git config --global user.email “your_email” →設定本機的 git email

git config --global user.name “your_name” →設定本機的 git name

git clone SSH →將整個 repositoy 複製到本機

git log →顯示 HEAD 與每個版本的內容(預設顯示最近四個)

[--oneline]：顯示所有版本的簡短 commit number & commit message

git status →顯示 staging area 狀態(新增或修改的檔案，有無被 git add 的都會列出)

git add +檔案：將檔案新增到 status area

+ . ：將所有修改的檔案新增到 status area

git reset +檔案：將檔案從 status area 移除

vi .gitignore →記錄到裡的檔案或目錄不會被 git 偵測到，也不會加進 staging area

git commit -m “commit_message” →將目前 staging area 中的檔案新增為新的版本

git checkout commit_number/分支 →依據 commit number(可簡短)切換版本或分支

main →切換至 main(最新)版本

HEAD~ →切換至目前位置的上一個版本

HEAD~n →切換至目前位置的上 n 個版本

-b 新分支 →創建新分支，並跳至新分支位置

git push origin main →上傳目前 repasitoy 的分支(main)到 GitHub 的自己帳號(origin)

: branch_name→刪除遠端 repositoy 的分支(用本地端空分支更新遠端 branch_name)

git branch →顯示目前分支狀態

new_branch ：建立一個新分支 new branch

-d branch_name ：刪除分支(branch_name)

git merge 其他分支 →合併其他分支至當前分支，會出現一個文字檔讓你輸入 commit message

git revert HEAD →新增一個 commit 來還原上一個 commit(不會刪除目前 commit)

git reset HEAD~ →回到上一個 commit，並且刪除目前 commit，只會刪除節點，不會刪除檔案

--mixed ：**預設狀況**，拆除 commit，但既有檔案不變(依然在 working directory)

--soft ：拆除 commit 後將檔案留在 staging area

會修改歷史 --hard ：拆除 commit 後復原到指定 commit 狀態(會影響檔案，不會有暫時節點)

git reflog →查看 git 的 log(各種 git 操作紀錄)，會列出操作紀錄編號

git reset HEAD@{1}　→回到 HEAD@{1}(看操作紀錄)這個操作的狀態(被刪掉或修改的 commit 會回復)

git restore 檔案 / .　　→救回檔案，「.」代表所有檔案

git rebase branch / main　　→把當前分支的基底狀態改為 branch / main，把當前分支合併至指定分支

commit number：sha1 加密演算法產生

Git flow：master、hotfix、release、develop、feature。