1. git 설정하기

Terminal 에서 버전을 관리할 디렉토리로 들어가 git config 커맨드를 입력해서 유저이름과 이메일을 추가한다.
유저네임과 이메일은 반드시 깃허브의 계정정보와 맞을 필요는 없음.

git config —global user.name “username” // 유저 네임 추가
git config —global user.email “user email” // 유저 이메일 추가
git config —list // 현재설정된 유저네임과 유저 이메일을 확인 가능

```
Yeoui-MacBook-Pro:gitsample Hun$ git config --global user.name yeohunjeon
Yeoui-MacBook-Pro:gitsample Hun$ git config --global user.email 21500630@handong
Yeoui-MacBook-Pro:gitsample Hun$ git config --list
credential.helper=osxkeychain
user.name=yeohunjeon
user.email=21500630@handong.edu
```

2. 저장소 만들기

2-1. 깃허브에서 만들기
깃허브에서 새로운 repository 를 만들거나 원래 있는 repository에서 파일을 수정할 컴퓨터로 clone을 한다.

git clone “repository URL” // 레파지토리에 있는 데이타들을 지정된 디렉토리로 복사

2-2. Terminal에서 만들기
Terminal 에서 버전관리에 이용할 디렉토리를 새로 만들거나 사용할 디렉토리로 들어가 init 커맨드를 입력한다.

git init // 이 디렉토리를 깃 레파지토리로 사용하겠다는 선언. .git 폴더가 생성됨
git remote add origin “repository URL” // 레파지토리로 이용할 폴더를 깃허브의 레파지토리와 연결한다.

2-3 Repository 연결 끊기

.git 폴더를 삭제해야함.
rm -rf .git // 폴더이기 때문에 r옵션과 f 옵션을 추가해서 삭제해야함.
- r : 리커시브 옵션, 디렉토리의 최하위 디렉토리부터 삭제
- f : 포스 옵션, 존재하지 않는 파일이 있더라도 무시하고 계속 진행

3. git 상태

git status : 이 명령어를 입력하면 현재 저장소의 정보를 알 수 있다. 변경된 파일이 있으면 변경되었다고 알려주고, add는 되었지만 commit 이나 push가 되지 않은 파일들도 알려준다.

git log : 이 명령어를 입력하면 현재 저장소의 git 히스토리를 보여준다.

4. 저장소 업데이트

git clone “원격저장소 repo URL” : 원격 저장소에 있는 데이타를 사용자의 컴퓨터에 있는 로컬 저장소로 복제시킨다.

git remote -v : 현재 컴퓨터의 로컬 저장소에 연결되어 있는 워격저장소의 주소를 보여준다.

```
Yeoui-MacBook-Pro:2019-1-OSS Hun$ git remote -v
origin  https://github.com/jeonyeohun/2019-1-OSS.git (fetch)
origin  https://github.com/jeonyeohun/2019-1-OSS.git (push)
```

git remote add origin “repo’s URL” : 현재 디렉토리를 입력한 원격저장소에 origin 이라는 이름으로 연결한다.

git push origin master: 로컬 저장소에서 변경해서 commit한 내용을 원격저장소에 업데이트한다.

git pull origin master: 원격 저장소에서 변경된 내용을 로컬 저장소로 업데이트 한다.

git fetch : 원격 저장소에서 내용을 로컬 저장소로 가져온다. pull과의 차이점은 fetch 현재 작업중인 내용을 변경(merge)하여 가져오지는 않는다.

git branch 이름: 새로운 브랜치를 만든다. git branch 다음에 브랜치 이름을 설정하면 설정한 브랜치로 만들어준다. 디폴트는 branch

git branch -d 이름: 해당하는 브랜치를 삭제한다.

git checkout 브랜치이름 : 해당하는 브랜치로 이동하여 작업을 수행한다.

git checkout -b 이름 : 입력된 이름으로 브랜치를 새로 만들고 그 브랜치에서 작업을 시작한다.