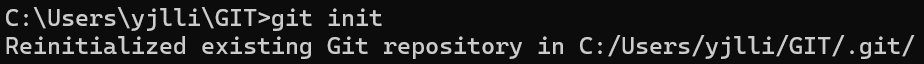
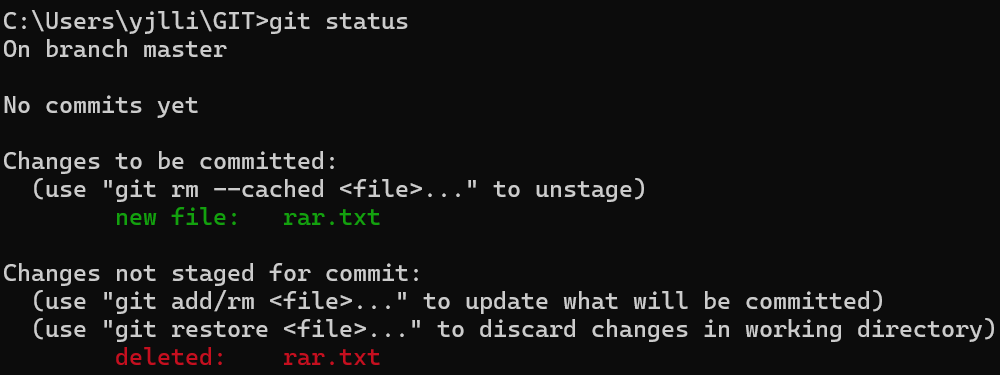
1. cоздать локальный репозиторий в текущей папке: git init



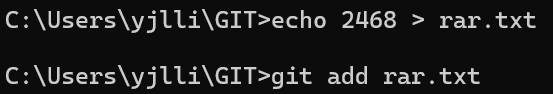
2. посмотреть статус текущего репозитория: git status



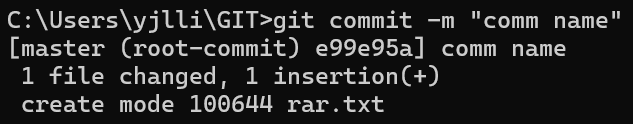
3. что такое ветка и какая ветка является обычно основной:

*Ветка в Git - это линия разработки,* параллельная версия репозитория. Ветки позволяют работать над отдельными функциями проекта, не влияя на основную версию. Закончив работу над новой фичей, можно объединить эту ветку с основной версией проекта*. Обычно основной веткой является master или main.*

4. добавить файл в контекст, который будет коммититься: git add [назв.]

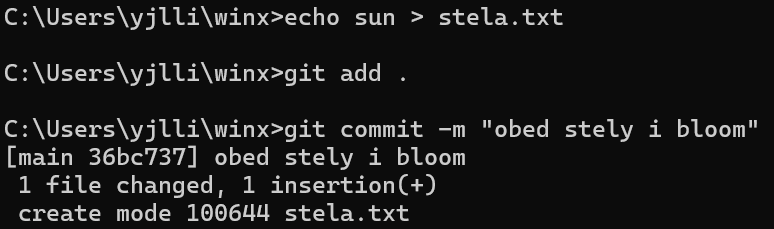


5. создать коммит на основе текущего контекста и указать для него комментарий: git commit -m “сообщение коммита”

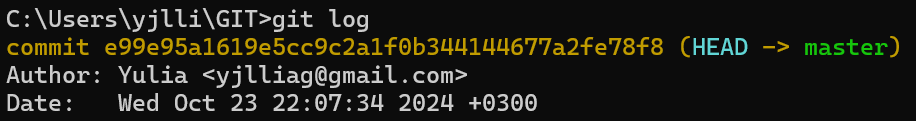


6. создать коммит, включающий изменения всех наблюдаемых файлов и указать для него комментарий: git add .

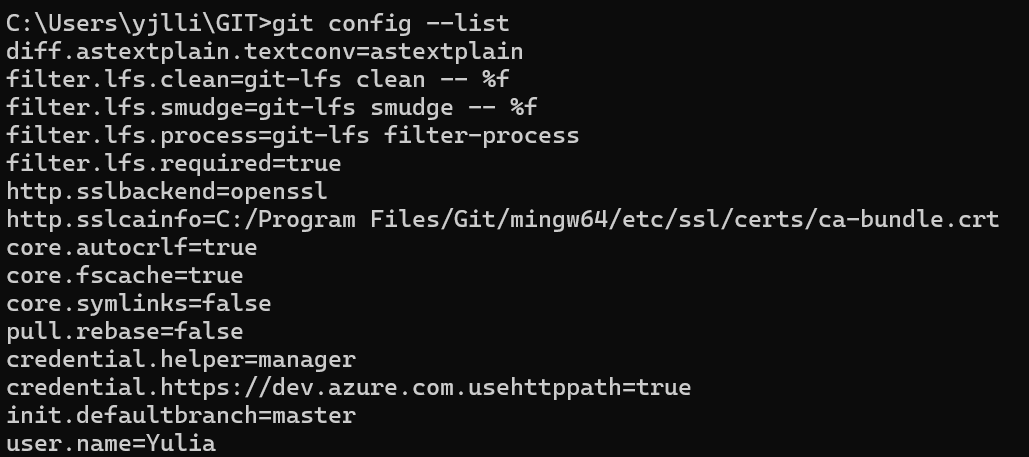
git commit -a -m "сообщение коммита"



7. посмотреть протокол(лог) коммитов: git log



8. посмотреть информацию о текущих настройках: git config --list

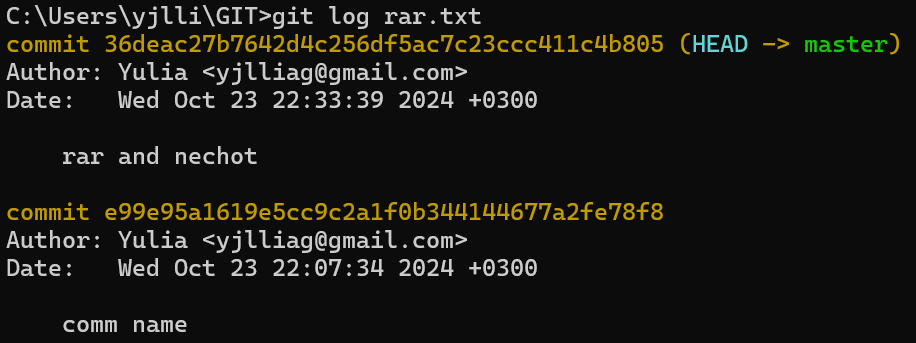


9. убрать файл из контекста: git reset (назв).\*



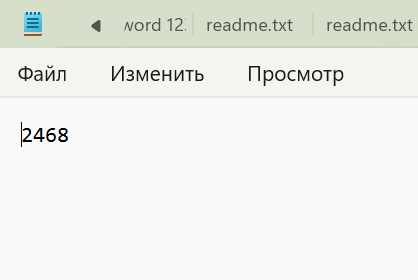
10. посмотреть изменения в файле по сравнению с последним коммитом:

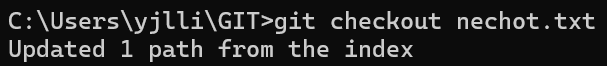
git log (назв).\*

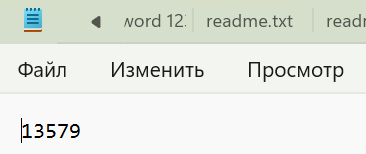


11. убрать изменения относительно последнего коммита из файла: git checkout наз.\*









12. добавить в контекст коммита все измененные и созданный файлы: git add .



13. изменить глобальные/локальные настройки:

*Глобальные git config --global user.name(email) “имя”*





*Локальные git config user.name(email) "имя"*

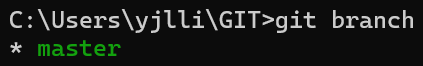




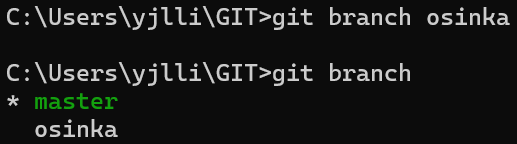
14. переписать имя пользователя: *git config --global user.name “имя”*



15. просмотреть существующие ветки: git branch



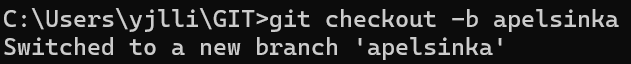
16. создать новую ветку: git branch название



17. переключиться на другую ветку: git checkout название



18. создать новую ветку и сразу же переключиться на неё: git checkout -b название



19. удалить ветку/удалить ветку, даже если она не примержена:

git branch -d название

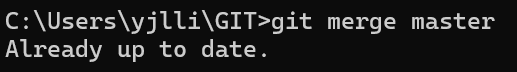


git branch -D название



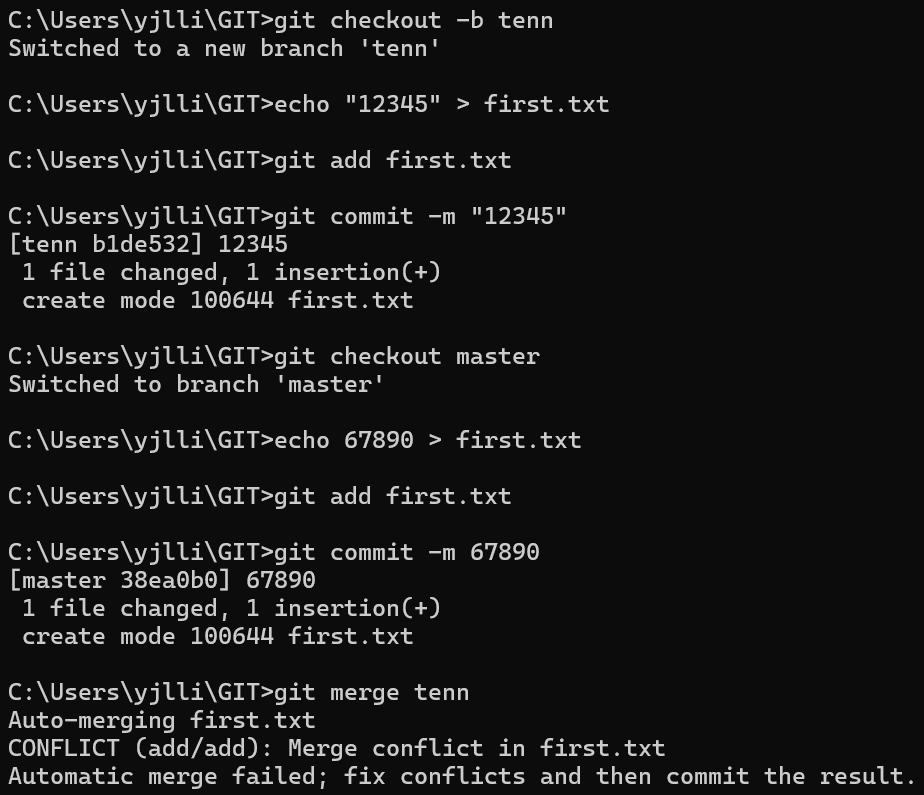
20. Примержить(переместить) изменения из указанной ветки в текущую:

git merge название ветки

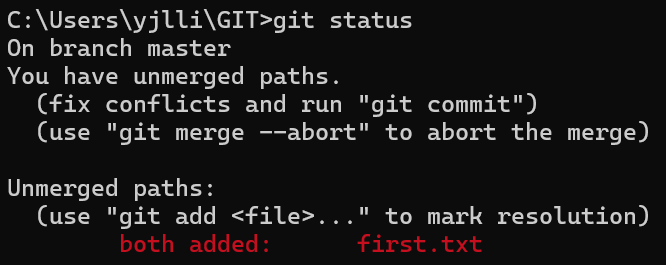


21. в каком случае могут появиться конфликты? сделать конфликт:

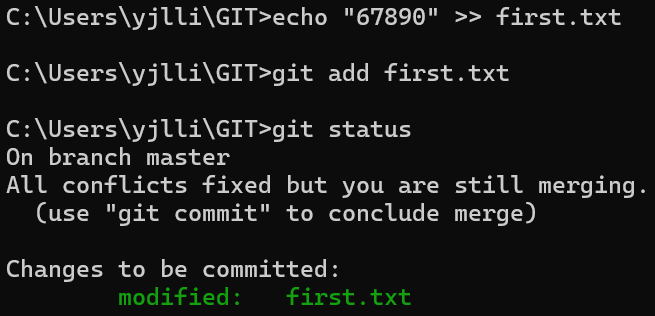
Конфликты в Git возникают, когда изменения в разных ветках затрагивают одну и ту же строку кода или файл, и Git не может автоматически объединить эти изменения.

**

22. как посмотреть в каких файлах конфликты: git status



23. как устранить конфликты: >> вместо >



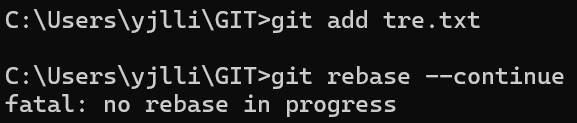
24. как переключиться на указанный коммит: git commit -m “”



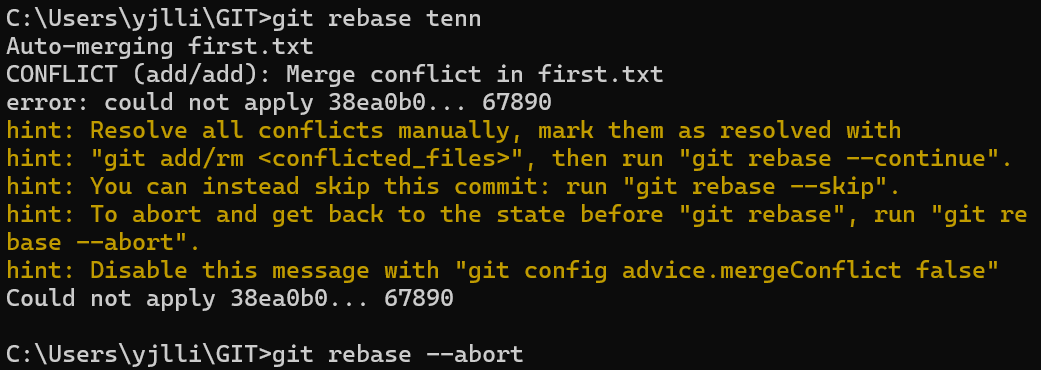
25. сделать ребазирование(объединение) текущей ветки: git rebase назв



26. устранение конфликтов во время ребазирования: git rebase --continue



27. отменить ребазирование во время конфликтов: git rebase --abort

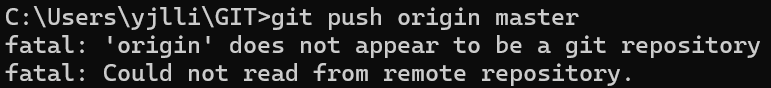


28. пропустить текущий конфликтный коммит и перейти к следующему:

git rebase --skip



29. отправить изменения из локального репозитория для указанной ветки в удалённый(дистанционный) репозиторий: git push origin назв



30. забрать изменения из репозитория, для которого были созданы удалённые ветки по умолчанию: git fetch origin/pull



31. забрать изменения удалённой ветки из репозитория по умолчанию, основной ветки: git fetch origin master/pull



32. создание копии репозитория: git clone https://github.com/имя/repo.git



33. переименовать последний коммит: git commit --amend -m “новое название”

34. переименовать не последний коммит: git rebase -i HEAD~3

35. скрыть изменения по сравнению с последним коммитом: git checkout -- файл