

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет**

**«СТАНКИН»**

**(ФГБОУ 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-21-06 Ушков П.Н.

(дата) (подпись)

Проверил:

к.т.н. доцент Ковалёв И.А.

(дата) (подпись)

**Москва 2024**

**Оглавление**

[Создание локального репозитория и отгрузка данных 3](#_Toc167536626)

[Создание новой ветки 5](#_Toc167536627)

[Слияние веток 7](#_Toc167536628)

[Просмотр изменений и разрешение конфликтов 11](#_Toc167536629)

[Удаление веток на сервере 12](#_Toc167536630)

[Возврат к предыдущему состоянию 13](#_Toc167536631)

[Исправление коммита 16](#_Toc167536632)

[Изменение комментария 19](#_Toc167536633)

[Отправка только нужных файлов на сервер 20](#_Toc167536634)

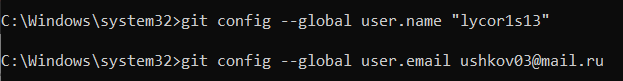
[Совместная работа с git 22](#_Toc167536635)

[Вывод 26](#_Toc167536636)

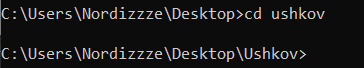
[Список литературы 27](#_Toc167536637)

**Цель работы:** ознакомиться с системой контроля версий Git.

# **Создание локального репозитория и отгрузка данных**

**1.** Указываем имя пользователя и почту:  


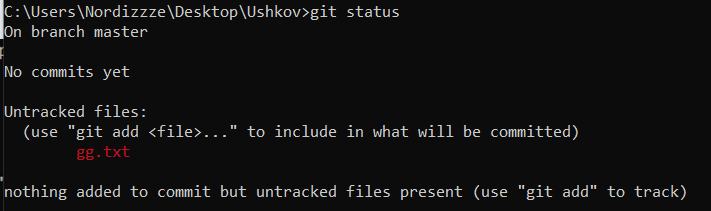
**2.** Переходим в созданную папку в командной строке используя команду *cd.*



**3.** Теперь необходимо проинициализировать эту папку как git репозиторий: *git init*.

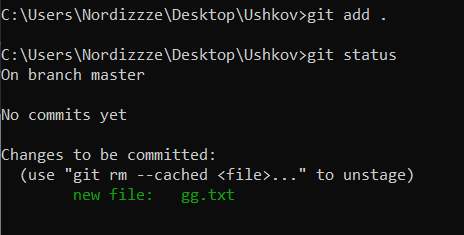


**4.** Теперь используя проводник windows создаём в своей папке любой текстовый файл и прописываем в него имя. Чтобы проверить, нужно ввести команду: *git stаtus.*

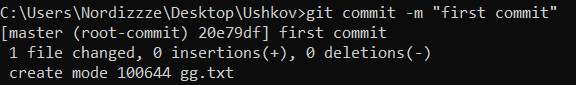


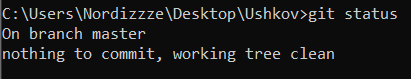
Отобразилось сообщение, что есть новый файл, но он не отслеживается.

**5.** Для того, чтобы начать отслеживать новый файл, нужно его специальным образом объявить. Если хотим добавить только один файл, можно написать: ‘*git add text.txt’*.Если измененных файлов много, то можно сделать так: ‘*git add .’* . Выполнив нужную команду, снова посмотрим статус репозитория:



**6.** Теперь сделаем commit и назовём его:

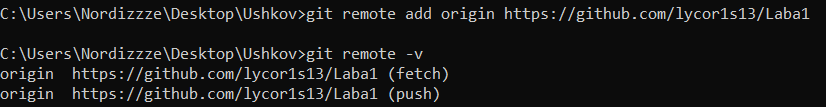




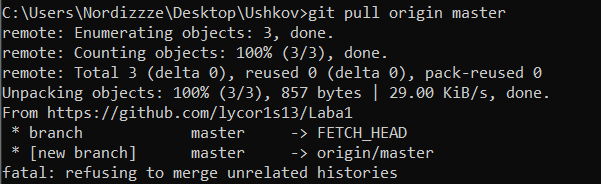
**7.** Отправка коммита на сервер:

1)Привязка ссылки на git.

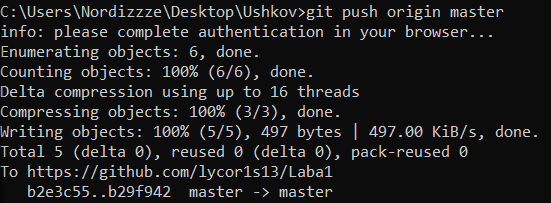
*git remote add origin …*



2) Запрос изменений с сервера:



3) Отправка на сервер:



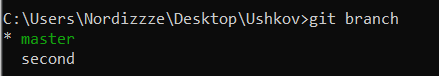
# **Создание новой ветки**

**1.** Создаём новую ветку:



Можно посмотреть в какой ветки находимся, используя команду:

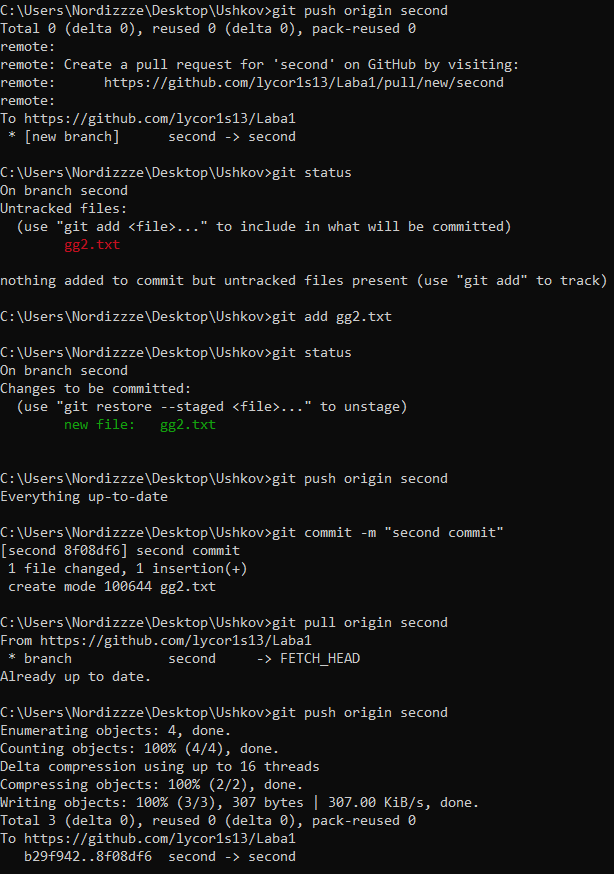
*git branch*



**2.** Переключение на другую ветку:

****

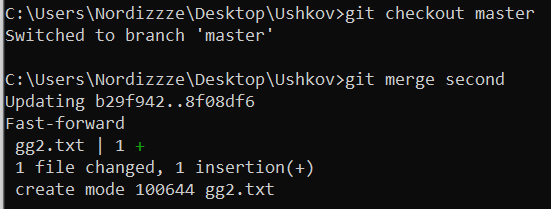
**3.** Создаём новый файл в нашем локальном репозитории и пишем в нем свою фамилию, добавляем в область подготовленных файлов, комитим и отправляем на сервер, с учётом того, что у нас теперь ветка second, а не master:

****

# **Слияние веток**

**1.** Переключаемся снова на ветку master. Если посмотреть через проводник windows в папку с проектом, то файла test\_new.txt не будет виден, потому что мы переключились обратно на ветку master, в которой такого файла не существует. Чтобы он появился, нужно воспользоваться merge для объединения веток (применения изменений из ветки second к основной версии проекта).

*git merge second*

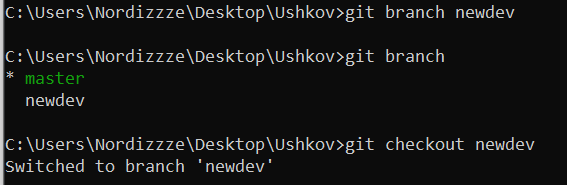


**2.** Если все прошло без ошибок, можно удалить ветку:

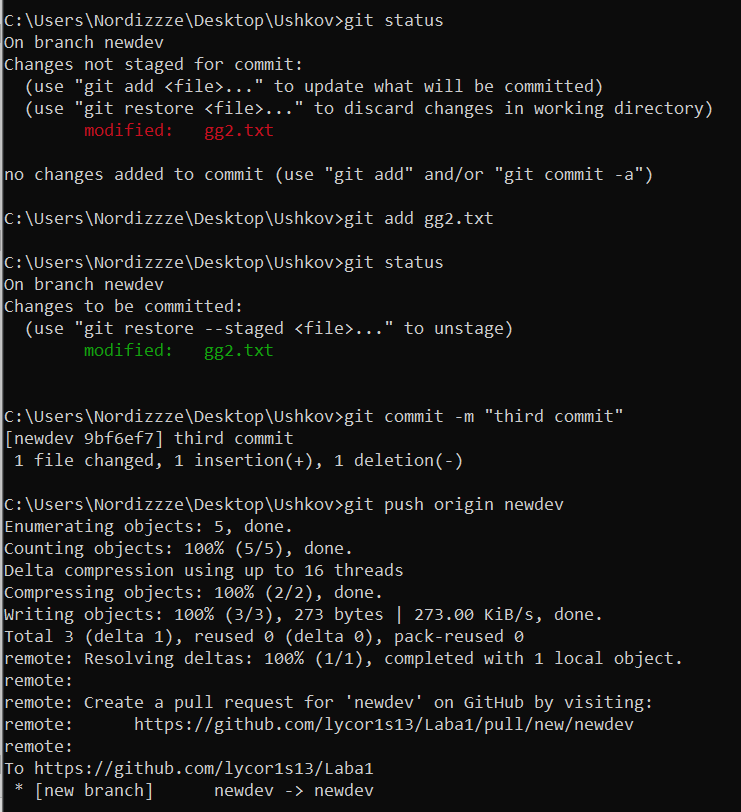
*git branch –d second*

****

**3.** Следующеезадание усложняется, в двух ветках могут быть одинаковые файлы и над ними работают разные разработчики. Создаём ветку с названием newdev:

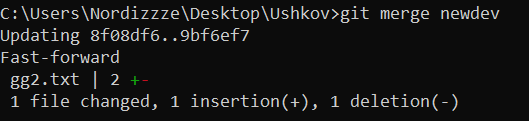


**4.** Добавляем в новую ветку файл с таким же название как в ветке master:



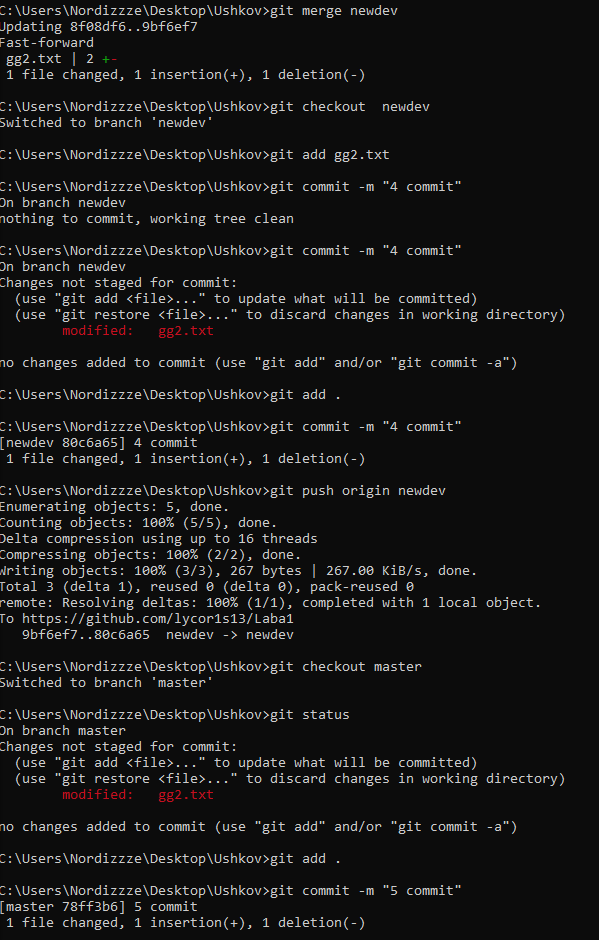
**5.** Теперь, когда переключимся на ветку master не увидим отчества в файле. Попробуем объединить их:

*git merge newdev*



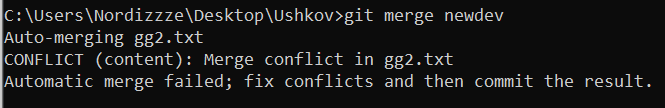
Если мы зайдём в наш файл увидим имя и отчество.

**6.** Переключимся снова в ветку newdev и удалим в отчестве несколько букв. Зафиксируем и закомитим. Потом переключаемся назад в ветку master и добавляем несколько букв в отчество. Теперь ничего не получается, возникает конфликт:



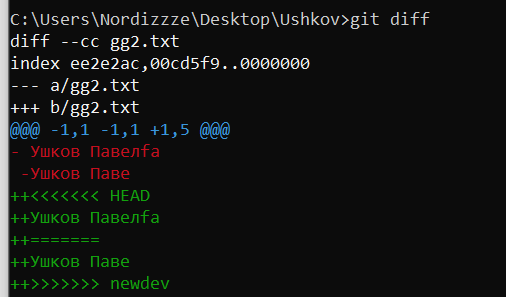
# **Просмотр изменений и разрешение конфликтов**

**1.** Нужно разрешить конфликт:

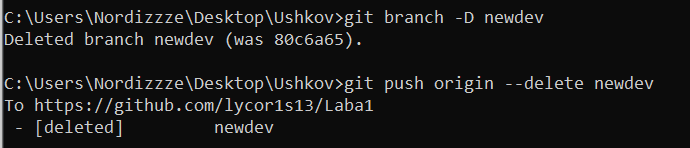


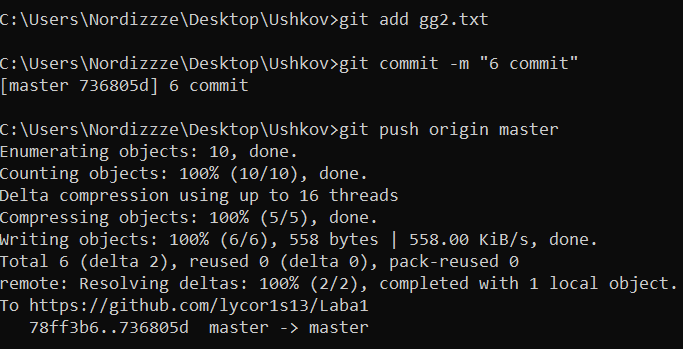
**2.** Набираем команду, для просмотра изменений:

*git diff*



**3.** Сделаем всё очень просто, удалим ветку newdev и поправим файл в ветке master:



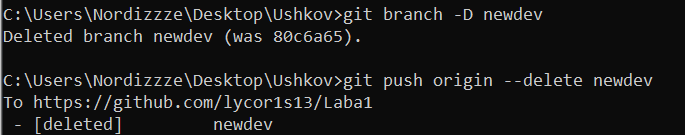


# **Удаление веток на сервере**

**1.** Удаление из github:

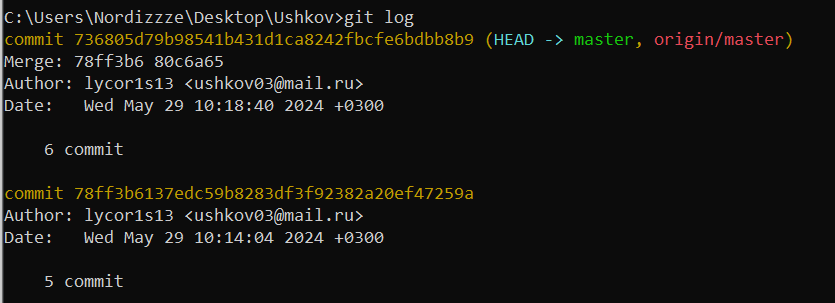
*git push origin --delete newdev*

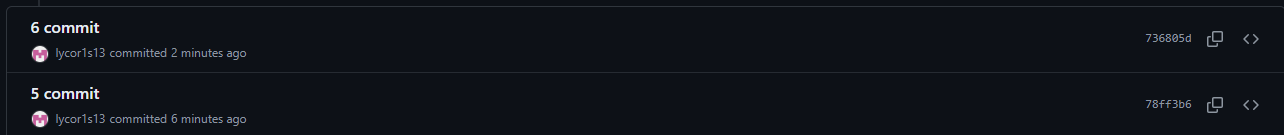
*git push origin --delete second*

****

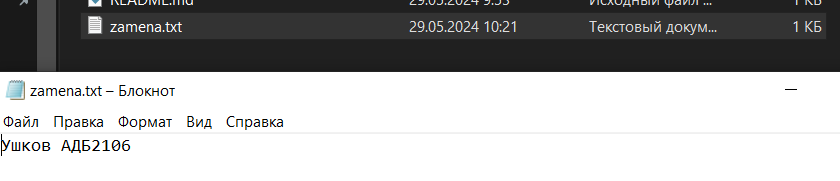
# **Возврат к предыдущему состоянию**

**1.** Есть два коммита: commit и second commit. Попробуем с commit откатится на second commit.



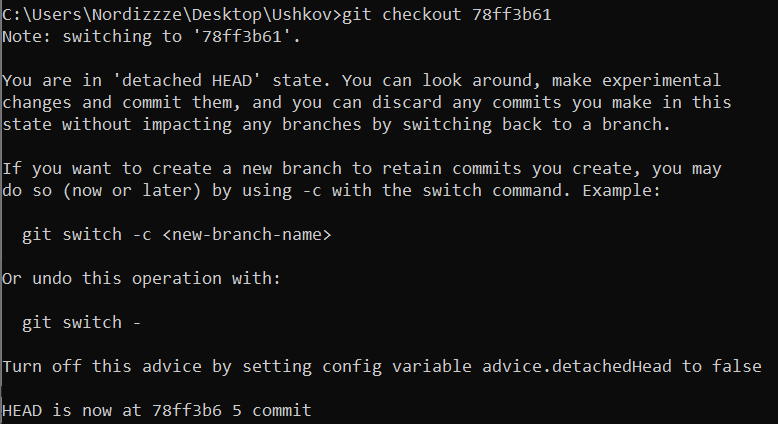


**2.** Создадим файл и запомним, что в нём находится (7 commit):

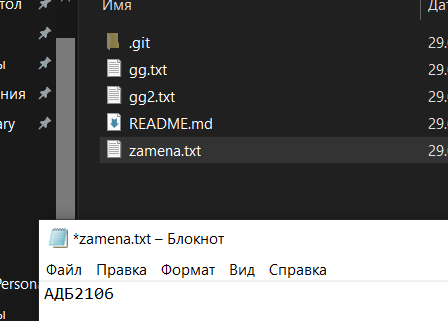


**3.** Нужно указать первые несколько символов коммита (перед выполнением посмотрите свои файлы, чтобы запомнить, что там находится)

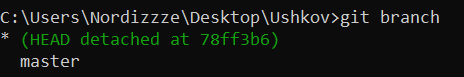
*git checkout [hash commit]*



**4.** Если посмотреть файлы теперь, то увидим, что они поменялись – вернулись назад (6\_1 commit):



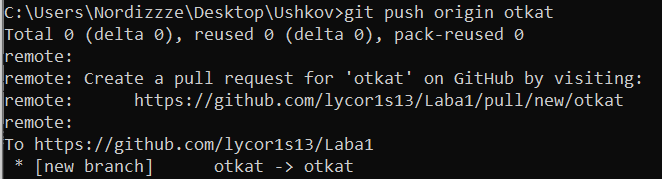
**5.** Чтобы это произошло создалась псевдо-ветка начинающаяся на этом коммите (6\_1 commit):

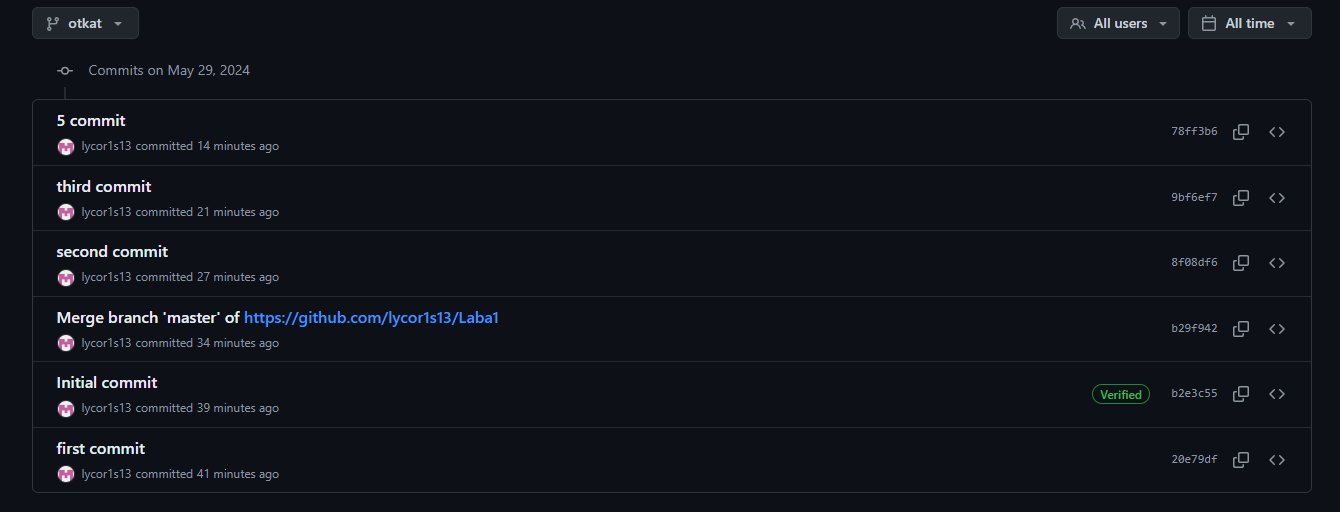


**6.** Создаём новую ветку, чтобы сохранить в ней наш откат. Далее будем работать в ней, если нам это необходимо:



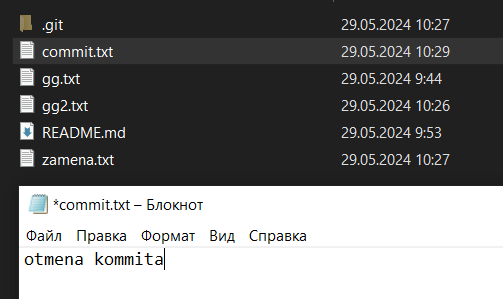
**7.** Запушим новую ветку:



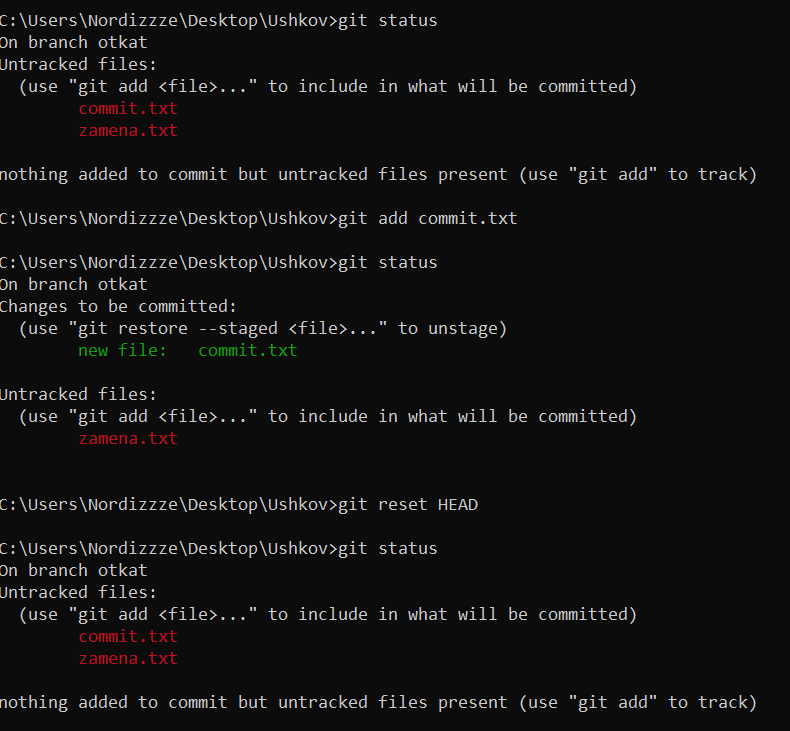
**8.** Теперь имеем ветку с откатом на один commit назад:  


# **Исправление коммита**

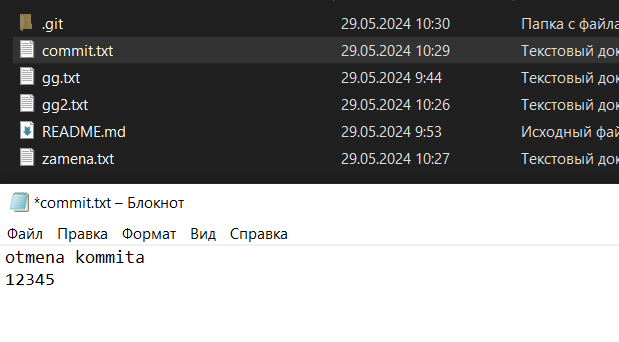
**1.** Создаём файл:

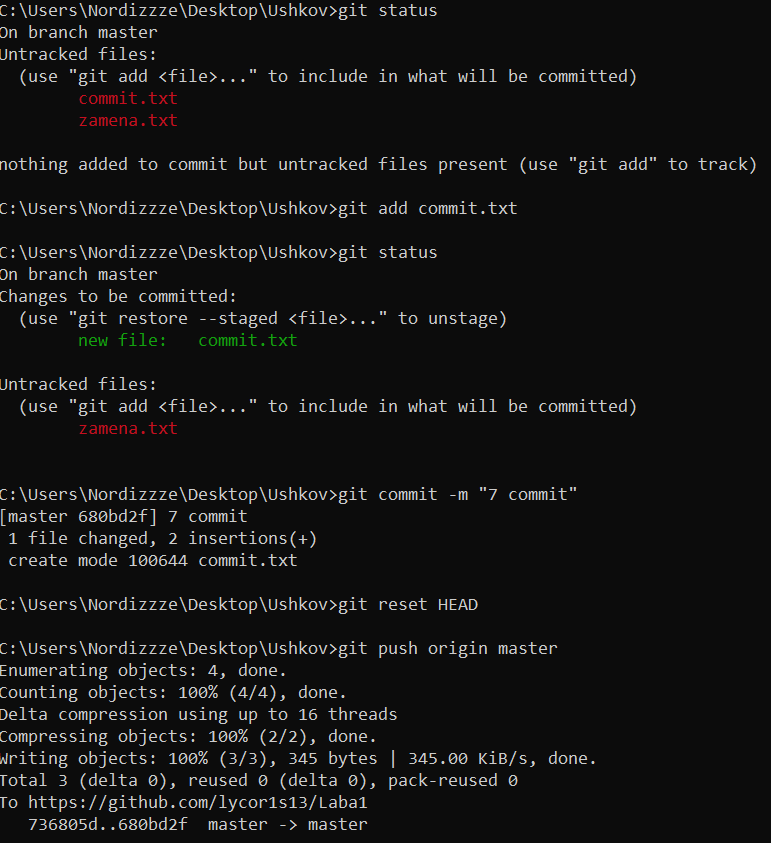


**2.** Отмена добавления файла до коммита:

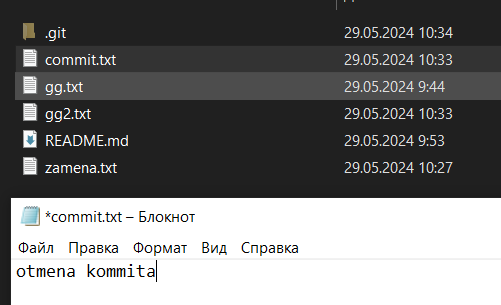


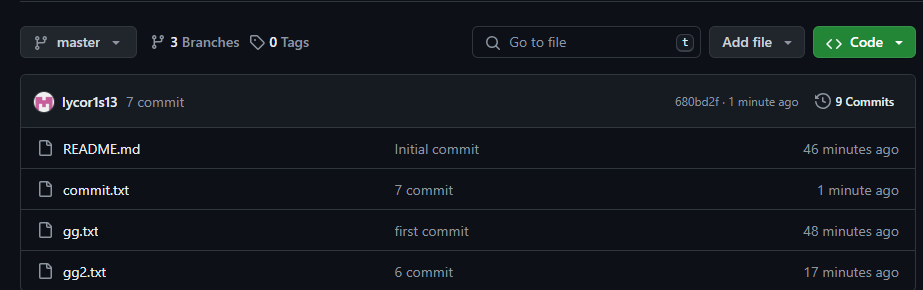
**3.** Отмена добавления файла после коммита:





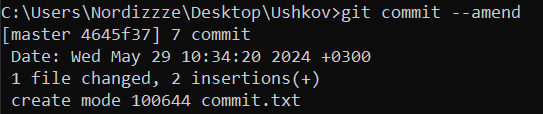
**4.** Файл открепился от коммита до отправки на сервер, изменения не произошли:



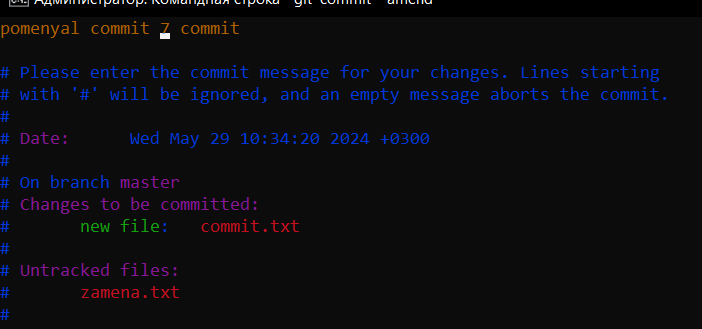


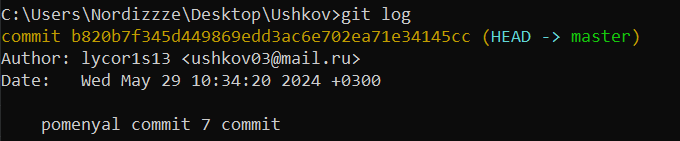
# **Изменение комментария**

**1.** Пишем команду git commit –amend:

****

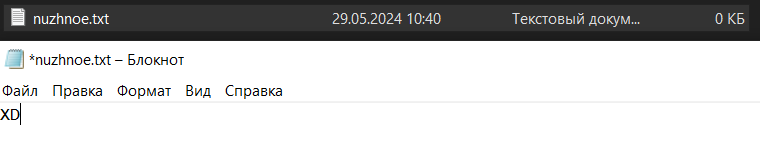
**2.** Вносим изменения:

****

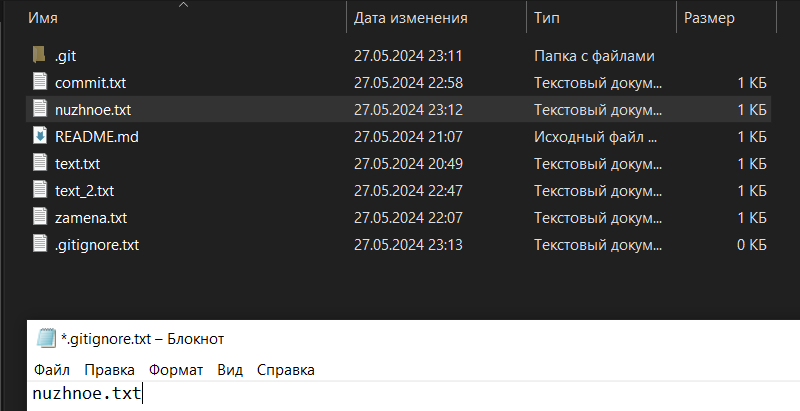
****

# **Отправка только нужных файлов на сервер**

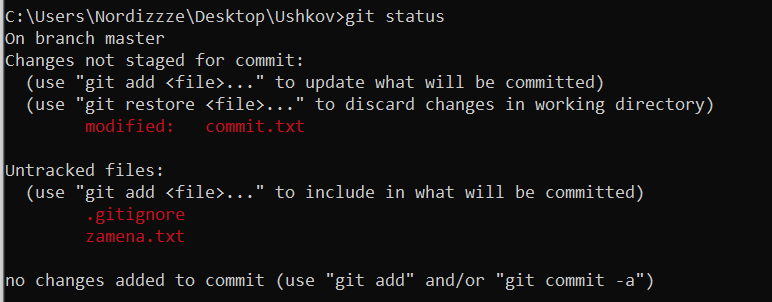
**1.** Создаём новый файл:

****

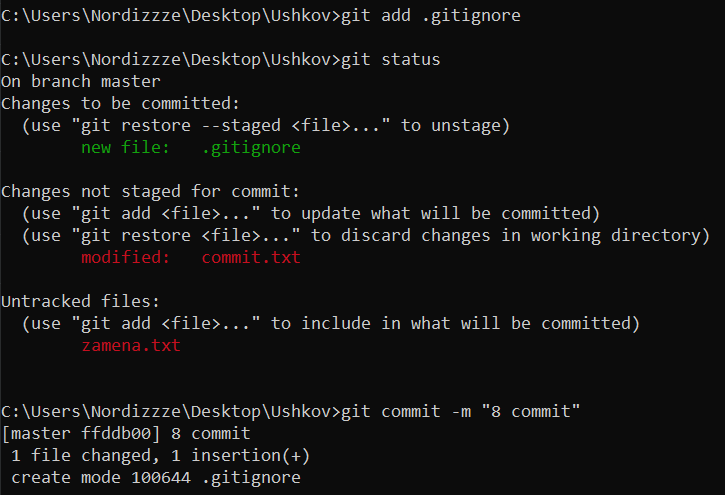
**2.** Создаём файл .gitignore и добавляем в него запрет на коммит нового созданного файла:

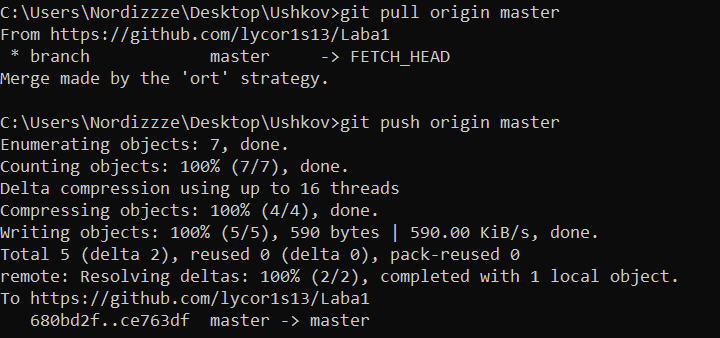
****

**3.** Уже на этапе просмотра директории, файл нельзя трогать (его просто нету):

****

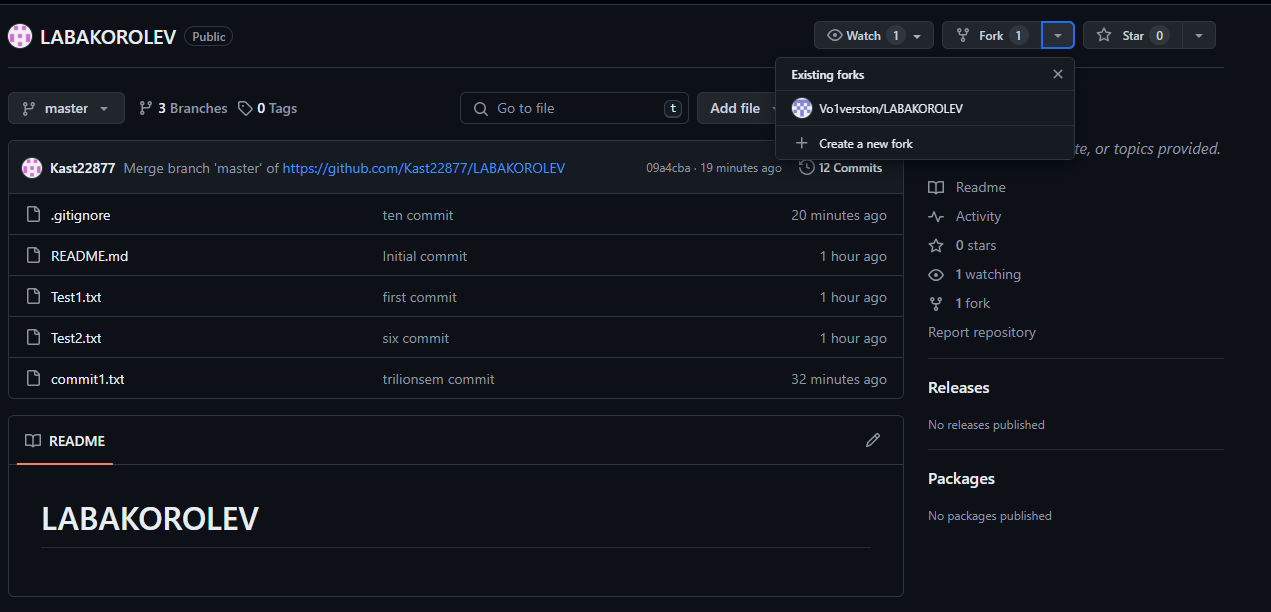
**4.** Отправляем .gitignore на сервер:

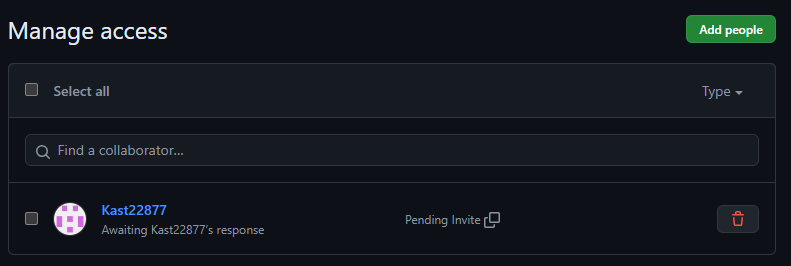
****

****

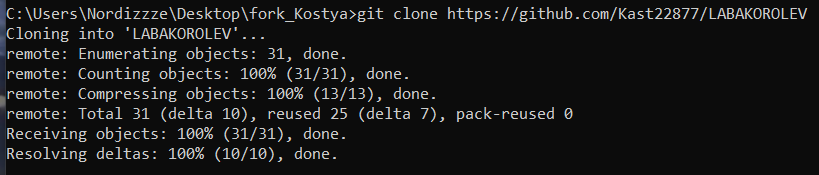
# **Совместная работа с git**

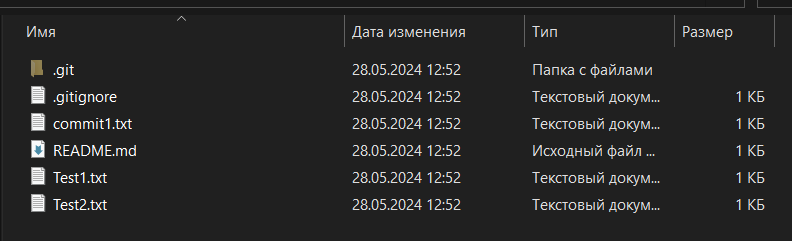
**1.** Берём одногруппника и делимся репозиториями:

****

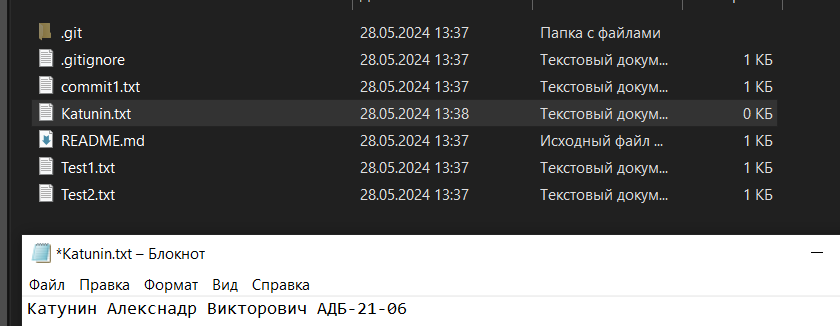
****

**2.** Создаём новую папку и добавляем туда клон репозитория одногруппника:

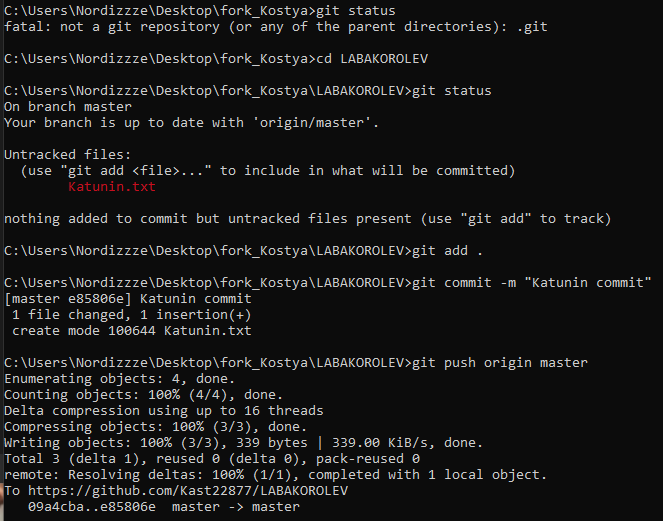
****

****

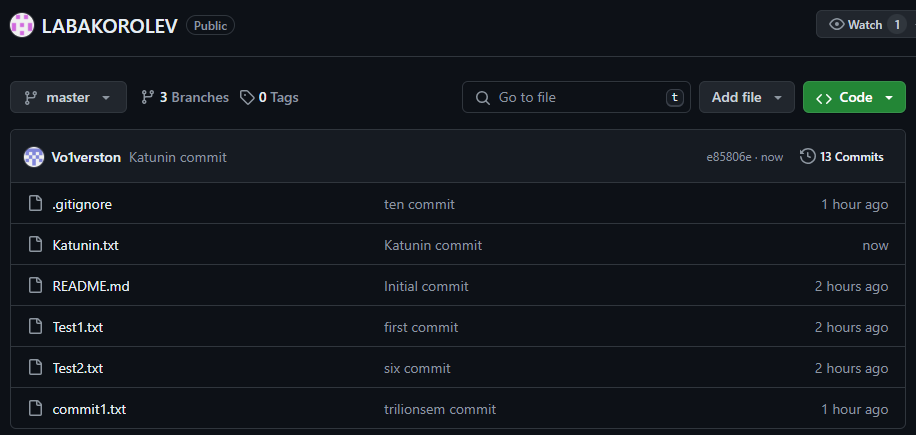
**3.** Создаём файл со своим ФИО и номером группы:

****

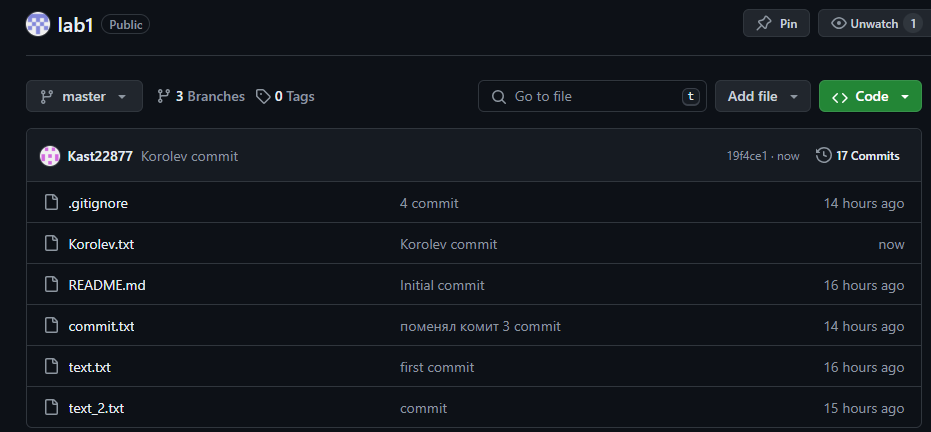
**4.** Оправляем файл:

****

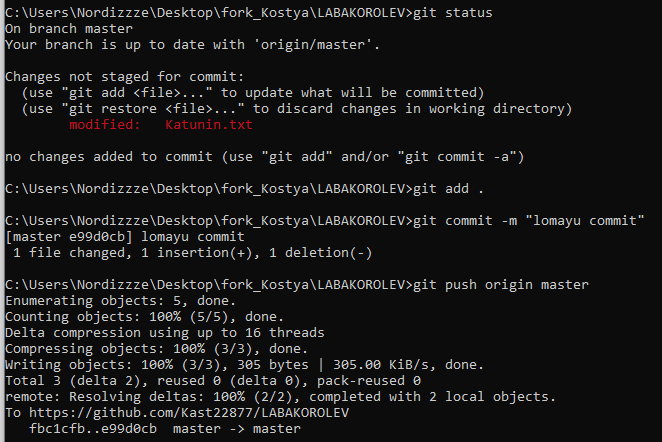
**5.** Файл появился в репозитории друга:

****

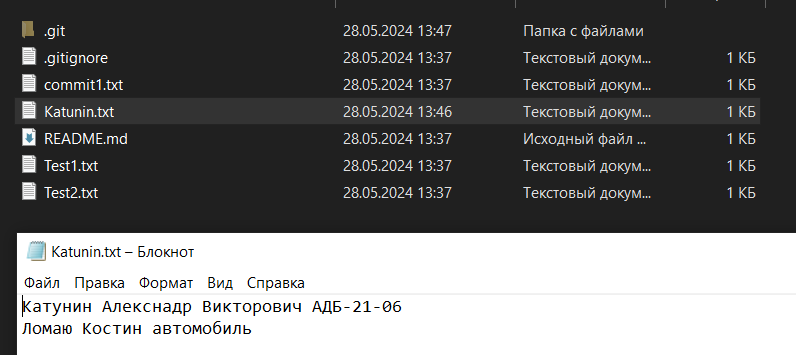
**6.** Файл друга появился в нашем репозитории:

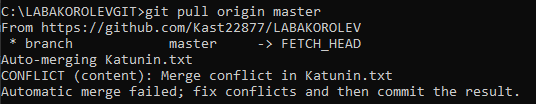
****

**7.** Попробуем создать конфликт:

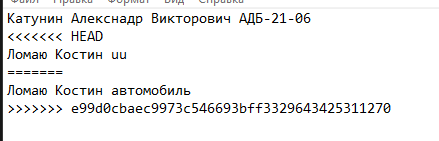
****

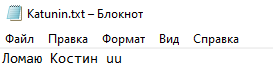
**8.** Открываем один файл и меняем его (друг тоже меняет и отправляет на сервер первее):

****

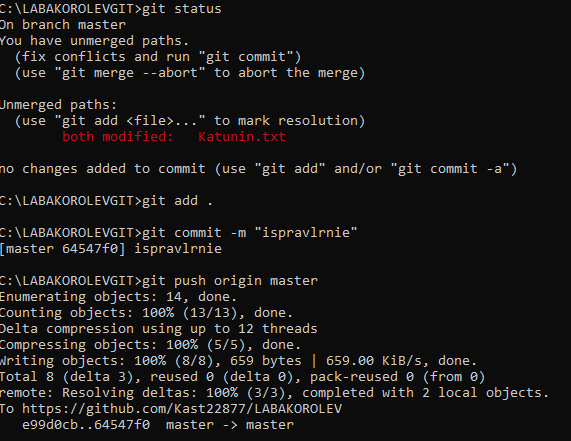
**9.** Получаем конфликт:

**10.** Разрешаем конфликт:

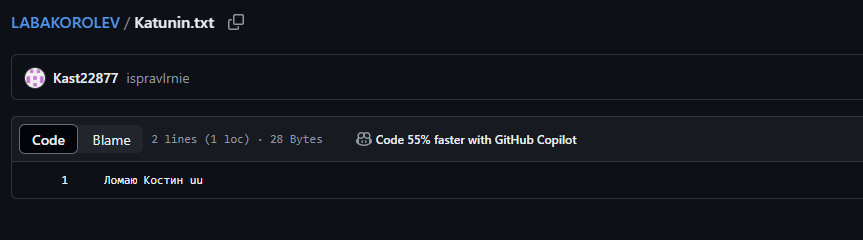
****

****

**11.** Пушим изменения:

****

**12.** Файл принял одно из состояний:

****

# **Вывод**

Приобрел навык работы с git и GitHub, поработал над общим репозиторием в команде.

# **Список литературы**

**Интернет-ресурсы:**

1. ЭБС «Университетская библиотека онлайн» издательства «Директ-Медиа»
2. Электронные издания научно-технической библиотеки, размещенные в разделе университета в ЭБС «Университетская библиотека онлайн»
3. Электронная образовательная среда в сети Интернет
4. Сайт университета в сети Интернет по адресу
5. Электронная библиотека научных публикаций «Российский индекс научного цитирования»
6. ЭБС «IPRBOOKS» группа компаний IPR MEDIA
7. Интернет-ресурс кафедры «Компьютерные системы управления» МГТУ «СТАНКИН»