**Оглавление**

[Часть 1. Основные команды git 2](#_Toc158975504)

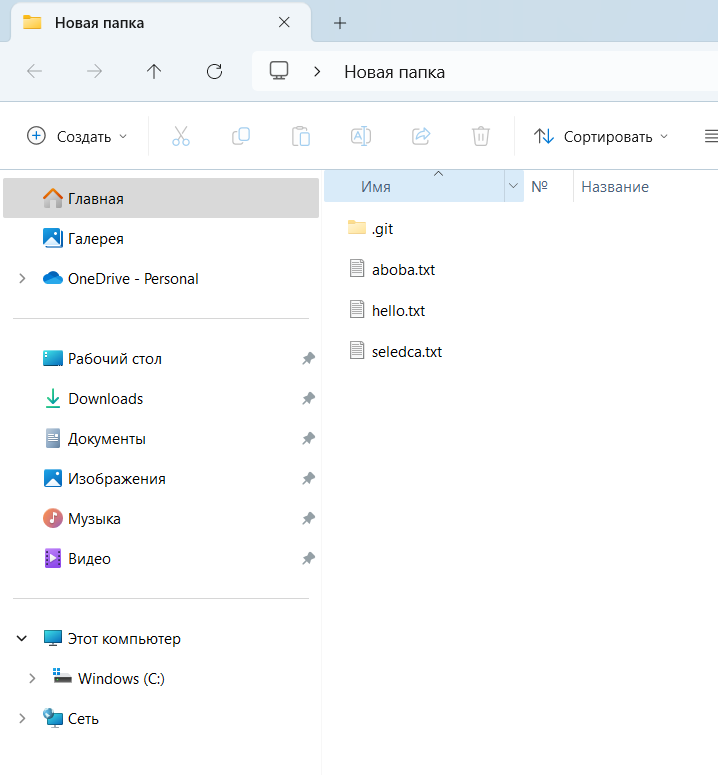
[Часть 2. Системы управления репозиториями 10](#_Toc158975505)

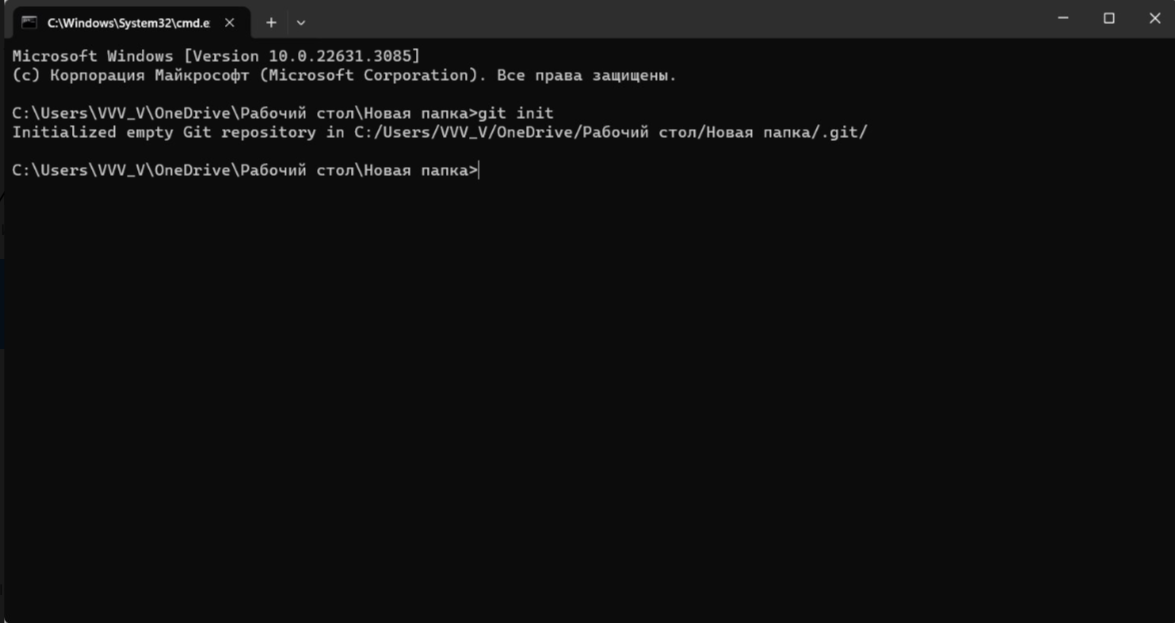
[Часть 3. Работа с ветвлением и оформление кода 16](#_Toc158975506)

[Контрольные вопросы 20](#_Toc158975507)

Часть 1. Основные команды git

Создайте локальный репозиторий и добавьте в него несколько файлов.

Рис. 1. Создания папки репозитория и файлов.

Рис. 2. Создание репозитория.

Внесите изменения в один из файлов.

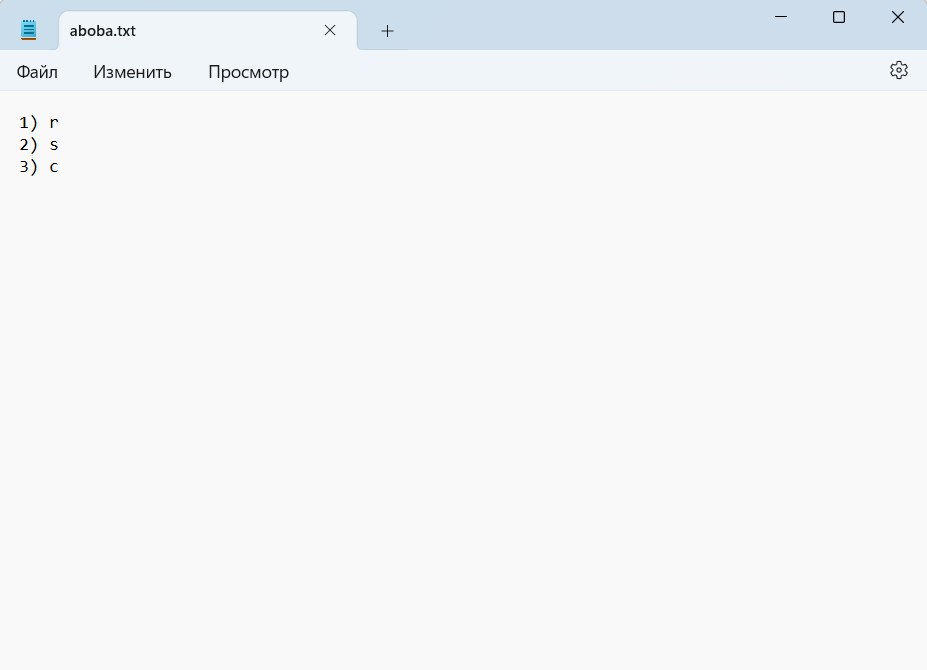
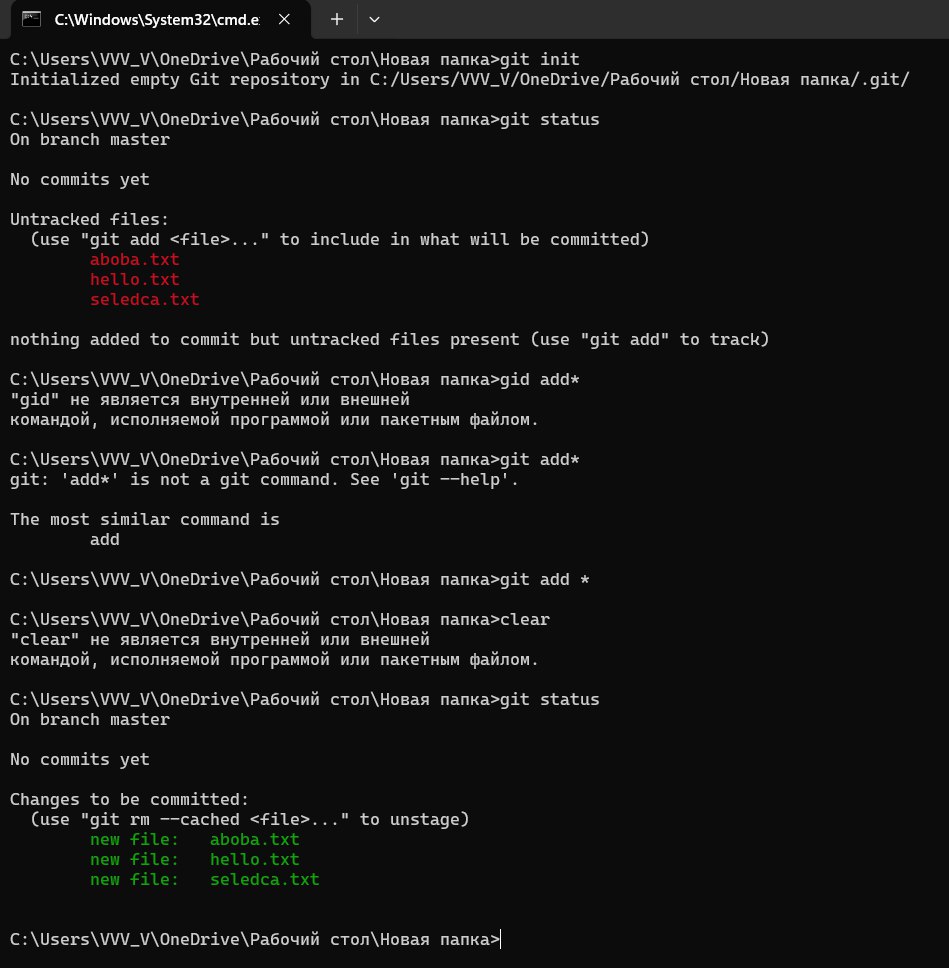
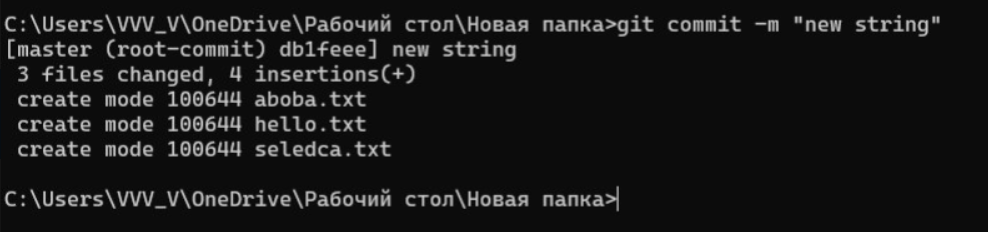


Рис. 3. Изменение в одном из файлов.

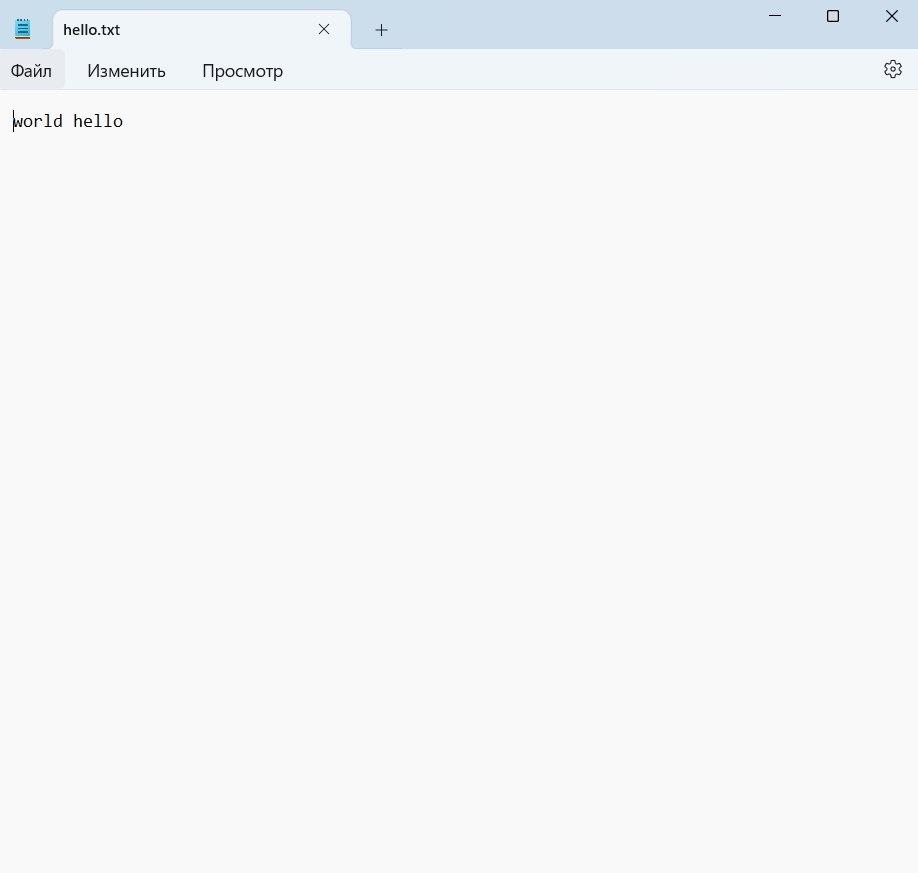
Проиндексируйте изменения и проверьте состояние.

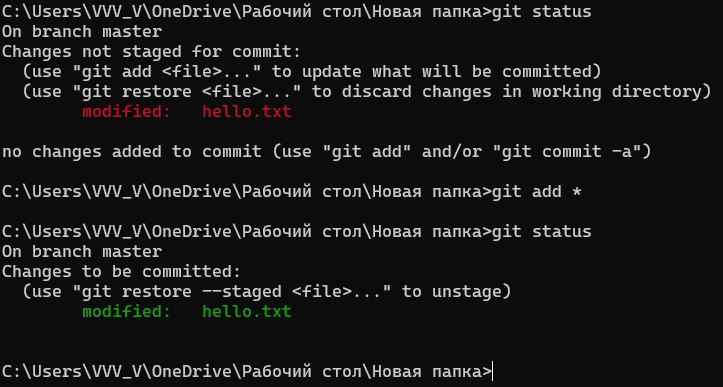
Рис. 4. Проиндексированные элементы и проверка состояния

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

Рис. 5. Коммит с комментарием

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.

Рис. 6. Изменение файла

Рис. 7. Производим коммит индексации

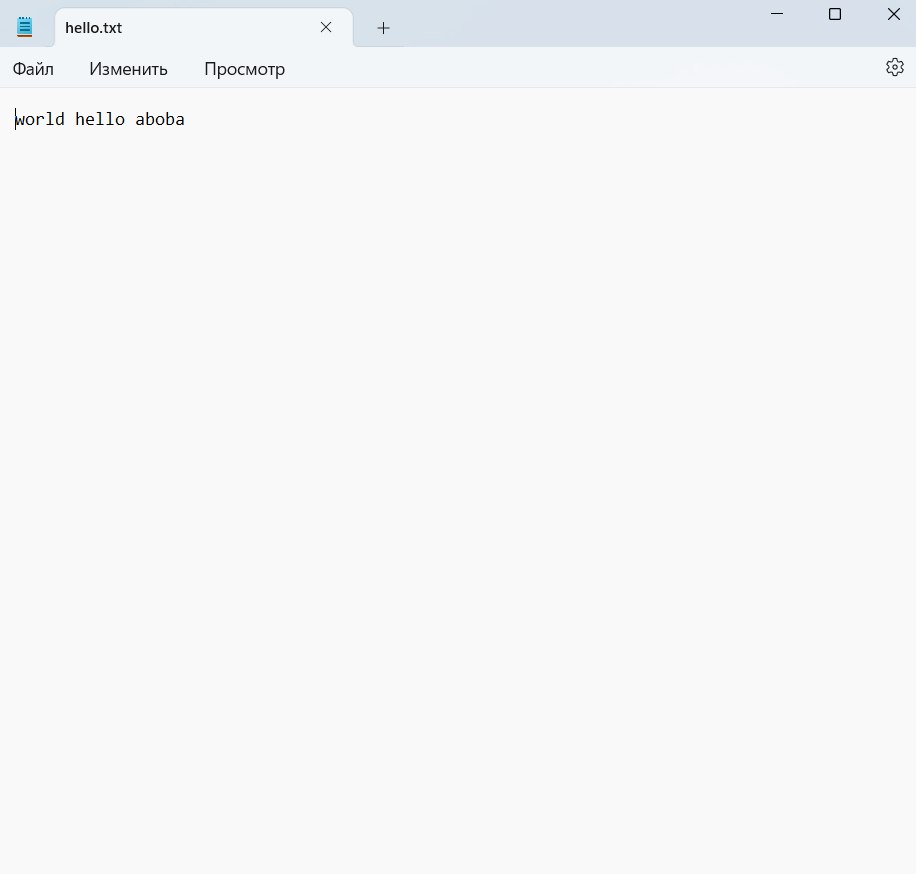
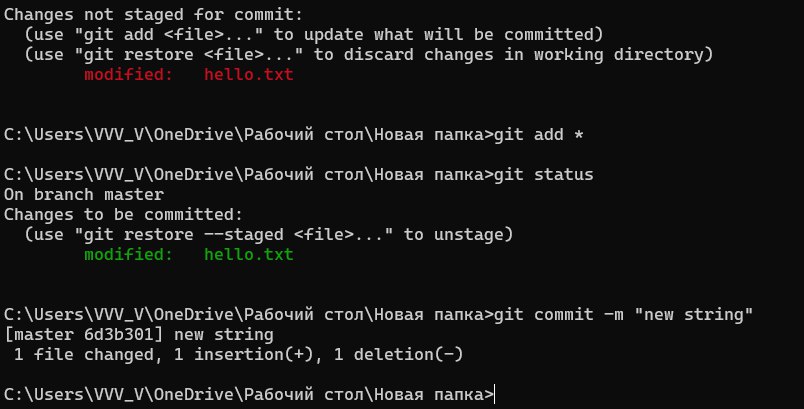
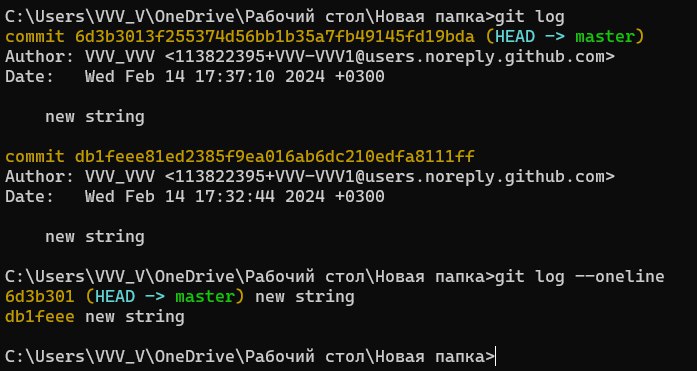


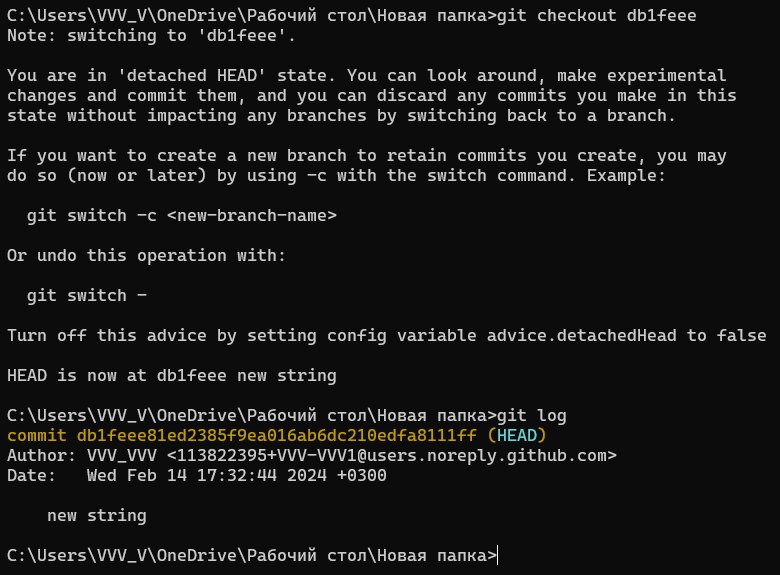
Рис. 8. Повторное изменение файла

Рис. 9. Коммит и проверка изменения файла

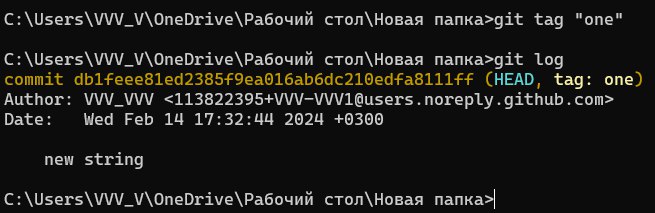
Просмотрите историю коммитов с помощью команды git log. Ознакомьтесь с параметрами команды и используйте некоторые из них для различного формата отображения истории коммитов.

Рис. 10. Использование команды git log и git log –oneline

Верните рабочий каталог к одному из предыдущих состояний.

Рис. 11. Возвращение рабочего каталога

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

Рис. 12. Создание тега

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

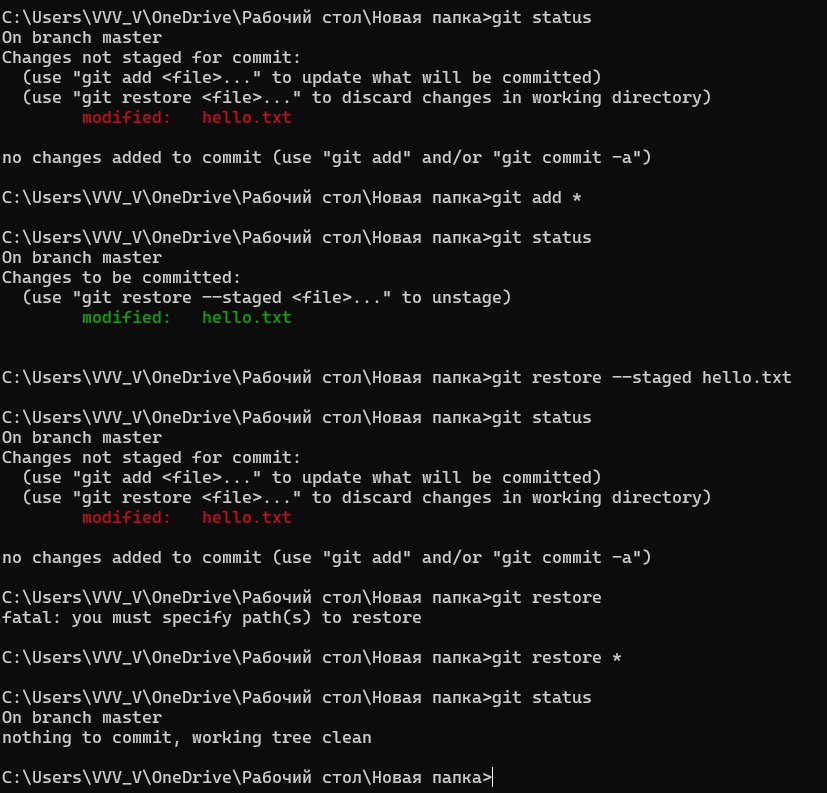


Рис. 13. Отмена некоторых изменений

Отмените один из коммитов в локальном репозитории.

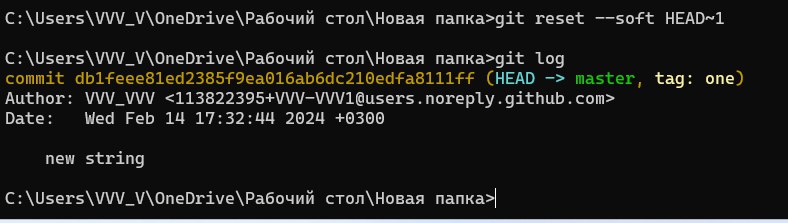


Рис. 14. Отмена одного из коммитов

Часть 2. Системы управления репозиториями

Создайте аккаунты на GitHub.

Создайте репозиторий согласно варианту. Вариант №7. Место по списку 29.

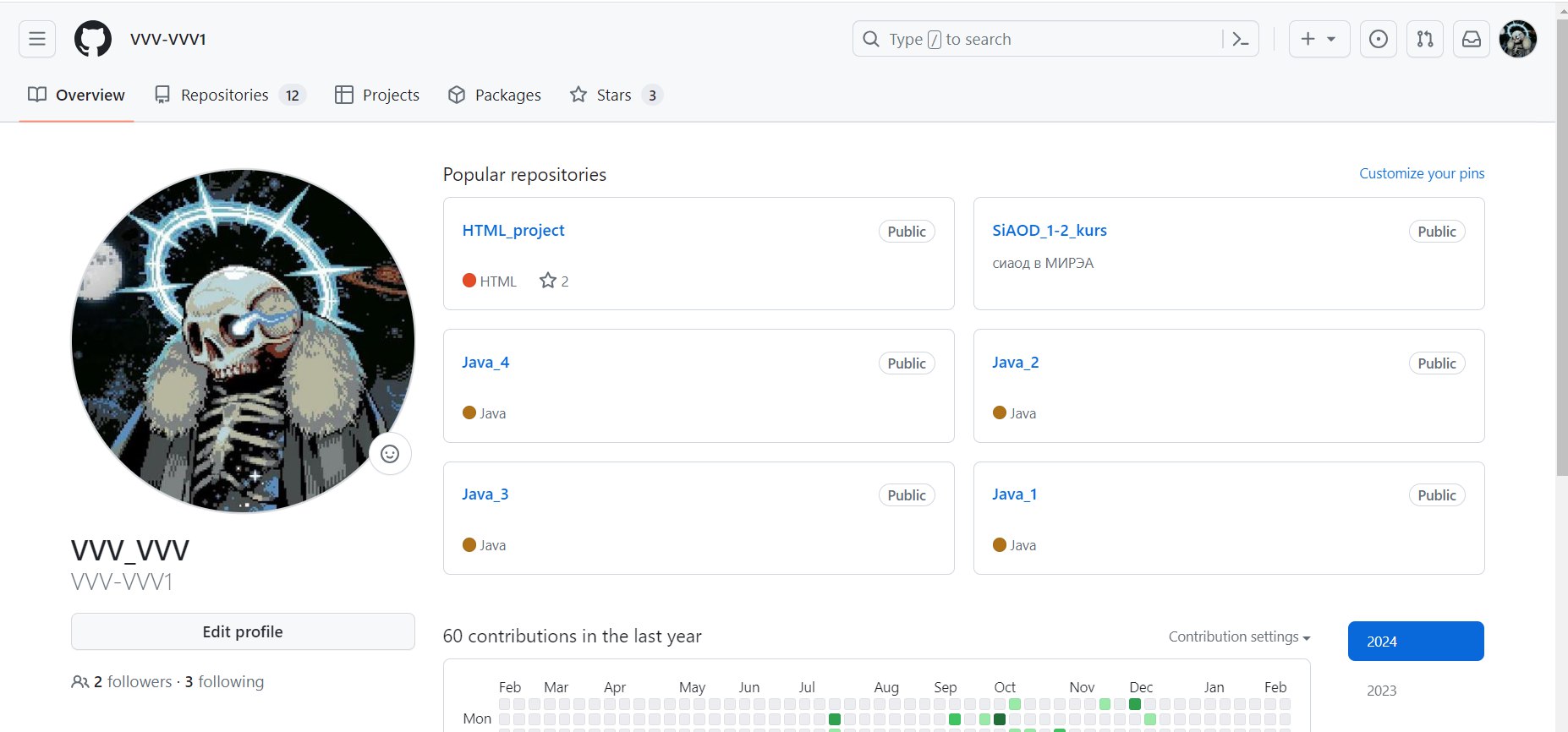


Рис. 15. Создание аккаунта GitHub и репазитория

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

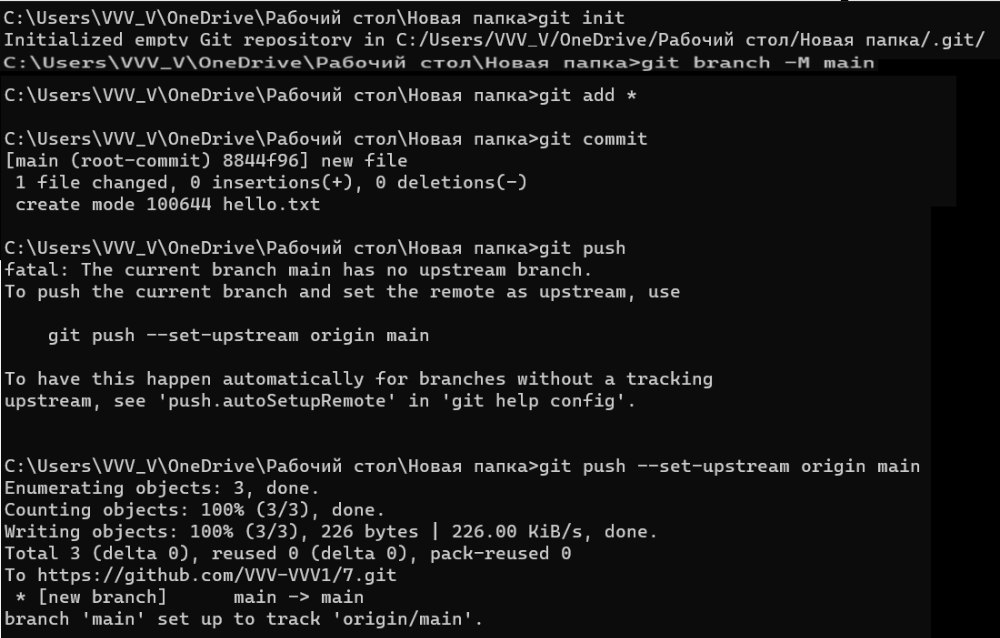


Рис. 16. Создание нового локального репозитория

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

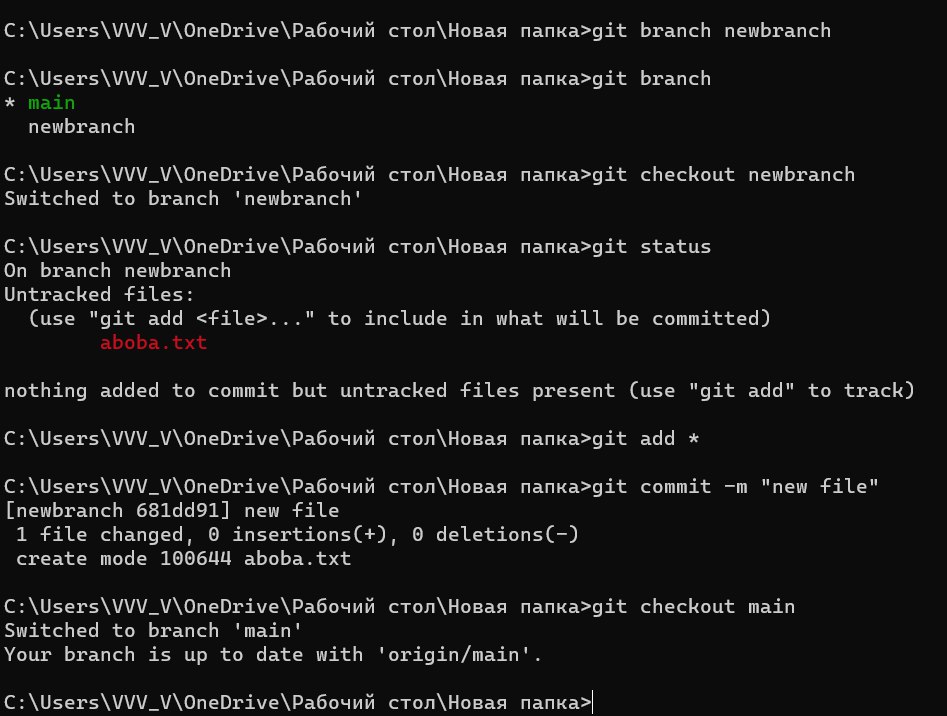


Рис. 17. Создание новый ветви

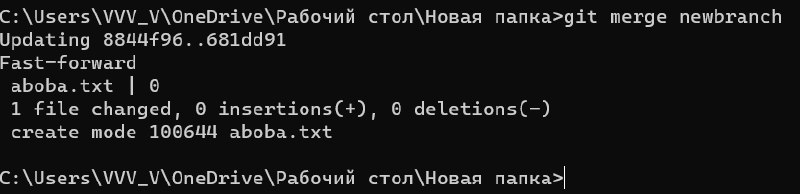


Рис. 18. Слияние ветвей

Клонируйте непустой удаленный репозиторий на локальную машину

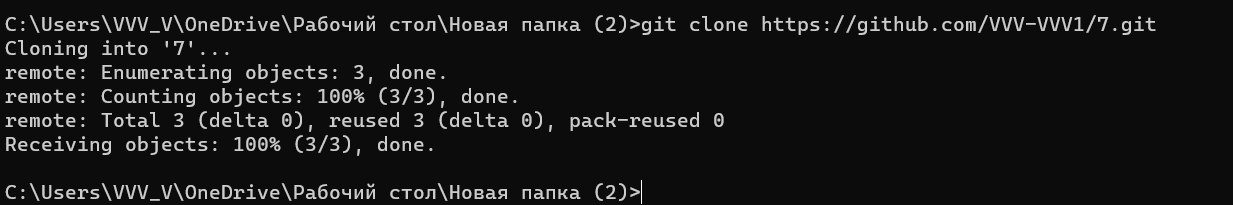


Рис. 19. Клонирование репозитория

Создайте две новых ветки и выведите список всех веток

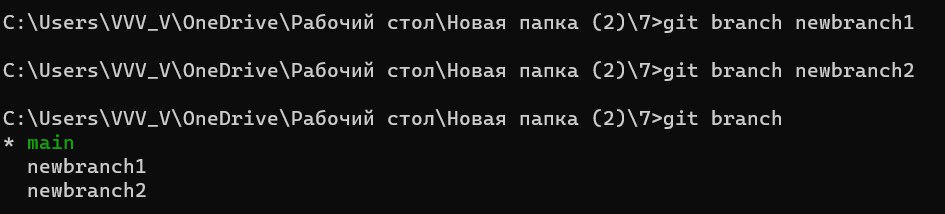


Рис. 20. Создание двух веток

Произведите 3 коммита в первой ветке

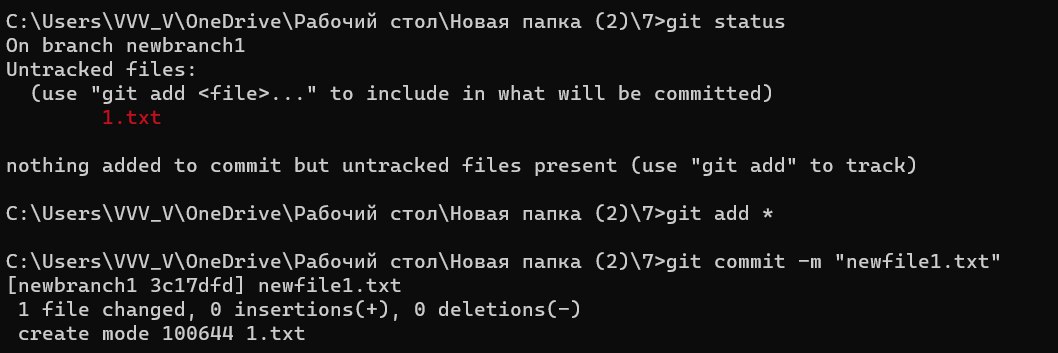


Рис. 21. 1 коммит

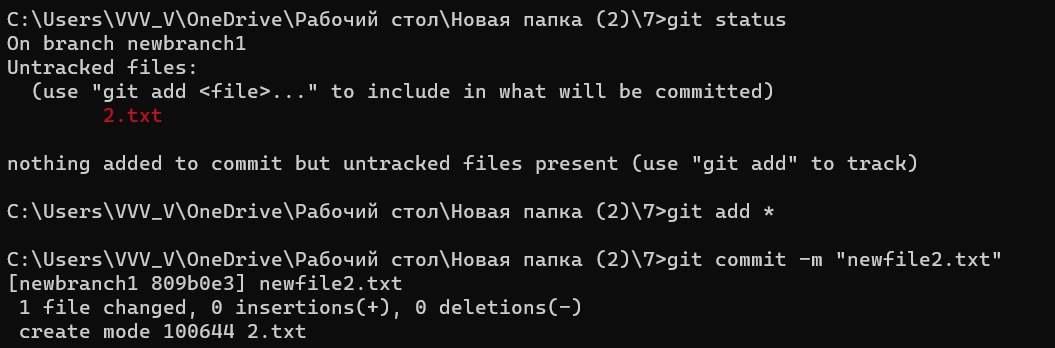


Рис. 22. 2 коммит

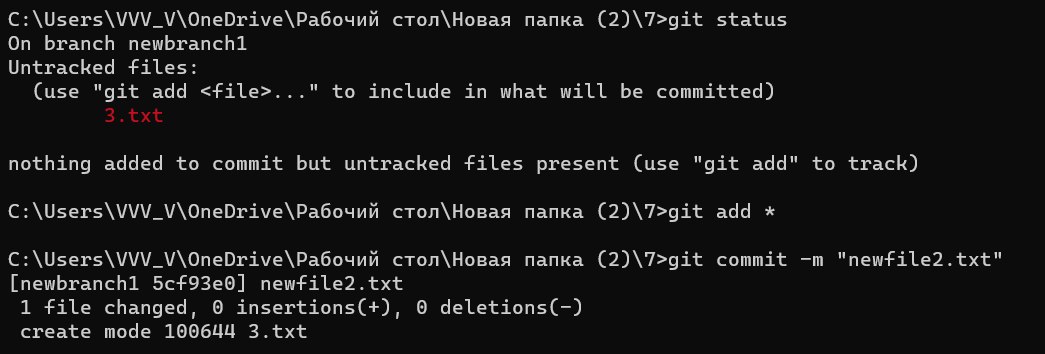


Рис. 23. 3 коммит

Переместите (cherry pick) первый коммит во вторую ветку

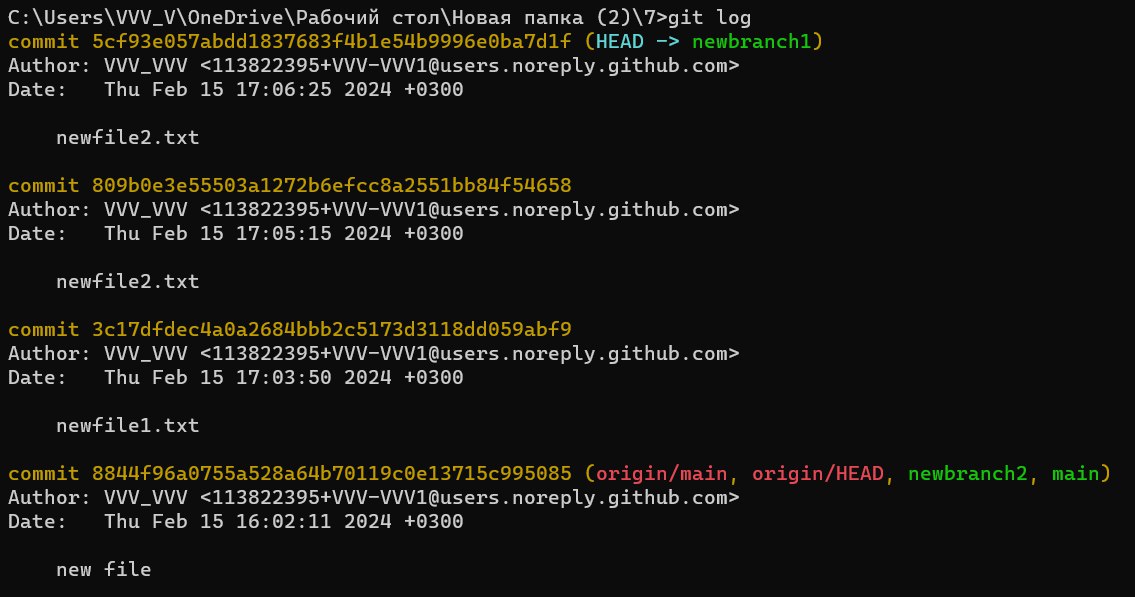


Рис. 24. Узнаем хеш коммитов

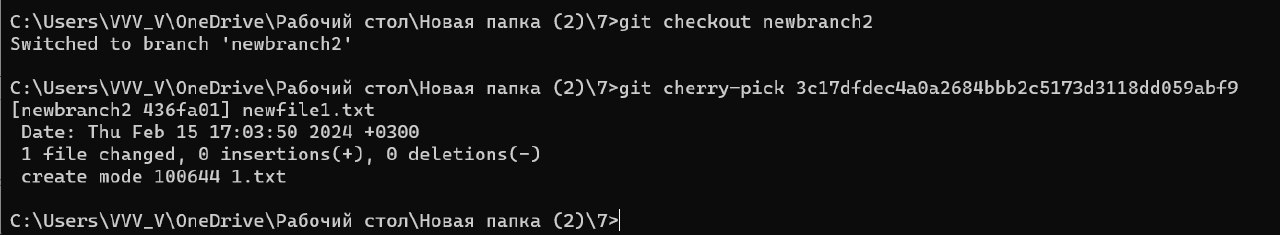


Рис. 25. Перемешаем 1 коммит в новою ветвь

Выгрузите изменения в удаленный репозиторий

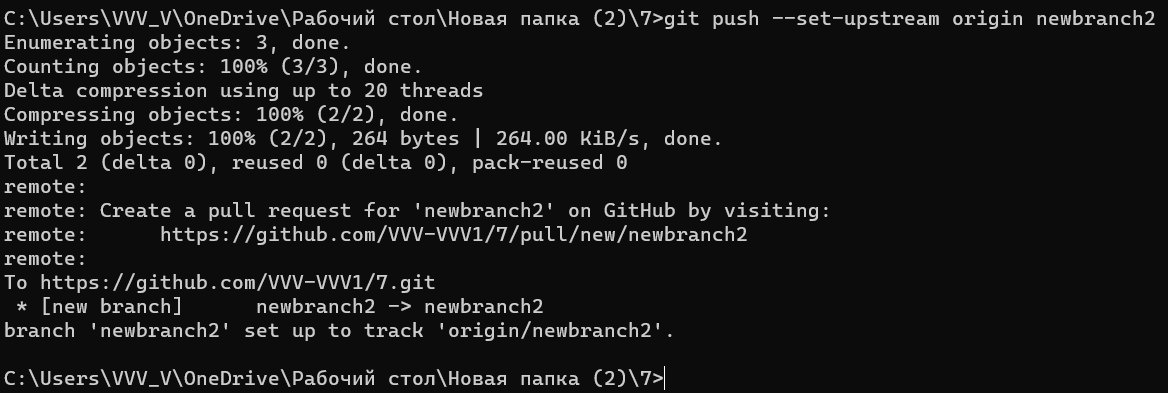


Рис. 26. Выгружаем репозиторий

Выведите в консоли различия между веткой master и новыми ветками

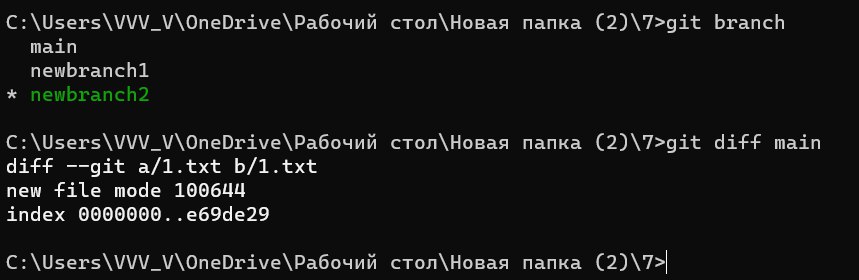


Рис. 27. Разница между main и newbramch2

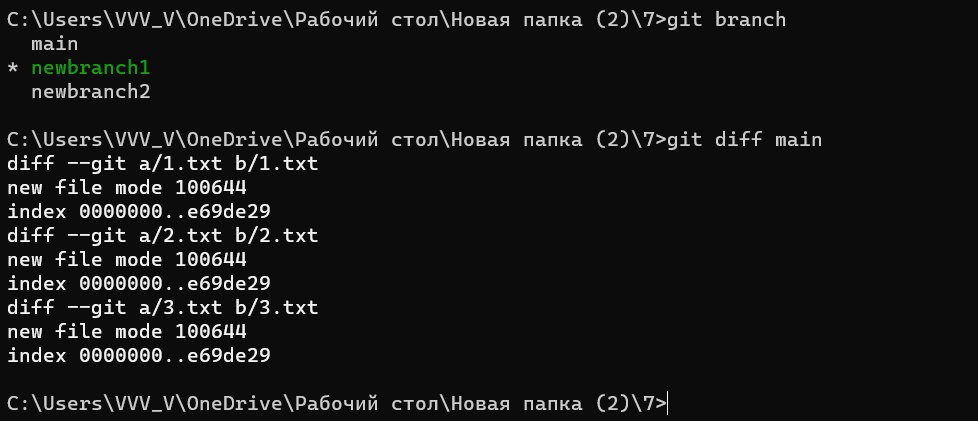


Рис. 28. Разница между main и newbramc1

Слейте обе ветки с master при помощи merge

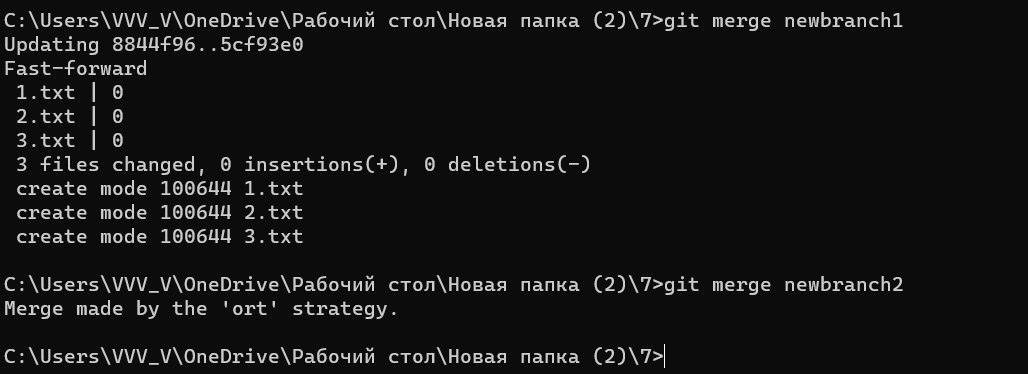


Рис. 29. Слияние ветвей

Часть 3. Работа с ветвлением и оформление кода

Сделайте форк репозитория в соответствии с Вашим вариантом. Вариант №9. Место по списку 29.

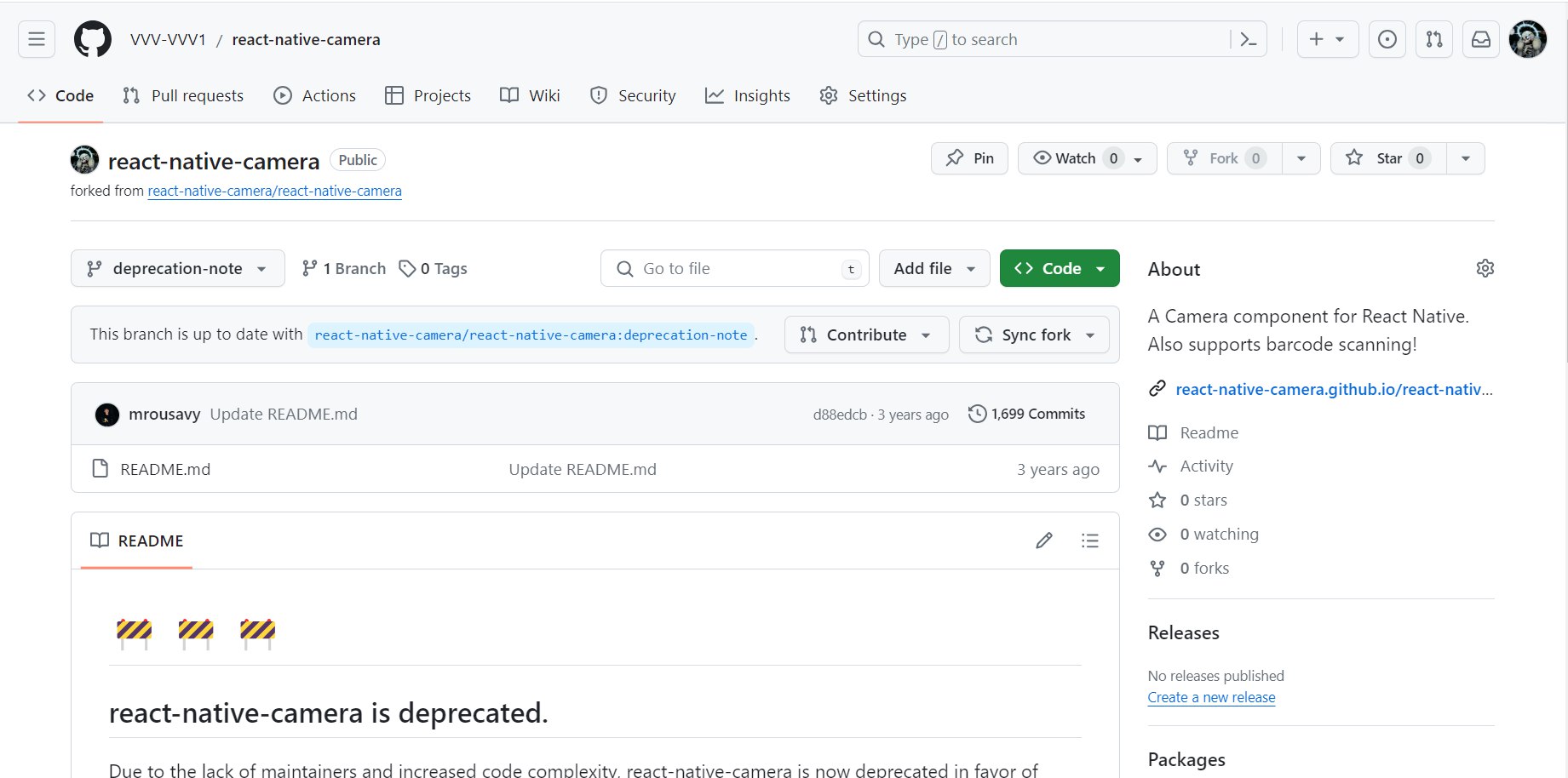


Рис. 30. Создание форк

Склонируйте его (форк репозитория) на локальную машину.

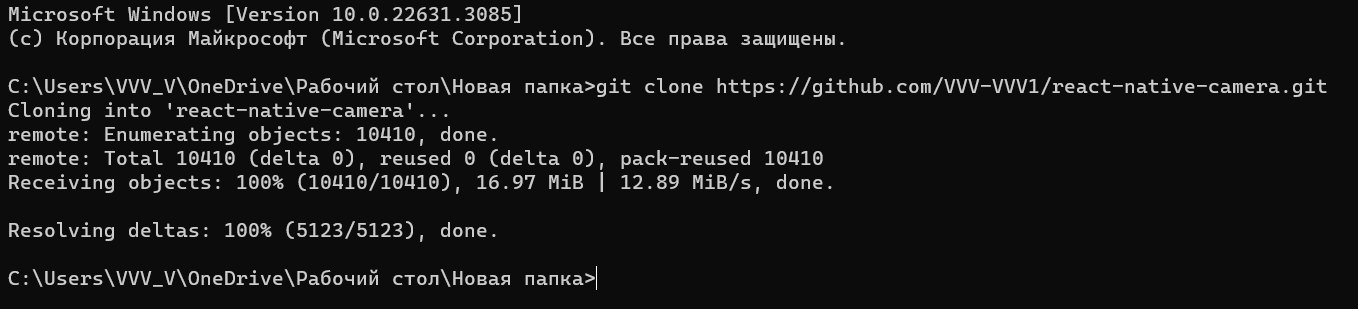


Рис. 31. Создание клона

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

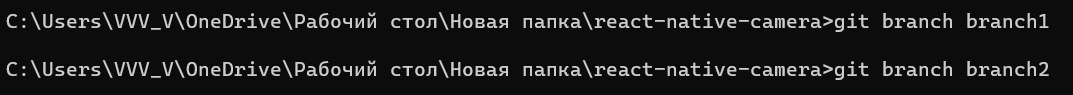


Рис. 32. Создание ветвей

Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла. Выполните слияние ветки branch1 в ветку branch2, разрешив при этом конфликты.

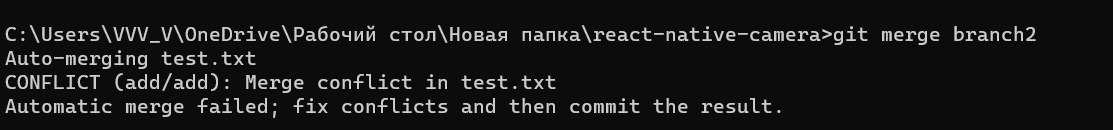


Рис. 33. Вывод ошибки при слиянии

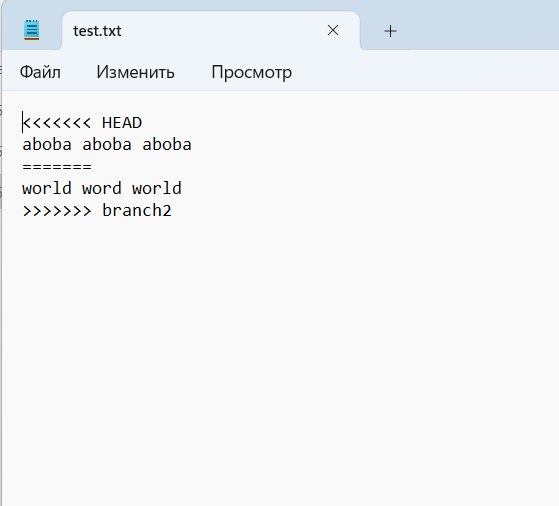


Рис. 34. Файл с ошибкой

Выгрузите все изменения во всех ветках в удаленный репозиторий.

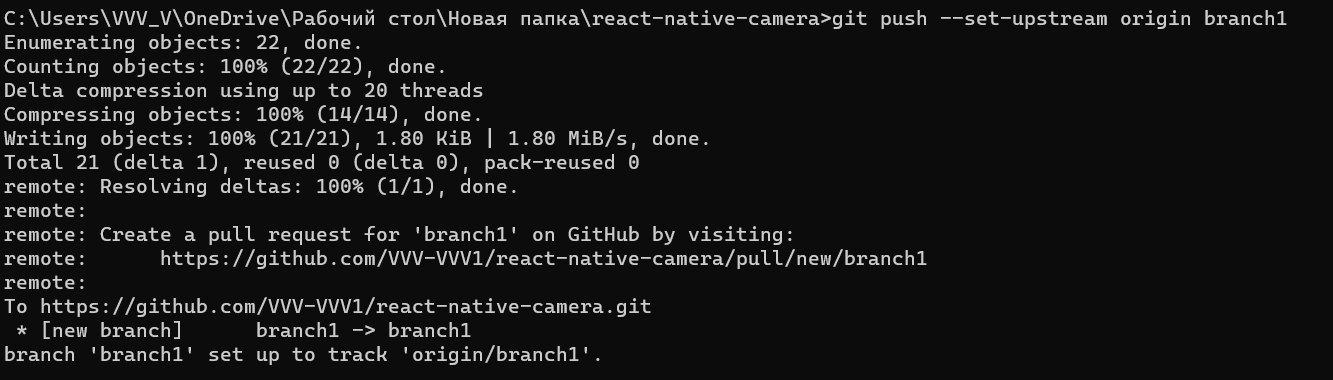


Рис. 35. Выгрузка изменений

Проведите еще 3 коммита в ветку branch1.

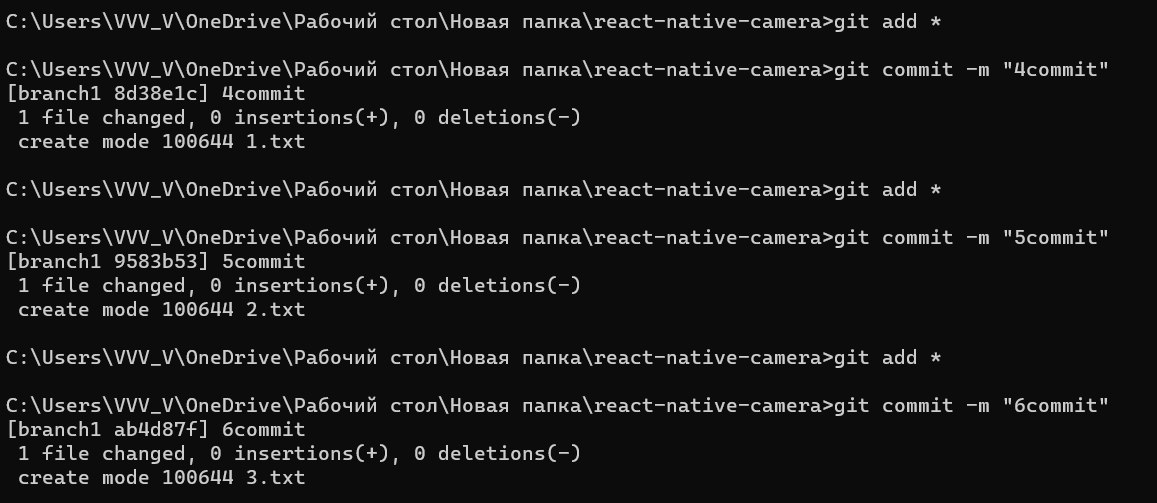


Рис. 36. Создание 3 коммитов

Склонируйте репозиторий еще раз в другую директорию.

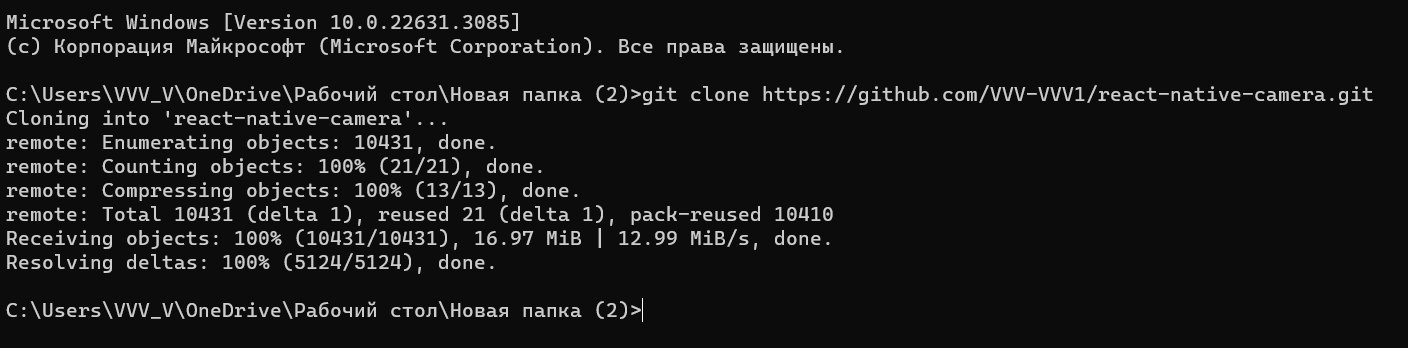


Рис. 37. Клонирования удаленного репозитория

В новом клоне репозитории сделайте 3 коммита в ветку branch1.

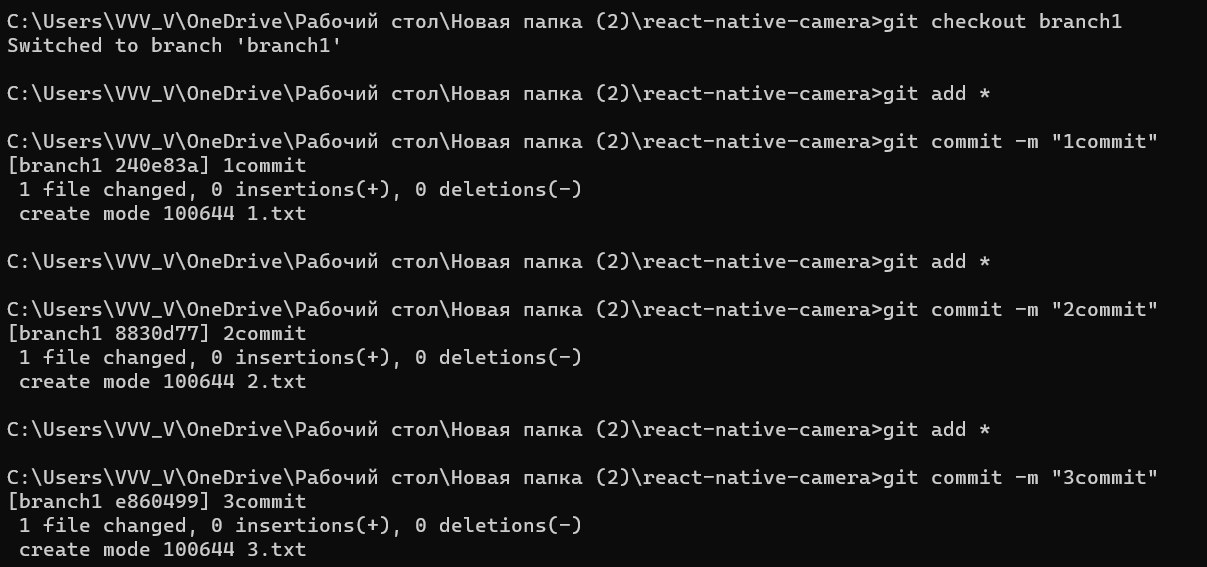


Рис. 38. Создание 3 коммитов

Выгрузите все изменения из нового репозитория в удаленный репозиторий.

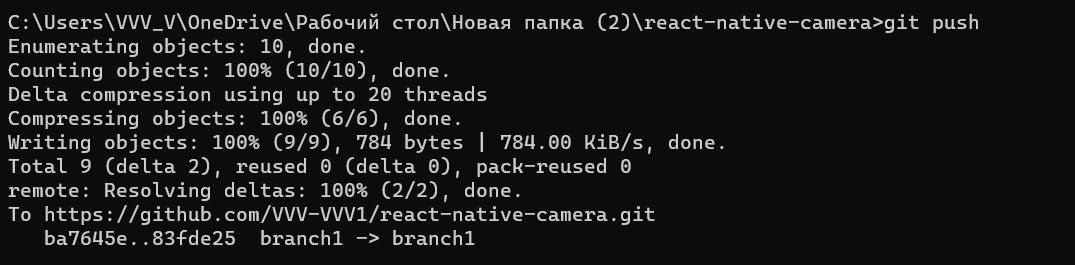


Рис. 39. Выгрузка изменений

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

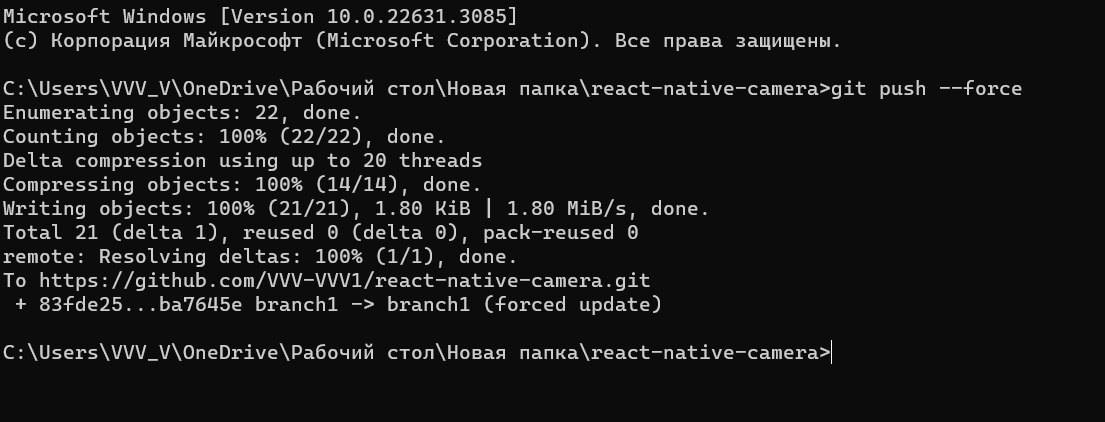


Рис. 40. Выгрузка изменений с опцией –force

Получите все изменения в новом репозитории.

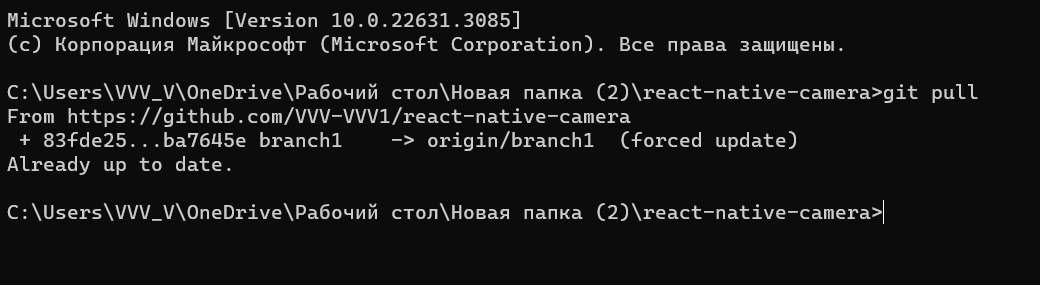


Рис. 41. Получить изменения в новый репозиторий

Контрольные вопросы

1. Какие существуют типы систем контроля версий? Приведите примеры к каждому типу.
   * Локальные системы контроля версий. – RCS(Revision Control System)
   * Централизованные системы контроля версий. - Subversion (SVN)
   * Распределённые системы контроля версий. - Git,
2. К какому типу систем контроля версий относится Git?
   * Распределённые системы контроля версий.
3. Что такое репозиторий Git?
   * Репозиторий Git — это папка, в которой в Git отслеживаются изменения.
4. Что такое коммит?
   * Фиксация набора связанных изменений.
5. Что такое ветка в репозитории Git?
   * Ветка в Git — это набор коммитов, расположенных в хронологическом порядке.
6. Что такое тег в репозитории Git?
   * Git-теги – этоссылки на конкретные версии вашего кода в репозитории Git.
7. Что такое слияние двух веток?
   * Слияние веток или merge (мердж) — это создание коммита в основной ветке
8. Что такое конфликт в Git? Как его решить и почему они бывают?
   * Конфликт в Git происходит при слиянии ветвей, имеющих конкурирующие фиксации.
   * Конфликт можно решить вручную изменив изменений файл

Вывод.

Были изучены основные команды git, а также работа с разными версиями проектов и решение конфликтов при слиянии ветвей.