문제 상황1: push 함수에서 무한루프에 빠지는 문제

문제 원인1



tailNode의 next가 이미 존재함

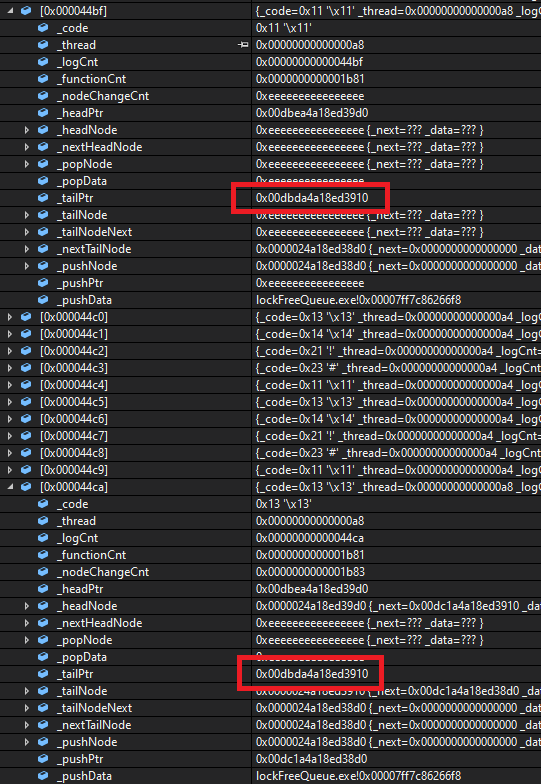
문제 상황2: tail node의 next가 null이 아닌 문제

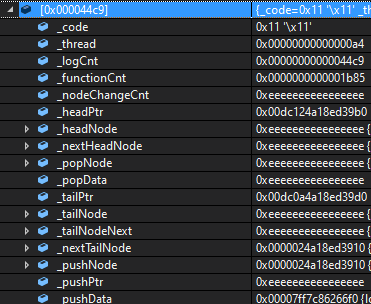
문제 원인2: tail을 반영하기 전에 다른 곳에서 tail을 선반영해서 반영할 수 없었음

테이블이(가) 표시된 사진

자동 생성된 설명

Thread A에서 tail을 [3910]으로 확인하고 next를 [38d0] 노드로 연결했다.

하지만 이것은 꽤 과거의 tail 이였고, 이 사이에 queue의 변화가 꽤 많았다.



그 사이 로그들을 살펴보면 tail 이였던 [3910] 노드가 pop 되어서 다시 push가 되는 상황이 와버렸다.

Push 하기 위해 next를 null로 수정했었고, 여기에 연결하고 말았다.

Tail을 변경하고자 했지만 tail은 알고있던 tail인 [3910]이 아니기에 변경에는 실패한다.

[3910] 노드는 실제 next가 null인 [39d0]의 다음 노드로 삽입되며, [39d0]이 실제 tail이기 때문에 tail또한 [3910] 노드로 변경한다.

테이블이(가) 표시된 사진

자동 생성된 설명

결과적으로 [39d0] -> [3910] -> [38d0] 의 형태가 되었으나, tail은 [3910]인 문제가 있다.

문제 해결 방안

[Tail의 next가 null이 아니라서 compare exchange에 실패하였다]고 했을 때

이 상황이 발생할 수 있는 경우는 두 가지 입니다.

1. 다른 스레드에서 next에는 연결했지만, 아직 tail을 변경하지 않았다.
2. 위와 같은 문제가 발생해서 tail을 변경할 수 없었다.

1번의 경우는 가까운 미래에 tail이 변경될 것이지만, 2번은 기다려도 변경되지 않습니다.

지금 당장 tail이 변경되기까지 마냥 기다리고 있을 수는 없기 때문에, 내가 tail을 변경합니다.

내가 변경하든 기존 작업하던 함수에서 변경하던 어짜피 tail은 기본적으로 next가 null인 상황인 것이 정상이기 때문입니다.

기존 tail을 변경하던 곳은 그대로 둡니다.

Compare exchange가 실패했을 때에는 while문으로 tail의 next가 null이 될 때까지 tail을 next로 변경한다. 이것을 반복하여 next가 항상 null이 되도록합니다.