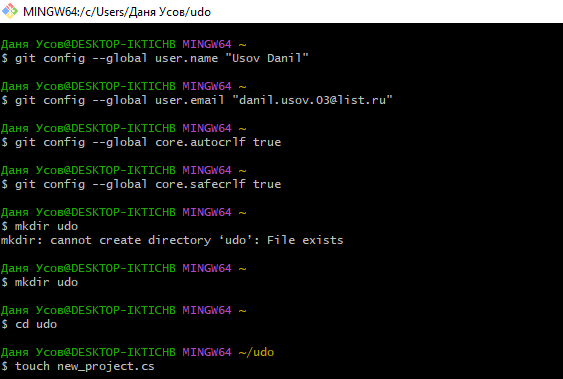
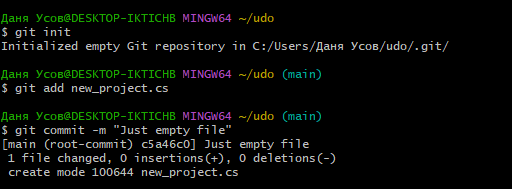
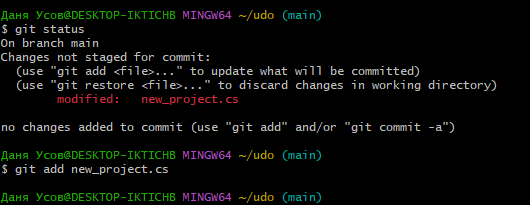
**Отчёт по лабораторной работе №3**

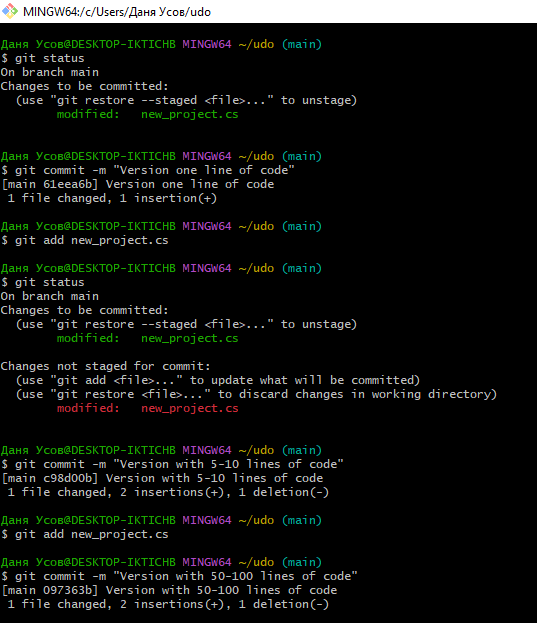
**Усов Данил БПИ22-01 Вариант-6**

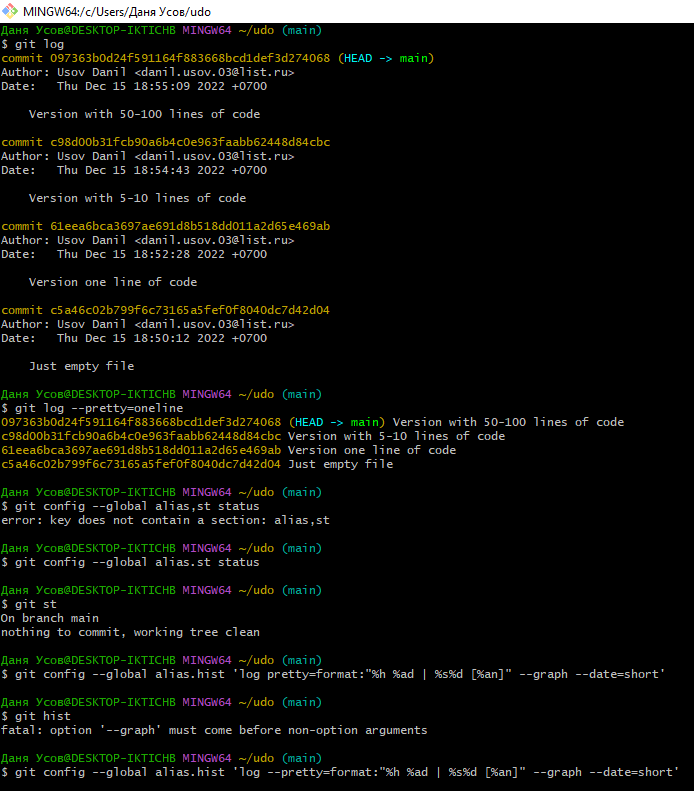
Первоначальная настройка, создание директории и файла

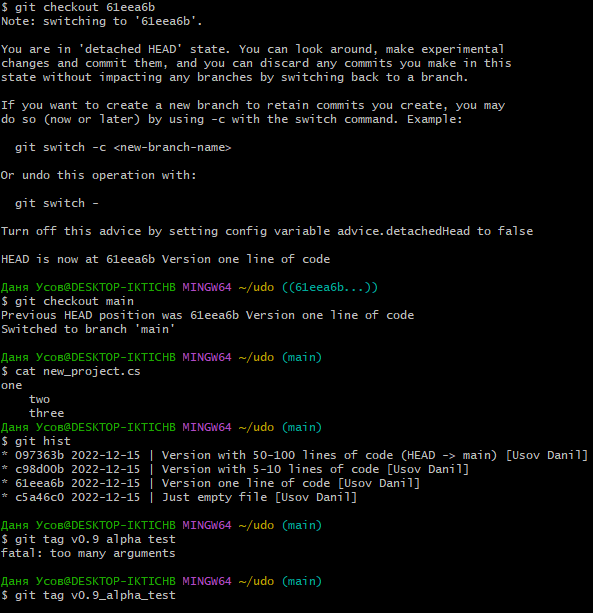
****

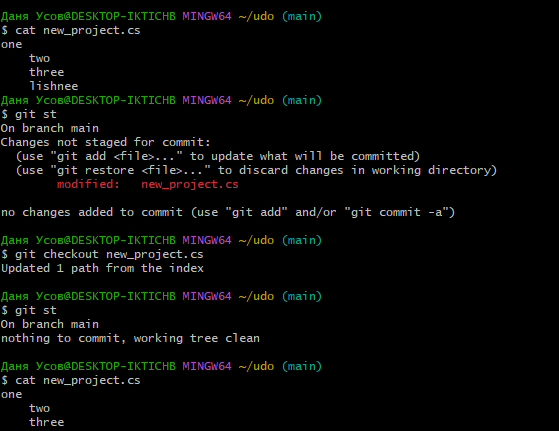
Создание репозитория, добавление туда файла, проверка состояния

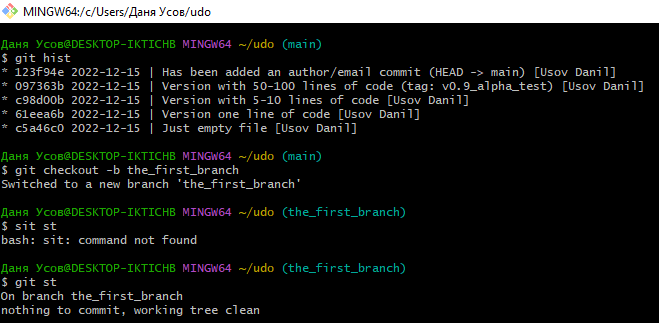
Проверка состояния после изменения файла, коммит измененного файла

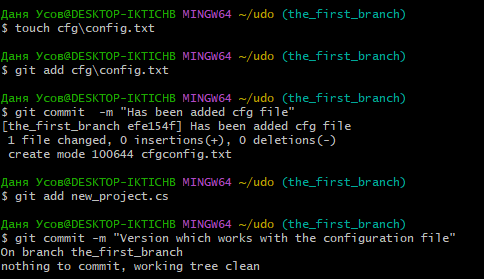


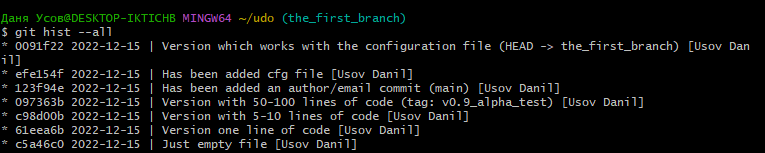
Просмотр изменений через стандартную команду и через созданный пользователем алиас

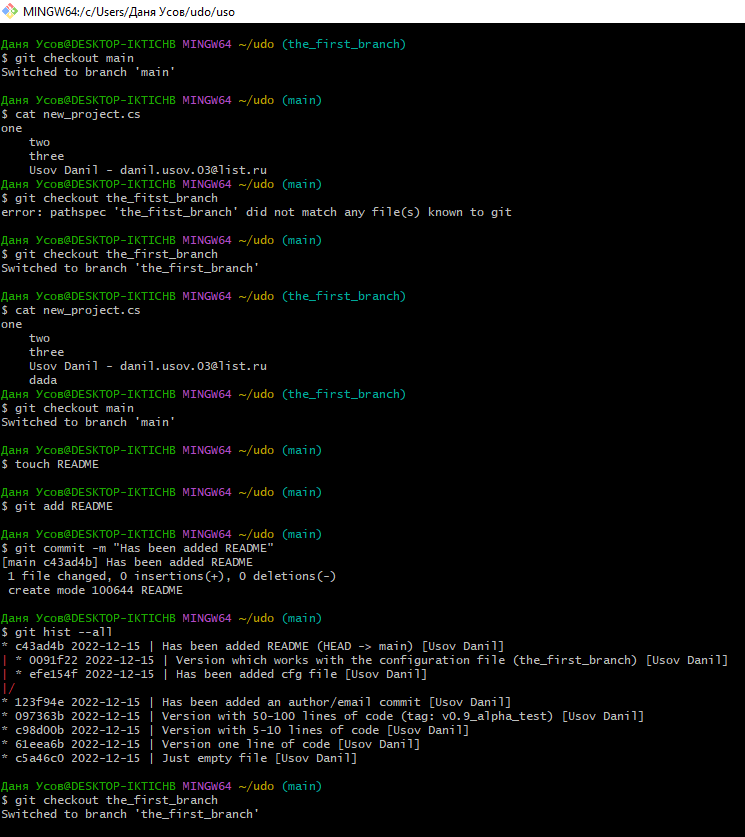
Возвращаемся к предыдущей версии файла, а затем снова к новейшей, добавление тэга

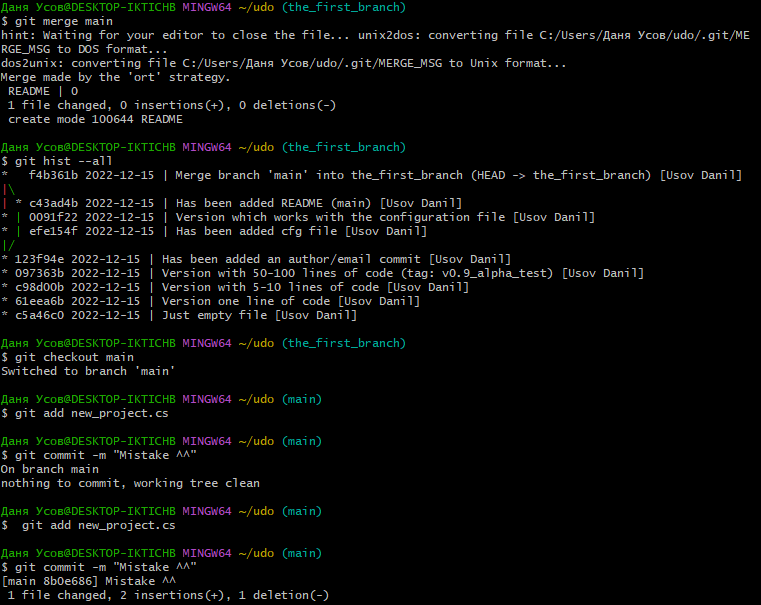
Убираем ненужную строку (коммита не было), откатываясь до последнего коммита

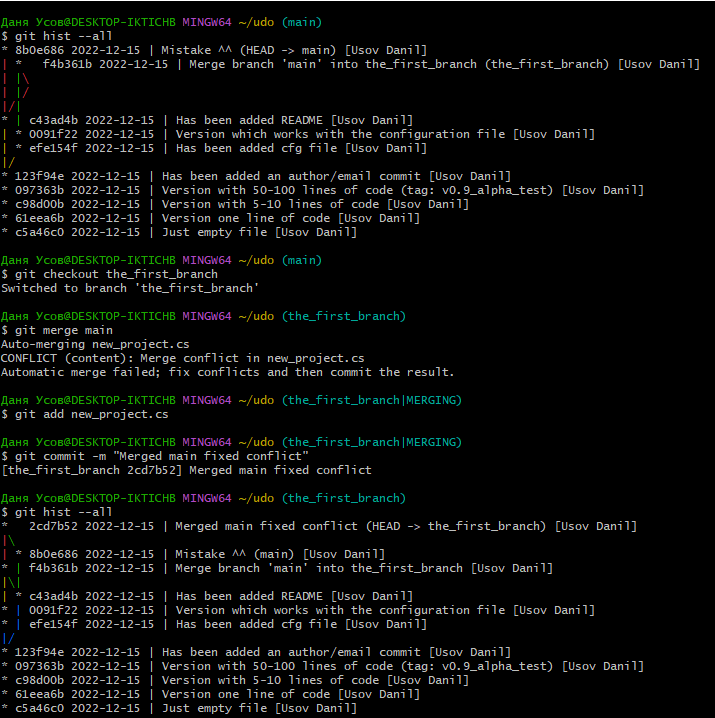
Создаём новую ветку, работаем с ней

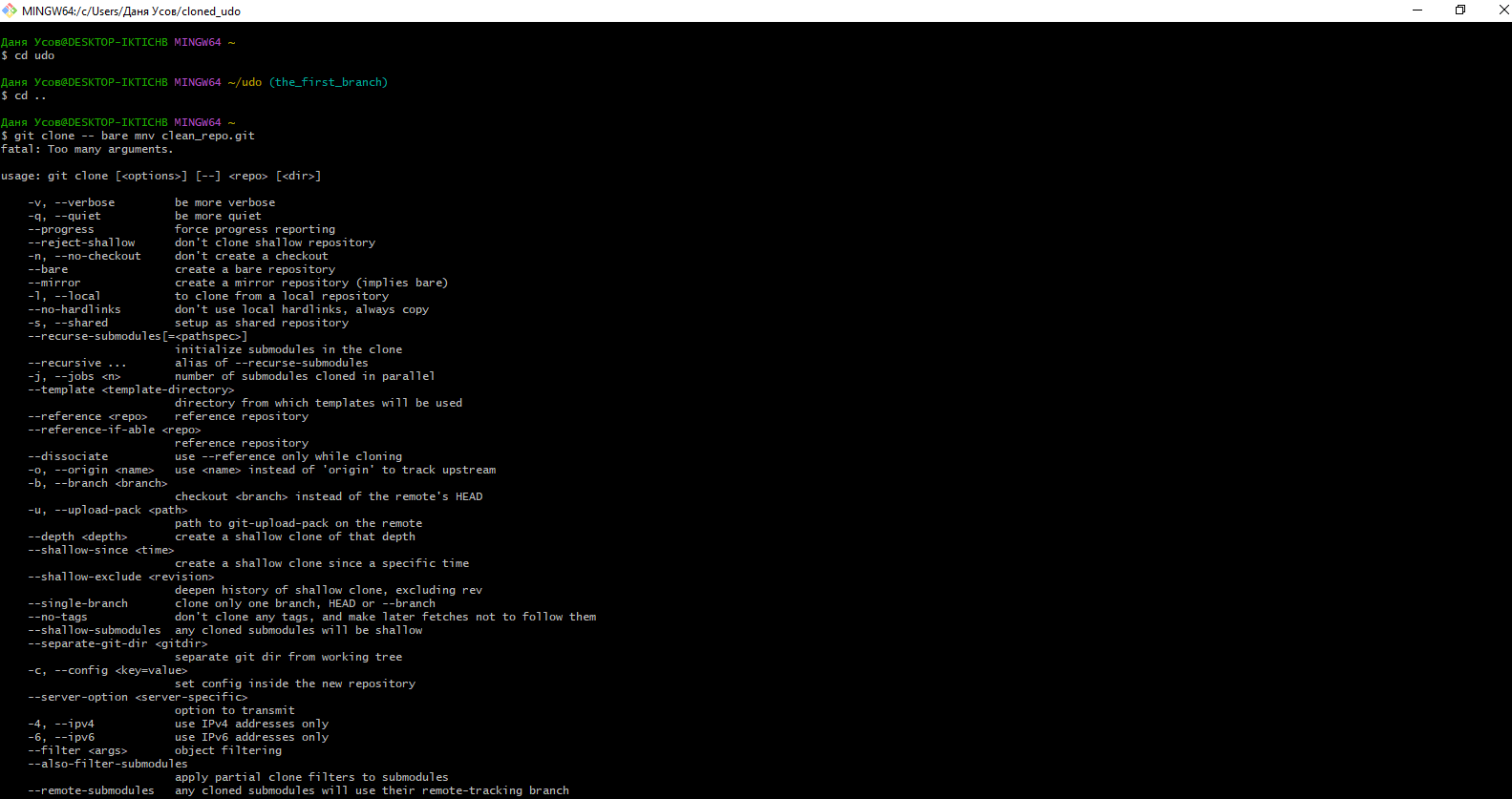




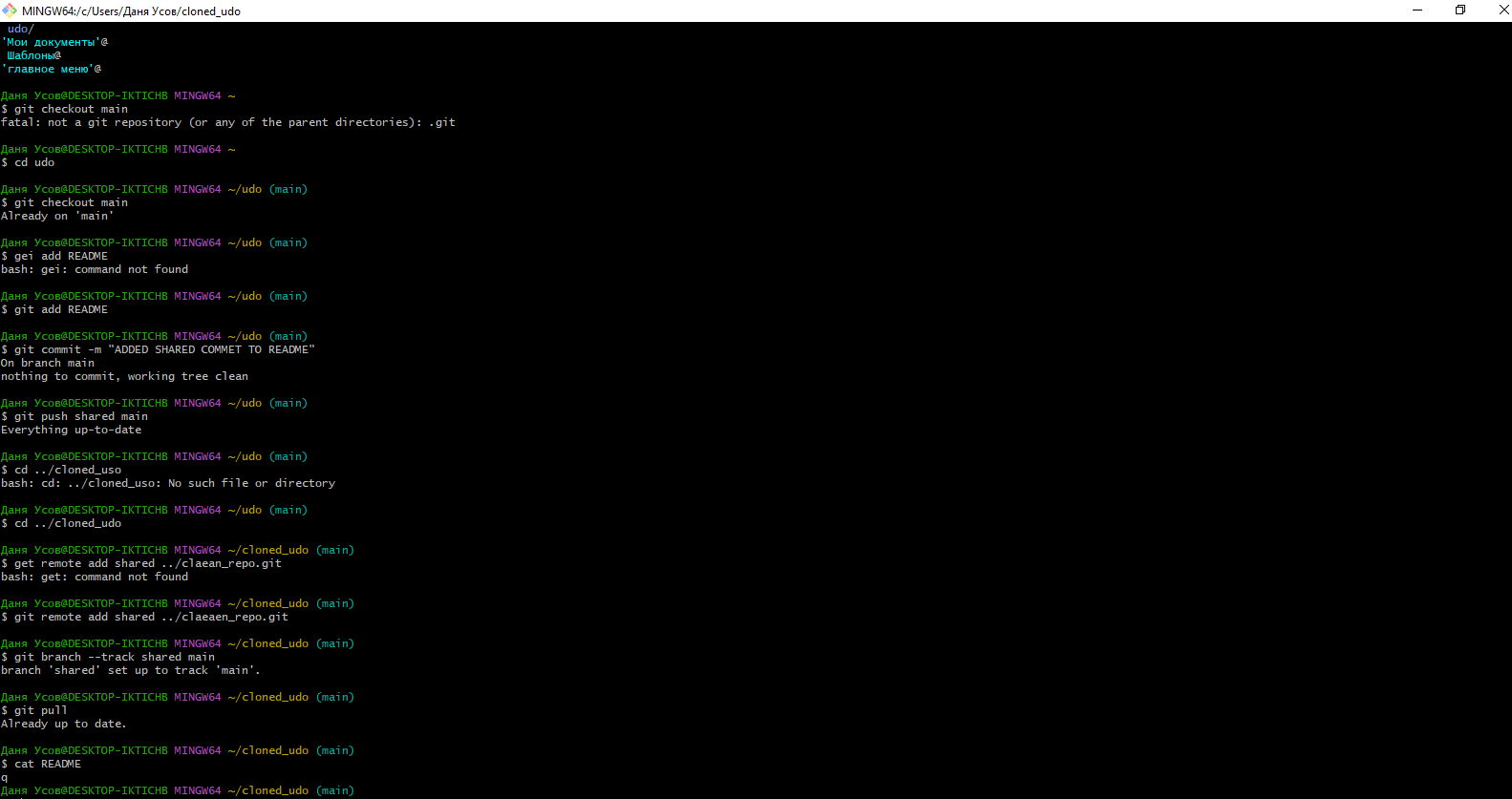


Слияние веток, создаем конфликт и решаем его

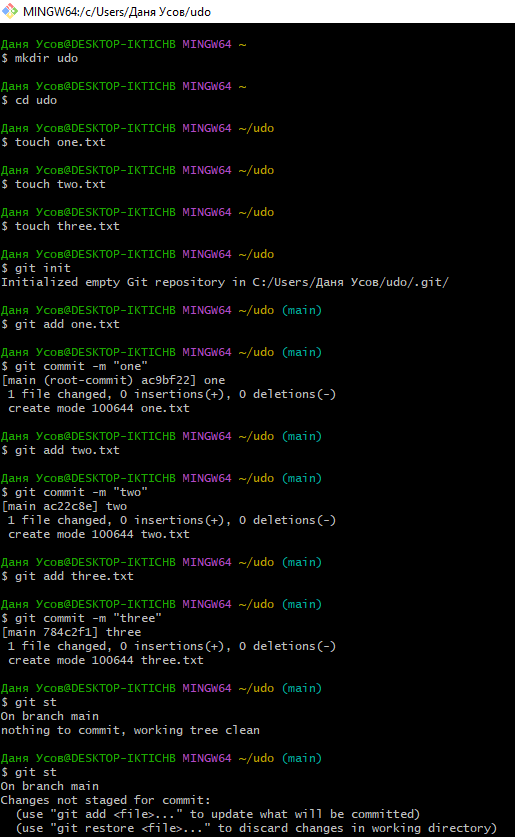
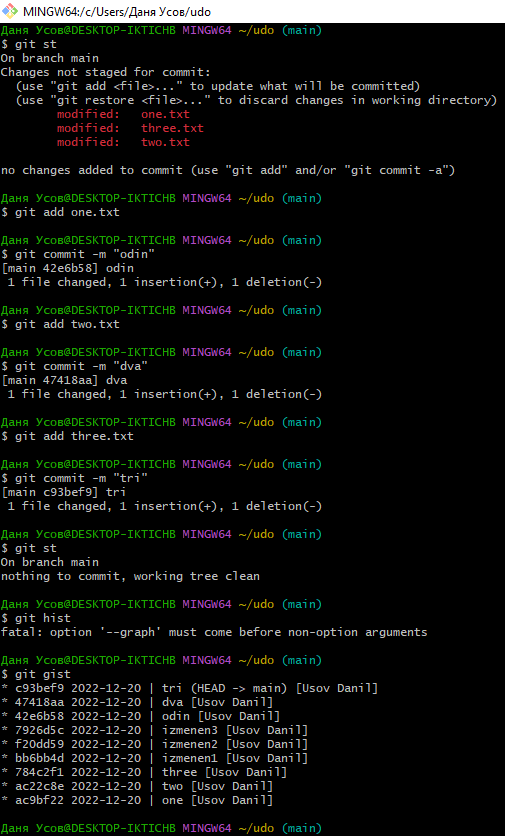
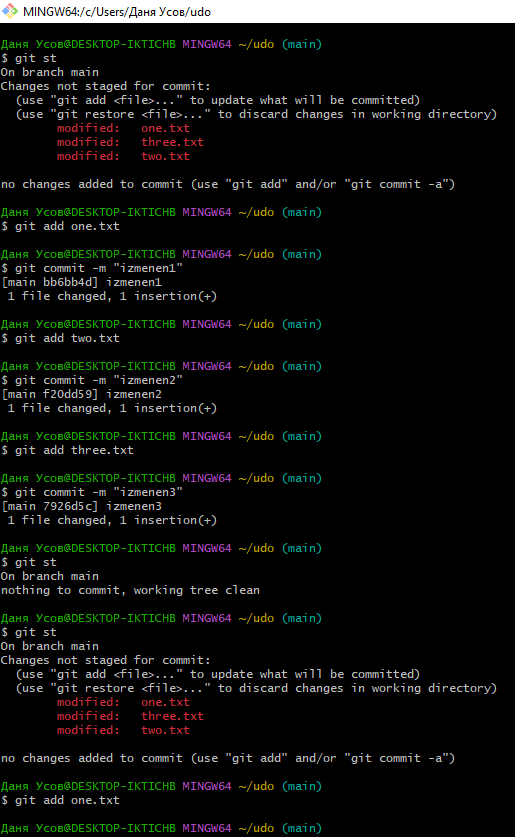


Создание чистого репозитория, работа с ним



Создание удалённого репозитория, работа с ним

**Задание 6 варианта:** Использовать в работе 3 разных файла, и количество версий каждого файла не должно быть меньше 3

**1**

**Ответы на контрольные вопросы:**

1. GIT – Система контроля версий, которая позволяет контролировать изменения в файлах, откатывать эти изменения, регистрировать, кто и когда внёс изменения и т.д.
2. Отслеживает изменения в файлах, предоставляет возможность создания новых веток и слияния существующих, проводит контроль доступа пользователей к проекту, определяет, кто, когда и какие внёс изменения
3. Коммит – операция отправки изменений в систему контроля версий, он выполняется, когда пользователь хочет добавить текущую обновленную версию файла в систему контроля версий.
4. С помощью команды git log или заданных вручную алиасов с аналогичным функционалом
5. С помощью команды git init
6. Архитектура деревьев (ветвления)
7. git checkout -b “название ветки”, для соединения git merge “ветка, с которой нужно соединиться”
8. stage это место, куда отправляется необходимый набор файлов, в которые были внесены изменения. Они находятся там до коммита.
9. С помощью команды git diff

Вывод: Я понял, что система контроля версий может быть очень полезной в личных проектах для возможности возврата к контрольным точкам при появлении непредвиденных ошибок, также можно сделать несколько путей развития своей программы с помощью веток. Ещё больше пользы эта технология может принести в командных проектах, которые создают несколько программистов, так как создавать структуру программы можно одновременно, далее сливая все наработки в одну ветку.