1. 깃허브가 뭐하는 친구일까?

- 버전관리에 정말정말 용이한 프로그램!
- 누가 어느 코드를 언제 고쳤는지까지 세세하게 다 나온다.
- 개발자들끼리 협업할 때 필수템

2. git bash 기본 사용법

git bash는 Linux 터미널과 거의 비슷한 환경을 제공해주는 shell이다. (짱편함)

- git bash 설치

- git clone [리포지토리 주소] (현재 디렉토리에 주소에 있는 리포지토리 복사)
- git branch [브랜치 이름] (브랜치 생성, 브랜치 이름이 없다면 현재 깃에 있는 브랜치 목록을 보여줌)
- git checkout [브랜치 이름] (선택한 브랜치로 변경)

- git add [디렉토리] (설정한 디렉토리 내에서 변경된 모든 점을 업데이트해준다.)
- git commit (add를 통해 업데이트된 내용들을 적용시킨다. commit 메시지가 없다면 commit은 취소되니 뭐라도 적자.)
=> 보통 커밋 메시지는 동사로 시작함. (ex. 'Fix' line #233~235, 'Upload' solution for #1234, ...)

- git push (git --set-upstream origin 뭐시기 나오는 설정들 입력해주고, 깃헙 아이디 비번 입력하면 자기 아이디로 변경사항을 깃헙에 저장 가능)

(혹시라도 user name이나 user email치라는거 나오면 그건 어차피 깃허브에서 쓸 거 아니니까 아무거나 적어도 됨)
이 정도만 알아도 큰 지장은 없겠지만... 호오오오오오오옥시 깃이 뭐하는 친구인지 저어어엉말 궁금하다면 무리하지 말고 시간이 남아돌때
https://missing.csail.mit.edu/2020/version-control/ <= 여기 홈페이지 들어가서 보시길. 따끈따끈한 mit 2019 겨울학기 보충수업~~

3. 깃허브 리포지토리 사용하는 팁

- branch는 버전마다 하나씩 있어야 하긴 하지만.. 적어도 마스터 브랜치로 전부 때려박진 말자
- 같은 문제를 풀었으면 같은 폴더 안에 넣어놓자.
    너무 흩어져있어서 누가 무슨 문제를 풀었는지 모르겠다.
    1234_chayhyeon.c 이런식으로 파일 형식을 통일하는 것도 방법.
    ps_ 이거 앞에 붙인건 GAS 스터디에서 알고리즘 말고 다른 것도 하려고 올린거라 굳이

붙일 필요는 없다.

여튼 통일성이 중요하지! 스터디 이름도 지어보고, 이런 규칙들도 몇 가지 정해보자.

- 추가 질문

1. branch는 뭔가요?

원본을 손상시키지 않으면서 자신의 작업을 할 수 있도록 도와주는 친구랍니다.

처음 repository를 만들면 master branch로 시작을 하게 되는데요, 이 master branch는 정말 금고 속에 소중히 다뤄야 하는 존재랍니다! 자신의 branch에서 일하다가 master branch로 merge하는 것을 일종의 ctrl+s처럼 생각할 수 있는데요, 행여나 master branch에서 작업 도중 심각한 버그가 발생한다면 항상 ctrl+s를 해와서 되돌릴 수 없는 강을 건넌 것이나 마찬가지겠죠?

그래서 master branch가 아니라 version 1.0.0, hotfix 1.0.3등의 이름을 가진 브랜치로 업데이트를 할 수 있는 것입니다! 사실상 코드가 따로따로 독립되어있고, 짧은 코드들만으로 구성된 이 스터디 repository는 master branch에만 저장해도 큰 손색이 없지만, 추후에 개발 관련 직종에서 종사할 때는 branch 관리까지 필수로 해줘야 한답니다!

2. git pull과 pull request의 차이점이 뭔가요?

git pull은 git hub에 있는 repository로부터 저의 local repository를 업데이트 시켜주는 역할을 한답니다!

pull request는 예시를 들어 설명하겠습니다. A라는 브랜치를 이용하여 local repository에서 작업을 완료했다고 합시다. 그러면 A에서의 변경사항이 모두 정상적이라면, master로 merge를 해도 되겠죠! 스케일을 조금 더 키워서, 이 local repository에서 git hub에 있는 repository로 저장을 한다고 합시다. 제가 git hub에게 업데이트한 내용을 주는 것은 “push”지만, git hub repository의 입장에서는 받는 것이 되니까 “pull”이겠지요? 그래서 이렇게 업데이트를 해도 되느냐고 다른 contributers에게 요청하는 것이 pull request랍니다.

짧은 시간이어서 내용이 많이 부실했지만 들어주셔서 정말 감사했습니다!