Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**ОТЧЕТ**

**ПО РАБОТЕ №1.3.**

**дисциплины «Основы кроссплатформенного программирования»**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнил:  Хачатрян Владимир Владимирович  1 курс, группа ИТС-б-о-22-1,  11.03.02 «Инфокоммуникационные технологии и системы связи», направленность (профиль) «Инфокоммуникационные системы и сети», очная форма обучения  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Руководитель практики:  Воронкин Р.А., канд. тех. наук, доцент, доцент кафедры инфокоммуникаций  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2023 г.

**Тема:** основы ветвления Git.

**Цель работы:** исследовать базовые возможности по работе с локальными и удаленными ветками Git.

**Ход работы:**

1. Создал репозиторий на GitHub.

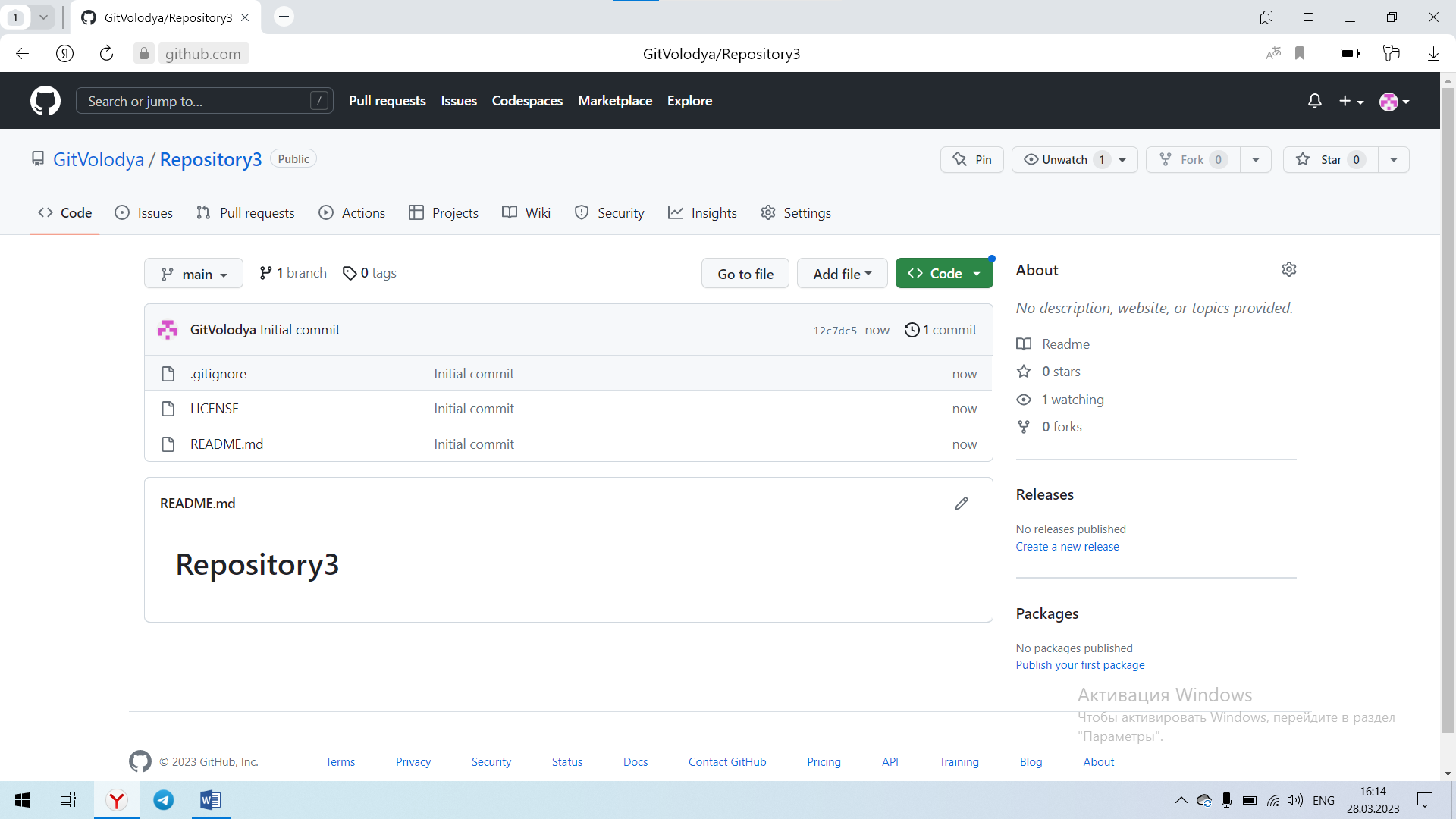


Рисунок. 1. Новый репозиторий.

1. Проклонировал репозиторий на свой компьютер.

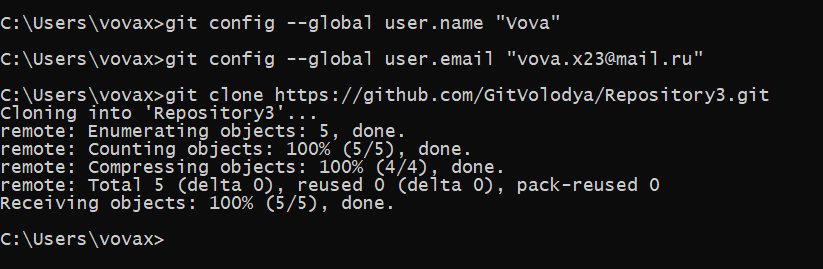


Рис. 2. Клонирование репозитория.

1. Добавил 3 текстовых файла.

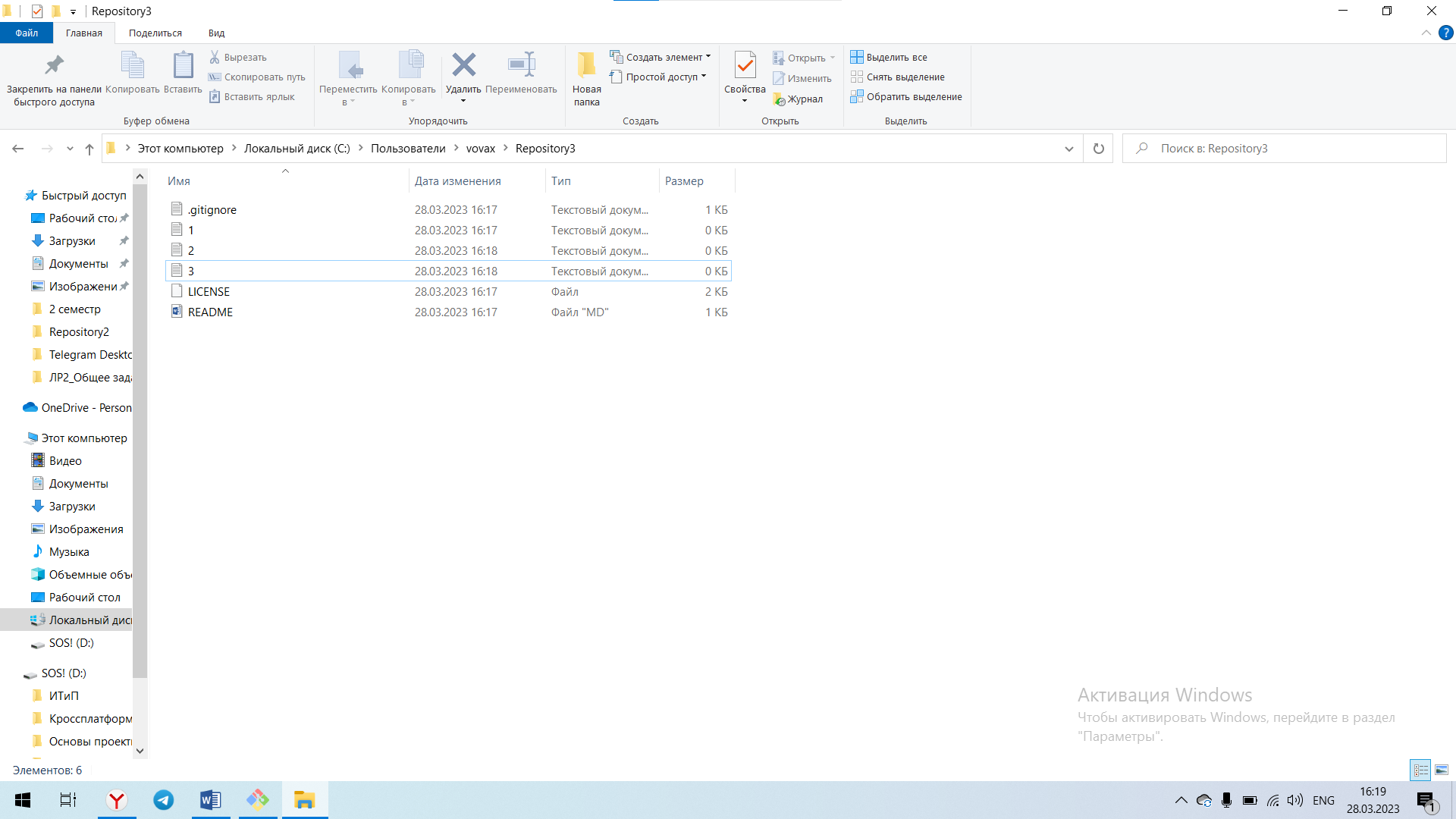


Рис. 3. Текстовые файлы.

1. Проиндексировал первый файл и сделать коммит.

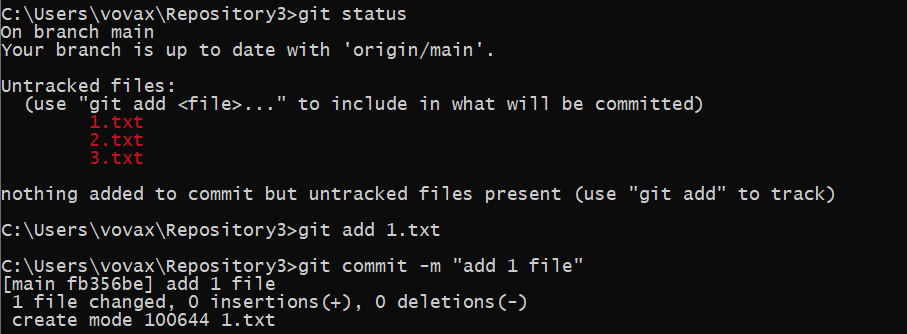


Рисунок. 4. Добавление изменений для перового файла и их фиксация.

1. Проиндексировал второй и третий файлы.

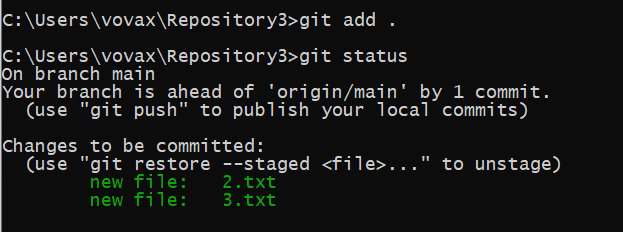


Рисунок. 5. Добавление изменений.

1. Перезаписал уже сделанный коммит с новым комментарием.

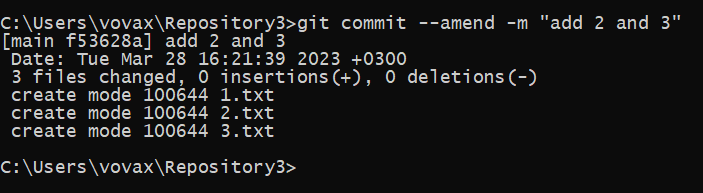


Рисунок 6. Редактирование существующего коммита.

1. Создал новую ветку my\_first\_branch.



Рис. 7. Новая ветка.

1. Перейти на ветку и создать новый файл in\_branch.txt, закоммитид изменения.

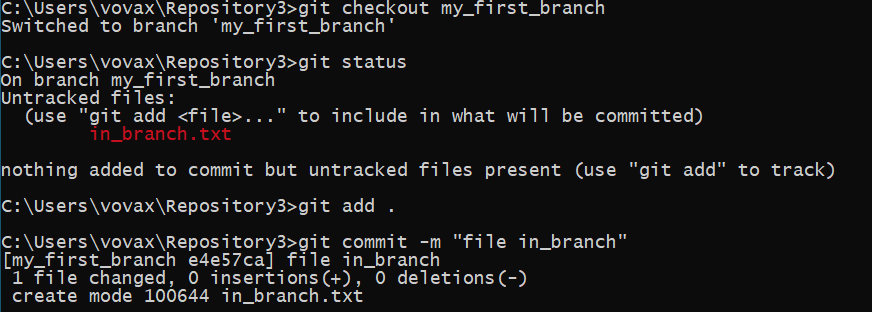
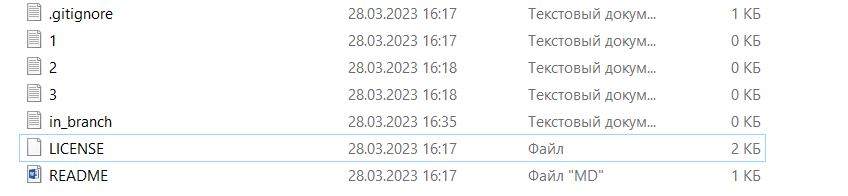


Рис. 8. Перешёл на ветку my\_first\_branch.



1. Перешёл на новую ветку main.



Рис. 9. Переход на главную ветку.

1. Создал и сразу перешел на ветку new\_branch.



Рис. 10. Создание и сразу переход на ветку.

1. Сделал изменения в файле 1.txt, добавил строчку “new row in the 1.txt file”, закоммитил изменения.

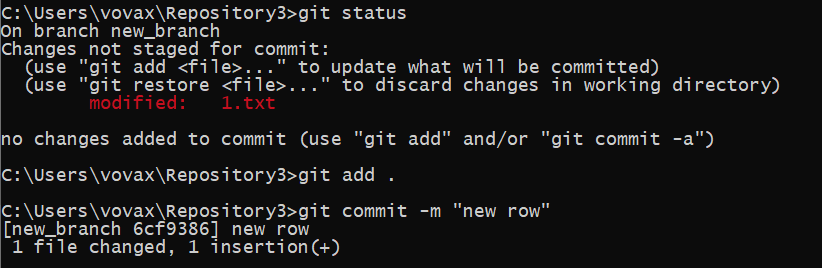


Рис. 11. Изменение в 1.txt фиксация этих изменений.

1. Перешёл на ветку main и слил ветки main и my\_first\_branch, после чего слил ветки main и new\_branch.

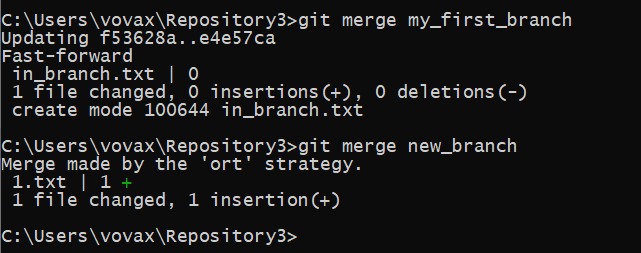


Рис. 12. Слияние веток.

1. Удалил ветки my\_first\_branch и new\_branch.

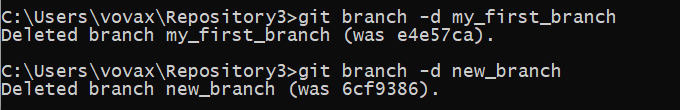


Рис. 13. Удаление веток.

1. Создал ветки branch\_1 и branch\_2.

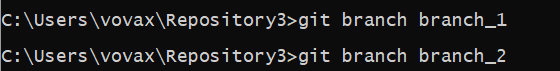


Рис. 14. Создание новых веток.

1. Перешел на ветку branch\_1 и изменил файл 1.txt, удалил все содержимое и добавил текст “fix in the 1.txt”, изменил файл 3.txt, удалил все содержимое и добавил текст “ fix in the 3.txt ”, закоммитил изменения.

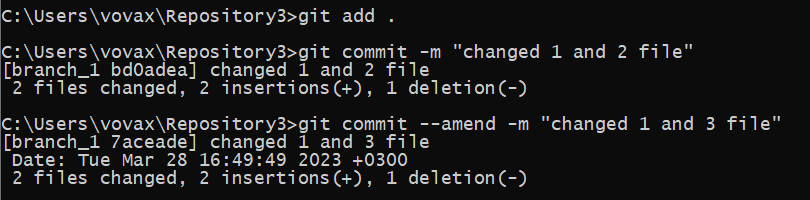


Рис. 15. Изменения в первой ветке и во второй ветка и фиксация изменений.

1. Перешел на ветку branch\_2 и также изменил файл 1.txt, удалил все содержимое и добавил текст “My fix in the 1.txt”, изменил файл 3.txt, удалил все содержимое и добавил текст “My fix in the 3.txt”, закоммитил изменения.

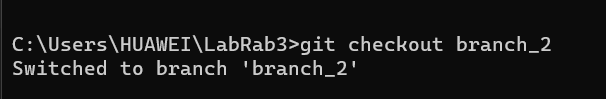


Рис. 16. Переход на вторую ветку.

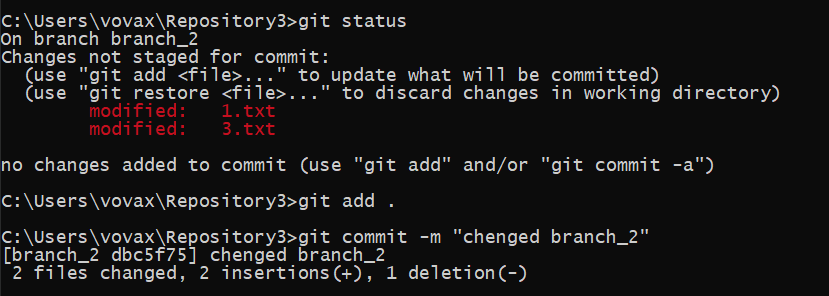


Рис. 17. Изменения файлов и их фиксация.

1. Слил изменения ветки branch\_2 в ветку branch\_1.

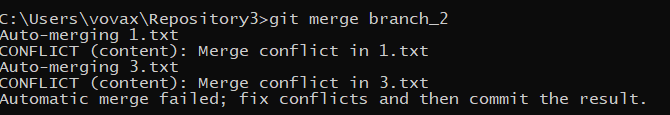


Рис. 18. Слияние веток.

1. Решил конфликт файла 1.txt в ручном режиме, а конфликт 3.txt используя команду git it mergetool c помощью одной из доступных утилит, например, vimdiff.

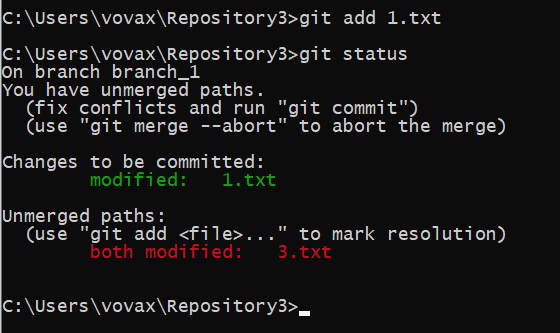
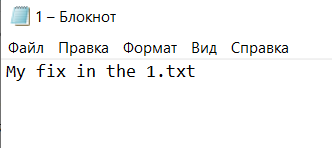


Рис. 19. Решение конфликта в ручную.

Изменю 3 текстовый файл с помощью инструмента vimdiff.

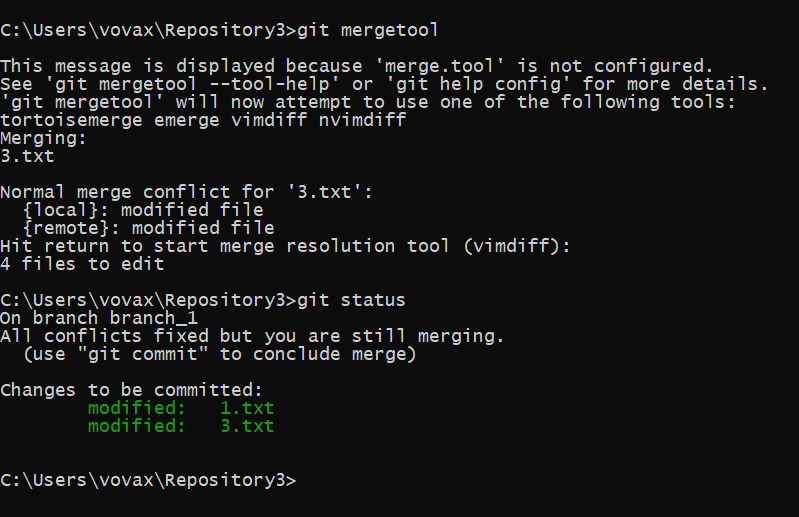


Рис. 20. Решение конфликта с помощью инструмента.

1. Отправил branch\_1 на удаленный сервер с помощью команды: *git push origin branch\_1.*

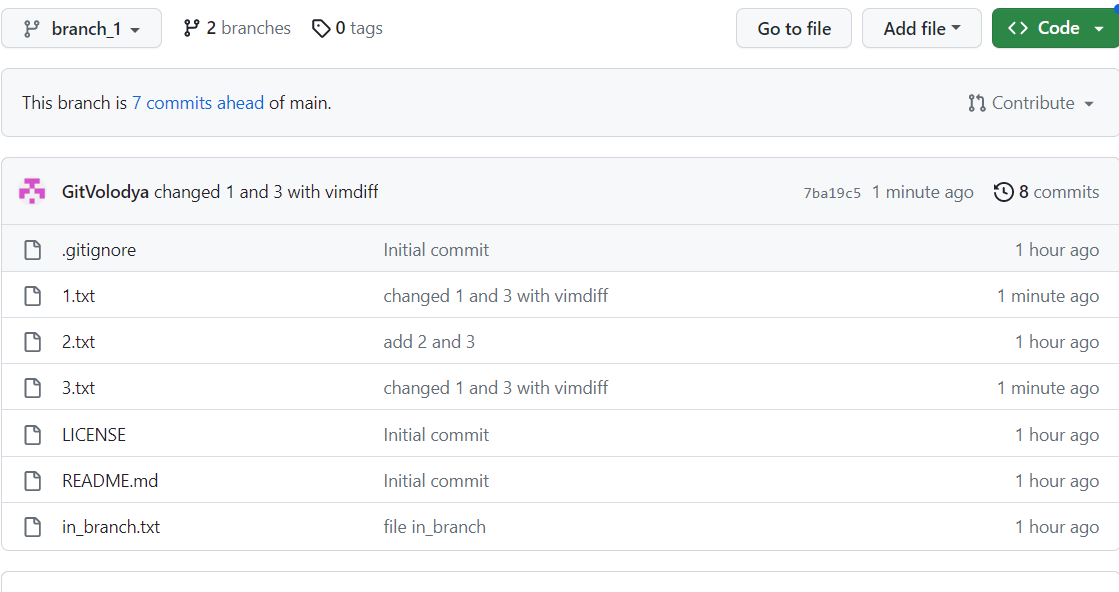


Рис. 21. Отправление ветку на удаленный сервер.

1. Создал средствами GitHub удаленную ветку branch\_3.

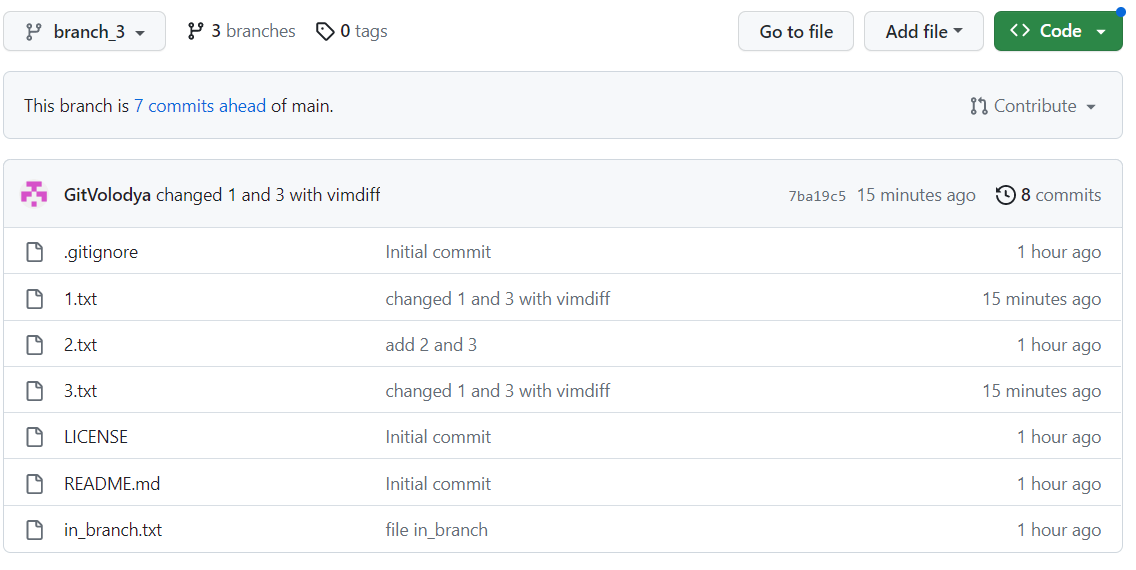


Рис. 22. Создание ветки на GitHub.

1. Создал в локальном репозитории ветку отслеживания удаленной ветки branch\_3.

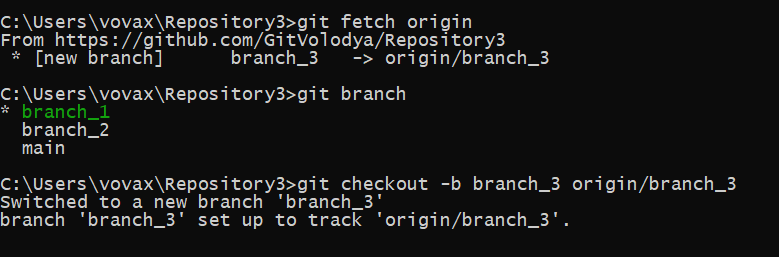


Рис. 23. Создание ветки отслеживания.

1. Перешел на ветку branch\_3 и добавил файл файл 2.txt строку "the final fantasy in the 4.txt file".

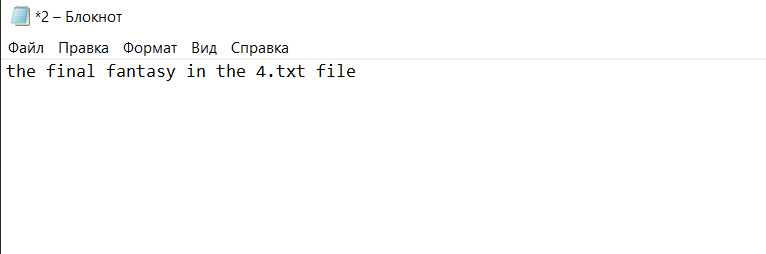


Рис. 24. Изменение файла во второй ветке.

1. Выполнил перемещение ветки main на ветку branch\_2.

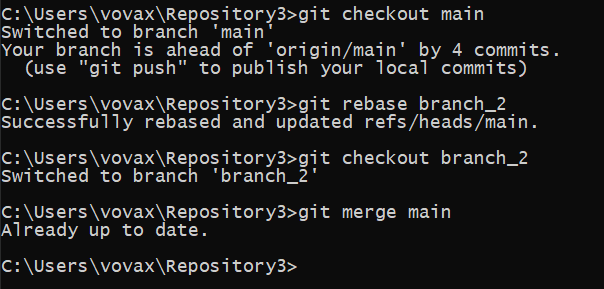


Рис. 25. Перемещение веток.

1. Отправил изменения веток master и branch\_2 на GitHub.

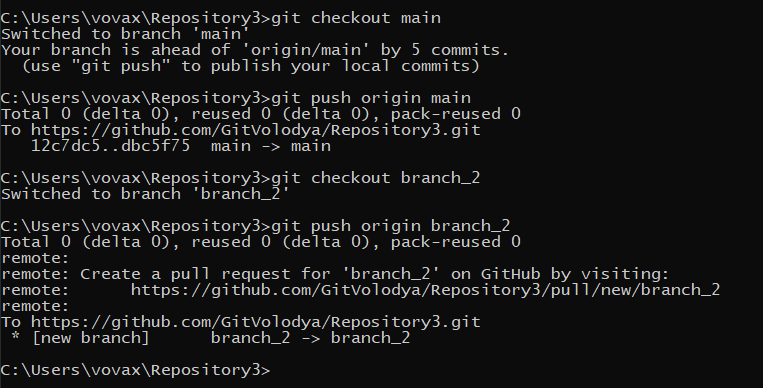


Рис. 25. Отправка изменения веток на GitHub.

Ссылка на репозиторий: https://github.com/GitVolodya/Repository3.git

**Ответы на контрольные вопросы:**

1. *Что такое ветка?*

Почти каждая система контроля версий (СКВ) в какой-то форме поддерживает ветвление. Используя ветвление, Вы отклоняетесь от основной линии разработки и продолжаете работу независимо от неё, не вмешиваясь в основную линию.

1. *Что такое HEAD?*

HEAD в Git – это указатель на текущую ссылку ветви, которая, в свою очередь, является указателем на последний сделанный вами коммит или последний коммит, который был извлечен из вашего рабочего каталога.

1. *Способы создания веток.*

Создать ветку можно с помощью двух команд. Команда, которая просто создает ветку: git branch “name\_branch”.

Команда, которая создает ветку и сразу же к ней переходит: git checkout –b “name\_branch”.

1. *Как узнать текущую ветку?*

Текущую ветку можно узнать с помощью команды: git branch.

1. *Как переключаться между ветками?*

Между ветками можно переключаться с помощью команды: git checkout “name\_branch”.

1. *Что такое удаленная ветка?*

Удаленные ветки - это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать.

1. *Что такое ветка отслеживания?*

Отслеживаемые ветки — это локальные ветки, которые напрямую связаны с удалённой веткой.

Если, находясь на отслеживаемой ветке, вы наберёте git push, Git уже будет знать, на какой сервер и в какую ветку отправлять изменения.

1. *Как создать ветку отслеживания?*

Ветку отслеживания можно создать с помощью команды: git checkout --track origin/<name\_branch>.

1. *Как отправить изменения из локальной ветки в удаленную ветку?*

Отправить изменения из локальной ветки в удаленную можно с помощью команды: git push origin <name\_branch>.

1. *В чем отличие команд git fetch и git pull ?*

Команда git fetch получает с сервера все изменения, которых у вас ещё нет, но не будет изменять состояние вашей рабочей директории. Эта команда просто получает данные и позволяет вам самостоятельно сделать слияние. Тем не менее, существует команда git pull , которая в большинстве случаев является командой git fetch , за которой непосредственно следует команда git merge.

1. *Как удалить локальную и удаленную ветки?*

Для удаление локальной ветки используется команда: git branch –d <name\_branch>

Для удаления удаленной ветки используется команда: git push –delete origin/<name\_branch>

1. *Какие основные типы веток присутствуют в модели git-flow? Как организованна работа с ветками в модели git-flow? В чем недостатки git-flow?*

Существуют следующие типы ветвей:

1. ветви функциональностей;
2. ветви релизов;
3. ветви исправлений.

Ветви функциональностей (feature branches), также называемые иногда тематическими ветвями (topic branches), используются для разработки новых функций, которые должны появиться в текущем или будущем релизах.

Ветви релизов (release branches) используются для подготовки к выпуску новых версий продукта. Они позволяют расставить финальные точки над i перед выпуском новой версии.

Ветви для исправлений (hotfix branches) весьма похожи на ветви релизов (release branches), так как они тоже используются для подготовки новых выпусков продукта, разве лишь незапланированных.

Недостатки git flow: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода.

Вторая проблема процесса git flow – сложности, возникающие из-за веток для патчей и для релиза. Подобная структура может подойти некоторым организациям, но для абсолютного большинства она просто убийственно излишяя.