

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГАОУ ВО «МГТУ «СТАНКИН»)**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | Институт цифровых интеллектуальных систем | Кафедра  компьютерных систем управления | |

Дисциплина «Основы системного программного обеспечения»

**Отчет по лабораторной работе №\_1\_**

|  |  |  |
| --- | --- | --- |
| **Выполнил**  **студент гр. АДБ-22-08:** | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  *(дата) (подпись)* | **Красномовец А.Н** |
| **Проверил**  **к.т.н., доцент** | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  *(дата) (подпись)* | **Ковалев И.А.** |

# Оглавление

[Цель работы 3](#_Toc198620642)

[Основные задачи 3](#_Toc198620643)

[Ход работы 4](#_Toc198620644)

[Вывод 17](#_Toc198620645)

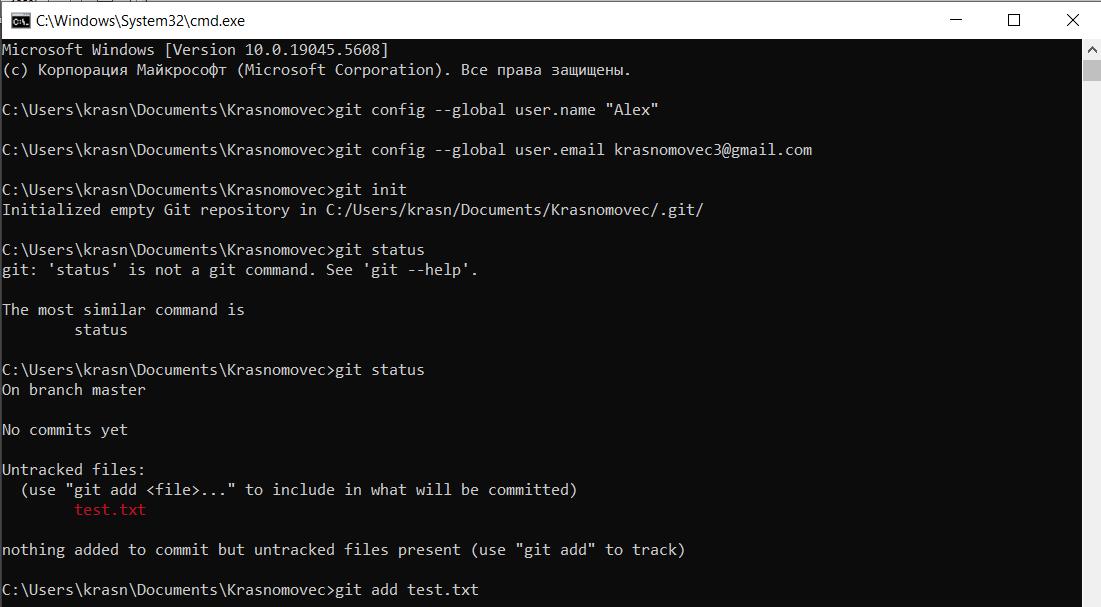
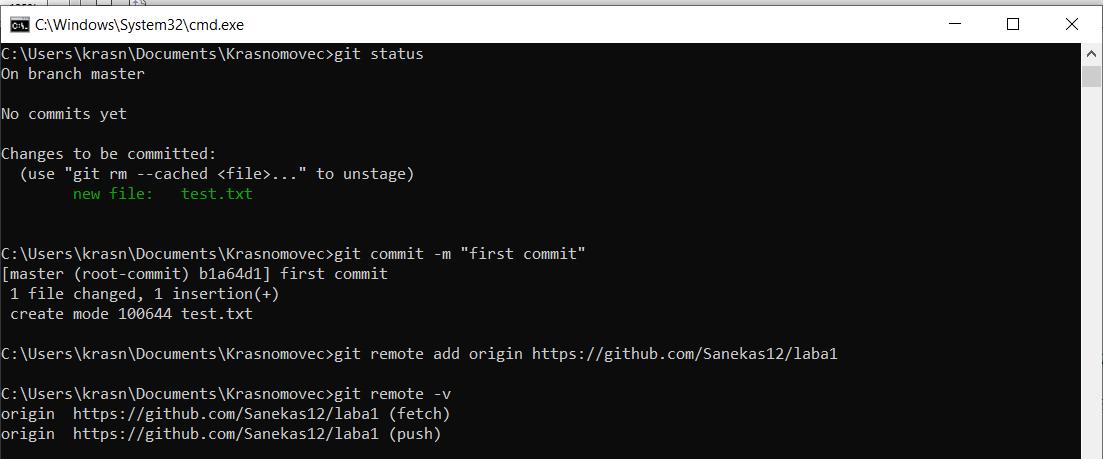
# Цель работы

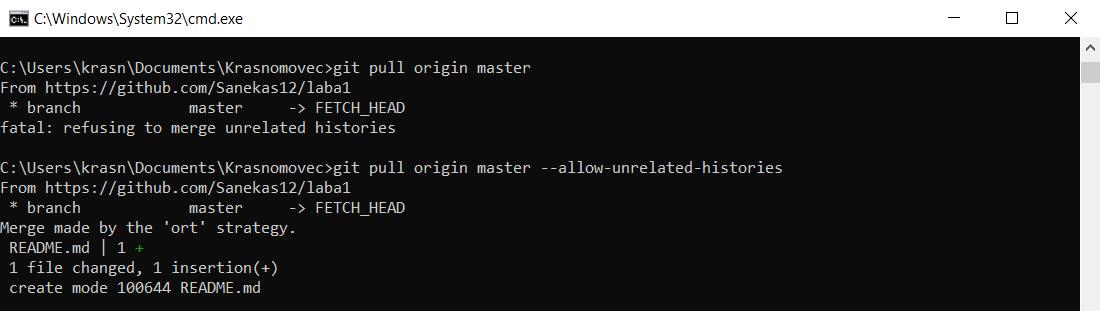
Научится работать с системами контроля версий на примере Git Hub

# Основные задачи

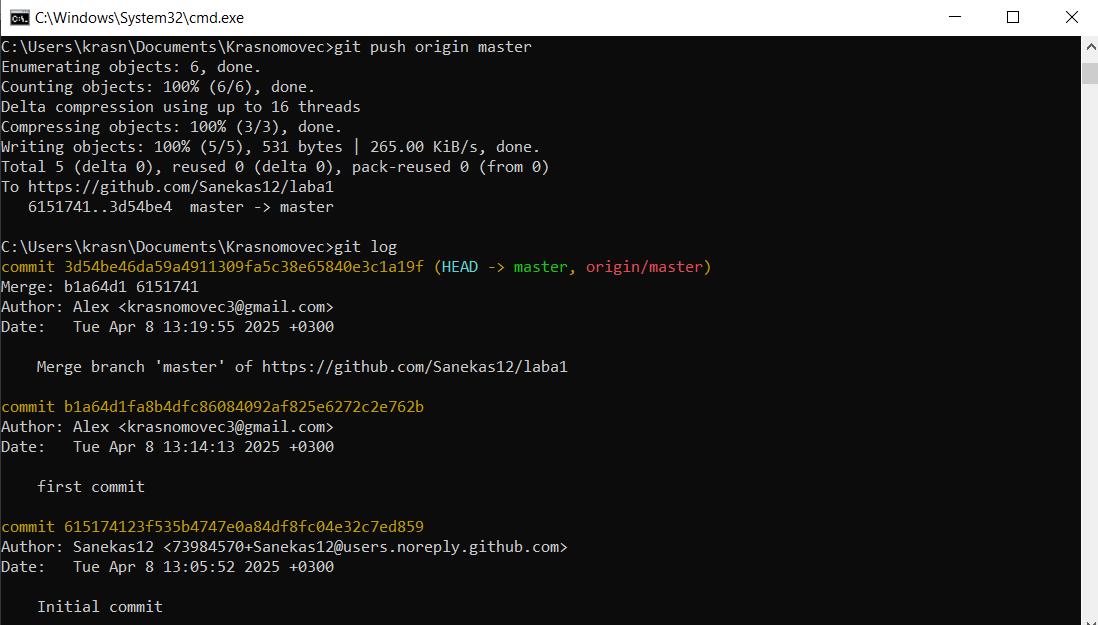
* Создать учетную запись на github.com
* Создать локальный репозиторий
* Научится фиксировать изменения файлов, пересылать локальный коммит на сервер, делать слияние веток
* Просмотреть как делать разрешение конфликтов

# Ход работы

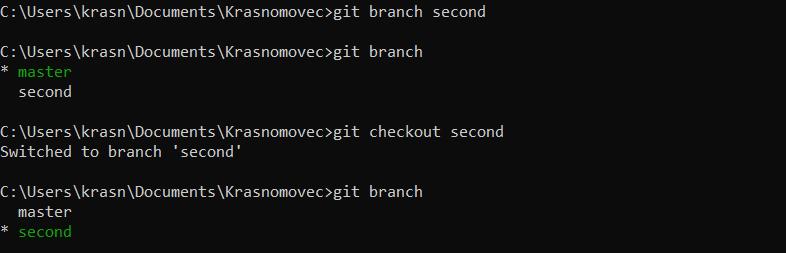
1. Заходим в уже имеющийся личный кабинет и создаем новый репозиторий.
2. Представляемся нашим именем и почтой. Создаем файл test.txt с именем. Открываем командную строку «cmd» в нужном каталоге и инициализируем пустую git директорию. Вводим команду: git stаtus и нам отображается сообщение, что есть новый файл, но он не отслеживается. Добавляем в подготовленные файлы с помощью add.
3. Видим, что файл подготовлен и коммитим изменения. Связываем наш локальный репозиторий с репозиторием на GitHub.
4. Запросим изменения с сервера



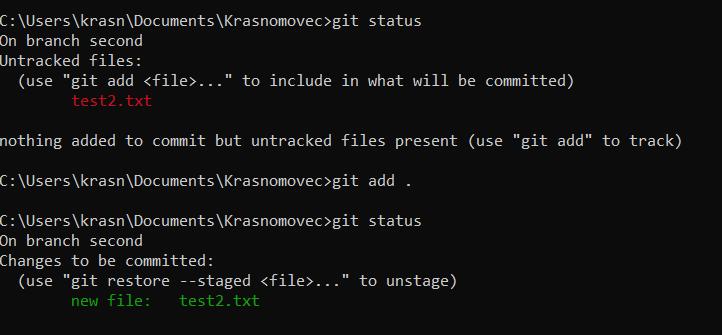
1. Посмотрим наши изменения

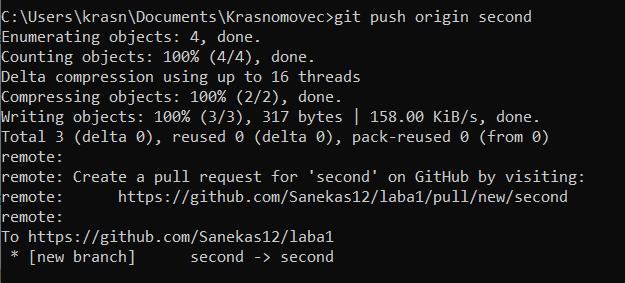


1. Создадим новую ветку second, посмотрим в какой находимся и перейдем на другую:

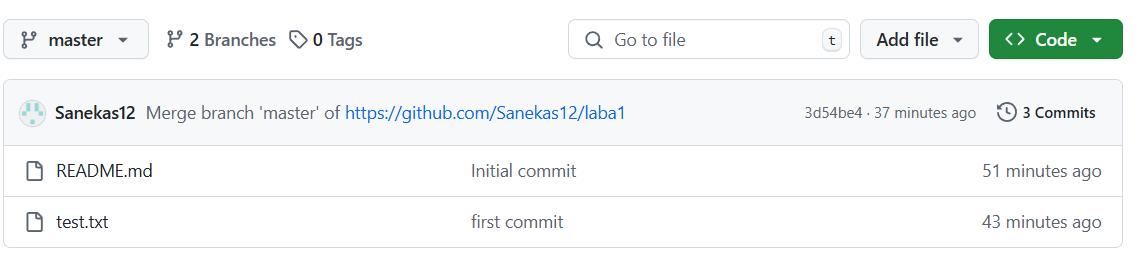


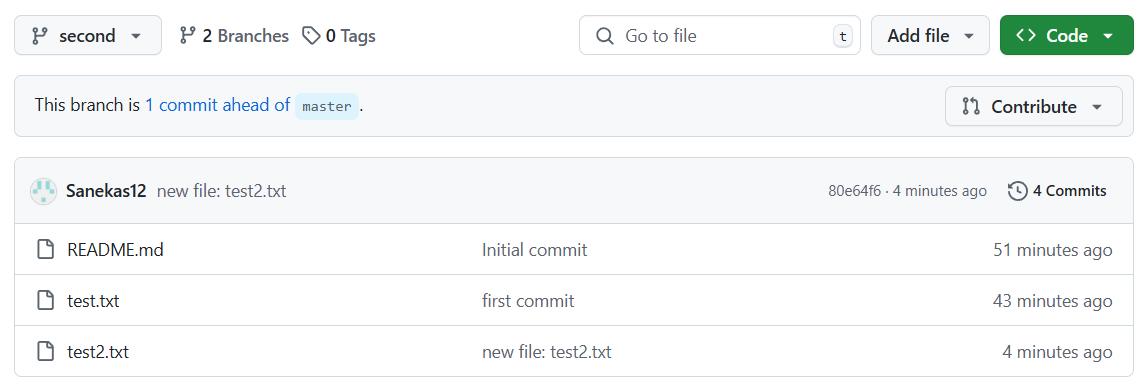
1. Создаем новый файл в нашем локальном репозитории и пишем в нем фамилию, добавляем в область подготовленных файлов, коммитим и отправляем на сервер



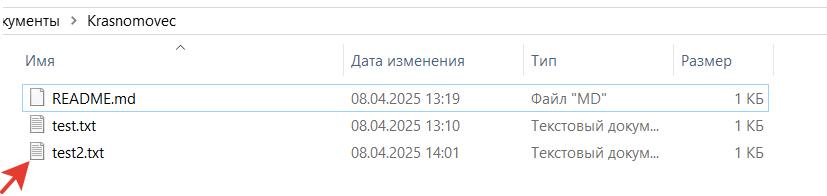
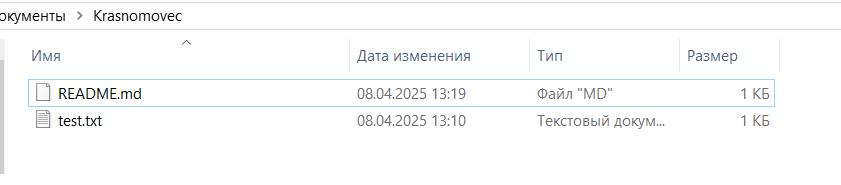


1. Видим, что на GitHub наши ветки выглядят так

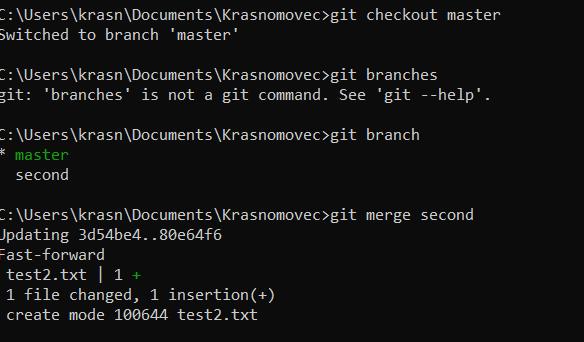




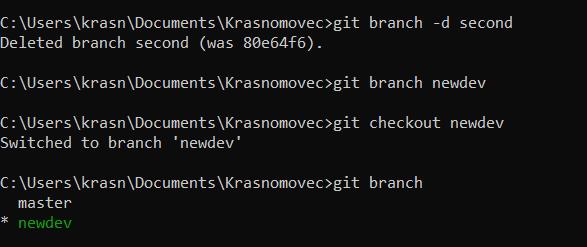
1. При переключении веток видим локально как меняются файлы в нашей рабочей директории



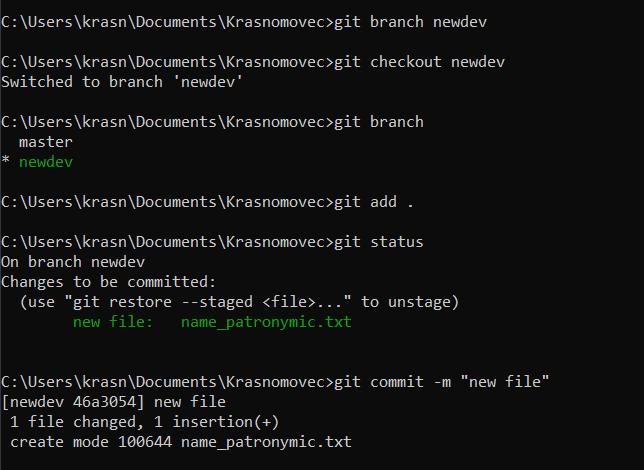
1. Делаем слияние веток



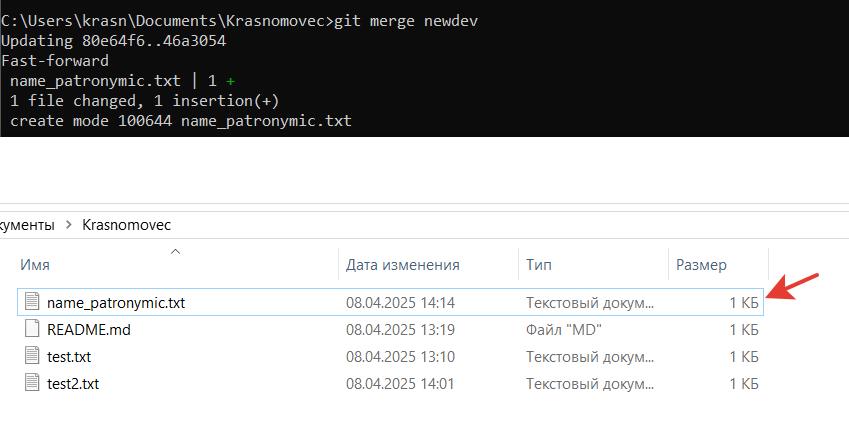
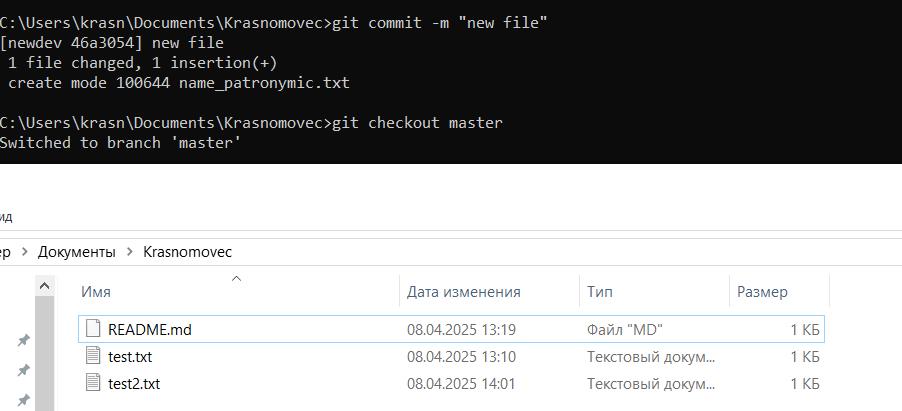
1. Удаляем ветку second и создаем новую newdev

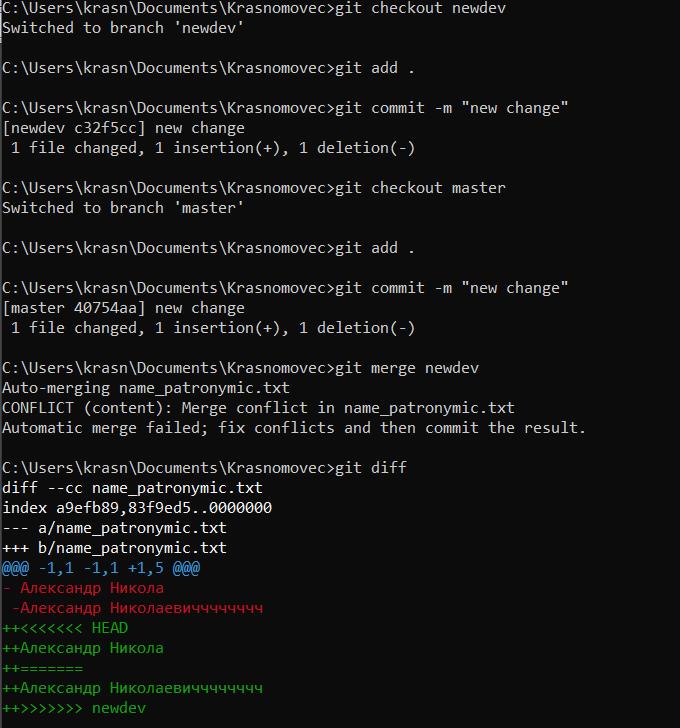


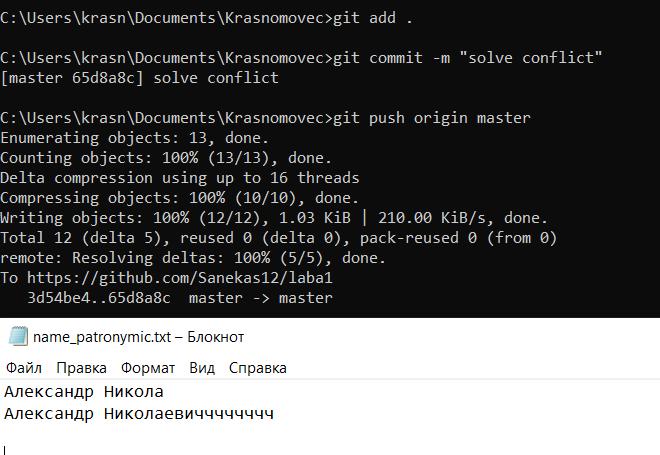
1. Переключаемся на нее. Добавляем в файл с именем и отчеством. Фиксируем и коммитим изменения.



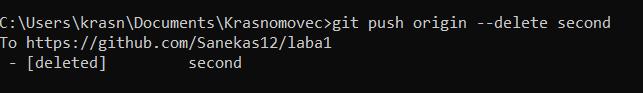
1. Объединяем ветки и меняем файлы в обоих ветках. Теперь если мы захотим объединить то, увидим конфликт. Нужно его разрешить



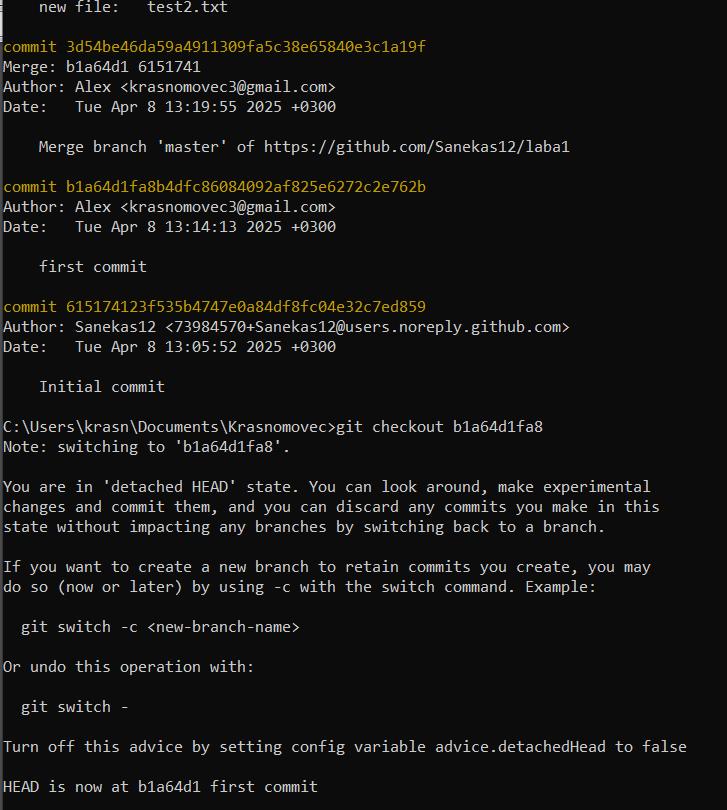


Git отметил строки, содержащие конфликт: над разделителем ======= мы видим последний (HEAD) коммит, а под ним — конфликтующий. Вручную исправляем файл и коммитим нужную информацию.

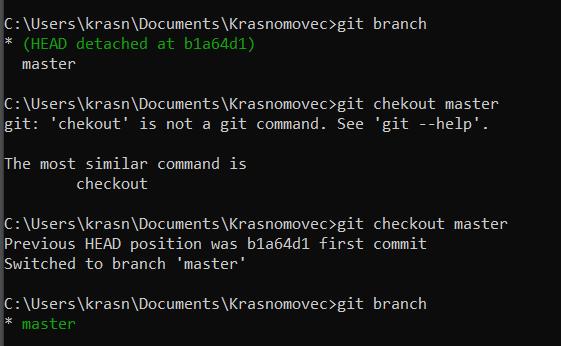
1. Удаляем ветки с сервера GitHub



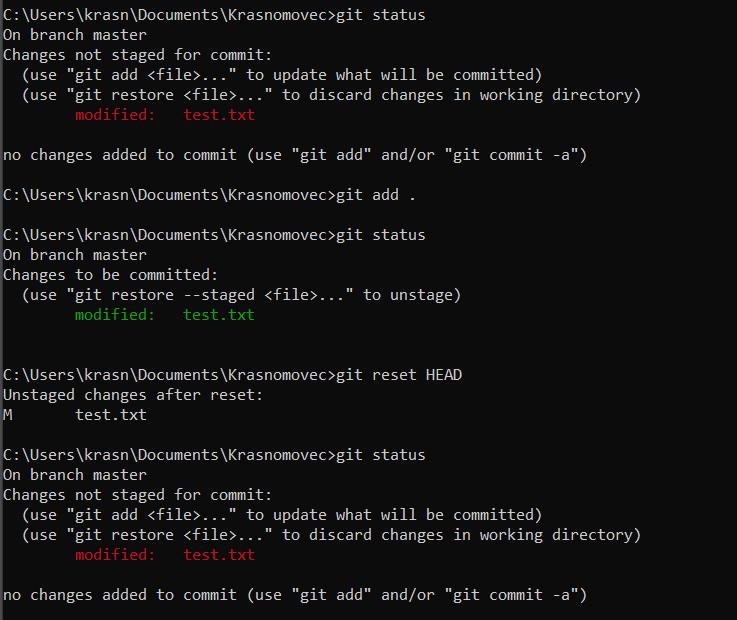
1. Чтобы посмотреть все коммиты, можно использовать команду git log. Можно выбрать коммит по хешу и откатиться.



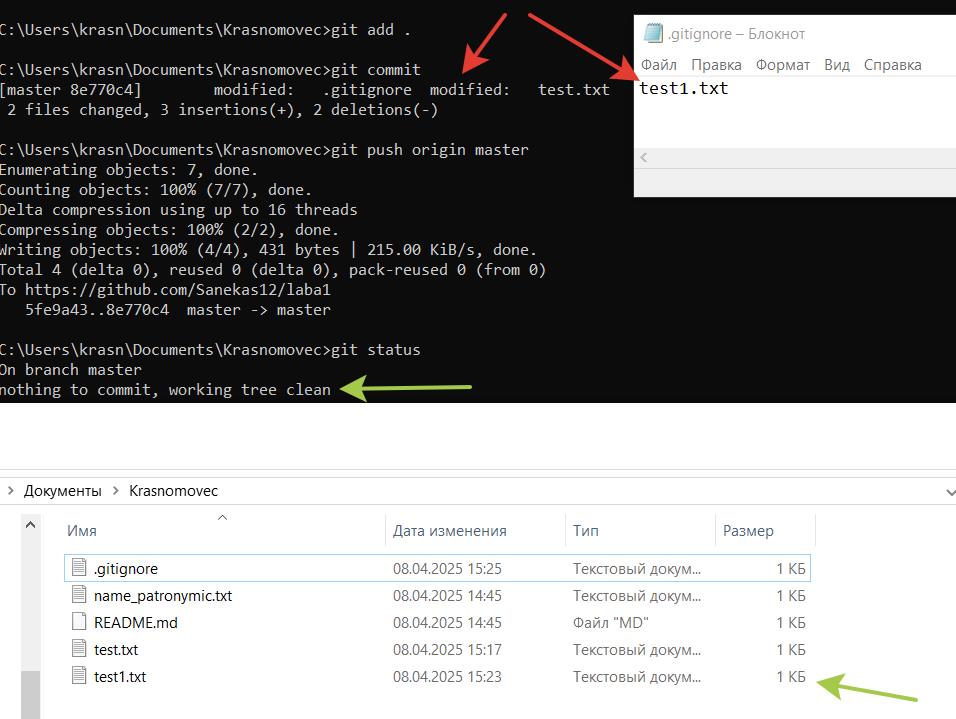
Возвращаемся в master



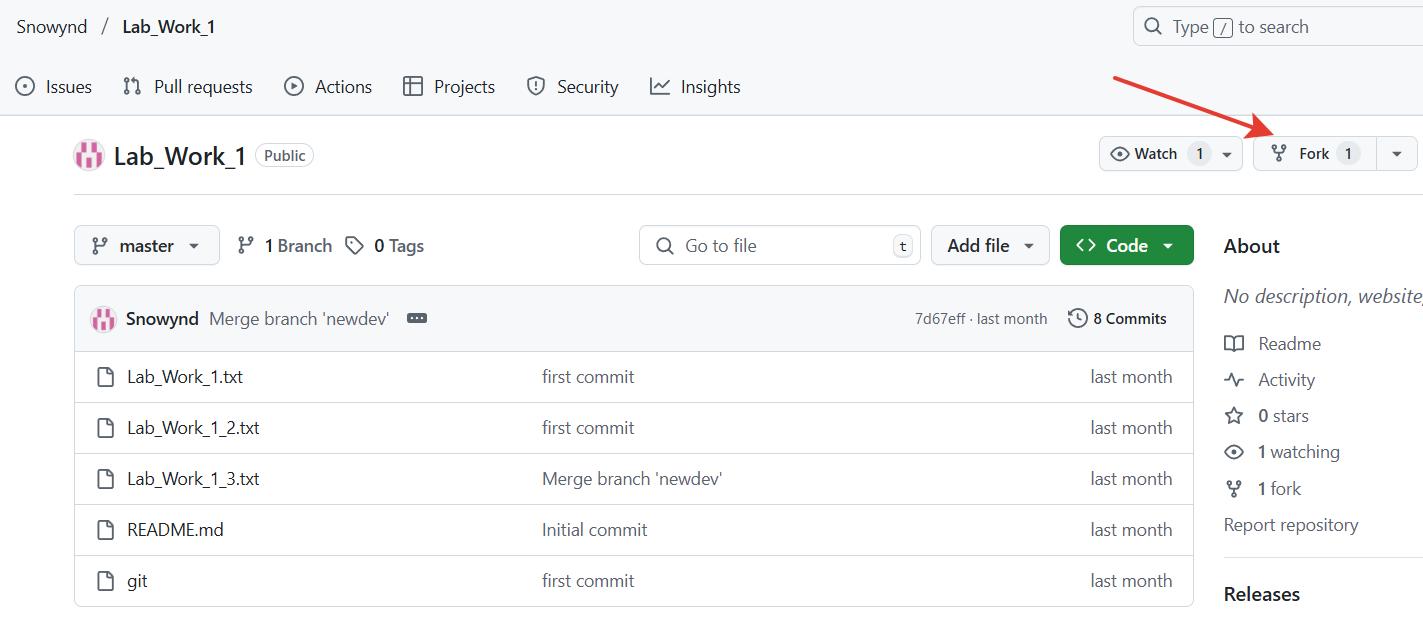
1. Мы можем исключить файлы из области зафиксированных с помощью git reset HEAD



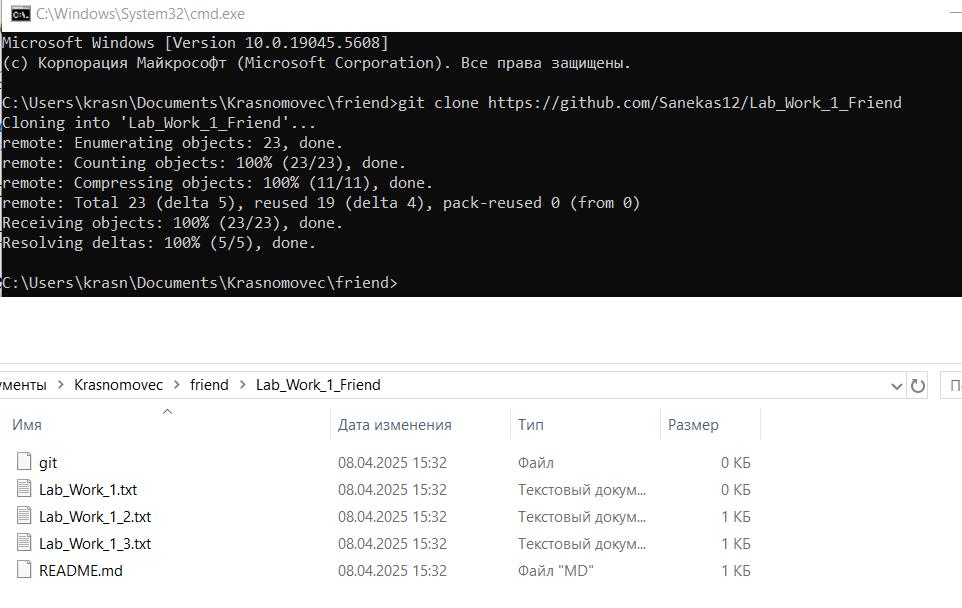
1. Создаем файл под названием «.gitignore» и сохраняем его в директорию проекта. Внутри файла перечисляем названия файлов/папок, которые нужно игнорировать. Файл «.gitignore» добавляем, коммитим и отправляем на сервер. Теперь видим, что измененный файл игнорируется git.



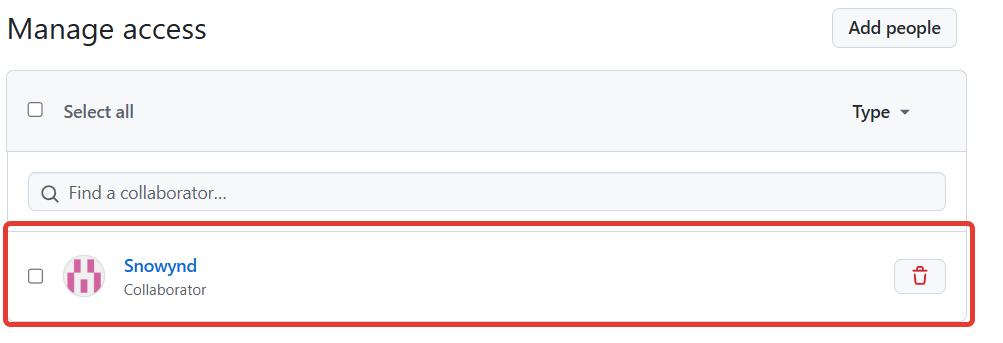
1. Переходим в репозиторий друга и делаем себе его копию с помощью «Fork»

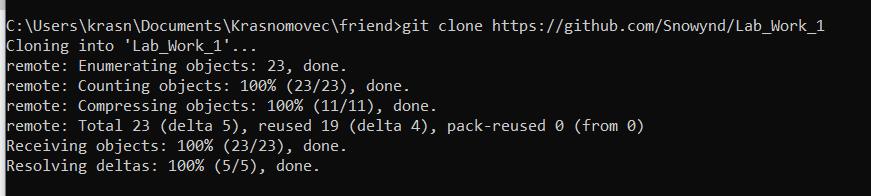


1. Клонируем этот репозитории на локальный компьютер, мы можем с ним работать, но изменения будут выгружать в наш скопированный репозиторий, а не оригинальный

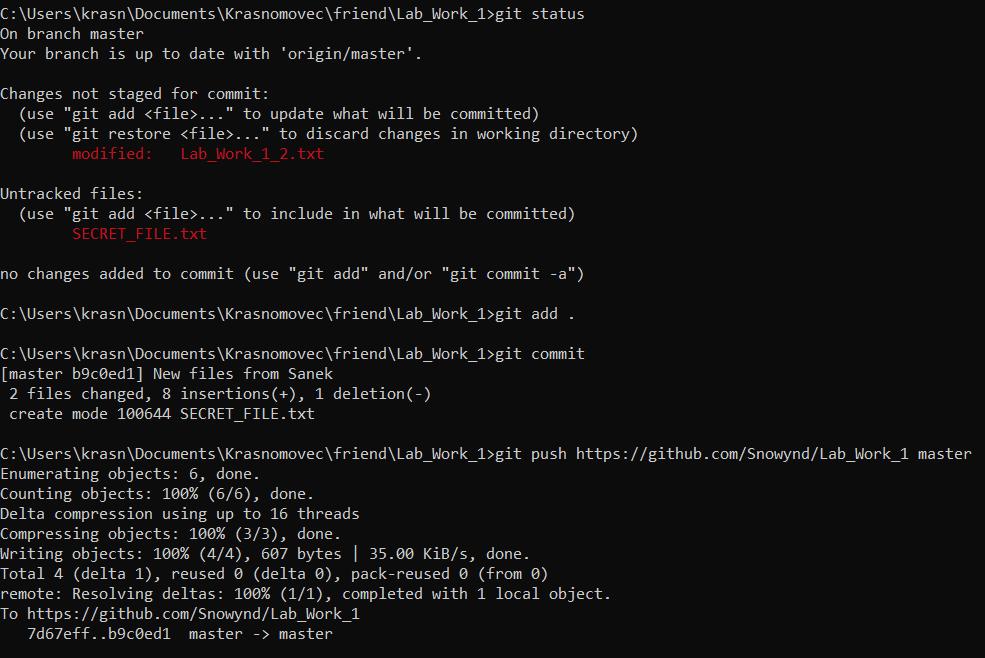


1. Разрешаем другу доступ к своему репозиторию и получаем доступ к его репозиторию, копируем его репозиторий себе с помощью git clone

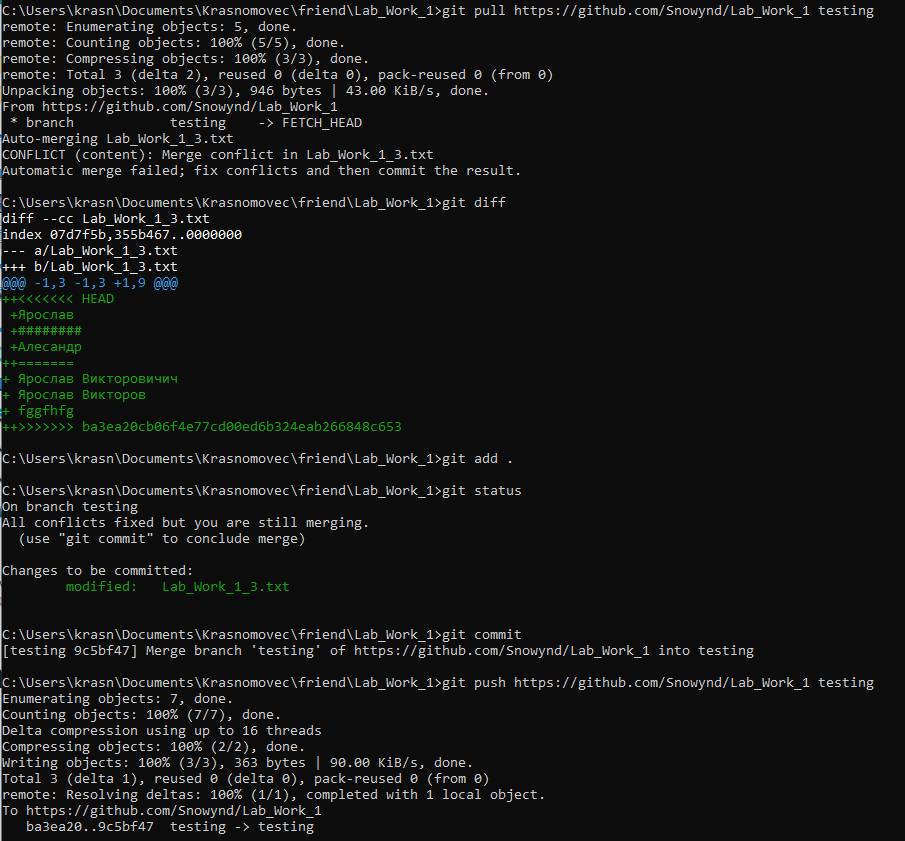




1. Меняем существующий файл и создаем новый, добавляем их, коммитим и отправляем в репозиторий друга



1. Во время совместной работы хотим сделать pull ветки с сервера, но получаем конфликт, т.к. друг запушил изменения, а мы локально тоже меняли файлы. Решаем его и пушим обратно на сервер



# Вывод

Мы провели работу с системами контроля версий на примере GitHub и научились создавать локальный репозиторий, пересылать локальный коммит на сервер, делать слияние ветоки работать в Git совместно.