**Ход решения**

**Часть 1. Основные команды Git.**

Задание 1. Установите и настройте клиент git на своей рабочей станции.

После установки git на рабочее устройство нам потребуется его настроить для этого добавим имя и почту для добавления имени используем команду: *git config --global user.name “Shilo Yuriy”*, а для изменения почты *git config --global user.email “shilo.y.s@edu.mirea.ru”*, а для проверки наших изменений введем *git config --list*. Выполнение данных команд предоставлено на рисунке 1.1.

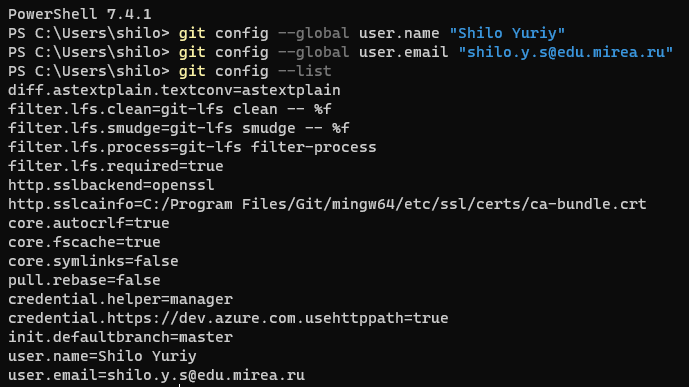


Рисунок 1.1 – Изменение глобальных настроек информации о пользователе

Задание 2. Создайте локальный репозиторий и добавьте в него несколько файлов

Для создание локальной репозитории мы должны ввести команду *git init*, находясь в папке, которую мы хотим сделать нашим репозиторием. После ввода данной команды консоль вывела следующее рисунок 1.2.



Рисунок 1.2 – Создание локального репозитория

Задание 3. Внесите изменения в один из файлов.

В пустой текстовый файл была написана фраза “Какой-то текст, который был добавлен. После создания репозитория.” рисунок 1.3.

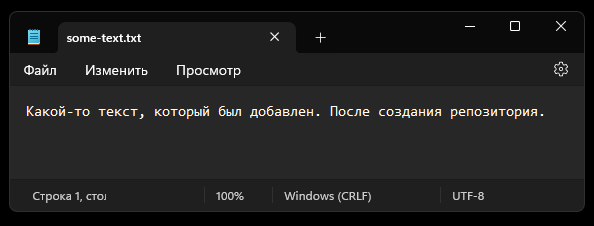


Рисунок 1.3 – Содержание текстового файла

Задание 4. Проиндексируйте изменения и проверьте состояние.

Для того что бы добавить изменный индекс в Git мы должны воспользоваться командой *git add .*, а чтобы проверить состояние репозитории мы должны использовать команду *git status*. После выполнения данных команд консоль выведет следующие рисунок 1.4.

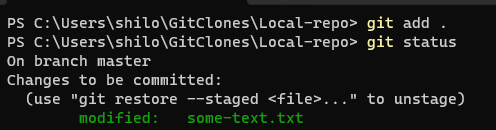


Рисунок 1.4 – Добавление индекса в Git и получение статуса локального репозитория

Задание 5. Сделайте коммит того, что было проиндексировано в репозиторий. Добавьте к коммиту комментарий.

Для того чтобы за коммитить изменения мы воспользуемся командой *git commit -m “…”* (за место трех точек мы пишем наш комментарий) выполнение данной команды показано на рисунке 1.5.

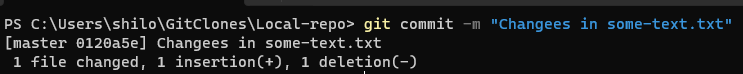


Рисунок 1.5 – Выполнение команды для коммита с комментарием

Задание 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.

Для выполнения данного задания поочередно введем команды, которые мы разбирали в предыдущих заданиях. Все введённые команды и их результат предоставлены на рисунке 1.6.

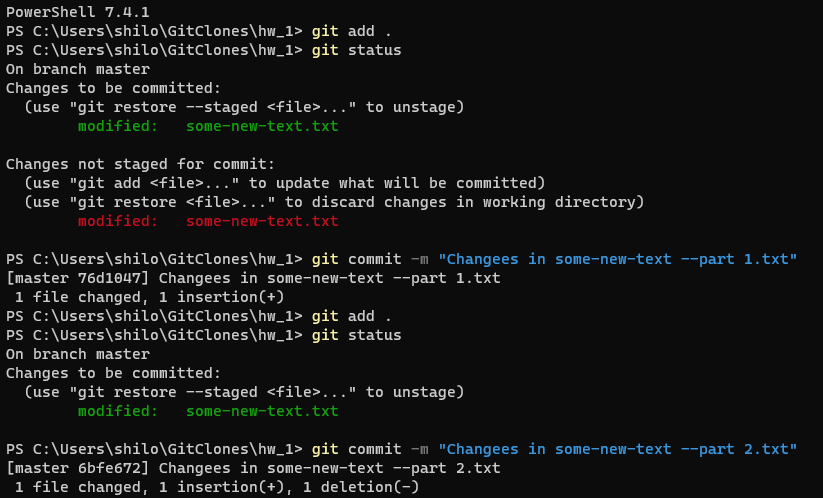


Рисунок 1.6 – Все операции произведенные в задание 6

Задание 7. Просмотрите историю коммитов с помощью команды *git log*. Ознакомьтесь с параметрами команды и используйте некоторые из них для различного формата отображения истории коммитов.

На рисунке 1.7 предоставлены три различных формата вывода коммитов: *git log, git log --inline, git log --graph*. Первый это вывод без форматирования, второй выводит каждый коммит в виде одной строчки и третий выводит список в виде графов.

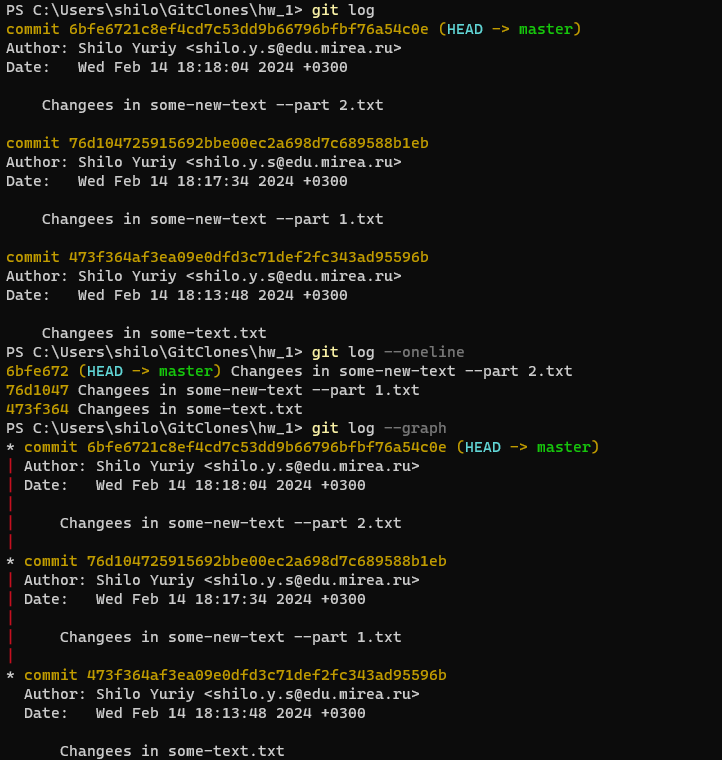


Рисунок 1.7 – Три варианта форматирования вывода коммитов

Задание 8. Верните рабочий каталог к одному из предыдущих состояний.

Для возврата к предыдущему коммиту мы воспользуемся командой *git checkout HEAD~*, также за место *HEAD~* мы можем написать хэш код коммита, имя ветки или имя ветки. Показ работы команды по возврату на один коммит назад предоставлена на рисунке 1.8.

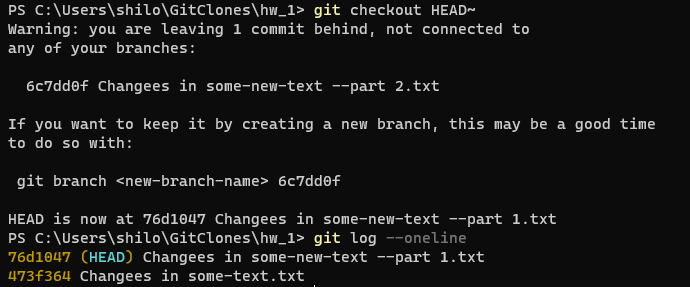


Рисунок 1.8 – Действия по возврату к предыдущему коммиту

Задание 9. Изучите, как создавать теги для коммитов для использования в будущем.

Для создания простого тега используется команда *git tag …*, за место троеточия нужно написать имя нашего тега. А для просмотра списка существующих тегов используется команда *git tag*. После добавления тега он будет отображаться в истории коммитов для удобного возврата. Отработка данных команд предоставлена на рисунке 1.9.

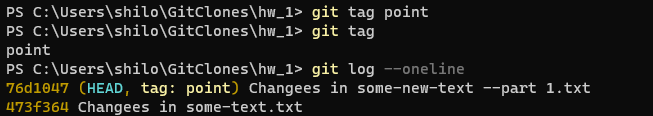


Рисунок 9 – Создание тега point

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

Для отката изменения в рабочей папки можно воспользоваться командой *git checkout point*, после выполнения данной команды я возвращусь на момент с тегом point, а для возврата к другой точке я воспользовался командой *git checkout tag\_02*. Выполнение данных команд предоставлено на рисунке 1.10.

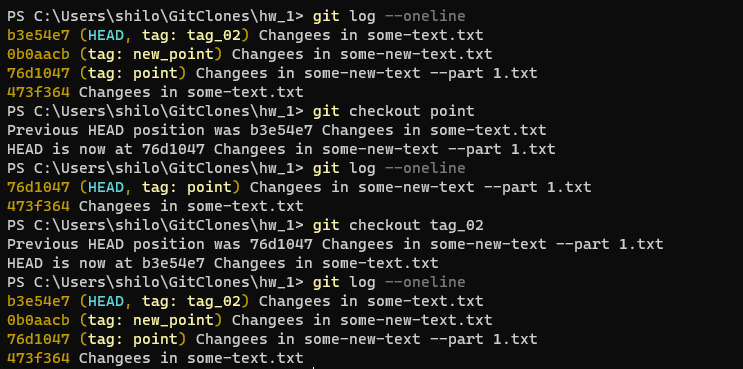


Рисунок 1.10 – Отмена изменений в рабочем каталоге

Задание 11. Отмените один из коммитов в локальном репозитории.

Для выполнения данного задания существует два способа отмены коммита первый способ использовать *git revert …* или *git reset …* где за место троеточия мы пишем идентификатор коммита. Первая команда создает новый коммит, который отменяет изменения, но при этом сохраняет историю, второй же удаляет историю. Действие одной из команд предоставлено на рисунке 1.11.

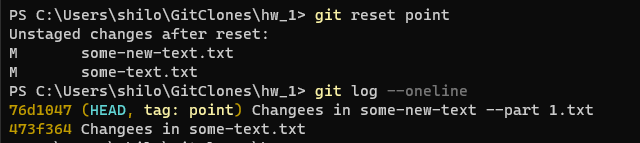


Рисунок 1.11 – Отмена коммита с удаление истории

**Часть 2. Системы управления репозиториями**

Задание 1. Создайте аккаунт на GitHub (у кого нет)

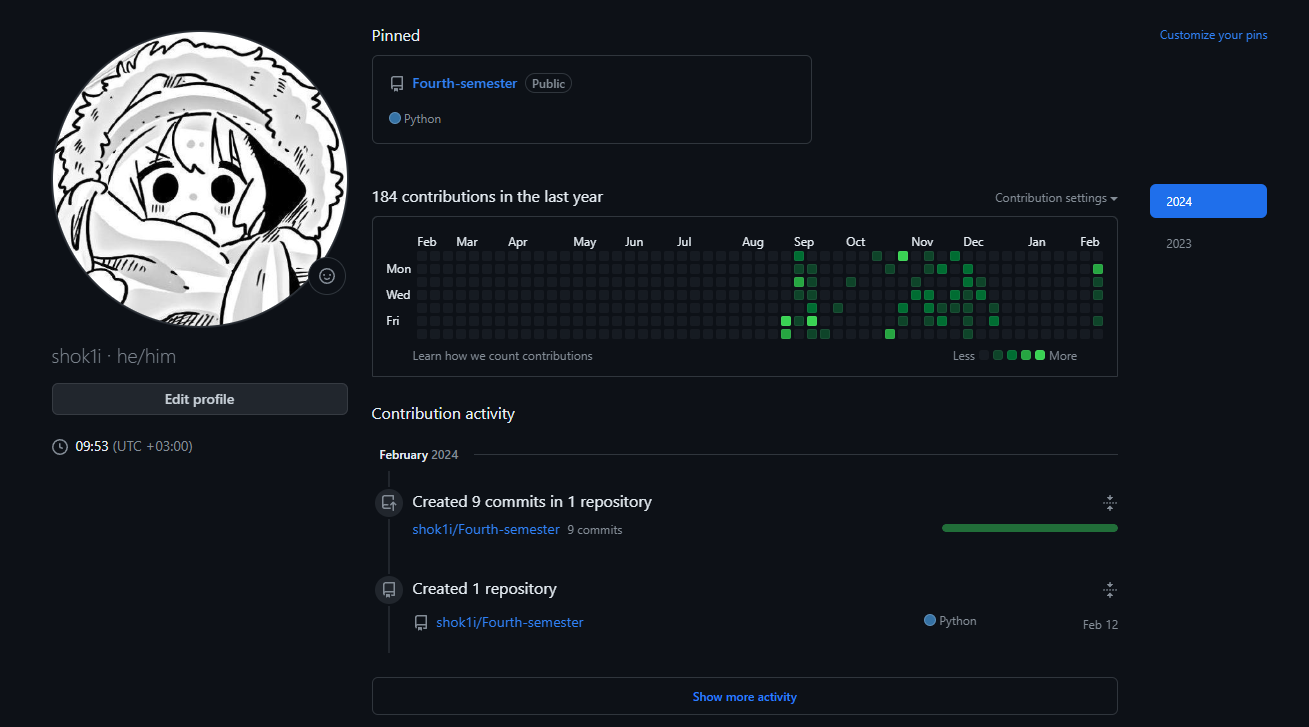


Рисунок 2.1 – Скриншот профиля на GitHub

Задание 2. Создайте репозиторий на GitHub и на локальной машине, согласно выбранной теме проекта

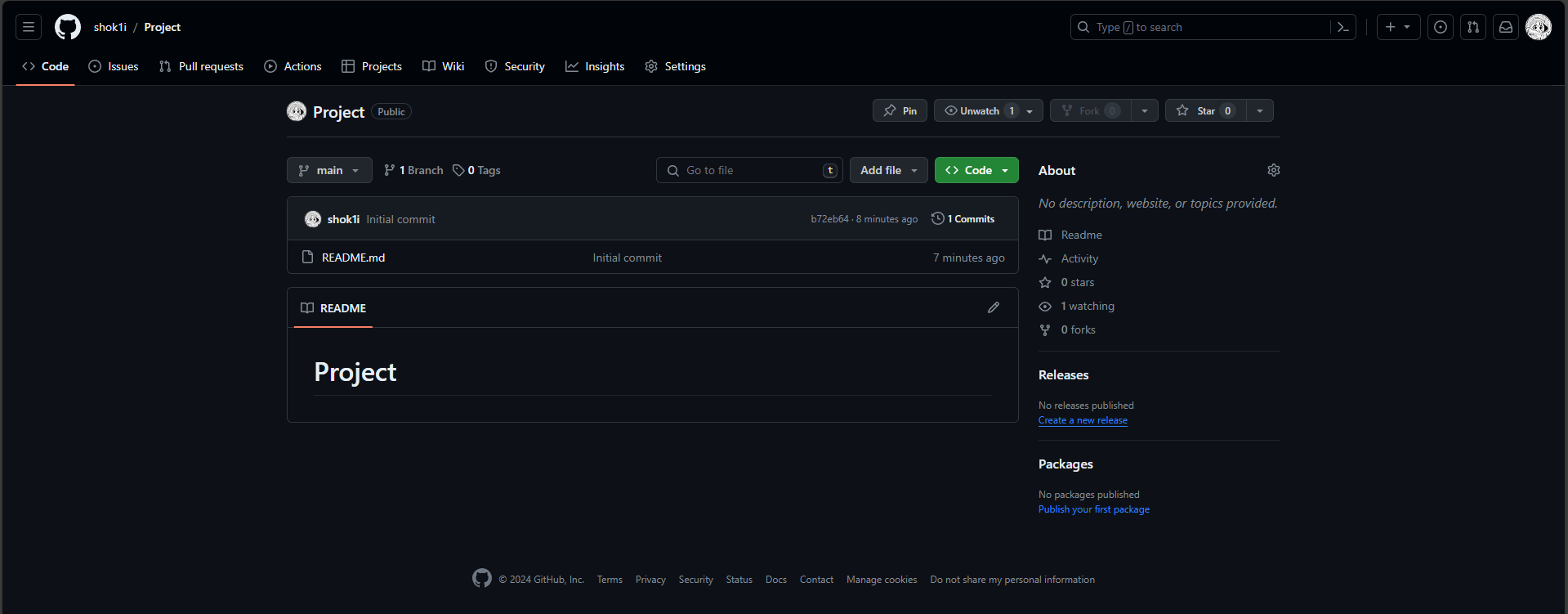


Рисунок 2.2 – Создан репозиторий на гитхабе

Задание 3. Создайте несколько файлов на локальной машине при помощи консоли

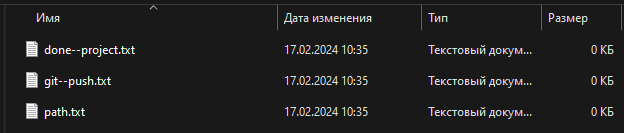


Рисунок 2.3 – Созданные фпйлы в папке проекта

Задание 4. Создайте SSH-ключ для авторизации

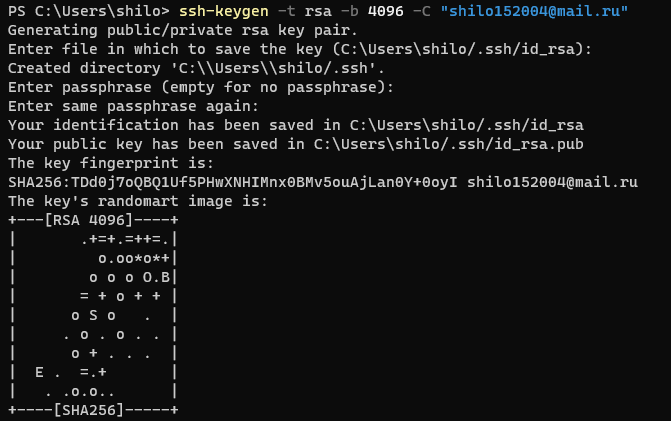


Рисунок 2.4 – Создание SSH ключа

Задание 5. Свяжите репозиторий локальной машины с репозиторием на GitHub при помощи консоли



Рисунок 2.5 – Связывание репозиториев

Задание 6.

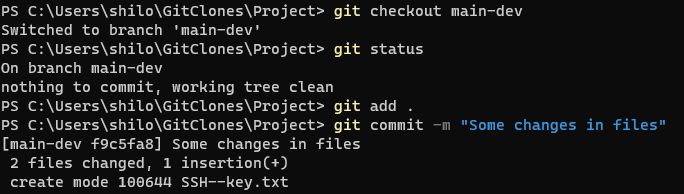


Рисунок 2.6 – Создание новой ветки и коммит в нее изменений

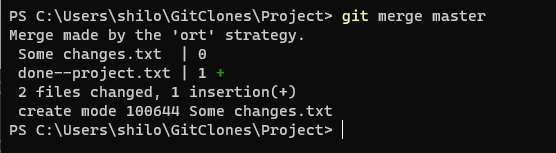
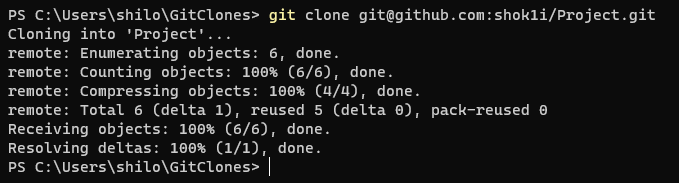


Рисунок 2.7 – Слияние ветки с веткой master

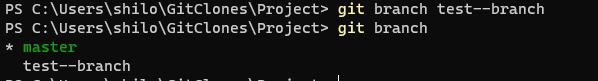
Задание 7.

**ВАРИАНТ 5**

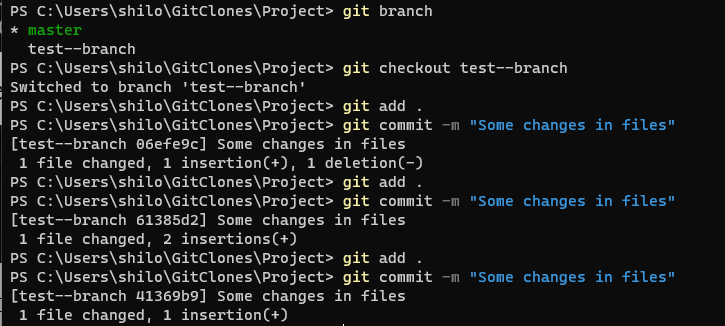
Задание 1. Клонируйте непустой удаленный репозиторий на локальную машину



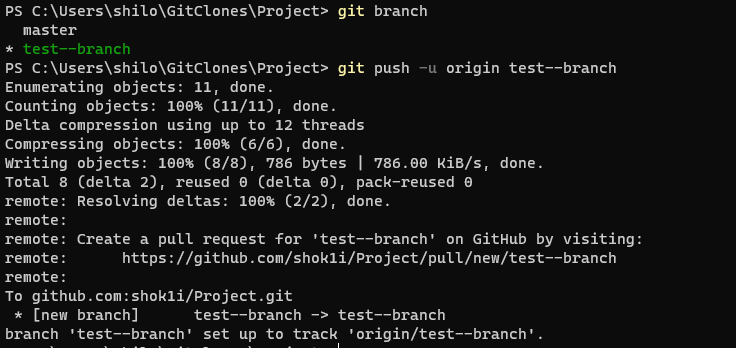
Задание 2. Создайте новую ветку и выведите список всех веток



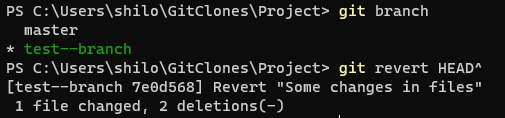
Задание 3. Произведите 3 коммита в новой ветке в разные файлы



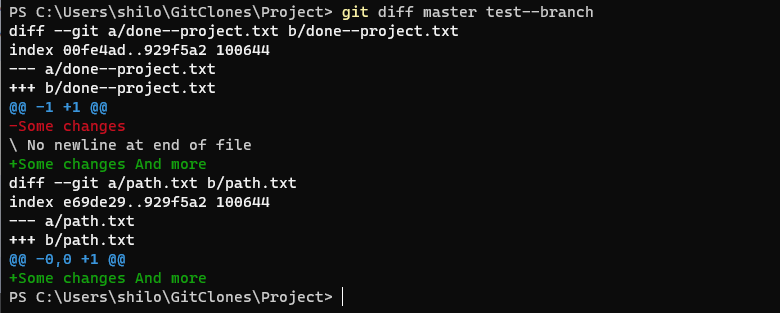
Задание 4. Выгрузите изменения в удаленный репозиторий



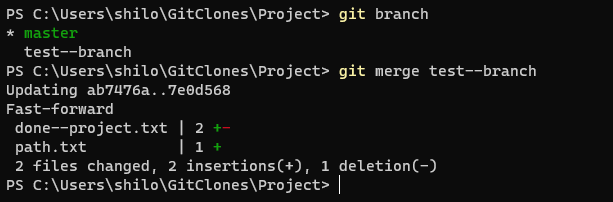
Задание 5. Произведите revert предпоследнего коммита в новой ветке



Задание 6. Выведите в консоли различия между веткой master и новой веткой



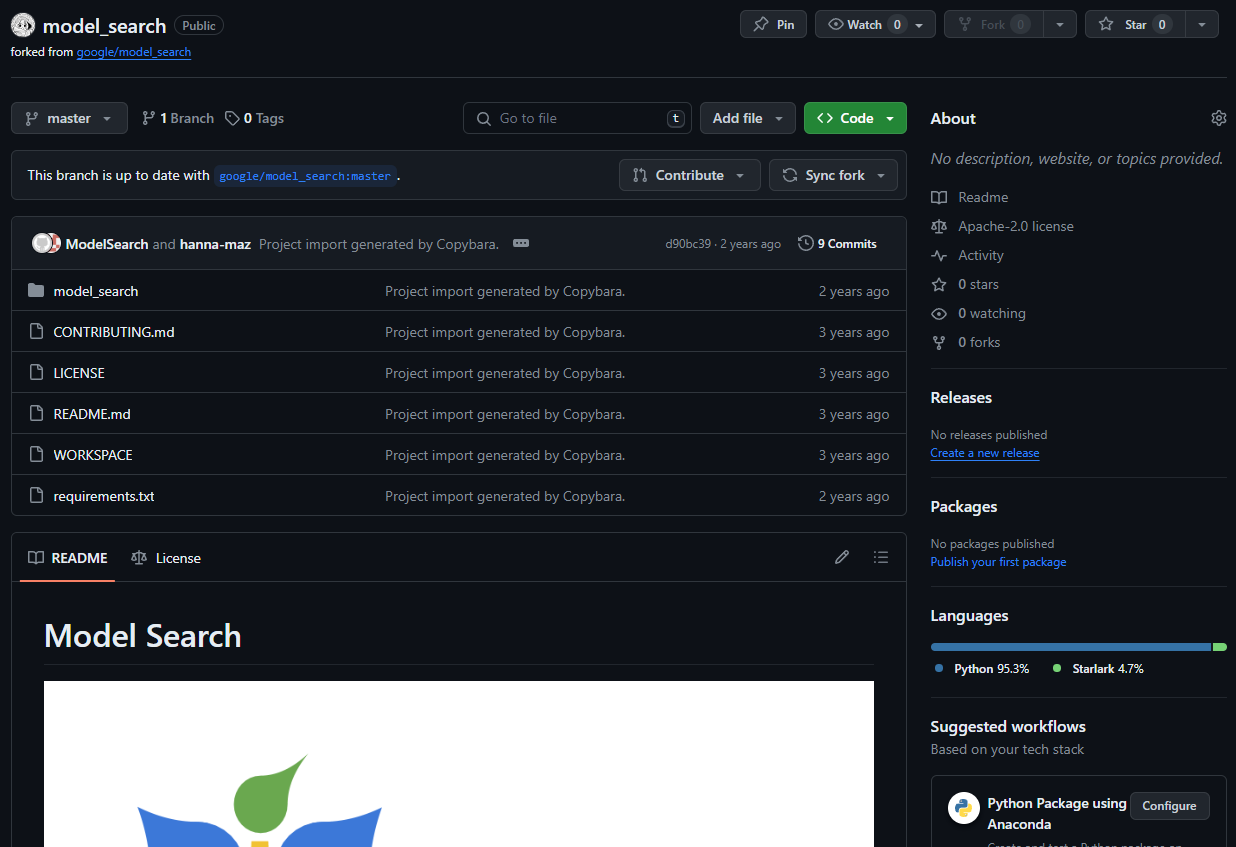
Задание 7. Слейте новую ветку с master при помощи merge



**Часть 3. Ветвление и оформление кода**

**ВАРИАНТ 2.** github.com/google/model\_search

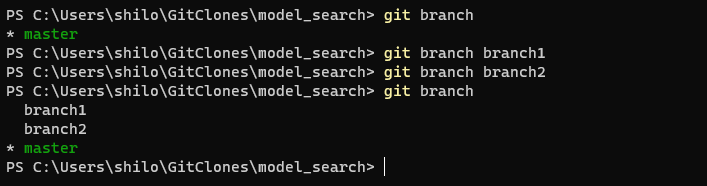
Задание 1. Сделайте форк репозитория в соответствии с вашим вариантом



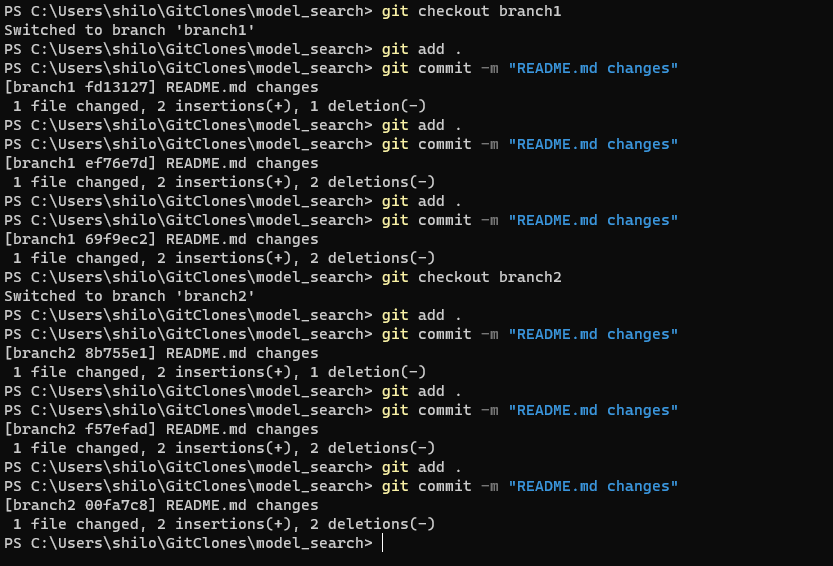
Задание 2. Склонируйте его на локальную машину



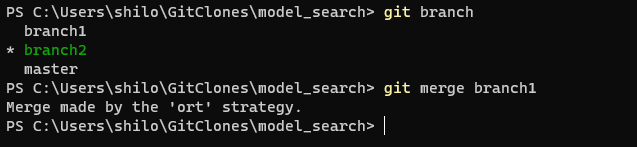
Задание 3. Создайте две ветки branch1 и branch2 от последнего коммита в master'е

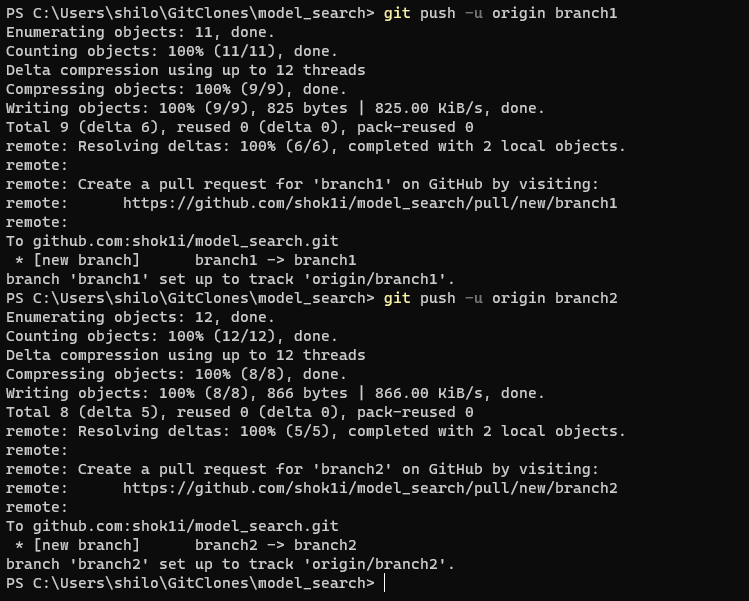


Задание 4. Проведите по 3 коммита в каждую из веток, которые меняют один и тот же кусочек файла

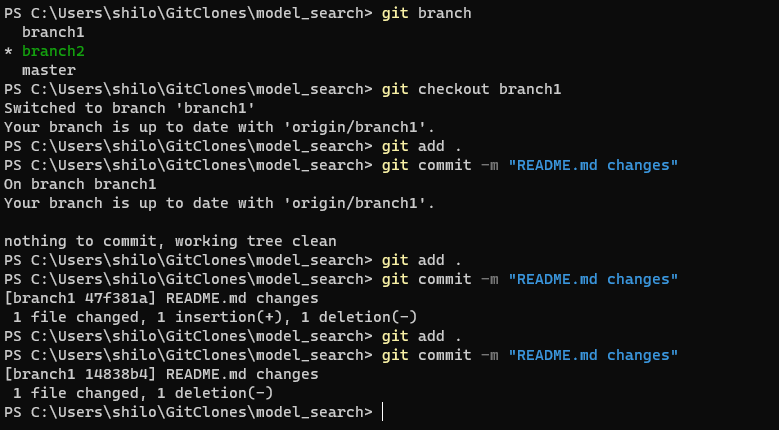


Задание 5. Выполните слияние ветки branch1 в ветку branch2, разрешив конфликты при этом

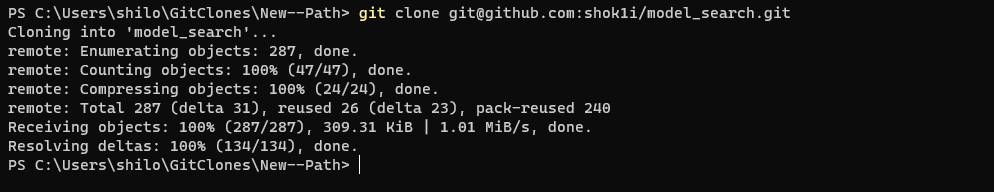
Задание 6. Выгрузите все изменения во всех ветках в удаленный репозиторий



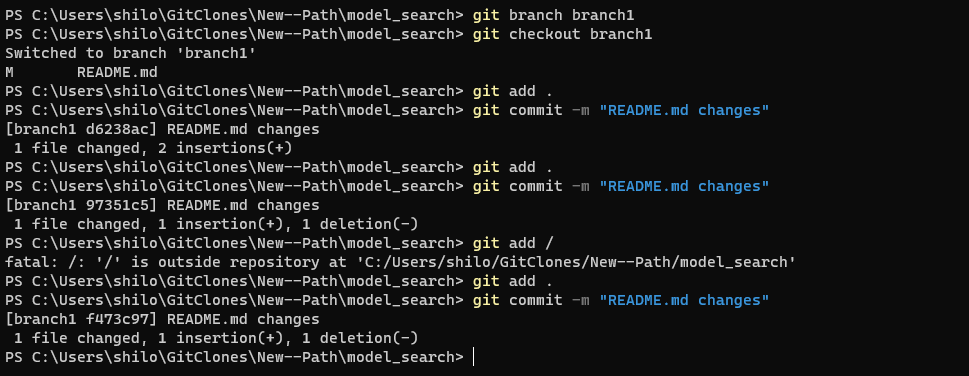
Задание 7. Проведите еще 3 коммита в ветку branch1



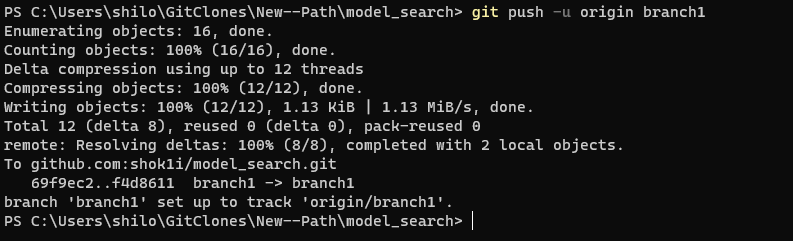
Задание 8. Склонируйте репозиторий еще раз в другую директорию



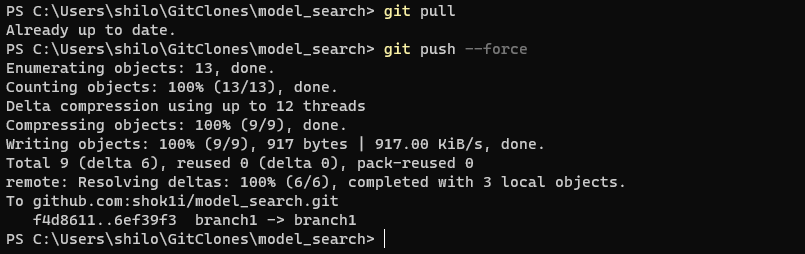
Задание 9. В новом клоне репозитории сделайте 3 коммита в ветку branch



Задание 10. Выгрузите все изменения из нового репозитория в удаленный репозиторий.



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



Задание 12. Получите все изменения в новом репозитории

