1. создать локальный репозиторий в текущей папке
   Git init
2. посмотреть статус текущего репозитория
   Git status
3. что такое ветка и какая ветка является обычно основной
   Master/ main
4. добавить файл в контекст, который будет коммититься
   Git add file
5. создать коммит на основе текущего контекста и указать для него комментарий
   Git commit -m “comment”
6. создать коммит, включающий изменения всех наблюдаемых файлов и указать для него комментарий
   Git commit -a -m “comment”
7. посмотреть протокол(лог) коммитов
   Git log
8. посмотреть информацию о текущих настройках
   Git config --list
9. убрать файл из контекста
   Git restore –staged file
10. посмотреть изменения в файле по сравнению с последним коммитом
    Git diff file
11. убрать изменения относительно последнего коммита из файла
    Git checkout file
12. добавить в контекст коммита все измененные и созданный файлы
    Git add –all / git add.
13. изменить глобальные/локальные настройки
    Git config --global
14. переписать имя пользователя
    Git config –global user.name {name}
15. просмотреть существующие ветки
    Git branch
16. создать новую ветку
    Git branch {name}
17. переключиться на другую ветку
    Git checkout {name} / git switch {name}
18. создать новую ветку и сразу же переключиться на неё
    git checkout -b {name}
19. удалить ветку/удалить ветку, даже если она не примержена
    Git branch -d {name}
20. примержить изменения из указанной ветки в текущую
    Git merge {name}
21. в каком случае могут появиться конфликты? сделать конфликт
    Конфликт в случае слияния веток с разными изменениями в одном файле
22. как посмотреть в каких файлах конфликты
    Git merge -> git status
23. как устранить конфликты
    устранить конфликт вручную изменив файл до удовлетворительного состояния
24. как переключиться на указанный коммит
    git checout {hash}
25. сделать ребазирование(rebase) текущей ветки
    git rebase {name}
26. устранение конфликтов во время ребазирования
    устранить конфликт → git add {file} → 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

30. забрать изменения из репозитория, для которого были созданы удалённые ветки по умолчанию

    git fetch → git merge / git pull

31. забрать изменения удалённой ветки из репозитория по умолчанию, основной ветки

    git pull origin {name}

32. создание копии репозитория

    git clone {name} {name}

33. переименовать последний коммит

    git commit --amend -m “comment”

34. переименовать не последний коммит

    git rebase -i HEAD~n

35. скрыть изменения по сравнению с последним коммитом

    git revert HEAD