1. 检查 git 版本git --version

2. 查看 git 相关命令git --help

3. 查看当前的 git 配置信息git config --list

4. 查看 git 用户名 或 邮箱git config user.name

5. 全局配置用户名(设置 git 使⽤者名称)git config --global user.name "username"

6. 设置 （配置）全局邮箱git config --global user.email "eamil@qq.com"

7.初始化 git 储存git init

8.需要提交的所有修改放到暂存区（Stage）

git add \* # 将工作区所有修改添加到暂存区

git add . # 将工作区所有修改添加到暂存区

git add <file-name> # 将指定文件添加到暂存区

git add \*.js # 提交所有 .js 格式文件

git add -f <file-name> # 强制添加 指定文件添加到暂存区

9.将暂存区的文件恢复到工作区

git reset <file-name> # 从暂存区恢复指定到工作区

git reset -- . # 从暂存区恢复所有文件到工作区

git reset --hard # 把暂存区的修改退回到工作区

10.查看工作区、暂存区的状态git status

11.移除暂存区的修改git rm --cached <file-name>

12.将缓存区的文件，提交到本地仓库（版本库 ）

git commit <file-name> ... "相关的记录信息" # 将缓存区的指定文件提交到本地仓库

git commit -m "相关的记录信息" # 将缓存区的所有文件提交到本地仓库

git commit -am '相关的记录信息' # 跳过暂存区域直接提交更新并且添加备注的信息

git commit --amend '相关的记录信息' # 使用一次新的commit，替代上一次提交，如果代码没有任何新变化，则用来修改上一次commit的提交记录信息

13.撤销 commit 提交

git revert HEAD # 撤销最近的一个提交(创建了一个撤销上次提交(HEAD)的新提交)

git revert HEAD^ # 撤销上上次的提交

14. 查看历史提交(commit)记录

git log # 查看历史commit记录

git log --oneline # 以简洁的一行显示，包含简洁哈希索引值

git log --pretty=oneline # 查看日志且并且显示版本

git log --stat # 显示每个commit中哪些文件被修改,分别添加或删除了多少行

16.查看分支合并图git log --graph

17.查看版本线图git log --oneline --graph

18.回到指定哈希值对应的版本

git reset --hard <Hash> # 回到指定 <Hash> 对应的版本

git reset --hard HEAD # 强制工作区、暂存区、本地库为当前HEAD指针所在的版本

19.git reset --hard HEAD~1 # 后退一个版本

20. 查看分支

git branch # 查看所有本地分支

git branch -r # 查看所有远程分支

git branch -a # 查看所有远程分支和本地分支

git branch --merged # 查看已经合并的分支

21.创建分支（依然停留在当前的分支）

git branch <branch-name> # 创建分支，依然停留在当前的分支

22. 切换分支

git checkout <branch-name> # 切换到指定分支，并更新工作区

git checkout - # 切换到上一个分支

23.git checkout -b <branch-name> # 创建一个新的分支，并切换到这个新建的分支上

24.合并分支（合并某一个分支到当前分支）

git merge <branch-name> # 合并<branch-name>分支到当前分支

25.删除分支

git branch -d <branch-name> # 只能删除已经被当前分支合并的分支

git branch -D <branch-name> # 强制删除分支

26.删除远程分支

git push origin --delete <remote-branch-name>

# 注：<remote-branch-name> 远程分支名

27.克隆远程仓库（从远程仓库拉取代码）git clone <url>

28.本地库与远程库进行关联

git remote add origin <url>

29.查看远程仓库地址别名git remote -v

30.新建远程仓库地址别名git remote add <alias> <url>

31.删除本地仓库中的远程仓库别名

git remote rm <alias>

32.重命名远程仓库地址别名

33.把远程库的修改拉取到本地

git fetch <alias/url> <remote-branch-name> # 抓取远程仓库的指定分支到本地，但没有合并

git merge <alias-branch-name> # 将抓取下来的远程的分支，跟当前所在分支进行合并

git pull <alias/url> <remote-branch-name> # 拉取到本地，并且与当前所在的分支进行合并

34.将本地的分支推送到远程仓库

git push <alias/url> <branch-name> # 将本地的每个分支推送到远程仓库

git push <alias/url> --force # 强行推送 当前分支到远程仓库，即使有冲突

git push <alias/url> --all # 推送所有本地分支到远程仓库