깃은 "버전 관리 프로그램" 으로
내가 수정한 것들을 각각의 버전으로 관리한다
그 버전을 나누는 기준은 일단 로컬 깃에서는 commit 기점으로 지정한다.

**commit**

사전명으로 "약속하다" commition 많이 쓰잖아여
그래서 이 소스의 변경사항
혹은 이 폴더의 변셩사항 등등을 반영하겠다는 명령어

commit을 하면 내 깃에서 지금 내 소스의 최신 버전이 commit된 그 버전이 된다.
파일 단위로도, 폴더 단위 등 아주 작은 변화도 커밋으로 버전 관리가 된다.
되돌릴 수 있다. -> 버전으로 관리되니까.

**add**
commit을 하는 파일 or 폴더 단위 별로 지정해서 commit 직전
commit할 파일 목록에 넣는다
이 리스트에 add 된 애들만 commit 한다는 지정

위의 두 명령어는 깃 로컬에서 이루어 지는 것

이제 원격에서 사용하는 깃 액션들

pull

당겨오다. 인 것 처럼 원격 저장소에 저장된 소스 코드를 내 저장소에 당겨오는 행동

fetch
똑같이 가져오는데
remote origin 소스랑 비교만 한다. 그래서 내 로컬 저장소 내용은 변화는 없다

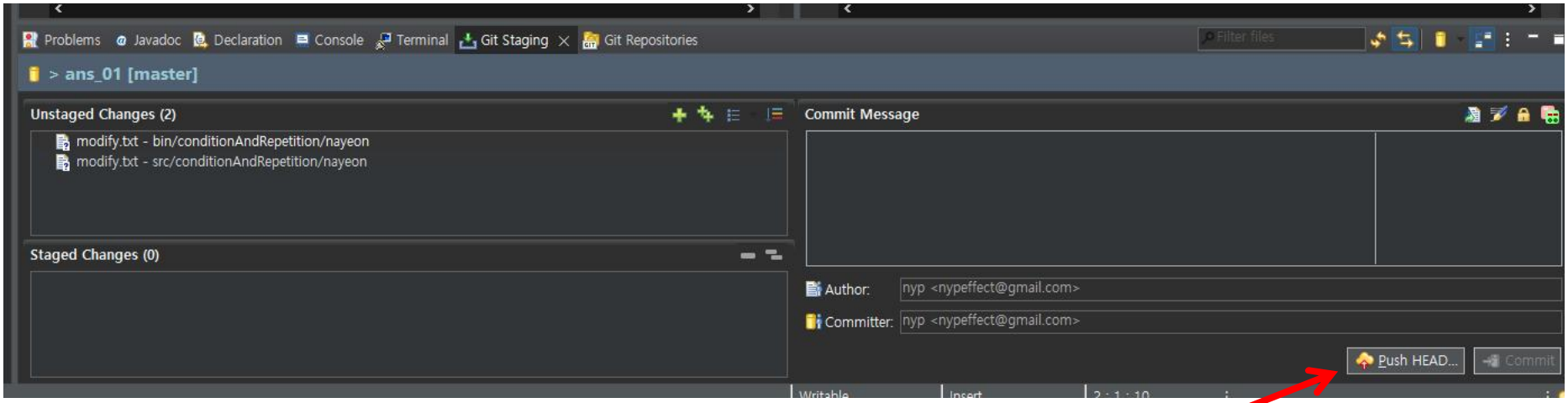
push
밀어 넣기
내 로컬 깃의 변경사항을 내 remote origin의 branch에 밀어넣는다

바뀐 소스를 변경하는거고 commit 된 애들만 올린다

가끔 내 컴의 소스랑 너무 많이 다르면 push가 안되고
commit해서 어쩌구 에러 뜨니
늘 새로 킬때마다 pull해서 내 저장소의 소스를 최신으로 유지하도록 노력하자

안되면... 복사 또 해와야 됨

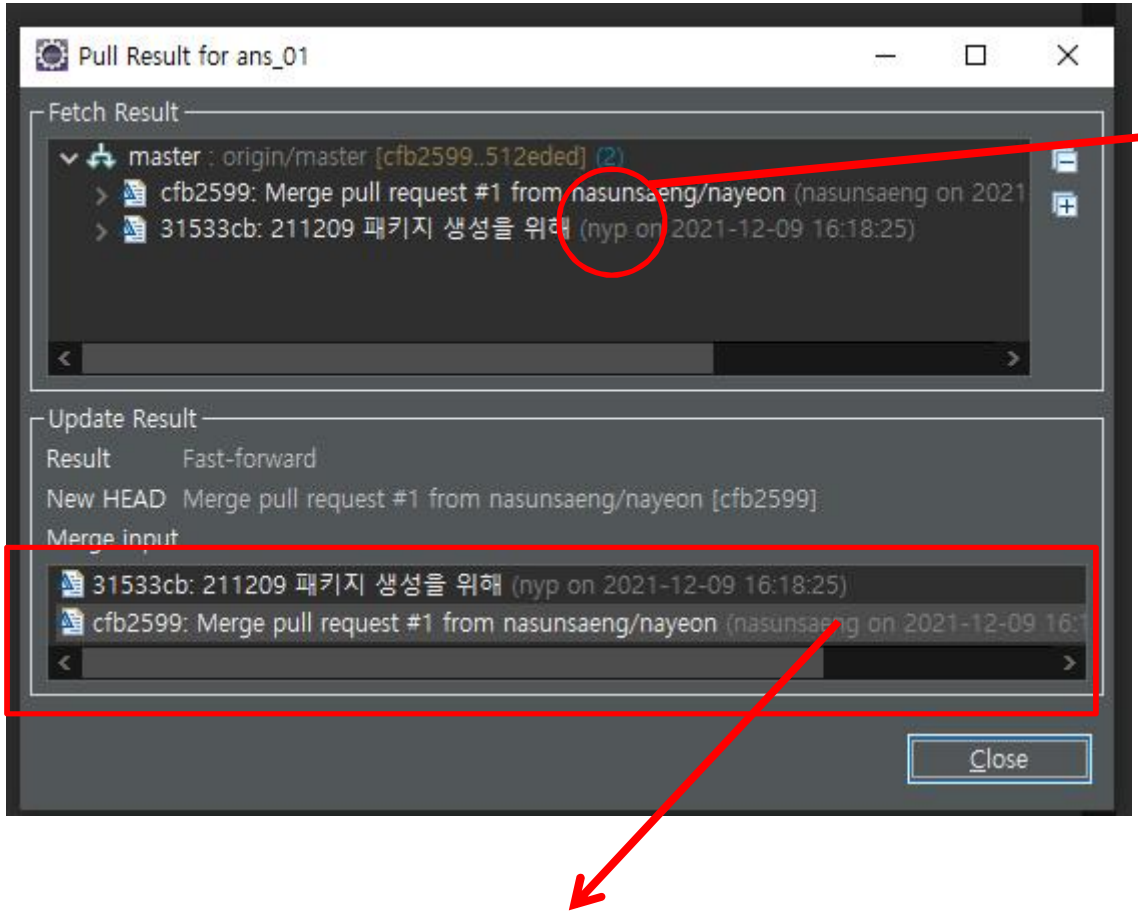
그래서 결국 우리는 우리의 소스가 변경되면
아래의 창에서 무엇을 commit 할 것인지 선택해서




+ 는 걍 add
++는 all add (git add .) 명령을 수행하여

commit message를 쓰고 난 뒤

요걸 눌러준다




아까
git config --global
user.name의 이름

내가 local서 저렇게
commit 했다고
기록이 남은 것

내가 원격 (깃헙)에서 merge 라는 행위를 했다
nasunsaeng<-이라는 깃헙 계정에서 nayeon 이란 브랜치로 한 작업이다.

이런 의미