


12/04 직선 열차

- `add <= 40,000`
- `remove <= 60,000`
-  `calculate <= 120,000`

1. 먼저 지금 열차로 종점 갈 수 있는지? \Rightarrow 가능하면 횡수 return
2. 안되면 이동한 후에 환승 가능한 역들을 찾아서 환승하고 다시 반복한다.

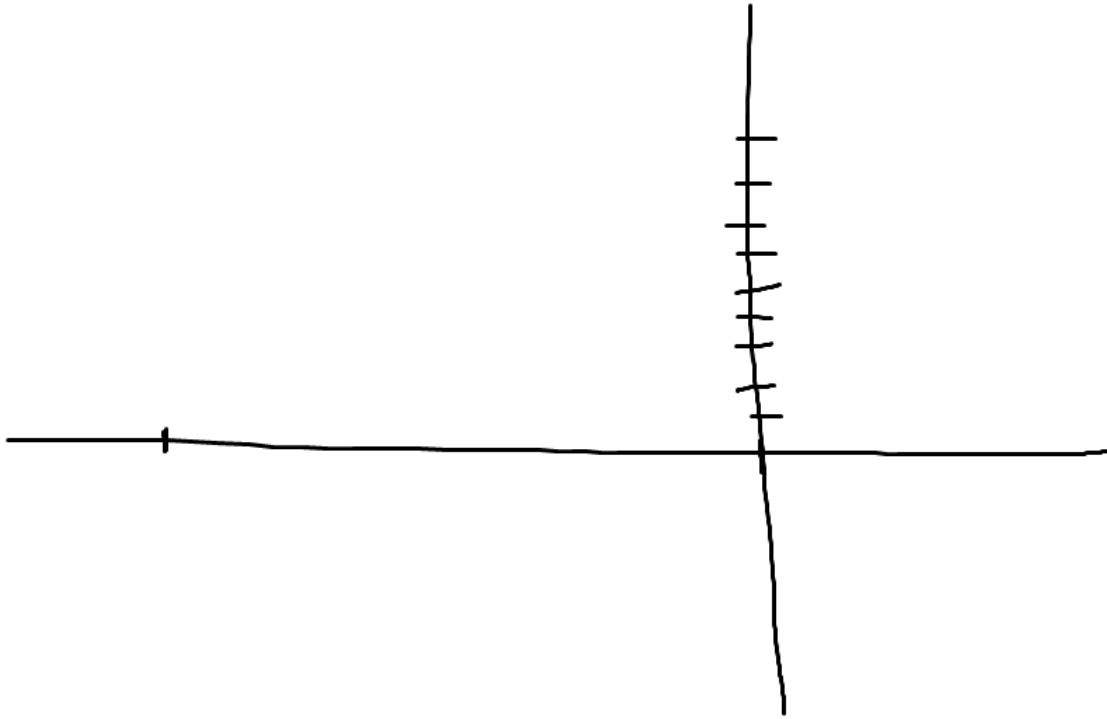
열차 최대개수 잘못 봄. \Rightarrow mid인줄.. K인데 \Rightarrow 문제 잠읽어야함.

(타고있는 열차, 현재 역, cnt) \Rightarrow 현재 역은 중복이 가능함.. 그럼 겹치는 역이 최악에 33000 개씩 200개 생길텐데 \Rightarrow 불가능함. 한번 확인한 역은 제외시킬 수 있긴함. 그래도 $33000 * 200$

역으로 환승할 수 있는지 판별하는 건 틀린 것 같다.

calculate가 시작점과 종점을 주고 있으니까 시작점을 지나는 열차들을 Queue에 넣어서 환승이 가능하면 횡수를 구하고 불가능하면 -1을 return하자

대충 max값으로 $200 * 200 \Rightarrow 40000$



문제점: 200* 200으로 찾고있는데 두 열차의 교집합 있는지 또 검증해야함.. ⇒ 어떻게 줄이지?

수학 계산으로 $O(1)$ 로 해보려했지만 x_1, x_2 가 달라서 안될 것 같다...

```
while(!trainQ.isEmpty()) {  
    int[] cur = trainQ.poll();  
    Train train = trainMap.get(cur[0]);  
    int cnt = cur[1];  
  
    // 먼저 현재 열차로 종착역 갈 수 있는지 판별 && 갈 수 있다면 cnt return  
    if((eld - train.start) % train.interval == 0) {  
        return cnt;  
    }  
}
```

```
    }  
  
    // 나머지 중에서 환승 가능하면 환승하기  
    for(int i = 0; i < trains.size(); i++) {  
  
    }  
}
```

줄이는 방법을 생각해내지 못하고 찾아보게 됨..

다른 함수에서 시간이 남으니까 두 열차의 환승여부를 거기서 계산할 수 있었다.

복기

- List나 원본 배열의 index가 변경될 것 같으면 isRemoved를 사용하자. 직접 삭제하지 않아도 될 경우가 있다.
- 연산 떠넘기는거 생각해보기