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

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №1**

по дисциплине «Технологии разработки программных приложений»

**Тема практической работы:** «Системы контроля

версий»

|  |  |  |  |
| --- | --- | --- | --- |
| **Выполнил:**  Студент группы ИКБО-41-23 | | Трофимов А.А. | |
| **Проверил:** | Доцент кафедры МОСИТ,  кандидат технических наук, доцент Жматов Д.В. | |

Москва 2025

Оглавление

[ЧАСТЬ 1. ОСНОВНЫЕ КОМАНДЫ GIT 3](#_Toc190951226)

[ЧАСТЬ 2. СИСТЕМЫ УПРАВЛЕНИЯ РЕПОЗИТОРИЯМИ 12](#_Toc190951227)

[ЧАСТЬ 3. РАБОТА С ВЕТВЛЕНИЕМ И ОФОРМЛЕНИЕ КОДА 21](#_Toc190951228)

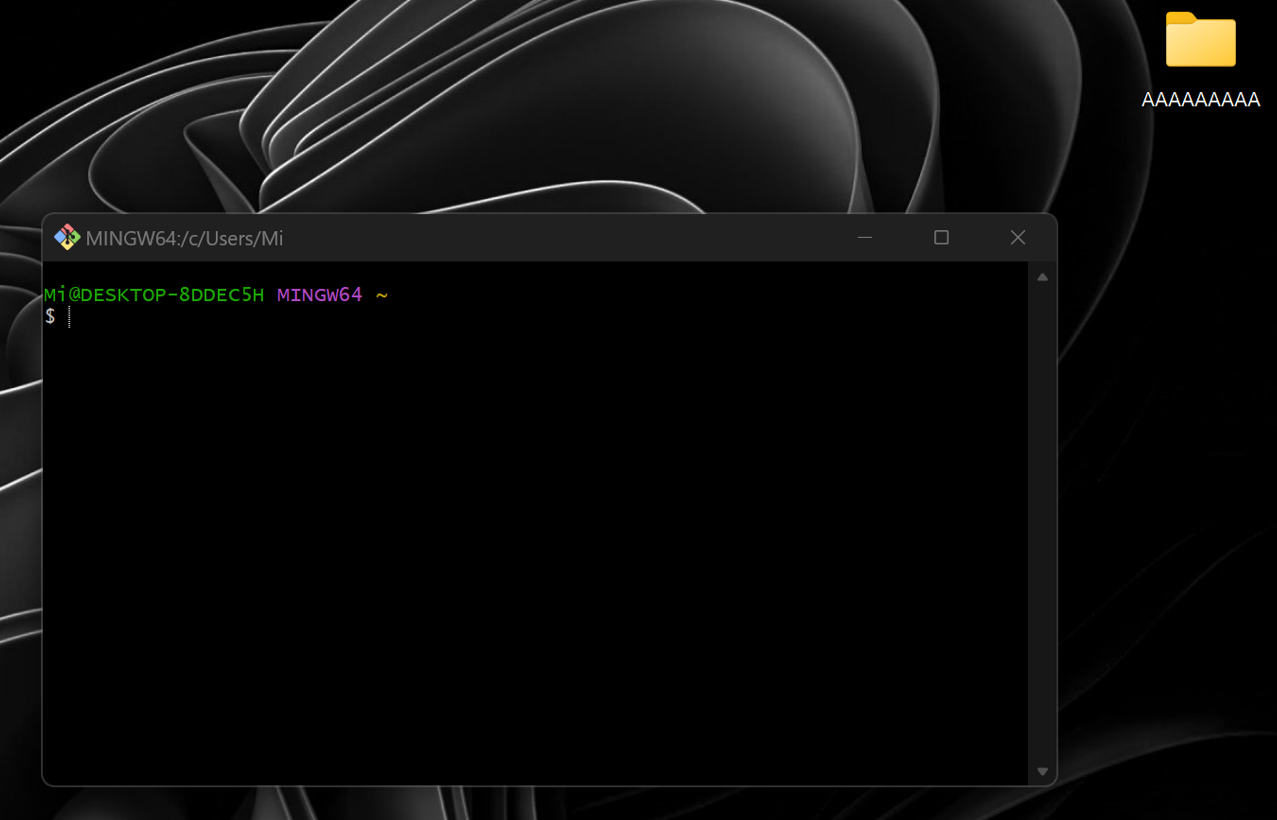
[КОНТРОЛЬНЫЕ ВОПРОСЫ 28](#_Toc190951229)

[ВЫВОД 29](#_Toc190951230)

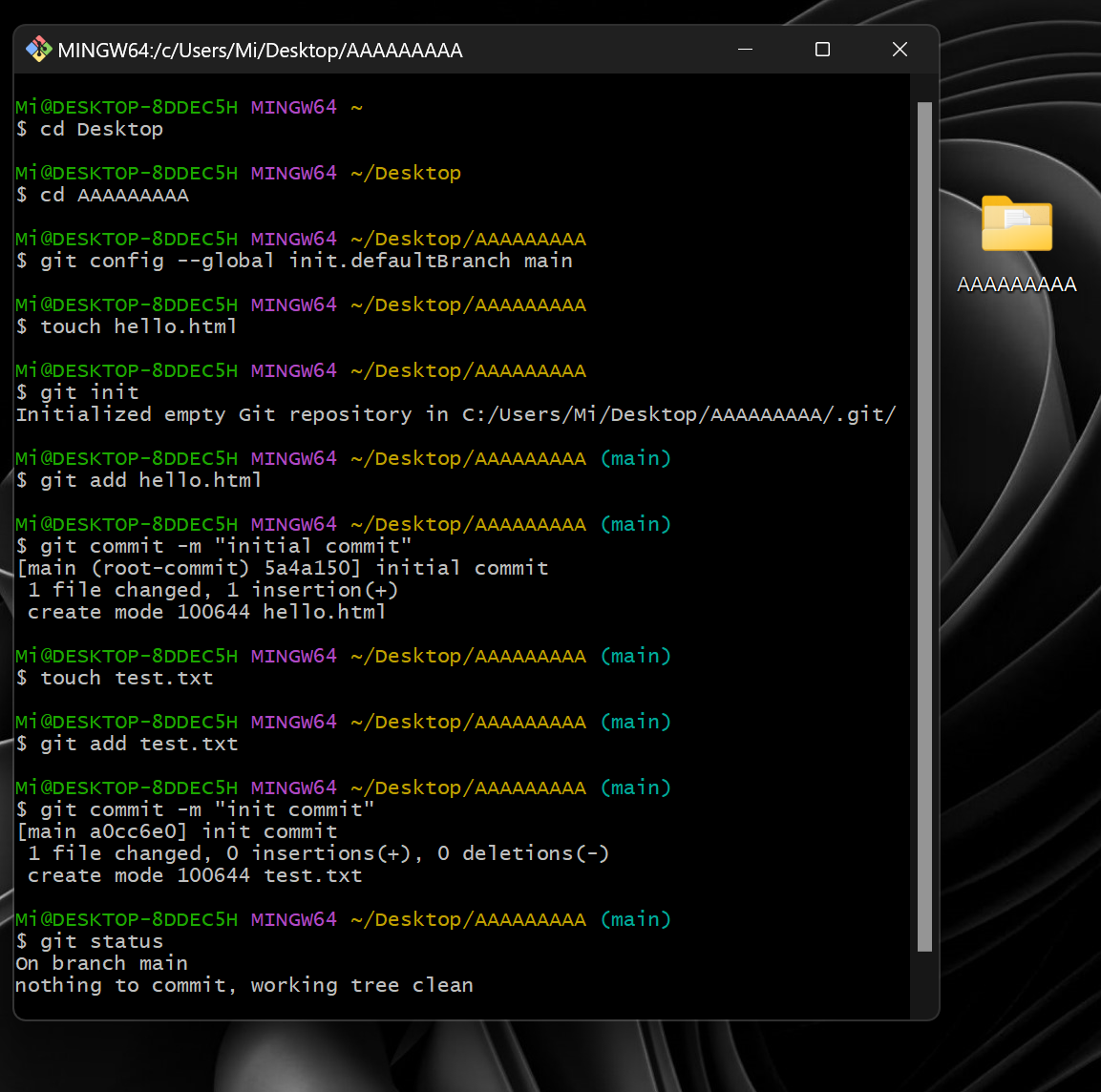
[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 30](#_Toc190951231)

**ЧАСТЬ 1. ОСНОВНЫЕ КОМАНДЫ GIT**

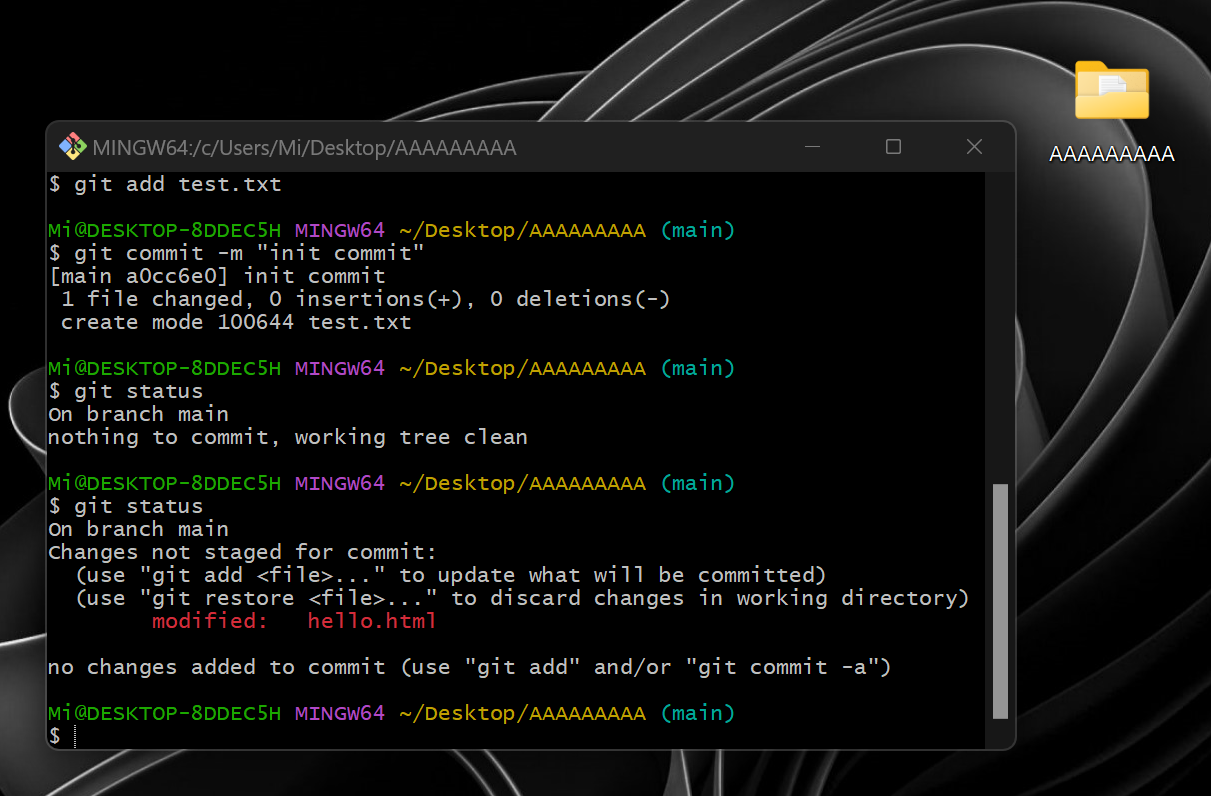
1. Установить и настроить клиент git на своей рабочей станции.



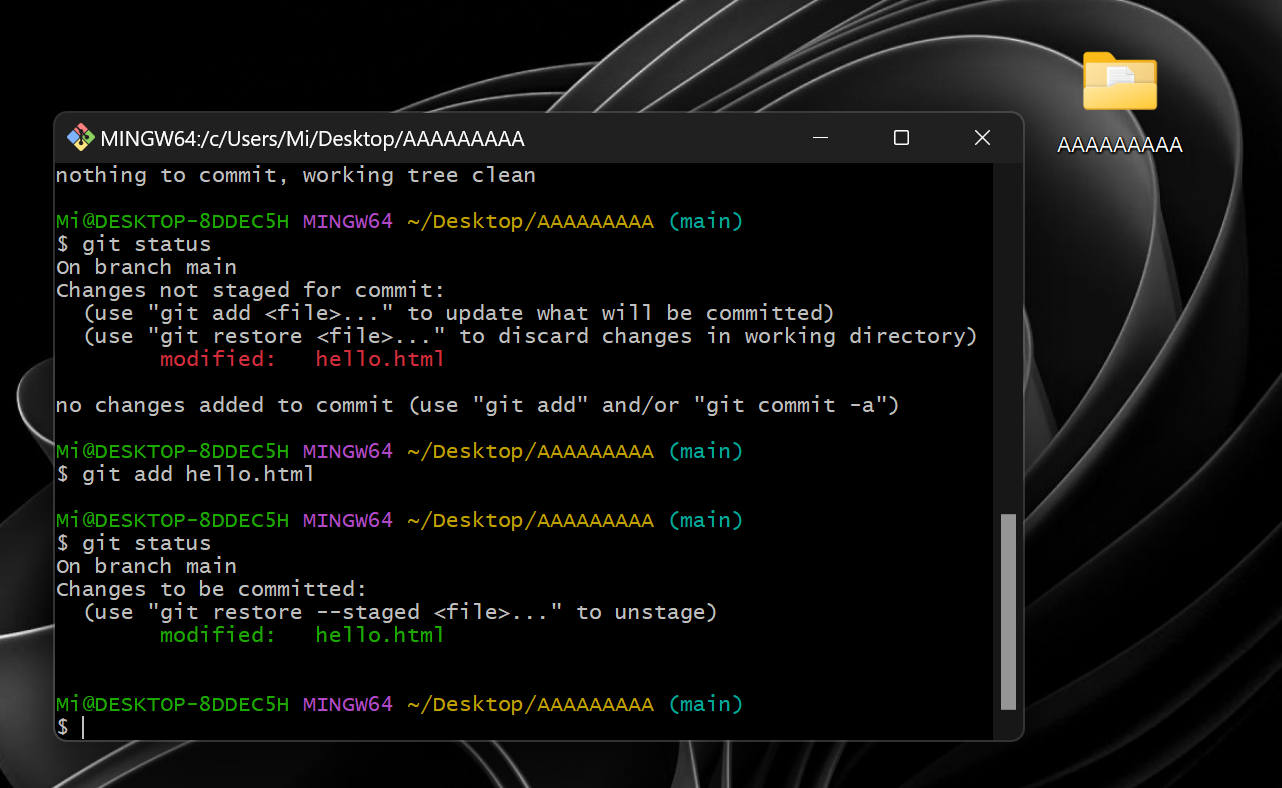
1. Создать локальный репозиторий и добавить в него несколько файлов.



1. Внести изменения в один из файлов

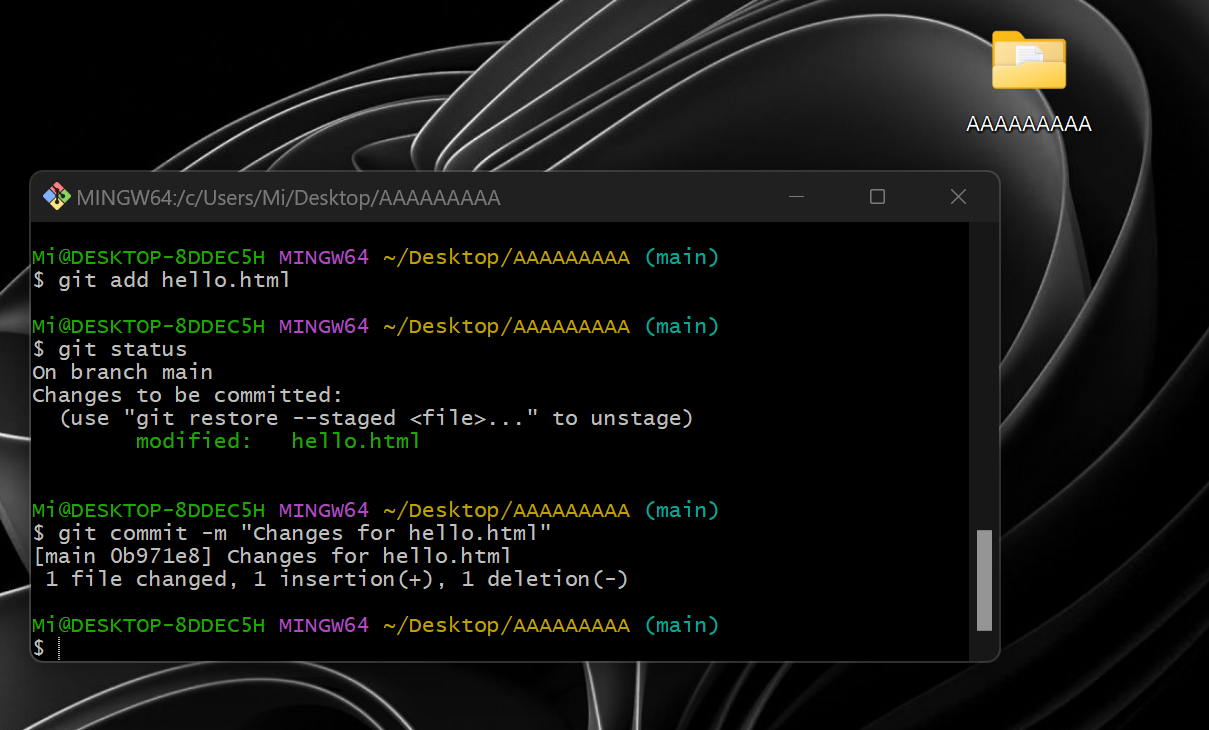


4. Проиндексировать изменение и проверьте состояние.

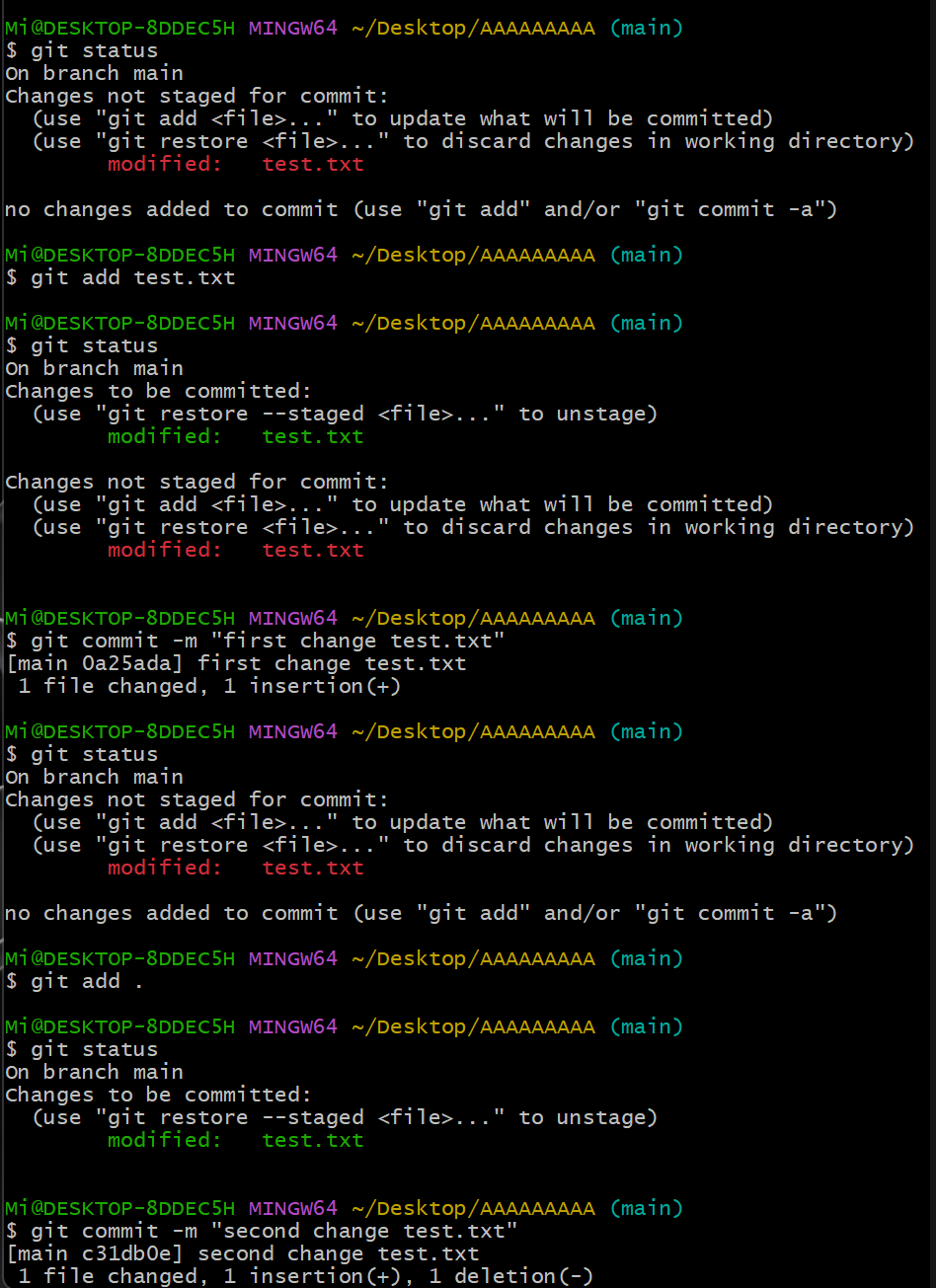


1. Сделать коммит того, что было проиндексировано в репозиторий.

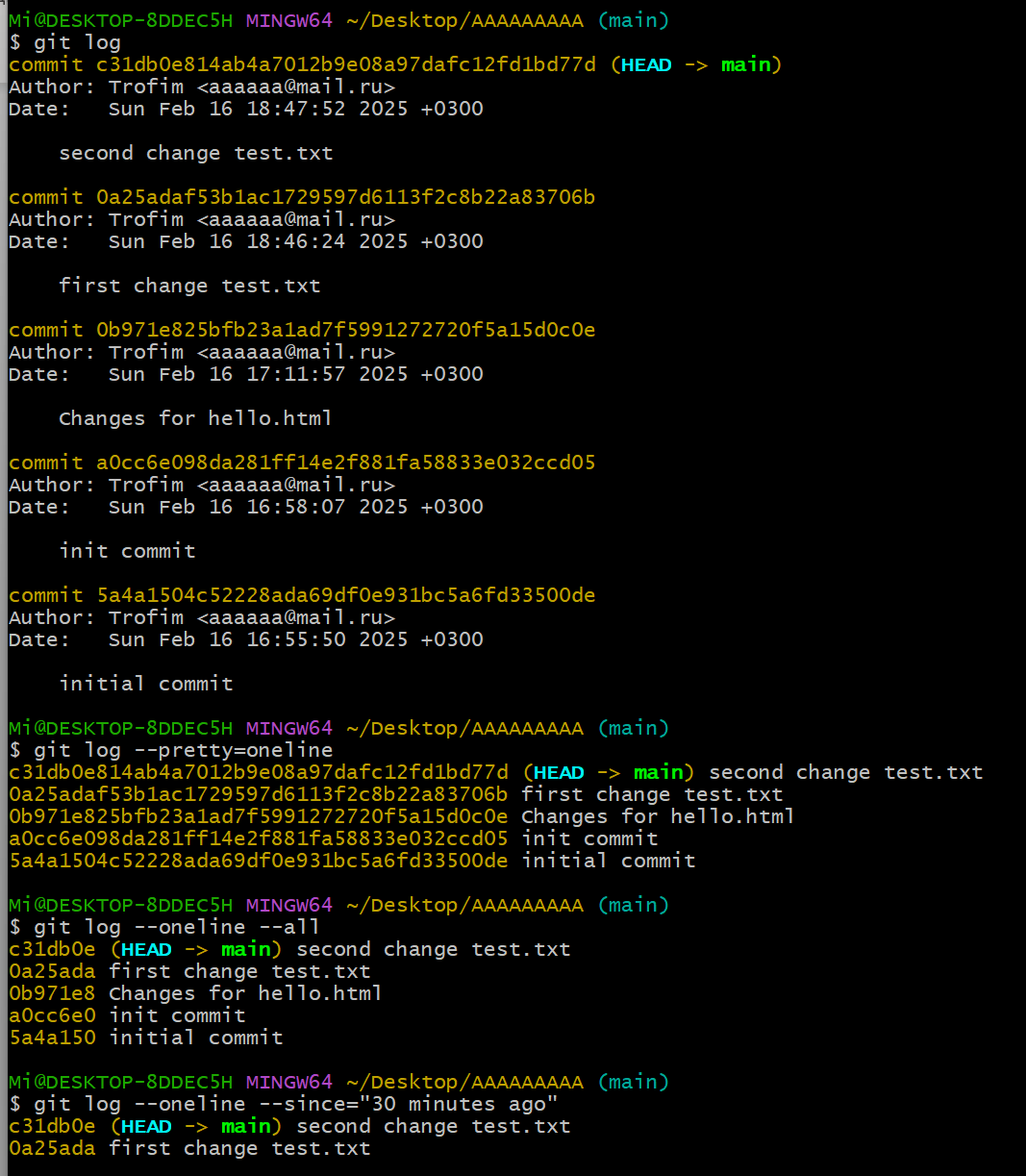
Добавить к коммиту комментарий.



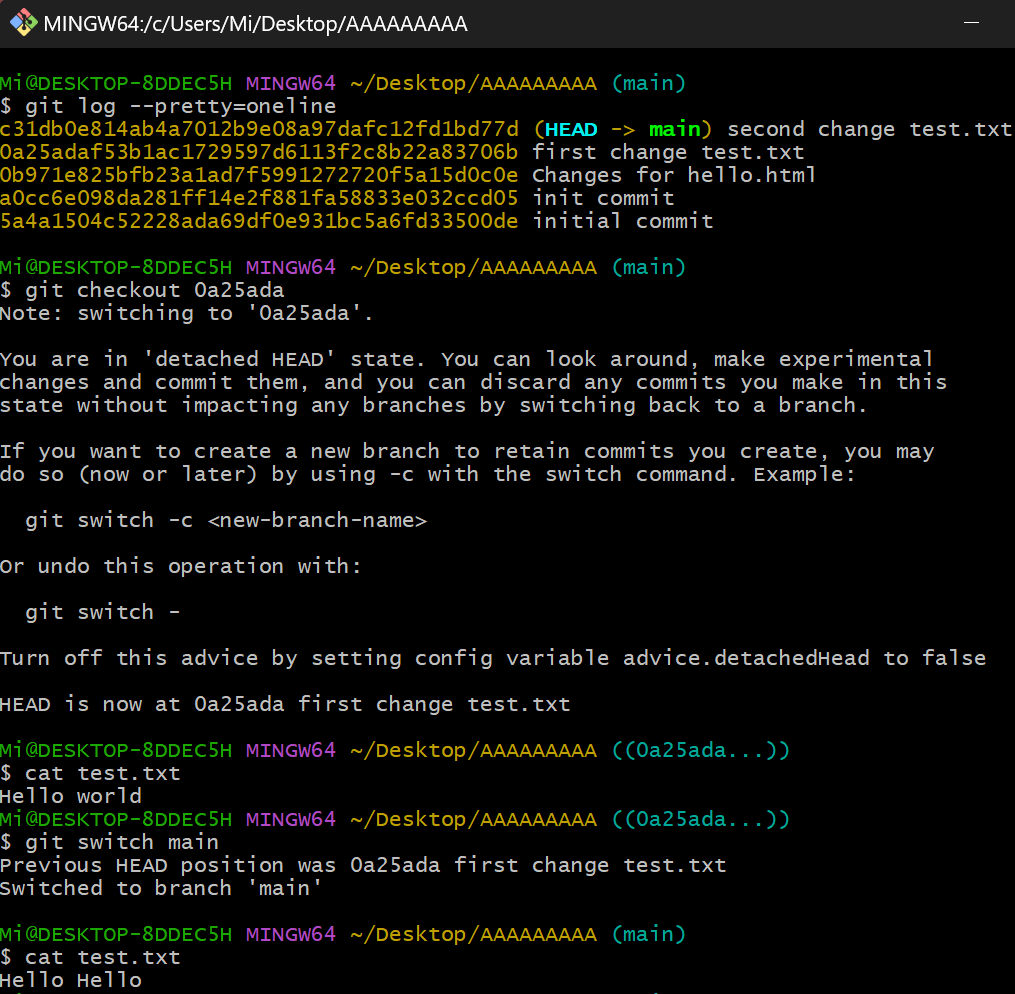
6. Изменить еще один файл. Добавить это изменение в индекс git. Изменить файл еще раз. Проверить состояние и произведите коммит проиндексированного изменения. Теперь добавить второе изменение в индекс, а затем проверить состояние с помощью команды git status. Сделать коммит второго изменения.



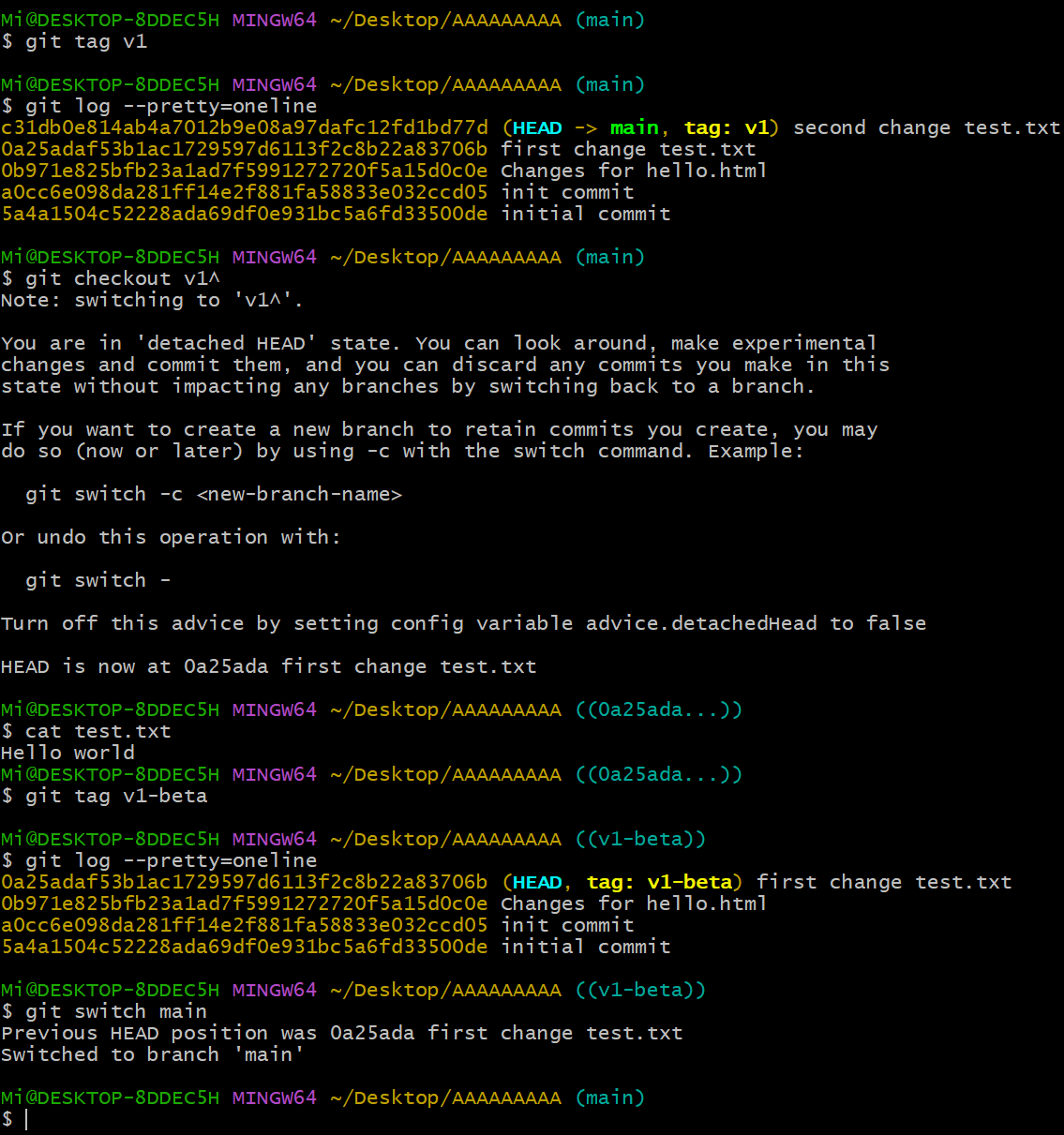
7. Просмотреть историю коммитов с помощью команды git log. Ознакомиться с параметрами команды и использовать некоторые из них для различного формата отображения истории коммитов.



8. Вернуть рабочий каталог к одному из предыдущих состояний.

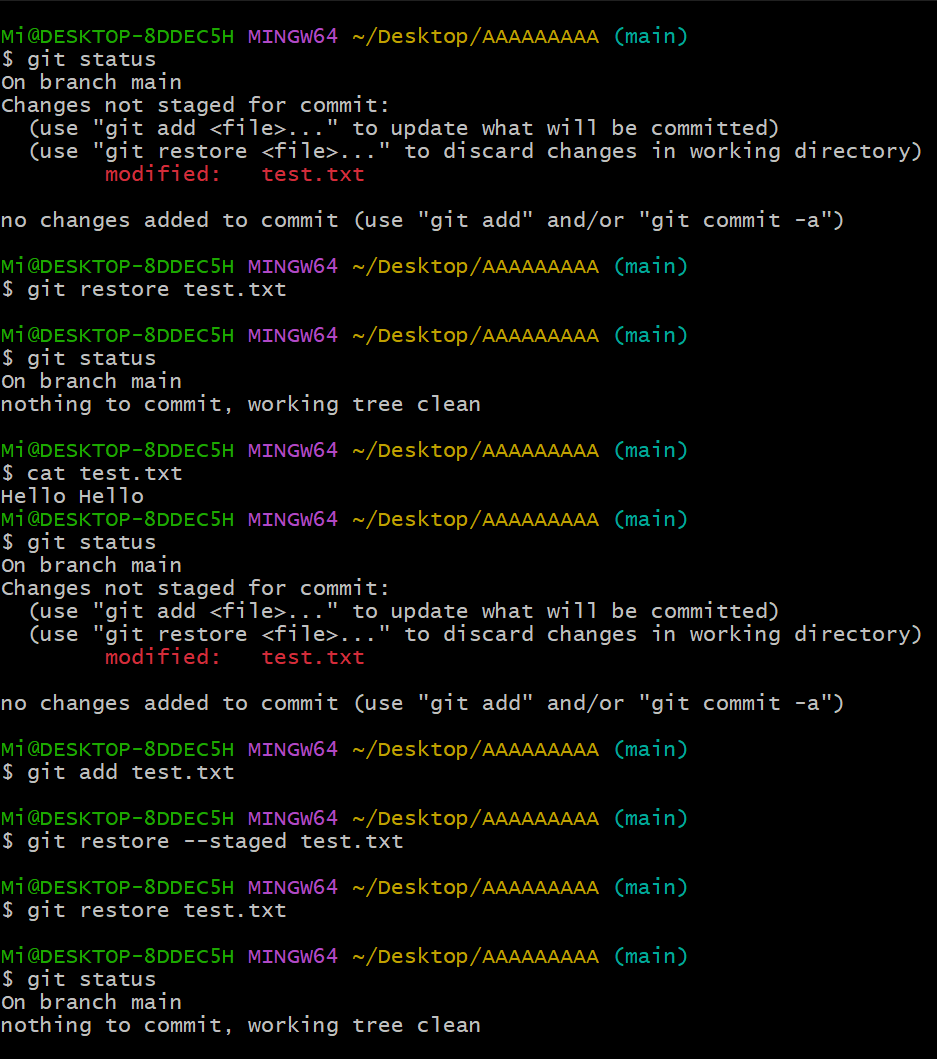


9. Изучить, как создавать теги для коммитов для использования в будущем.

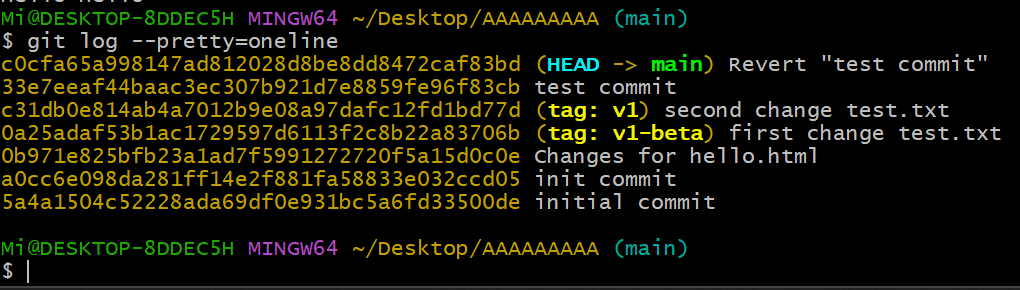


10. Отмените некоторые изменения в рабочем каталоге (до и после

индексирования)



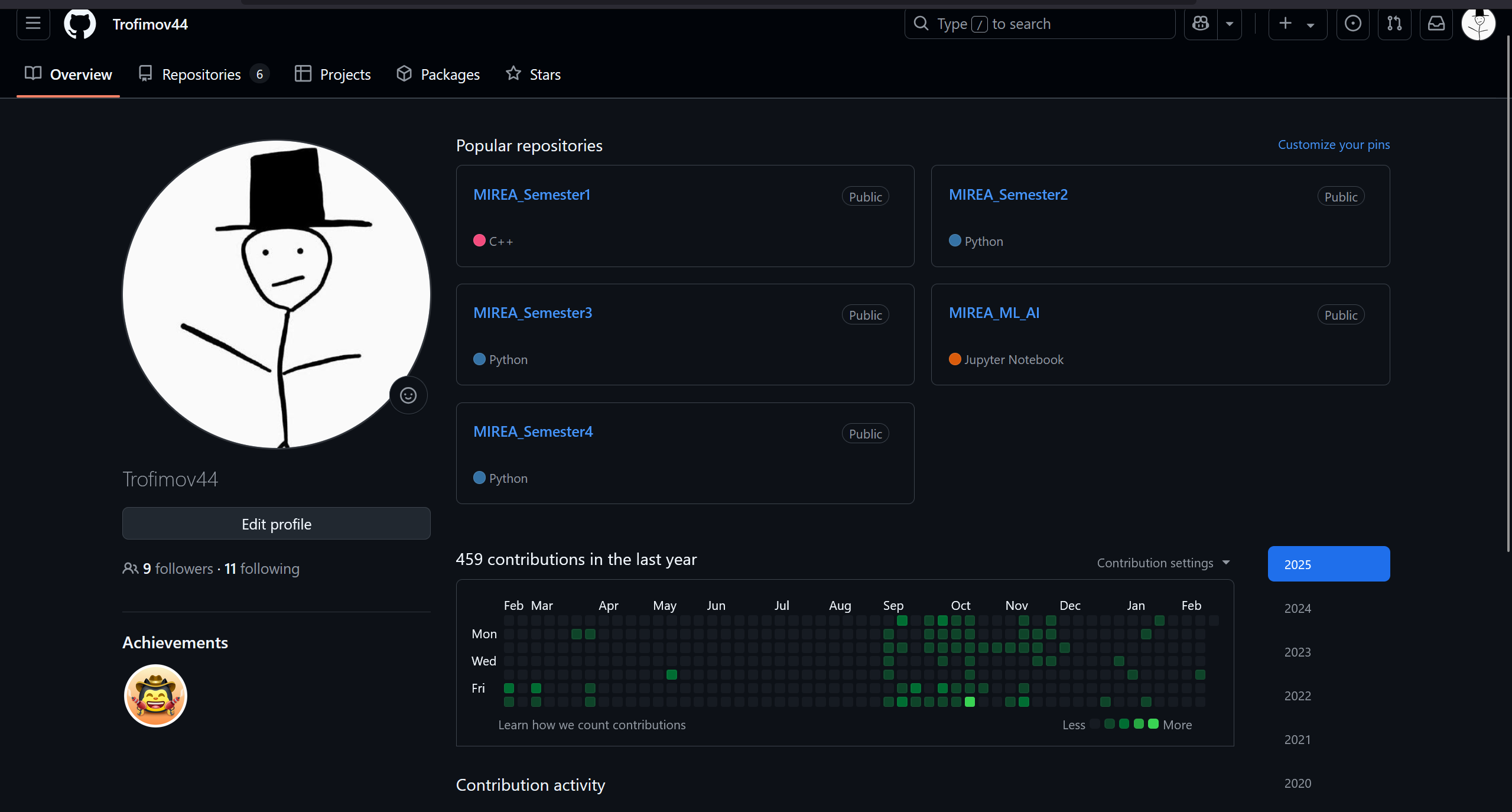
11. Отменить один из коммитов в локальном репозитории.



**ЧАСТЬ 2. СИСТЕМЫ УПРАВЛЕНИЯ РЕПОЗИТОРИЯМИ**

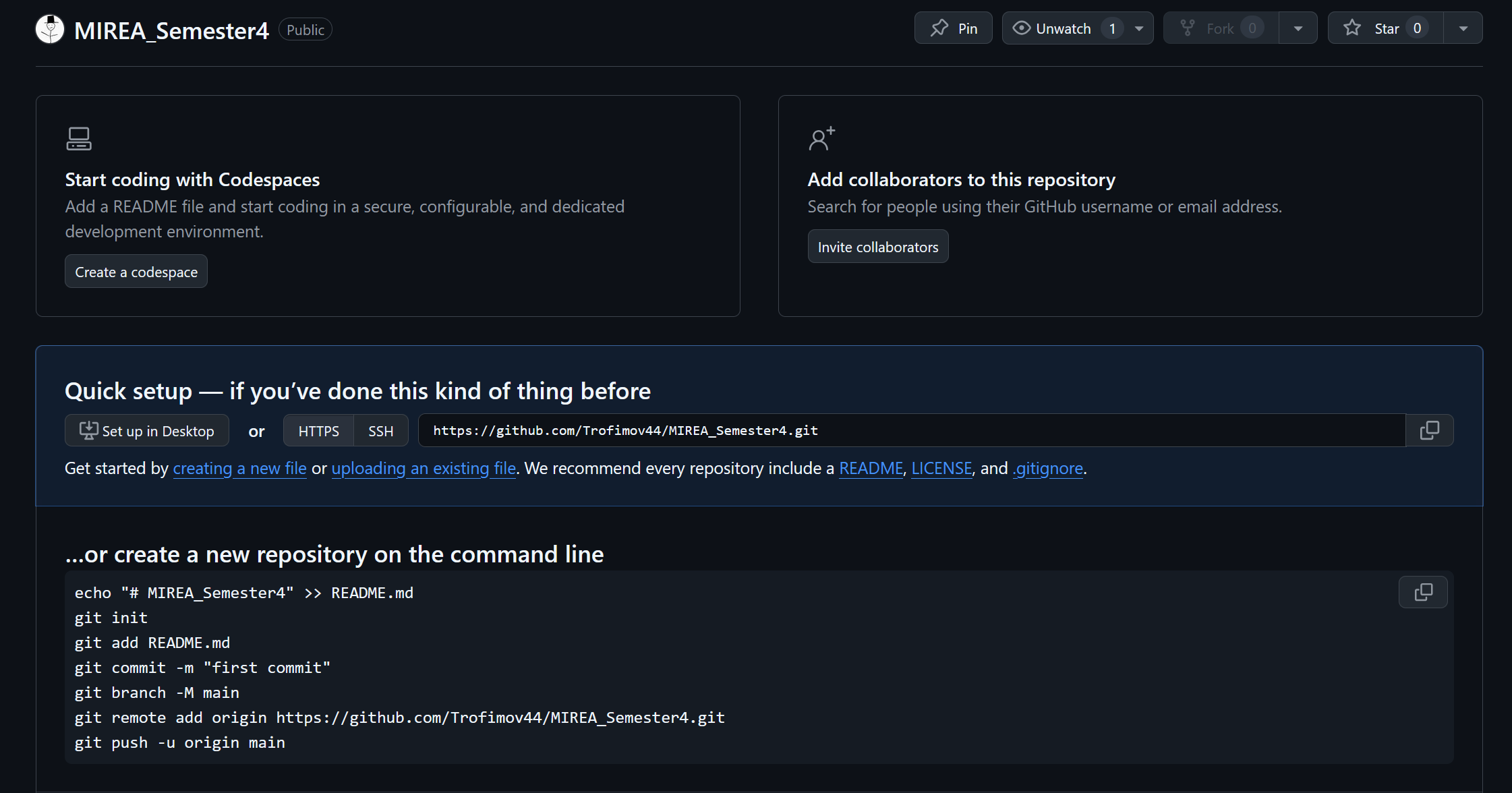
1. Создать аккаунты на GitHub.

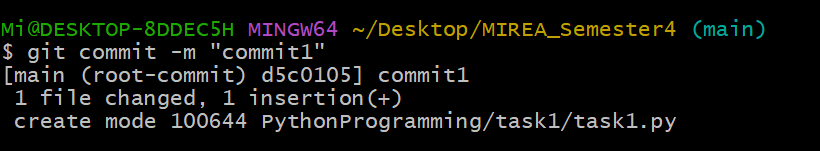
Уже 5 лет создан:

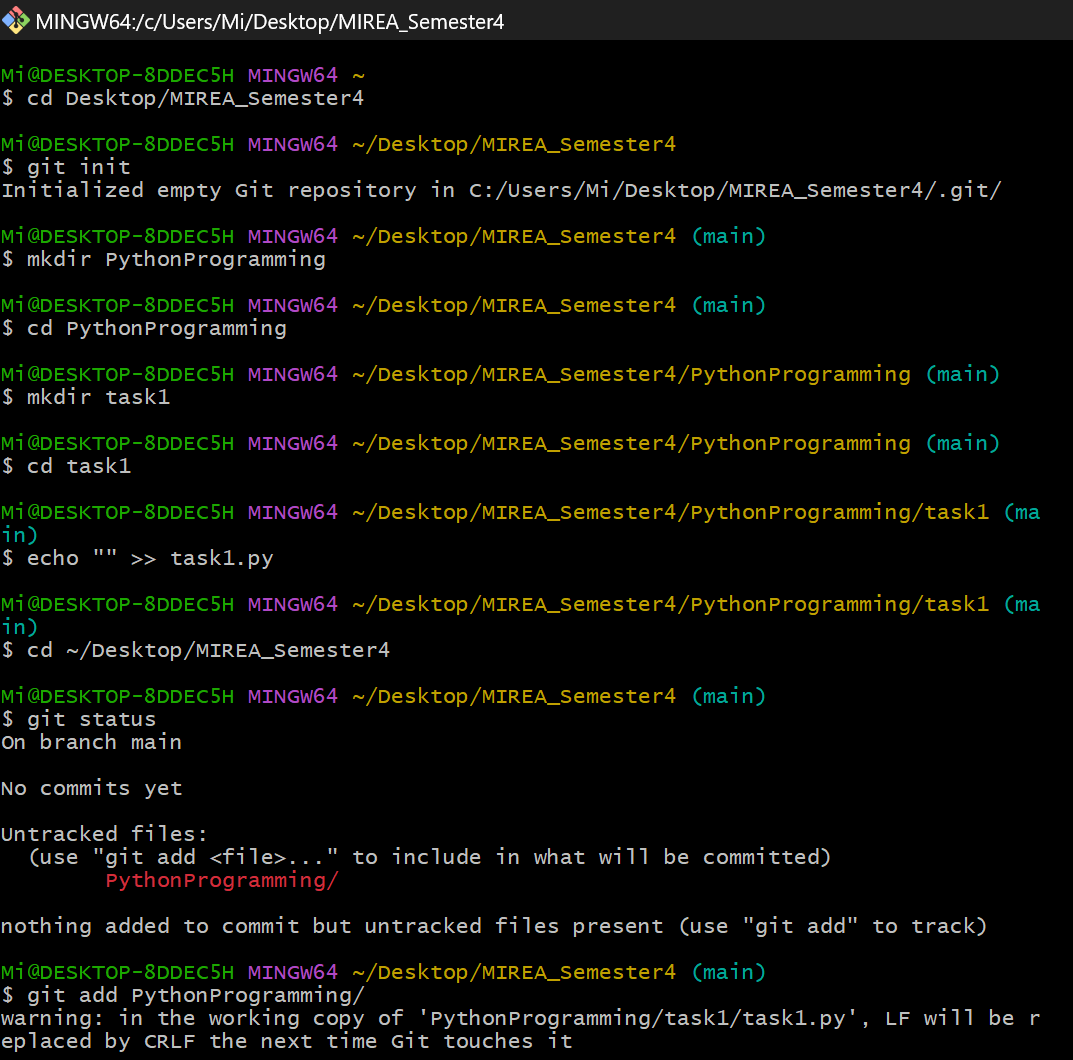


1. Создать репозиторий на GitHub и на локальной машине, согласно выбранной теме проекта

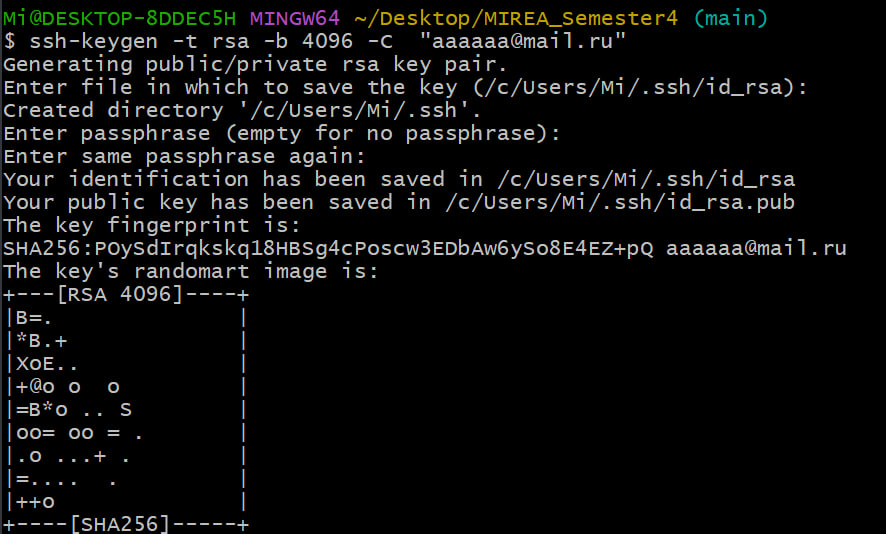
Репозиторий MIREA\_Semester4:

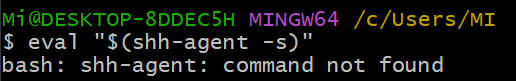


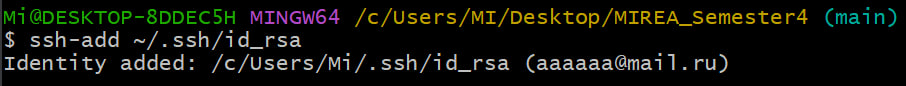
1. Создайте несколько файлов на локальной машине при помощи консоли

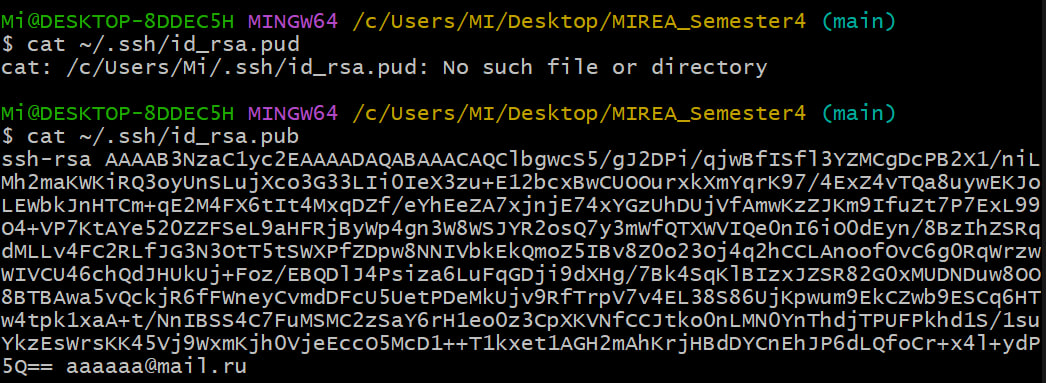


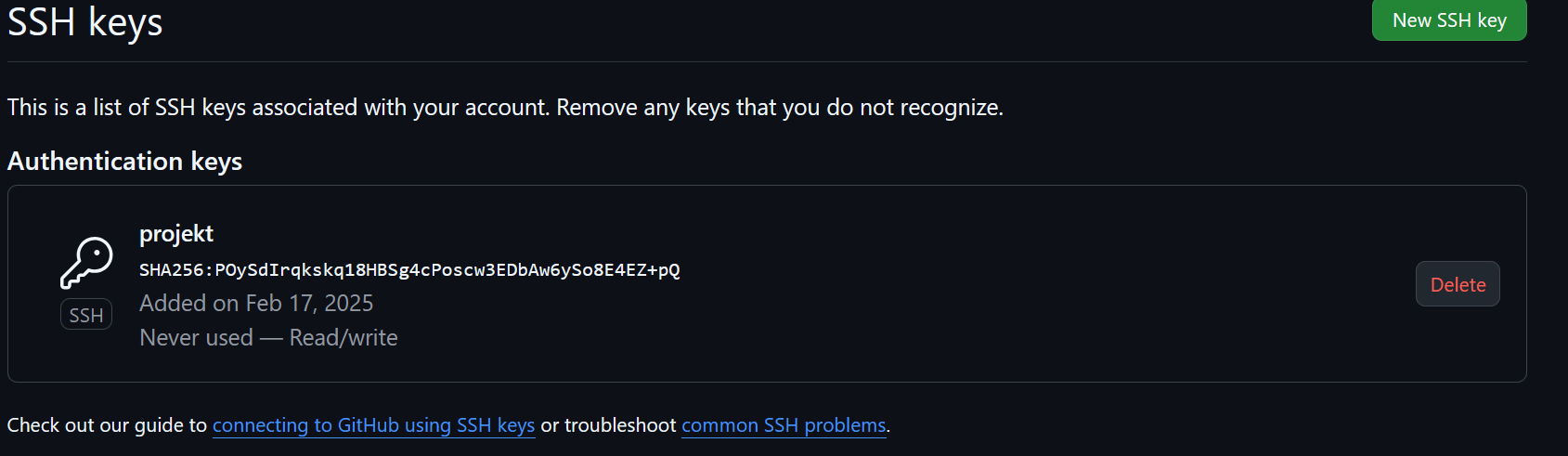
1. Создайте SSH-ключ для авторизации



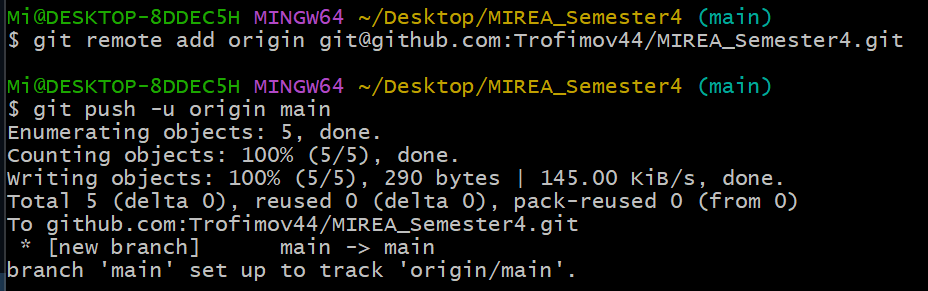




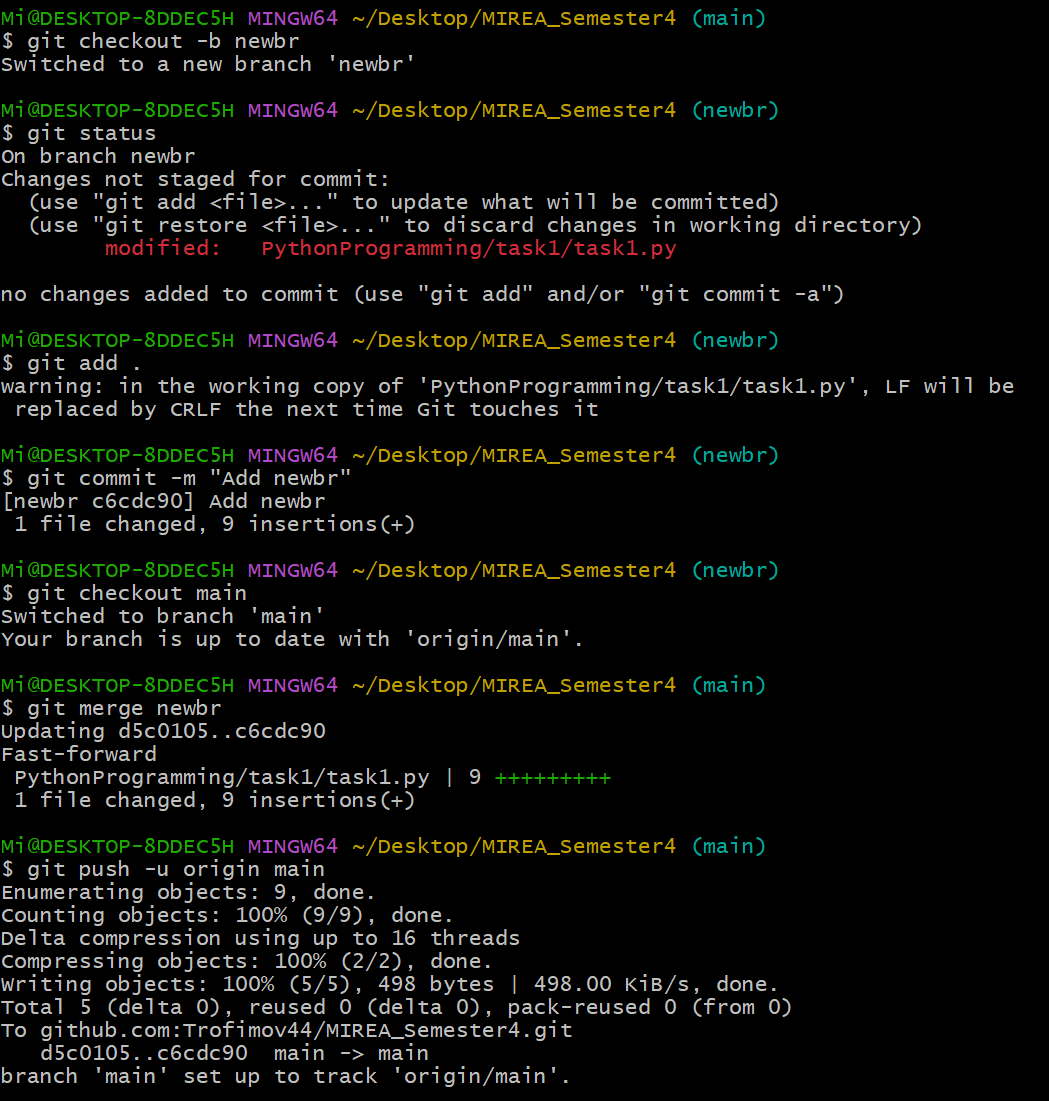




1. Свяжите репозиторий локальной машины с репозиторием на GitHub при помощи консоли

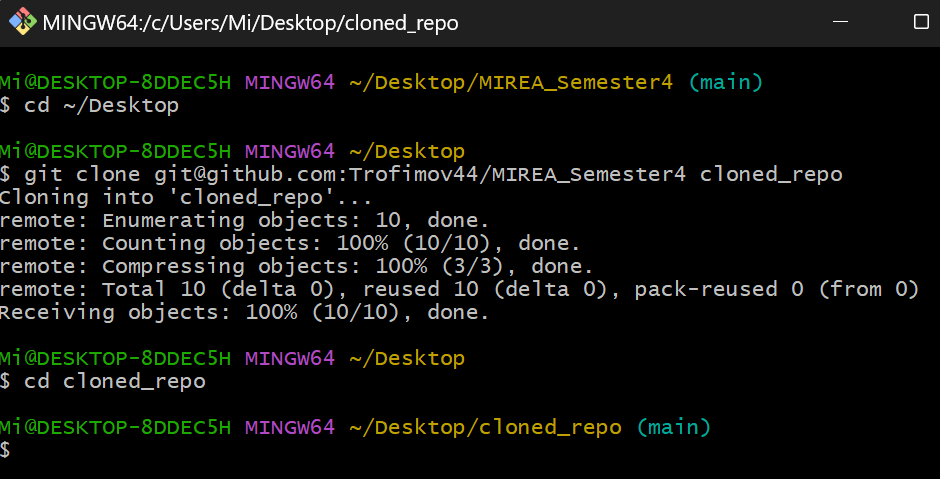


1. Создайте новую ветку в репозитории с помощью команды, произведите в ней какие-нибудь изменения, а после слейте с веткой main

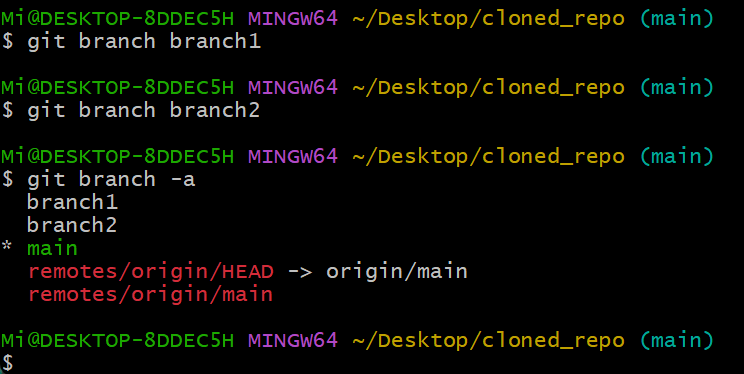


1. Выполните цепочку действий в репозитории, согласно вариантам

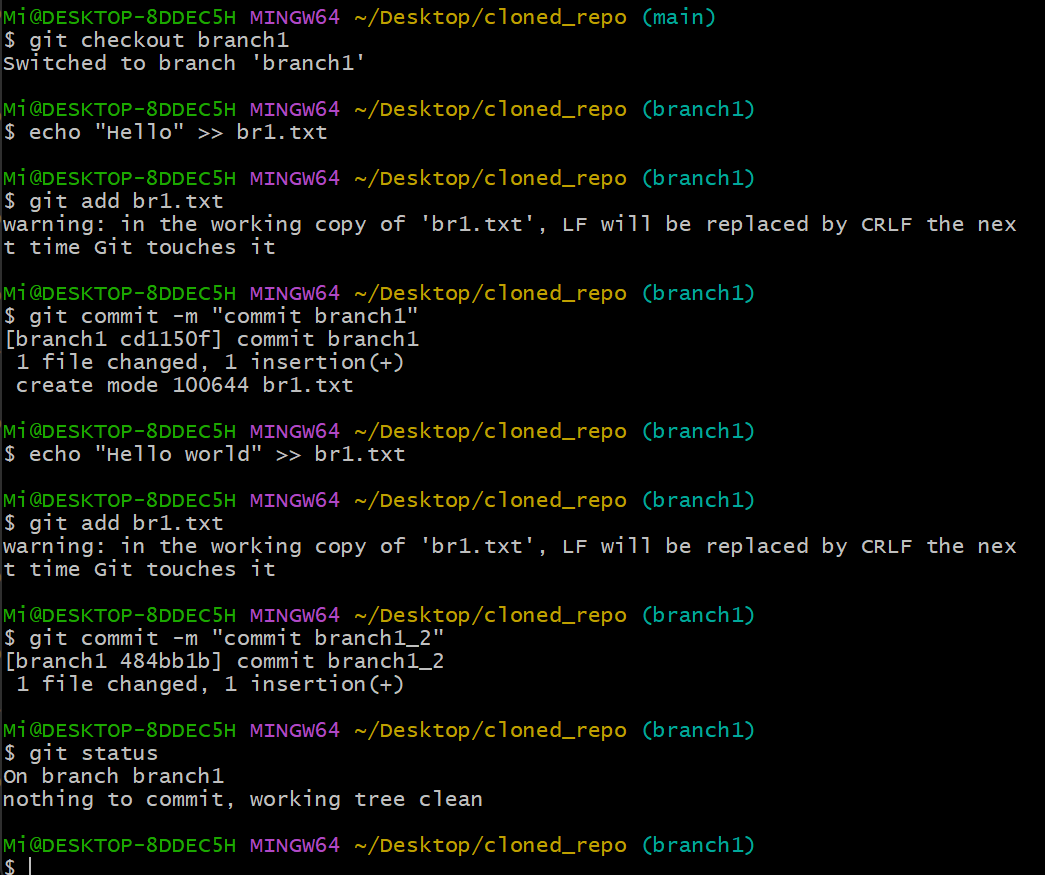
(Вариант 8):  
1) Клонируйте непустой удаленный репозиторий на локальную машину:

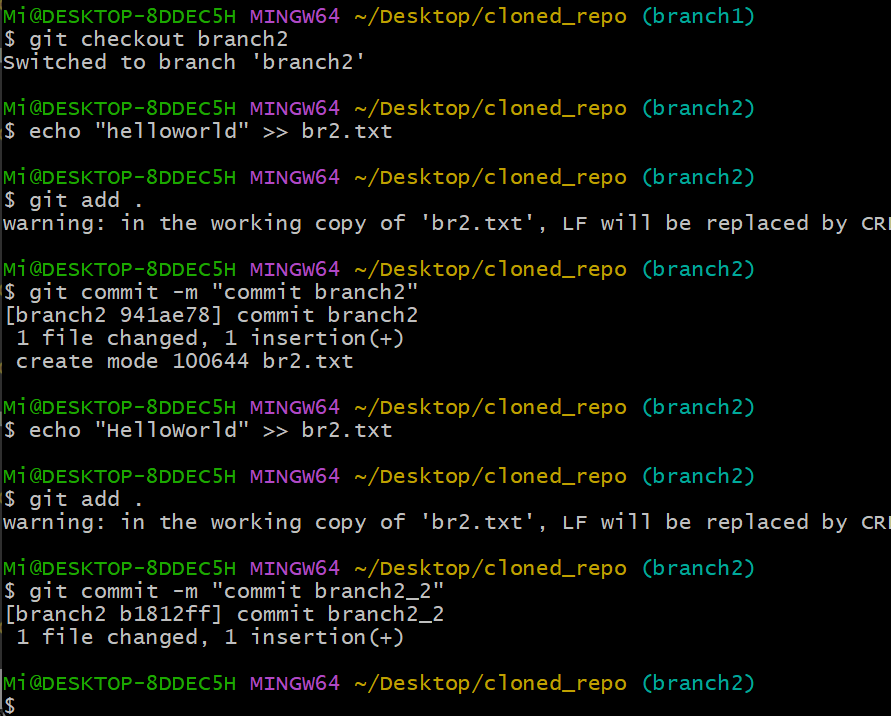


2) Создайте две новых ветки и выведите список всех веток

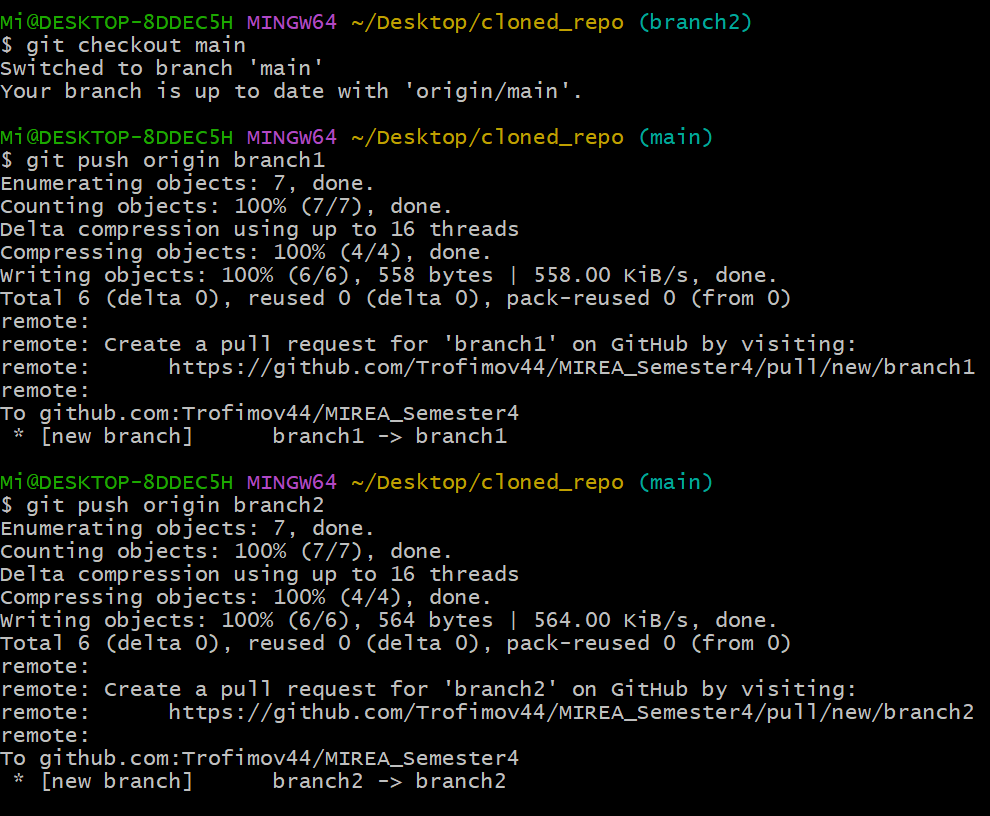


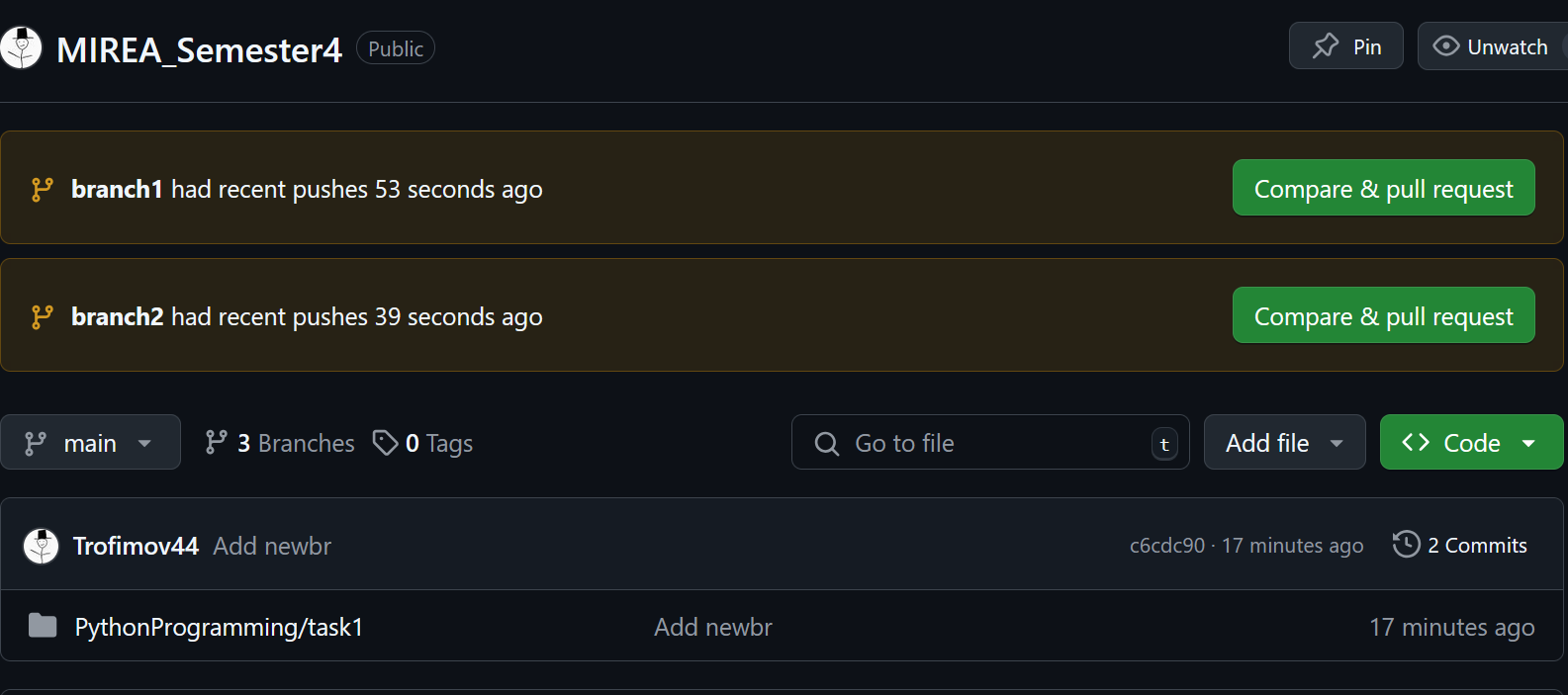
3) Произведите по 2 коммита в обоих ветках в новых файлах



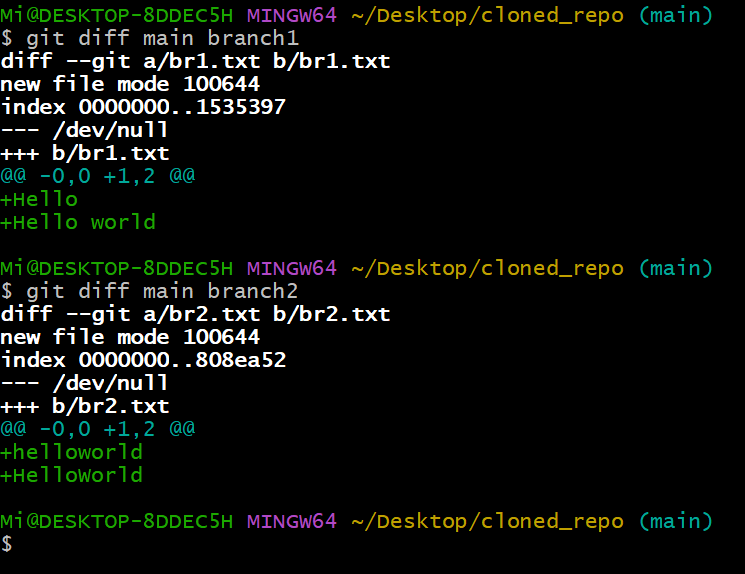


4) Выгрузите изменения в удаленный репозиторий

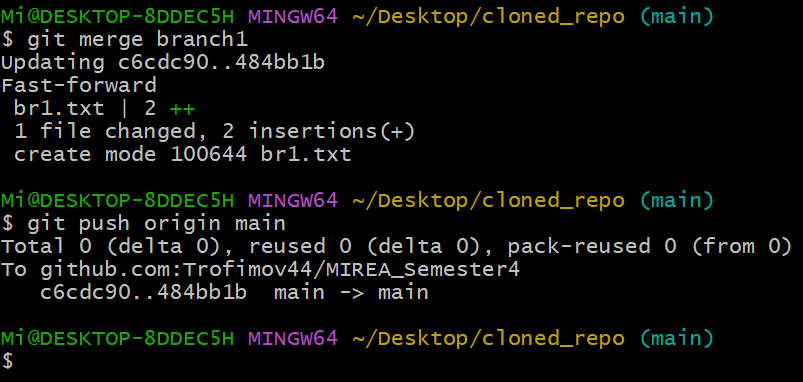


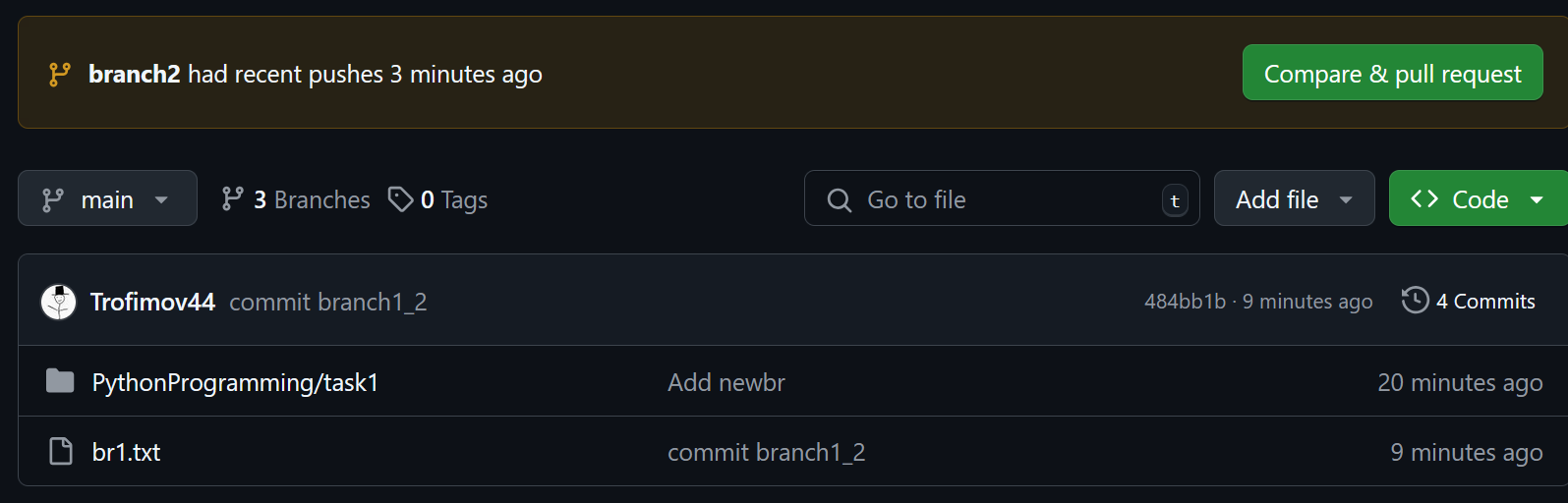


6) Выведите в консоли различия между веткой master и обоими ветками

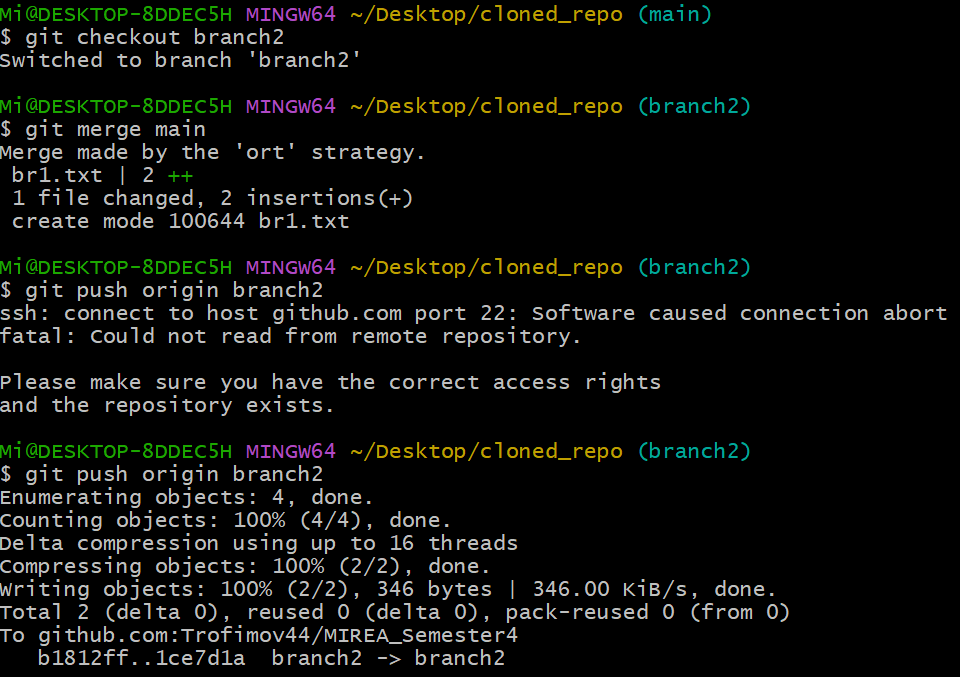


7) Слейте одну ветку с master при помощи merge

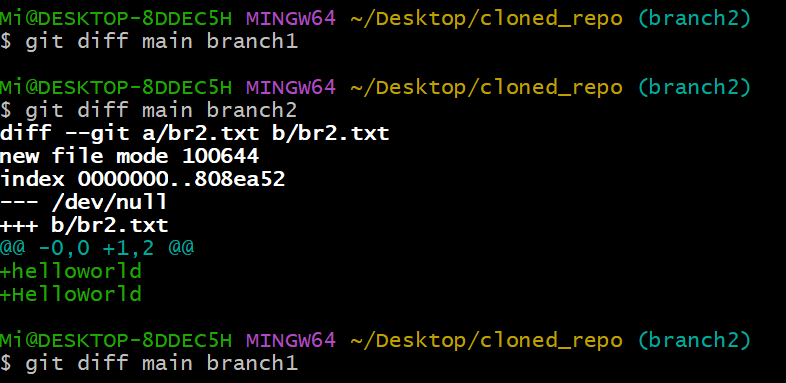




8) Подлейте master ко второй ветке

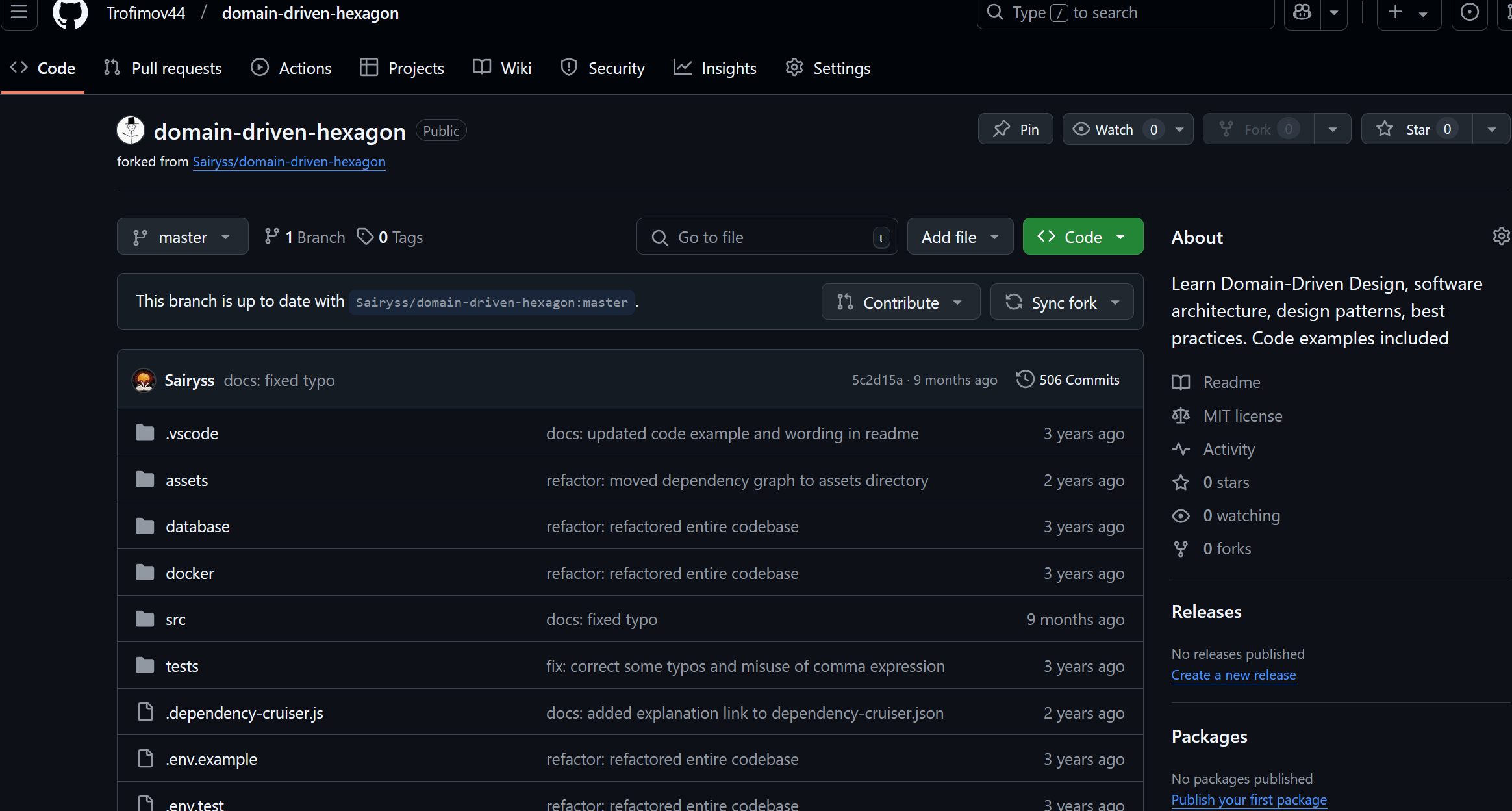


9) Выведите в консоли различия между веткой master и обоими ветками

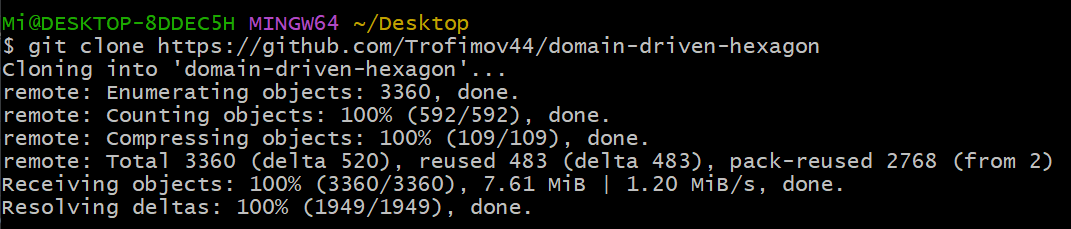


**ЧАСТЬ 3. РАБОТА С ВЕТВЛЕНИЕМ И ОФОРМЛЕНИЕ КОДА**

1. Сделайте форк репозитория в соответствии с вашим вариантом



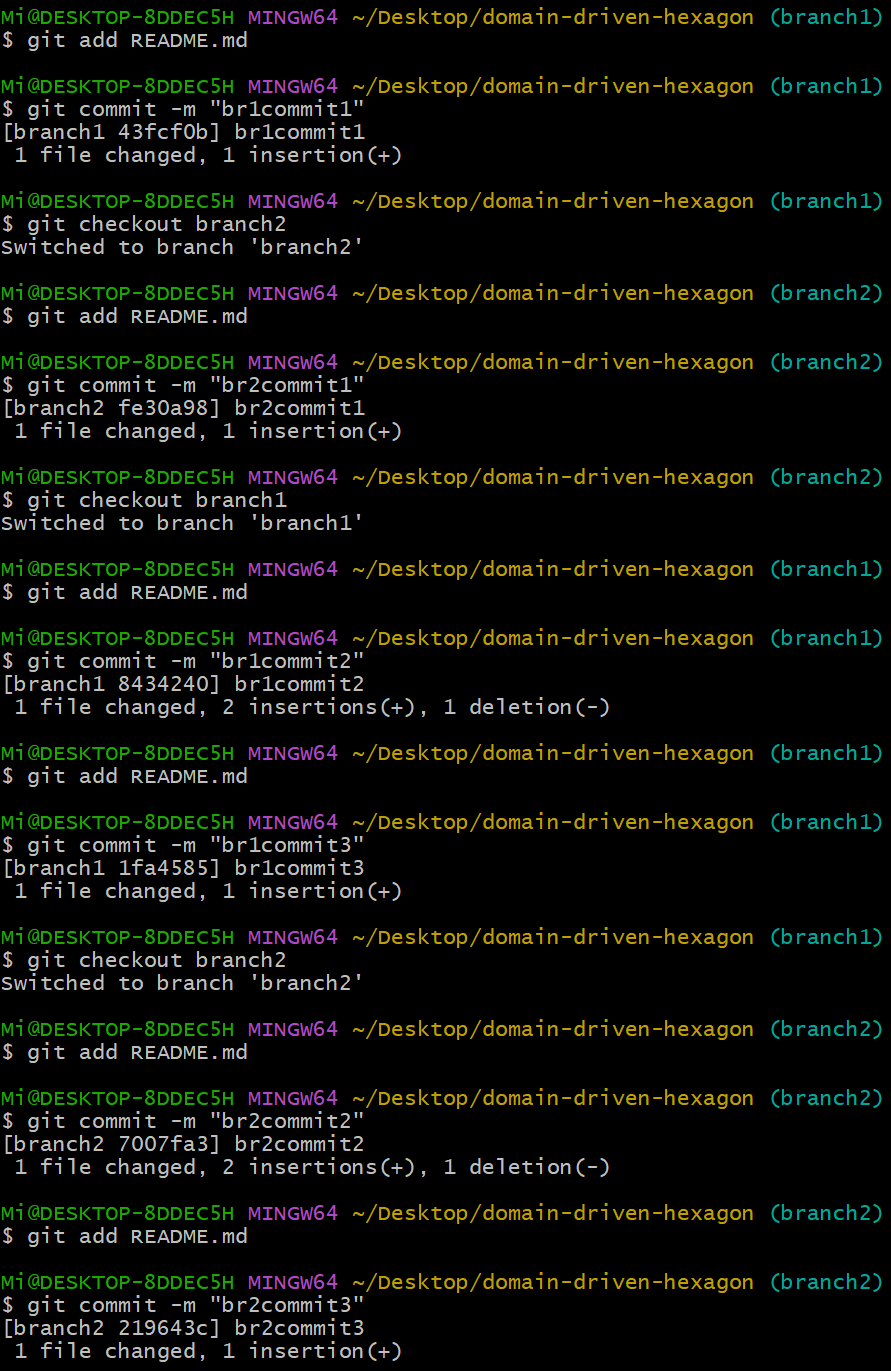
1. Склонируйте его на локальную машину



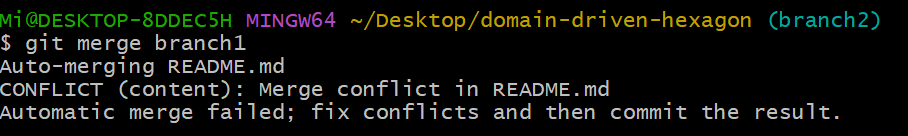
1. Создайте две ветки branch1 и branch2 от последнего коммита в master'е

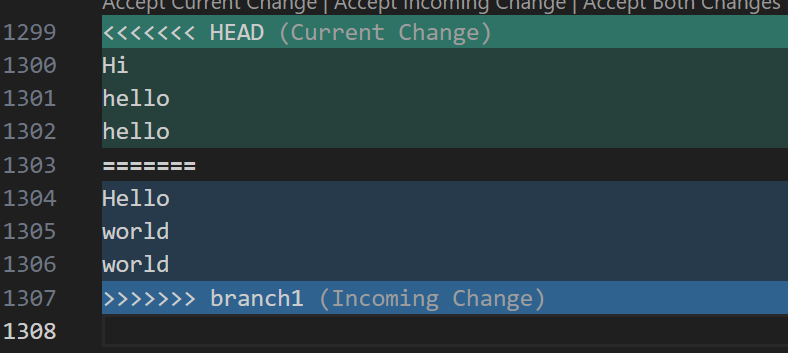


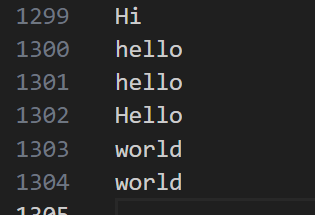
4. Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла

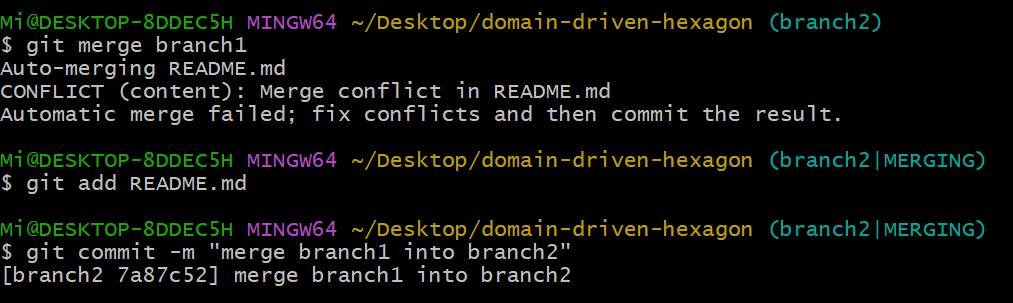


5.Выполните слияние ветки branch1 в ветку branch2, разрешив конфликты при этом

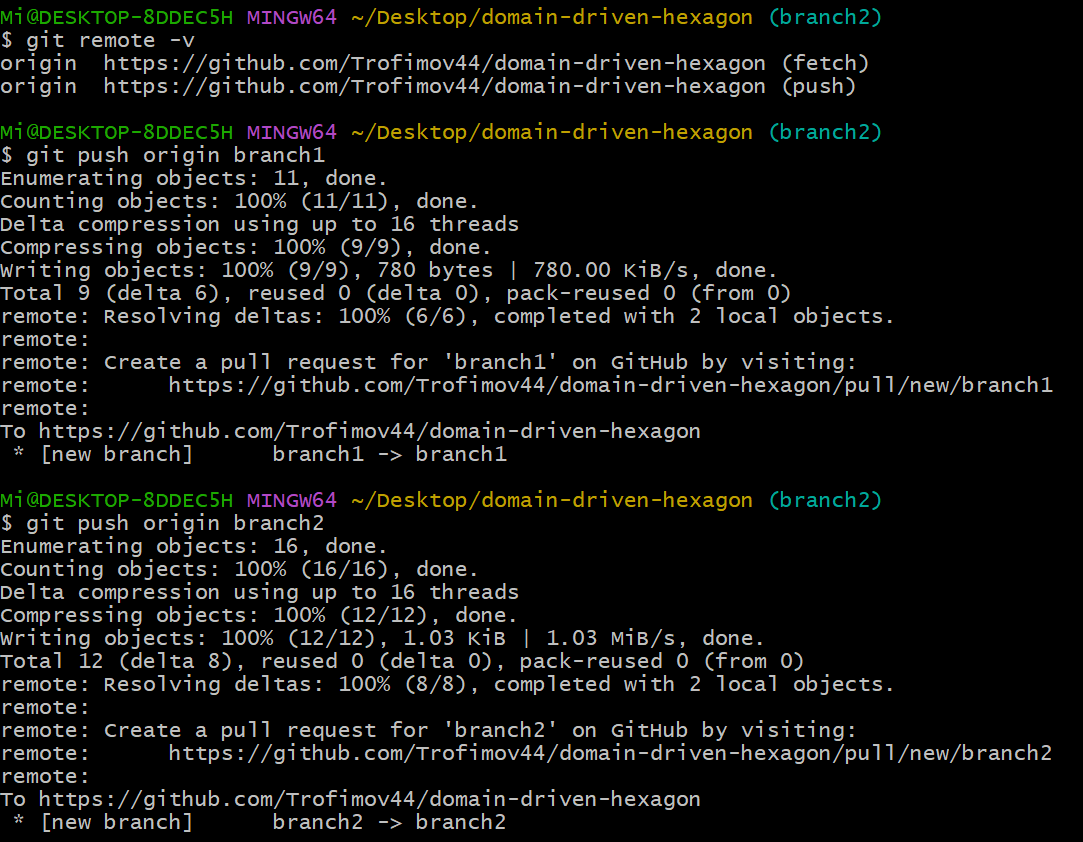




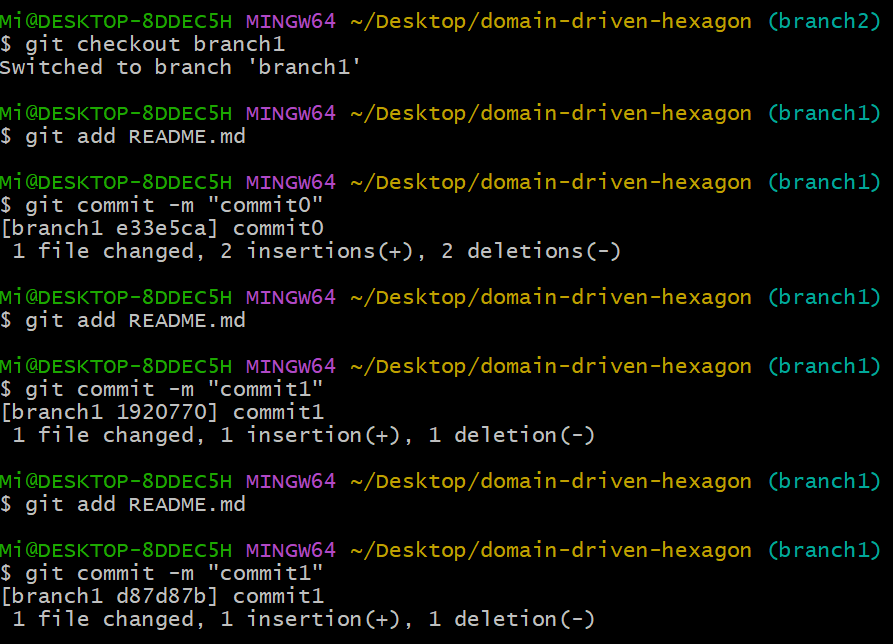




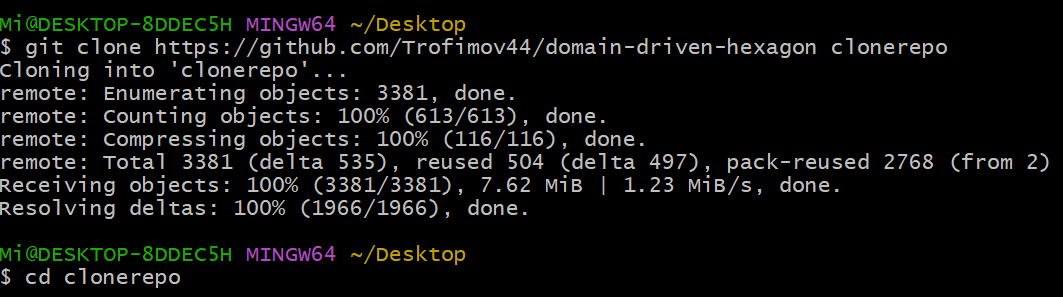
1. Выгрузите все изменения во всех ветках в удаленный репозиторий



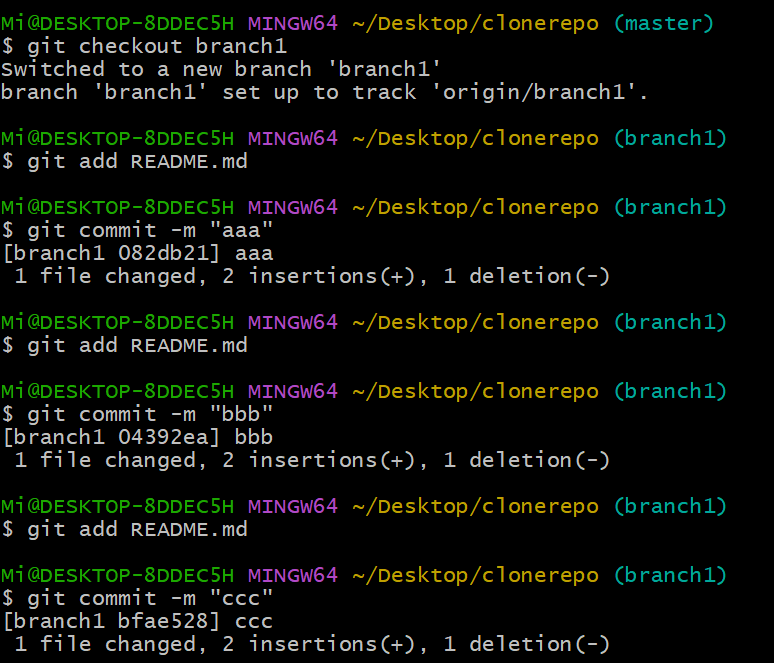
1. Проведите еще 3 коммита в ветку branch1



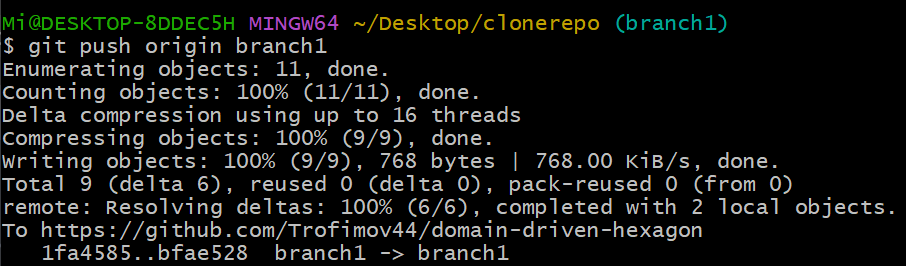
1. Склонируйте репозиторий еще раз в другую директорию

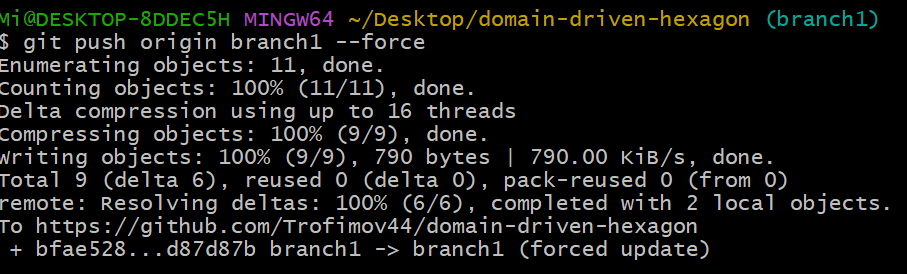


1. В новом клоне репозитории сделайте 3 коммита в ветку branch1

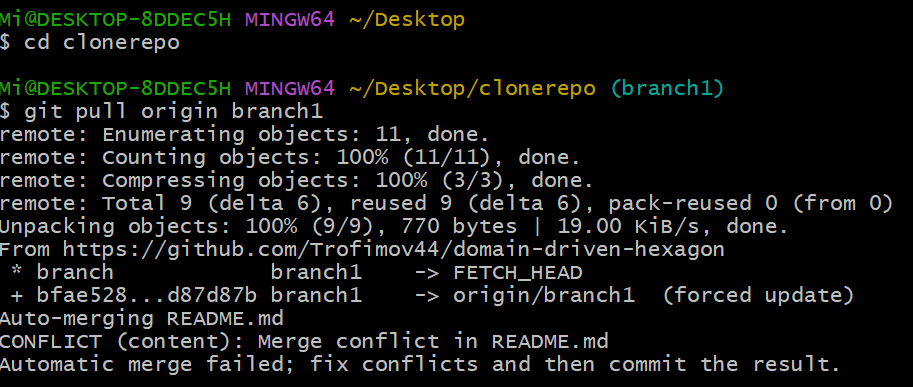


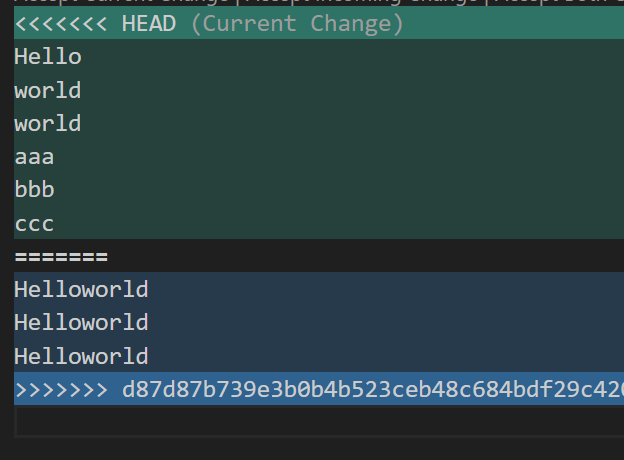
1. Выгрузите все изменения из нового репозитория в удаленный репозиторий



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


12. Получите все изменения в новом репозитории





**КОНТРОЛЬНЫЕ ВОПРОСЫ**

2. К какому типу систем контроля версий относится Git?

Git относится к распределительной системе контроля версий, в таких системах разработчик имеет локальную копию репозитория со всеми комитами

8. Что такое конфликт в Git? Как его решить и почему они бывают?

Конфликт это состояние репозитория, когда изменения в двух ветках затрагивают одни и те же строки в файле, и Git не может автоматически определить, какое изменение должно быть применено. Конфликт можно решить вручную, отредактировав файл и выбрав нужный вариант.

9. Как отменить слияние веток, если произошел конфликт?

Чтобы отменить слияние веток нужно воспользоваться командой

git merge –abort, это вернёт всё в состояния до начала слияния.

11. Что делает команда git status?

Команда git status показывает текущее состояние рабочего каталога и индекса, это помогает понять какие файлы были изменены и какие файлы нужно проиндексировать.

15. Что делает команда git show?

Показывает информацию об объекте(коммите, дереве или теге) для которого мы вызываем эту команду, выводит изменения введённые в последнем коммите и его автора, дату.

19. Чем отличаются команды "git push" и "git pull"?

git push отправляет изменения локального репозитория в удалённый, а git pull наоборот, забирает изменения из удалённого репозитория и объединяет с локальной веткой.

**ВЫВОД**

В ходе выполнения данной практической работы были изучены основные принципы работы с системой контроля версий Git, включая базовые команды, управление локальными и удалёнными репозиториями, работу с ветками и разрешение конфликтов.

Были рассмотрены такие важные аспекты, как создание и настройка локального репозитория, выполнение коммитов, откат изменений, работа с тегами, а также взаимодействие с удалёнными репозиториями на GitHub. Отдельное внимание было уделено управлению ветвлением, слиянию изменений и разрешению возможных конфликтов.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. GIT: Создание прочной основы для эффективной разработки. Ч. 1. : учебное пособие / Д. В. Жматов .— Москва : РТУ МИРЭА , 2024
2. Турнецкая Е. Л., Аграновский А. В.Программная инженерия. Интеграционный подход к разработке» (Турнецкая, Е. Л. Программная инженерия. Интеграционный подход к разработке / Е. Л. Турнецкая, А. В. Аграновский. — Санкт-Петербург : Лань, 2023.