сделать?***Поиск по коммитам*

git log --oneline | grep revert # поиск упоминания revert

git log --oneline | grep -i revert # независимо от регистра  
*Коммиты, затронувшие один файл*  
git log index.html

*Поиск по автору*

git log --author <author>

*Поиск по диапазону дат*

Опции --after и --before задают начальную и конечную даты коммитов

git log --after='2020-03-09 15:30' --before='2020-03-09 16:00'

1. **Что такое тегирование, и для чего его целесообразно использовать?**  
   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).
2. **Как можно исправить коммит, если он оказался неверным?**  
   git commit –amend -m “”
3. **Для чего используется ветвление?**Ветки нужны для того, чтобы программисты могли вести совместную работу над проектом и не мешать друг другу при этом.
4. **В каком случае необходимо исправлять файлы, при слиянии веток?**  
   Если они конфликтуют.
5. **Что такое алиасы? Для чего они применяются?**Алиасы позволяют сделать вызов громоздких команд очень простым, с легко запоминающимися названиями.
6. **Что такое режим «одной строки»?**  
   Вывод коммитов в одну строку. Показывает только хэш коммита и commit message.

**Вывод:** В процессе изучения системы управления версиями GIT я научился: индексировать изменения, просматривать историю версий и форматировать ее, создавать алиасы, возвращаться к предыдущим версия, тегировать версии, отменять локальные изменения, создавать и сливать ветки, а также разрешать конфликты.