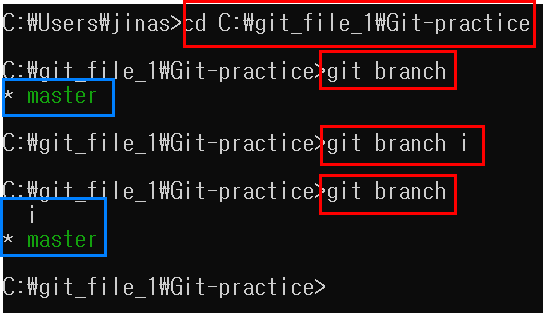
오늘 배울 내용

1. branch마다의 다른 저장으로 서로 정보로 일어나는 병합 오류 해결하기.



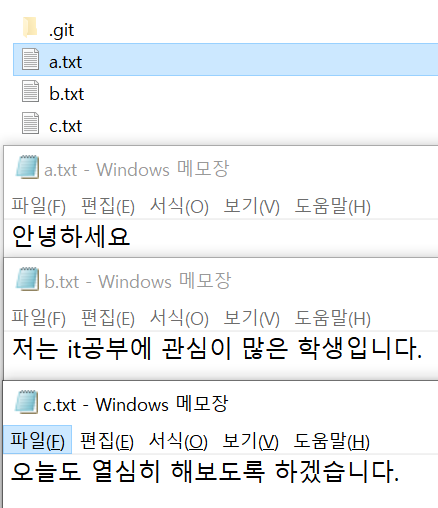
1. cd를 사용하여 나의 파일이 저장되어있는 경로로 가기.

2. git branch를 사용하여 현재 존재하는 branch를 확인.

3. git branch i를 써넣어 새로운 I.branch를 만들었다.

4. git branch를 확인하면 i가 잘 생성되었다.

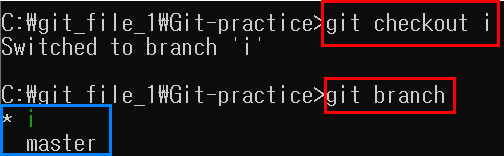
-------------



[Master.branch에 저장된 파일. ]

I.branch에서 수정사항을 추가하기 전에 미리보는 Master.branch에 저장되어있던 내용들.

------------



1. Git checkout i를 이용하여 HEAD가 i를 향하도록한다.

-이로써 I.branch로 넘어올수 있게 되었다.

2. git branch를 확인 결과 HEAD가 잘 I.branch로 넘어왔다는 것을 알 수 있다.

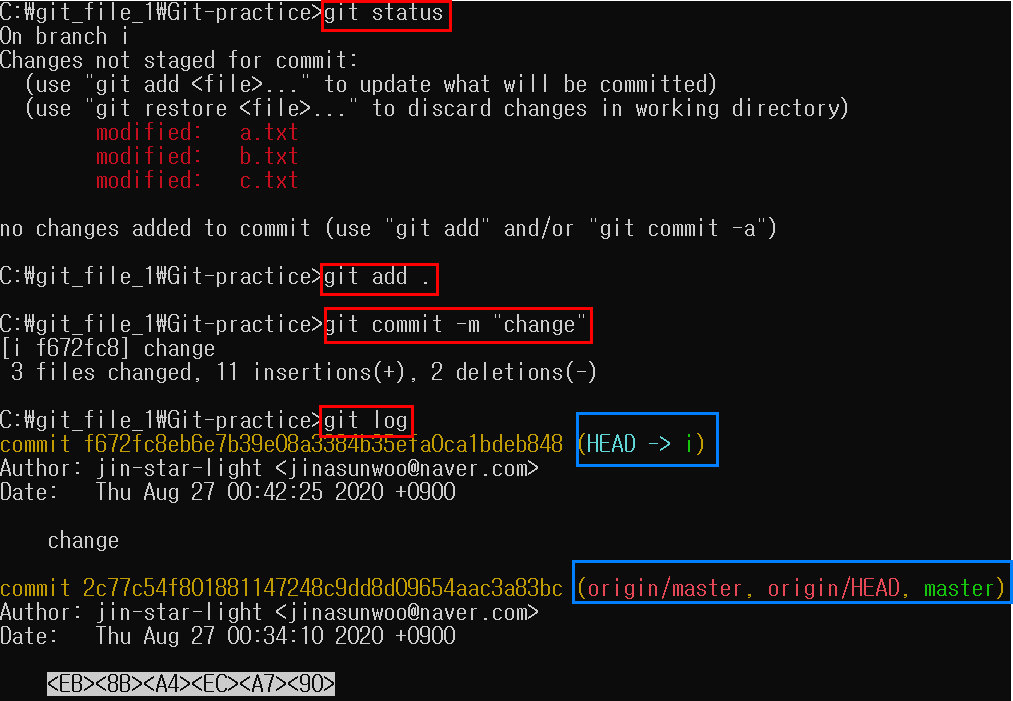
------



[I.branch에서 추가한 내용]

1. I.branch으로 넘어온 상태이니 이제 i의 용도에 맞게 새로운 내용을 써주었다.

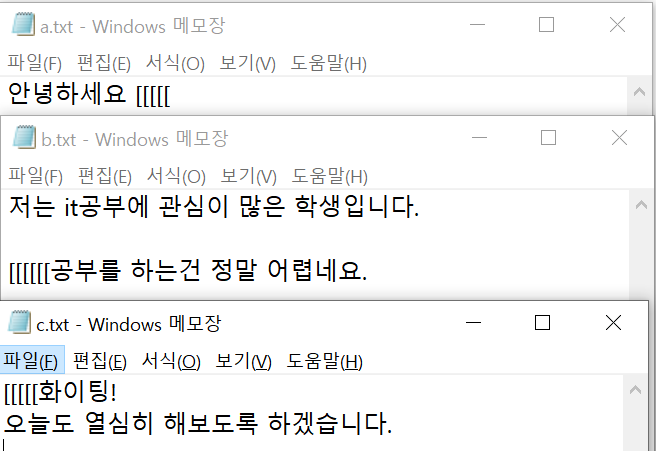
----------------------



1. git status를 사용하여 적용되지 않은 파일들을 모두 git add . 를 이용하여 넣어주고 commit까지 진행했다.

여기까지 i와 master의 branch를 저장해보았다. [실험을 위한 기본 환경 설정 끝]

이제 master.branch 와 I.branch가 서로 달라지기 위해서 master.branch에 가서 i와 다른 추가적인 요소를 추가해보겠다.



[master.branch] -행열 변화에 대한 차이를 확인하기 위해서 일부러 다양하게 추가해보았다.

[상황 정리하기]

현재 설정된 사항은

Branch : I 와 master가 있다.

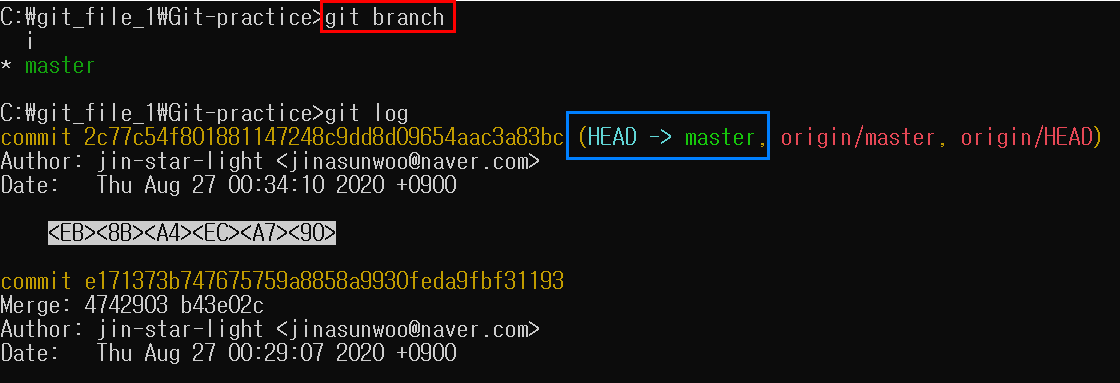
여기서 I.branch 생성후 I.branch에 master.branch의 추가적인 내용을 만들어 저장했다.

그리고 방금전에 master.branch에 추가적인 내용을 추가하여

결론적으로 I.branch와 master.branch의 저장 상태를 다르게 만들어 보았다.

그럼 이제부터 master.branch에 I.branch를 merge했을 때 저장소에서 나타나는 차이점을 해결하기 위해서 어떻게 대처해야하는가를 보도록 하겠다.

-----------------

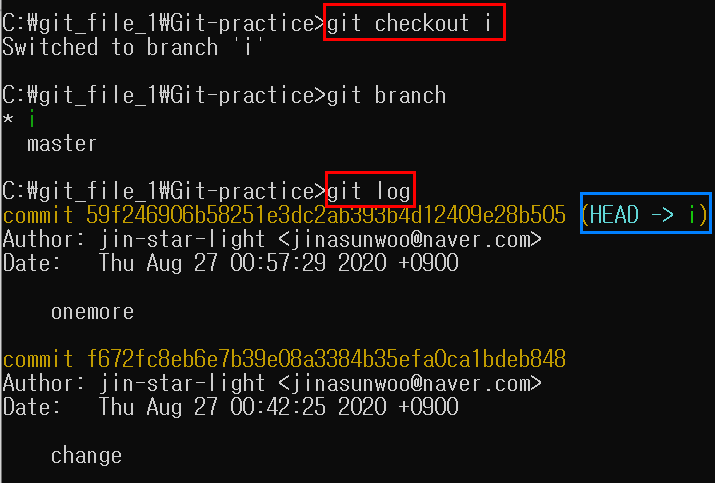


가장먼저 확인사항으로 master.branch에서 git log를 사용하여 branch를 확인해보았다.

여기서 잘보면 I브랜치가 없는 것을 확인할 수 있다.

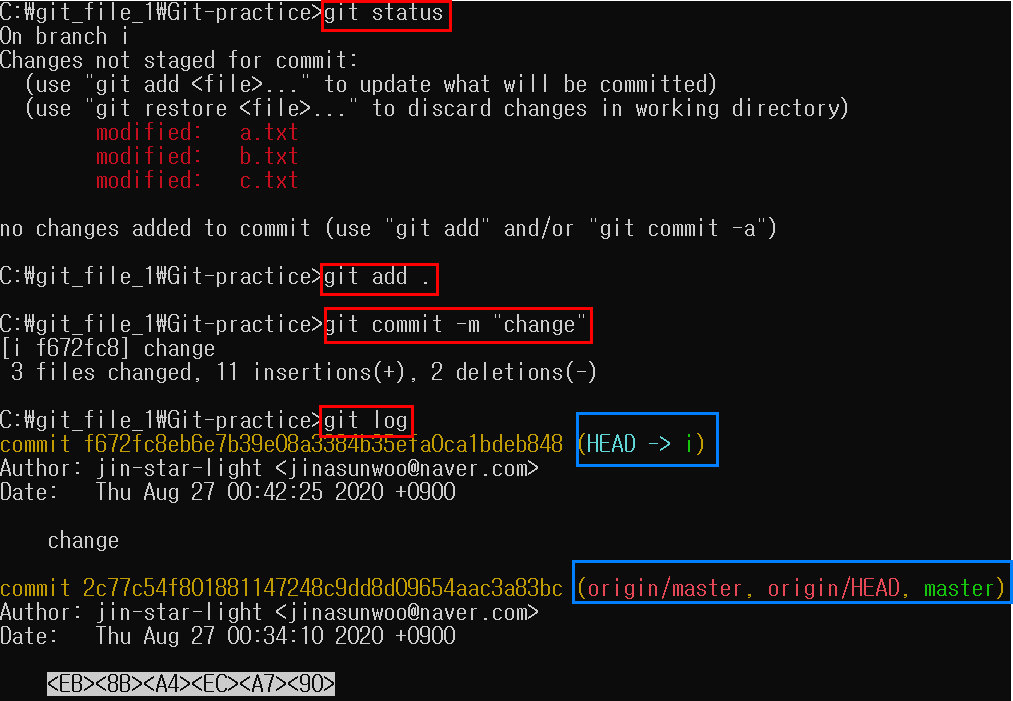
이유는 더 이상 한가지에서 속한 branch라고 판단하지 않아서 나타나는 현상이다.

-------------



I.branch 또한 바로 위 사진과 같은 상황이다.

Git log를 사용하여 branch의 상황을 살펴 보았을 때

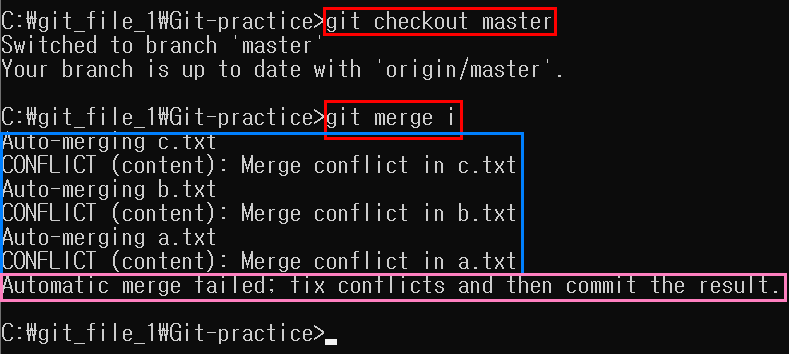


[ master.branch의 파일내용을 변경하기 전 I.branch의 git log 화면]

전과달리 master.branch의 모습을 볼 수가 없다.

이 이유는 컴퓨터가 판별했을 때 더 이상 I.branch와 master.branch가 같은 파일이라고 인식하지 않아서이다.

--------------

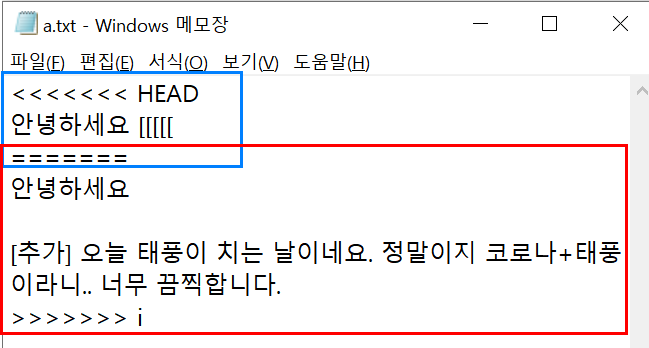


자 이상태에서 전 시간에 배운 merge를 이용하여 두 파일을 병합해 보았다.

merge를 시행했더니 (파란색 상자) 서로다른 사항에 대한 것들이 뜨면서

(핑크색상자) merge failed가 되었다고 뜬다. 잘 보면 다른점들을 고치고 나서 commit을 진행하라고 적혀있다.

---------------

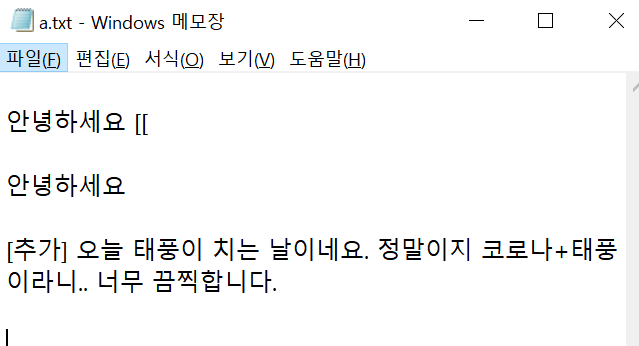


그후 파일에 들어가보면 이렇게 표가 되어있다.

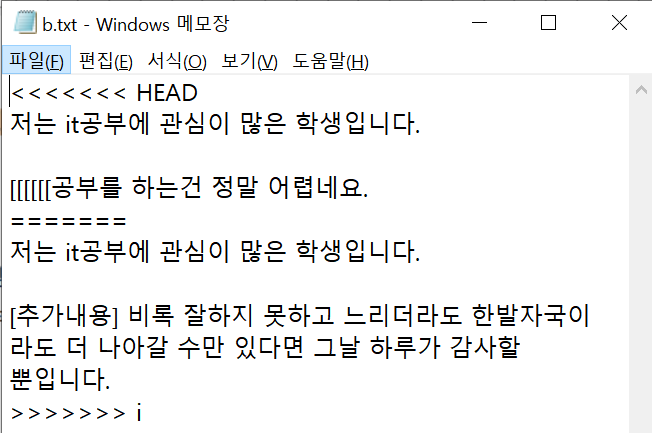
잘보면 파란색 상자에는 HEAD가 표시되어있고 빨간상자에소는 i라고 표기 되어 있는데

이것은 각각 master.branch와 I.branch를 뜻한다.

서로 다른 점을 표기해주는 것인데 여기서 자신이 남기고 싶은 것만 남기고 나머지는 지워주면 된다.

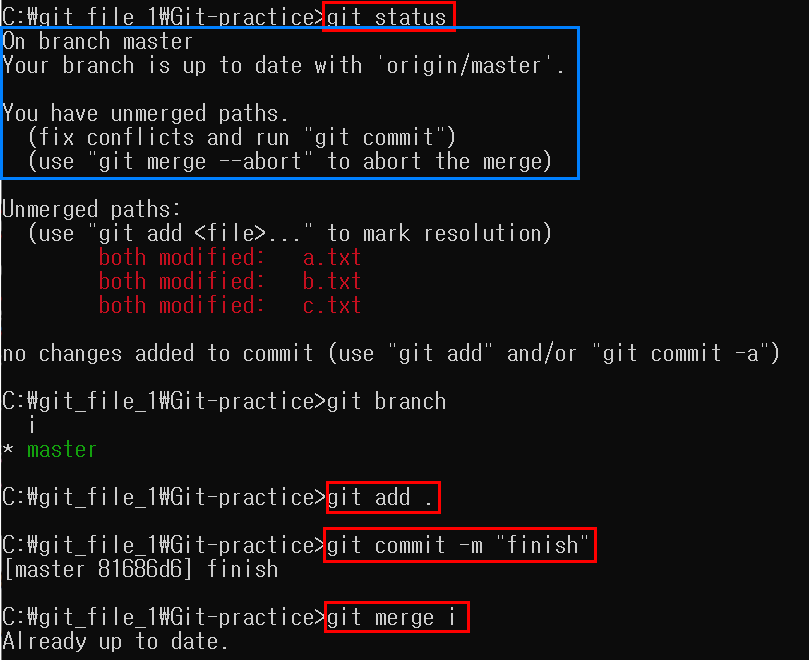


이렇게 남기고 싶은것만 지우면 된다. 참고로 i혹은 master 브랜치 둘중 한 개만 골라야하는 것은 아니다. 적절히 섞어도 되고



심지어는 이상태로 저장시킬 수 있다.

이런식으로 자신이 남기고 싶은 내용을 남기고 나머지는 지우고 수정을 하면서 파일과의 충돌 을 고칠 수 있다.

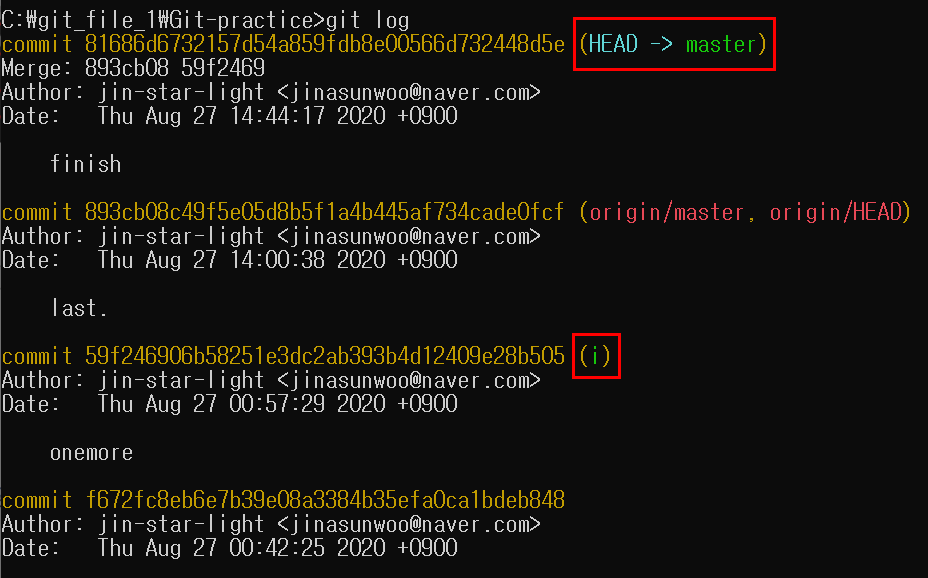


이제 다시 Remote Repository에 저장해 보도록하겠다.

위 사진과 같이 add . 와 commit 을 거친후 merge를 하여 다시 확인 해보면 이미 데이터가 업데이트 되었다는 글이 나온다.

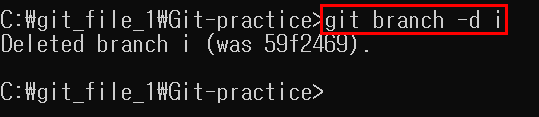
이로써 업데이트는 성공적으로 마쳤다.

------------



Git log를 확인해봐도 master 와 I 가 공존 하는 것을 볼 수있는데 이것은 master에 i가 merge 되어서 그렇다.

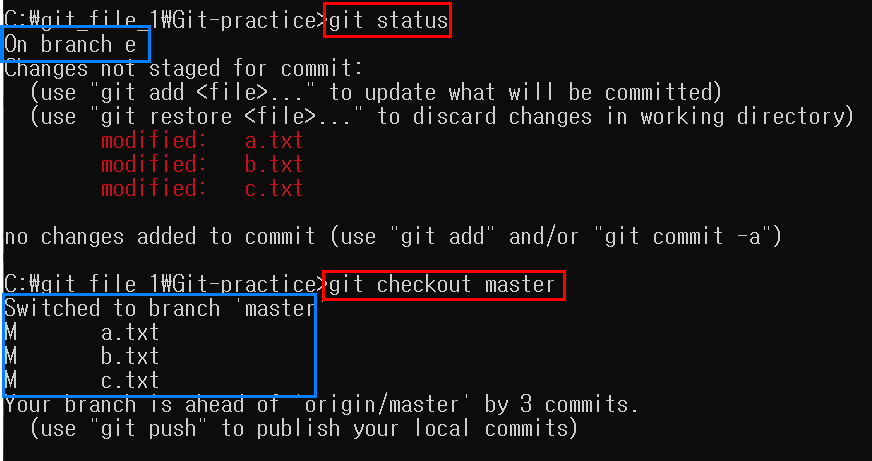
-----------



이렇게 병합이 끝난 브랜치는 더 이상 쓸모가 없으니 삭제해주도록 하자.

----------

[추가적인 발견사항]



1. 이번에는 새로운 e.branch를 만들어 파일들을 만들고 나서 git status로 상황 파악을 했다.

- 파란상자 :잘보면 on branch e가 있는 것을 볼 수 있다. 이것은 현재 내가 보는 storage area가 어떤 branch인지 알려주는 것이다.

2. 다양한 파일 변경사항이 있는 상태에서 git add와 commit을 하지 않은 상태로 다른 branch로 넘어가면 이 저장되지 않은 파일들이 함께 넘어온다. [파란상자 참조]

