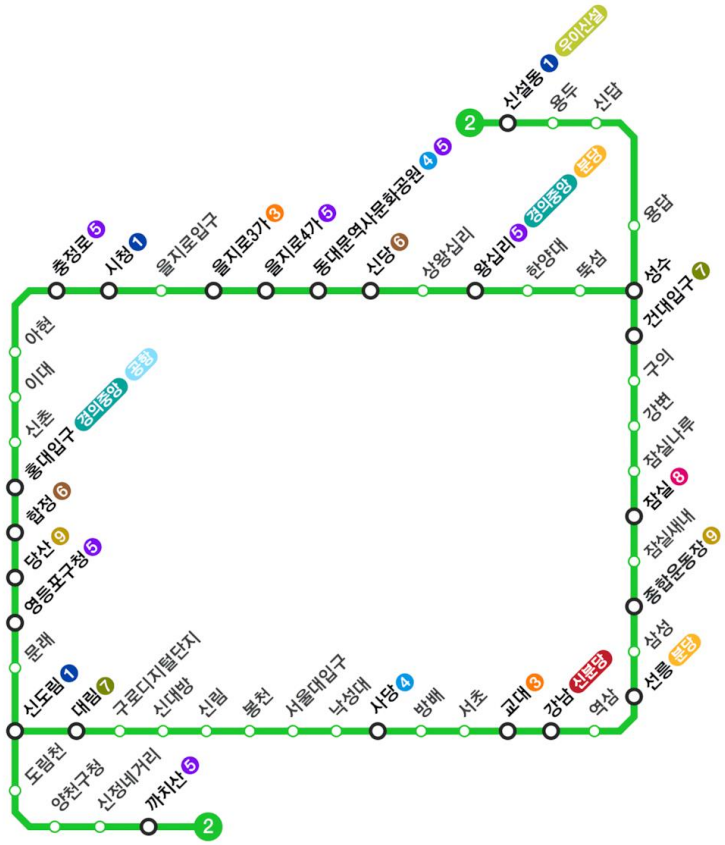
ICPC 19주차 문제풀이

서울 지하철 2호선

소프트웨어학과 201921017 이지우

Step1 - 접근

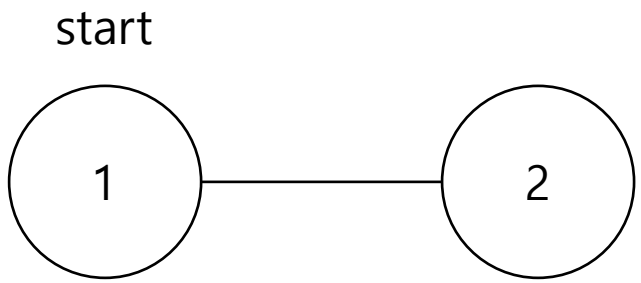


순환선 사이의 거리
- 순환선에 포함되는 역: 0
- 순환선에 포함되지 않는 역: 0보다 큰 값

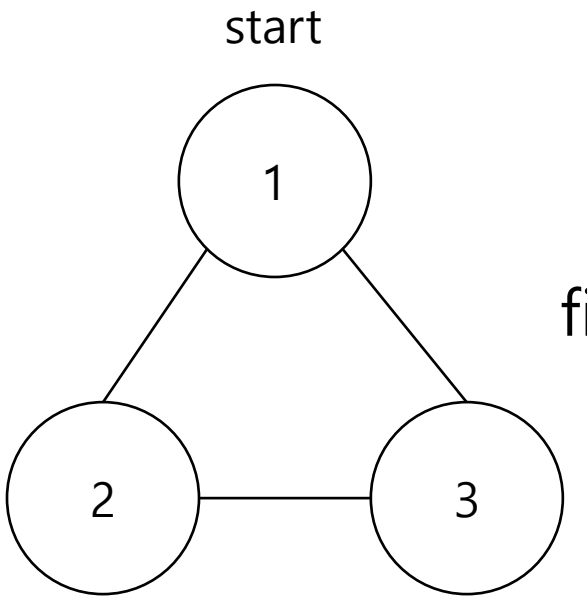
1. 순환선 찾기
   어떤 역이 순환선에 포함되는지 찾아야 함
2. 순환선까지의 (최소)거리 구하기
   순환선에 포함되는 역: 0
   순환선에 포함되지 않는 역: BFS로 거리 구하기

1. 순환선 찾기

DFS로 모든 역 탐색 중 현재 탐색 역이 처음의 역과 같으면 순환역!(돌고 돌아 다시 제자리로~)
*단 그 사이에 거쳐온 역이 2개 이상이어야 진짜 순환역



1, 2는 순환역이 아님



1-3는 순환역임

find_cycle(start, curr, prev_cnt)

```python
def find_cycle(start, curr, prev_cnt):
    visited[curr] = True

    for adj in adj_list[curr]:
        if not visited[adj]:
            find_cycle(start, adj, prev_cnt+1)
        elif adj == start and prev_cnt >= 2: # cycle
            is_cycle[adj] = True
            return
```

*모든 지하철 역에 대한 순환역 확인을 위해 메인함수에서 n번의 find_cycle() 호출이 필요함..
어쨌든 그렇게 해서 x역에 대한 순환 여부는 is_cycle 리스트에 저장됨

2. 거리 계산하기

1. Is_cycle[x] 값이 True인 역(순환역)들은 distance[x] = 0
2. 순환역들을 시작으로 인접한 역들의 '순환선까지의 거리' 계산
3. 현재 탐색역의 순환선까지의 거리: 3이면
현재 탐색역의 인접역들의 순환선까지의 거리: 3+1

→ 현재 탐색역의 모든 인접 역에 대해 거리를 구하고 나서 탐색역을 갱신
→ BFS 방식

```
def get_distances():
    dq = deque()

    for i in range(n):
        if is_cycle[i]:
            distance[i] = 0
            dq.append(i)

    while dq:
        curr = dq.popleft()

        for adj in adj_list[curr]:
            if distance[adj] == INF:
                dq.append(adj)
                distance[adj] = distance[curr] + 1
```

3. 입출력

```python
from sys import stdin, setrecursionlimit
from collections import defaultdict, deque

setrecursionlimit(10000)
```

defaultdict 사용 → 딕셔너리의 value 형식을 list로 지정함
distance의 초기값은 INF

```python
INF = 3000

n = int(input())
adj_list = defaultdict(list)
is_cycle = [False] * (n)
distance = [INF] * (n)

for _ in range(n):
    s1, s2 = map(int, stdin.readline().split())
    adj_list[s1-1].append(s2-1)
    adj_list[s2-1].append(s1-1)

for s in range(n):
    visited = [False]*(n)
    find_cycle(s, s, 0)

get_distances()

print(*distance)
```

Question…?

Yes

Thank you