git add	git add index.html index2.html index.html 과 index2.html 두 파일을 staging area 에 담음
	git add *.* 현재 폴더에 있는 모든 파일을 staging area 에 담음
	git add *.html .html 확장자를 가진 파일을 staging area 에 담음
git commit	git commit –m "commit msg" : 메시지와 함께 커밋
	git commit -am "commit msg" : staging area에 추가하고 commit msg를 커밋 메시지로 입력
	git commit : 어떤 메시지를 작성할 지 창이 생기면서 길게 입력 가능
	git commit --amend : 마지막 Commit 메시지 수정
git status	작업 디렉토리에 변경된 파일 보기
git clean -f	untracked 파일을 모두 지울 수 있음
git log	git log --pretty=oneline --graph : branch & merge 내역 시각화
	git log --author=Brandon : 저자 이름을 검색하여 보여줌
	git log --oneline : 커밋을 커밋 체크섬과 메시지를 한 줄 표시
git diff	커밋들을 비교하거나 commit과 working tree 변경사항 표시
	git diff

	: 워킹 디렉터리와 Staging area 비교
	git diff --cached : Commit 내용과 Staging area 비교
	git diff commit1 commit2 : Commit 과 다른 Commit 비교
git checkout	git checkout <file명> : Modified 상태를 unmodified로 변경(WD)
	git checkout issue2 : 현재 branch에서 issue2 branch로 이동
	git checkout –b issue1 : issue1 branch를 만들고 issue1 branch로 이동
	git checkout master : 현재 branch에서 master로 이동
git reset	특정 commit으로 되돌아 갈 수 있으며 특정 commit 이후의 버전들은 history에서 삭제됨
	git reset --soft HEAD~ : HEAD이동, 해당파일 staged, WD 파일 보존 Commit하면 원래 상태로 복원 가능
	git reset HEAD~2 : 뒤에 파일명이 없으면 add한 파일 전체를 취소 : 파일을 Unstaged 상태로 변경
	git reset --mixed HEAD~ : default, HEAD이동, 해당파일 unstaged, WD파일 보존
	git reset --hard HEAD~ : HEAD이동 해당파일 unstaged, WD 파일 변경사항 삭제

git rm	git rm sample.txt : 원격저장소 + 로컬 저장소 모두 삭제
	git rm --cached sample.txt : 원격저장소 삭제, 로컬 저장소 삭제하지 않음
git clone	git clone "Remote repo URL" : URL 의 내용을 clone
	git clone "Remote repo URL" abc : 디렉터리의 이름을 본래 디렉터리 이름이 아닌 다른 디렉터리 이름으로 clone
	git clone -b Lab1 "Remote repo URL" abc : Lab1 브랜치를 abc디렉터리로 clone
git remote	git remote : 현재 원격과 어떻게 되어 있는지 확인하는 명령어
	git remote -v : 어떤 링크와 연결되어 있는지 확인하는 명령어
	git remote add origin "remote repo URL" : remote 연결
	git remote remove origin : remote와 연결되어 있는 것을 제거
git pull	원격 저장소의 데이터를 로컬 저장소에 가져와 병합 git pull origin master
git push	로컬 저장소의 데이터를 원격 저장소로 밀어넣음 git push origin master
git fetch	: 원격 저장소의 데이터를 로컬에 가져오기만 하기
git fetch와	fetch : 원격 저장소의 데이터를 로컬에 가져오기만 하기

git pull 차이	pull : 원격 저장소의 내용을 가져와 자동으로 병합 작업을 실행
gitignore	git 저장소에서 관리할 필요가 없는 파일이나 폴더 지정 : ex) 0컴파일된 파일, 실행시간에 생성된 파일, Hidden file, 개인적인 파일, 실행파일 디렉터리
git branch	이미 만들어진 버전(master)을 다른 방향으로 개발을 할 때, 본 ver(master)은 유지하고 복사 본(branch)을 만들어 이어 나갈 수 있다.
	git branch –h : branch의 옵션을 보여주는 Help 기능
	git branch -a : 현재 repository
	git branch : 현재 local에 어떤 branch들이 있는지 보여줌
	git branch issue1 : issue1이라는 새로운 branch를 만듬
	git branch –d issue1 : issue1이라는 branch를 삭제함
	git branch issue2 : issue2 branch를 만듬
	git branch –D issue1 : branch 강제삭제
stash	현재 작업중인 것을 저장하고자 할 때 사용하는 명령어 완료되지 않은 작업을 commit하지 않고 잠시 저장하여 나중에 필요할 때 작업할 수 있도록 보관해 두는 작업 워킹 디렉토리에서 수정한 파일들만 저장
	git stash : 현재 작업을 저장

	git stash save : 현재 작업을 저장 (=git stash)
	git stash list : 각 branch별로 저장된 상태를 목록화해서 나타냄
	git stash pop : 마지막 저장 정보를 꺼냄 : 동시에 삭제됨
	git stash apply : 마지막 저장 정보를 꺼냄 : 단 삭제되지 않음
	git stash drop : stash에 저장된 이력을 삭제함
	git stash clear : list를 모두 지움
merge	branch를 병합하는 명령어
	git merge TopicA : master branch에 TopicA의 branch에 있는 변경사항과 이력을 master에 병합 **: Fast-forward** merge
	git merge –no -f TopicA : master branch에 TopicA의 branch에 있는 변경사항과 이력을 master에 병합 : **No fast-forward** merge
merge conflict	각각의 브랜치에서 변경한 내용이 myfile.txt의 같은 행에 포함되어 있기 때문에 발생 충돌이 있는 부분에 Git이 자동으로 충돌 정보를 포함하여 파일 내용을 변경

merge conflict 해결방법	1. 충돌이 일어난 부분을 확인해서 직접 수정 2. Git add 3. Git commit
revert	commit된 스냅샷을 취소하는 명령어 : commit이력에서 해당 commit에 의해 변경된 내용을 취소하는 방법을 찾아내고 그 결과를 새로운 commit으로 추가
revert vs reset	revert : commit 이력에서 해당 commit에 의해 변경된 내용을 취소하는 방법을 찾아내고 그 결과를 새로운 commit으로 추가
	reset : 특정 commit으로 되돌아갈 수 있으며 특정 commit이후의 버전들은 히스토리에서 삭제됨

[2020 OSS] – git 명령어 정리