[22.05.06] 지하철 노선 관리

TS버스 노선

김 태 현

문 제

- 0~4 번호의 5개 버스 노선, 0~N-1 번호의 N개의 정류장 존재
- 인접한 정류장 이동/다른 노선으로 환승 각 1분 씩 소요

```
void init(int N, int lastBusStop1[5], int lastBusStop2[5])
```

N(<=40,000)개의 정류장 존재
lastBusStop1 : 노선별 첫 번째 종점
lastBusStop2 : 노선별 두 번째 종점

```
void append(int line, int prevBusStop, int nextBusStop)
```

line 노선의 prevBusStop 다음 정류장으로 nextBusStop 추가
prevBusStop이 line에 포함되고 두 번째 종점이 아님을 보장
nextBusStop이 line에 포함되어 있지 않음을 보장

append() <= 40,000
minTime() <= 100
minTransfer() <= 400

```
int minTime(int start, int end)
```

start -> end 로 가는 **최소 시간** 반환
불가능 시 -1 반환

```
int minTransfer(int start, int end)
```

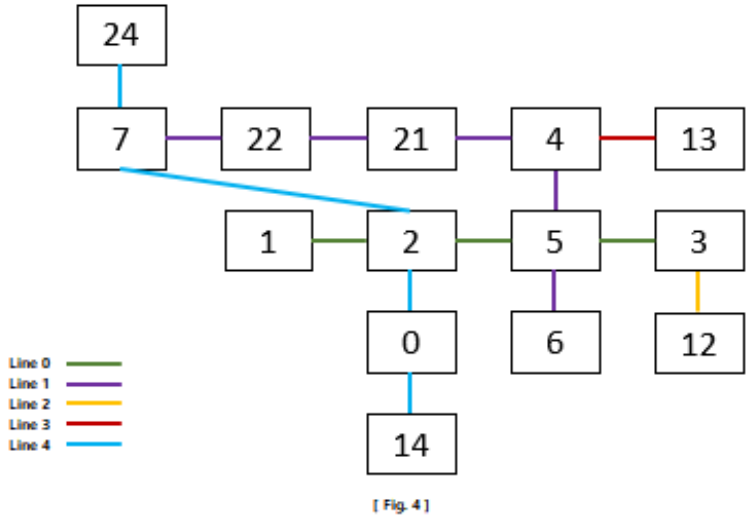
start -> end 로 가는 **최소 환승 횟수** 반환
불가능 시 -1 반환

문제 분석

- 노선 **5개**, 정류장 40,000개
- 노선별로 포함된 정류장의 인접 정류장 : 최대 **2개**
- minTime()<=100, minTransfer()<=400 호출횟수 적음

주요 포인트

1. 최소 시간을 어떻게 구할 것인가
2. 최소 환승 횟수를 어떻게 구할 것인가
3. 노선 관리를 어떻게 할 것인가



[Fig. 4]

최소 시간 구하기 : BFS

필요한 상태 : {정류장, 노선 번호, 소요 시간}

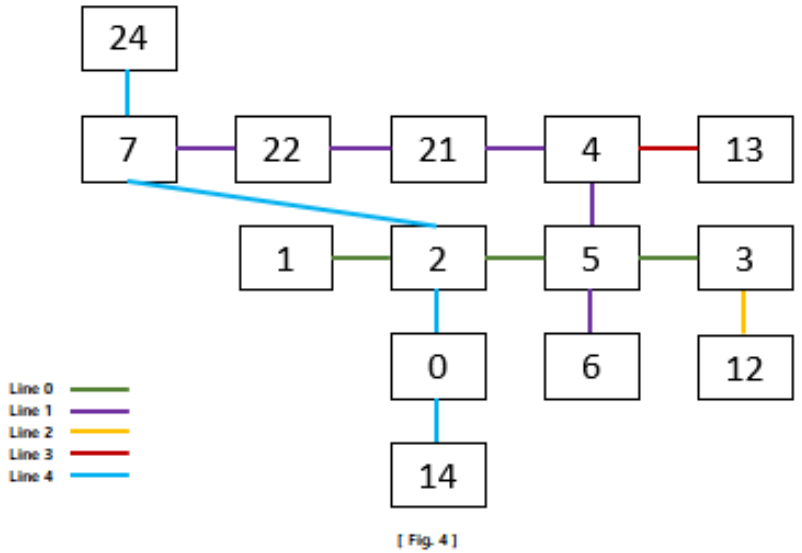
Queue Init

start 정류장이 포함된 모든 노선 line 에 대해 {start, line, 0} 등록

Queue Push

front 정보 {x, line, cnt} 에서 이동 or 환승 가능하고 방문하지 않은 상태를 cnt+1로 등록

1. 이동 : x 정류장 line 노선의 양쪽을 방문하지 않은 경우
2. 환승 : x 정류장이 다른 노선에 포함되어 있는 경우



[Fig. 4]

Result

push 과정에서 end 정류소가 나오면 해당 cnt 반환
queue가 다 빌 때까지 나오지 않으면 -1 반환

최소 환승 횟수 구하기 : BFS

필요한 상태 : {노선 번호, 환승 횟수}

Queue Init

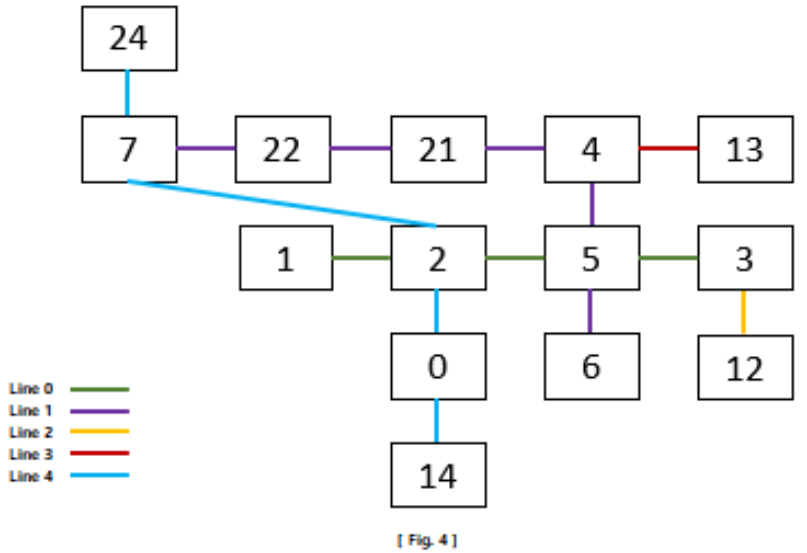
start 정류장이 포함된 모든 노선 line 에 대해 {line, 0} 등록

Queue Push

front 정보 {line, cnt} 에 대해 해당 노선에 포함된 모든 정류소에서 환승 가능한 노선 nextLine 이 최초인 경우에만 {nextLine, cnt+1} 로 등록

Result

push 된 line에 end가 포함되어 있는 경우 해당 cnt 반환
queue가 빌 때까지 결과를 못 찾은 경우 -1 반환



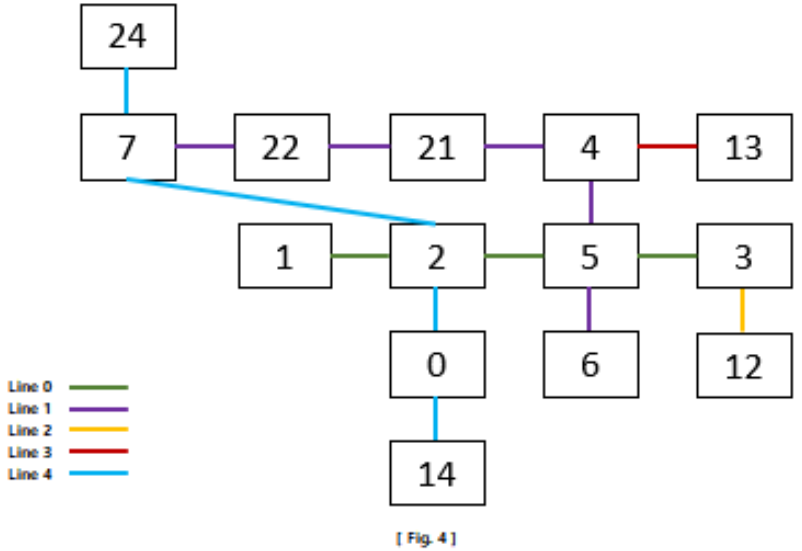
[Fig. 4]

노선 관리

노선 관리시에 필요한 정보

x: 정류장 번호, line: 노선 번호

- 노선에 포함된 모든 정류장 번호
- x가 line에 포함되어 있는지 판별
- line에서 x의 이전,다음 정류장



[Fig. 4]

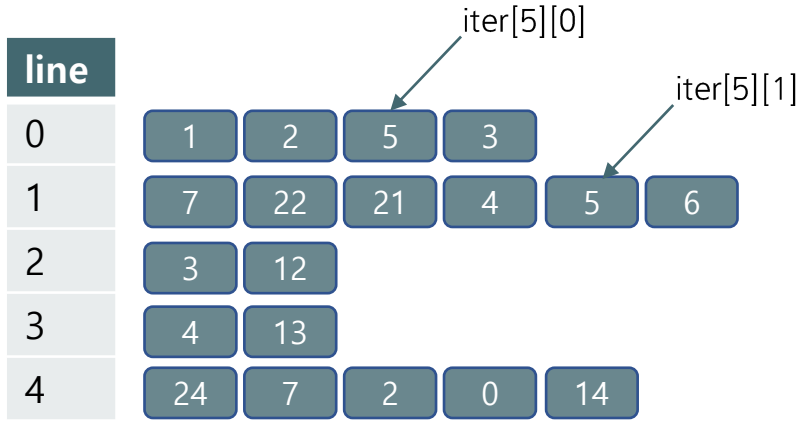
line 총 5개, line별 정류장이 갖는 인접 정류장은 prev, next 두개뿐이므로 그래프 구성 필요 없이

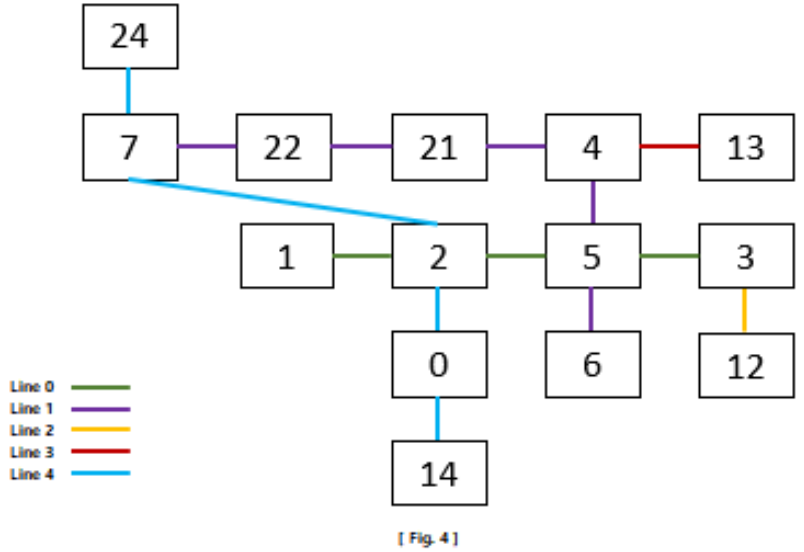
1. **노선 별 linked list**
2. **{정류장, 노선} 별 prev, next 저장**

노선 관리

1. 노선 별 linked list

list<int> route[노선]
list<int>::iterator iter[정류소][노선]





- iter[x][line] 초기값으로 route[line].end() 설정하여 해당 line에 x 존재 여부 파악
- iter[x][line] 통해 이전, 다음 정류장 파악 가능

노선 관리

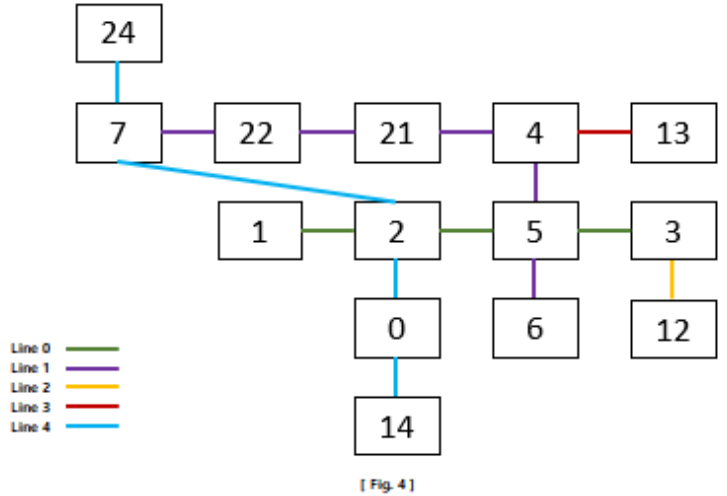
2. {정류장, 노선} 별 prev, next 저장

int lineStart[line]

line	start
0	1
1	7
2	3
3	4
4	24

pii adj[x][line] = { prevBusStop, nextBusStop}

x	line	prev	next
1	0	-1	2
7	1	-1	22
7	4	24	2
5	1	4	6
5	0	2	3



[Fig. 4]

- adj[x][line]의 prev, next 값이 없으면 -1로 설정하여 둘다 -1 이면 x가 line에 포함되지 않았음을 판별
- line의 시작점은 고정이므로 lineStart 통해서 시작점부터 adj를 타고 끝점까지 탐색 가능


감사합니다