BOJ

#1976. 여행 가자

https://www.acmicpc.net/problem/1976

25.03.03

Problem 시간: 1시간 15분

#1976. 여행 가자

여행 가자 성공

☆

골드 IV

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	128 MB	54306	21272	15485	37.463%

문제

동혁이는 친구들과 함께 여행을 가려고 한다. 한국에는 도시가 N개 있고 임의의 두 도시 사이에 길이 있을 수도, 없을 수도 있다. 동혁이의 여행 일정이 주어졌을 때, 이 여행 경로가 가능한 것인지 알아보자. 물론 중간에 다른 도시를 경유해서 여행을 할 수도 있다. 예를 들어 도시가 5개 있고, A-B, B-C, A-D, B-D, E-A의 길이 있고, 동혁이의 여행 계획이 E C B C D 라면 E-A-B-C-B-C-B-D라는 여행경로를 통해 목적을 달성할 수 있다.

도시들의 개수와 도시들 간의 연결 여부가 주어져 있고, 동혁이의 여행 계획에 속한 도시들이 순서대로 주어졌을 때 가능한지 여부를 판별하는 프로그램을 작성하시오. 같은 도시를 여러 번 방문하는 것도 가능하다.

📝Problem

Step 0. 입출력

예제 입력 1 복사

```
3
3
0 1 0
1 0 1
0 1 0
1 2 3
```

도시의 수 N

여행 계획에 속한 도시의 수 M

N개의 줄에는 연결정보

0: 연결X
1: 연결O

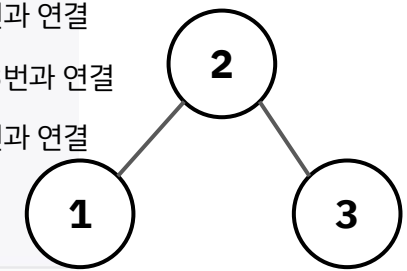
j=1 j=2 j=3

i=1 1번 도시는 2번과 연결

i=2 2번 도시는 1,3번과 연결

i=3 3번 도시는 2번과 연결

여행 계획



예제 출력 1 복사

```
YES
```

여행 계획에 속한 도시들을
순서대로 방문할 수 있는가?

YES or NO

💡 Hint

Step 1. 문제 접근 방향

💡 BFS? Union-Find?

이 문제는 **여행 계획에 있는 모든 도시가 서로 연결되어 있는지 확인하는 문제**다.
연결되어 있으면 "YES", 하나라도 끊겨 있으면 "NO"다.

- BFS로 접근하여 한 도시에서 연결된 모든 곳을 찾아갈 수 있다. 하지만 이 문제는 연결 여부만 확인하면 되는데, BFS는 매번 탐색을 새로 시작해야 해서 시간이 오래 걸릴 수 있다.

- **"Union-Find(유니온 파인드;분리 집합)"**라는 방법을 쓰면 적합한 문제이다. 유니온 파인드는 도시들을 연결된 그룹으로 묶어서 빠르게 확인하는 방법이다.

💡 Hint

Step 2. 유니온 파인드 (Union-Find)

> 💡 **여러 노드(도시)를 집합으로 묶어 두 노드가 같은 집합에 속하는지를 빠르게 확인하는 자료 구조이다.**

- **union** 연산과 **find** 연산으로 구성되어있다.
  1. **union** 연산은 **두 노드를 하나의 집합으로 합치는 연산**이다.
  2. **find** 연산은 **특정 노드가 속한 집합의 대표(루트)를 찾는 연산**이다.
     - 경로 압축 기법을 사용하여 반복되는 **find 연산의 효율성**을 크게 향상시킨다**.**
       → 경로 압축은 노드가 대표 노드와 바로 연결되도록 재배치하여, 이후 탐색 시간을 단축시킨다.

💡 Hint

Step 2. 유니온 파인드 (Union-Find)

💡 동작 과정

1. **초기화**
   모든 노드는 처음에 자기 자신이 대표 노드가 된다.
   → parent = [0, 1, 2, ... , n-1]
2. **union 연산**
   - union(a, b)는 두 노드 a와 b의 대표 노드를 찾은 후,
     두 집합을 하나로 합친다.
   - 대표 노드 값을 비교하여 더 작은 값(혹은 기준에 따라 결정된 값)을 새로운 대표 노드로 정한다.
     → 이 과정을 통해 집합 내의 모든 노드는 동일한 대표 노드를 가지게 된다.
3. **find 연산**
   - find(a)는 도시 a가 속한 집합의 대표 노드를 찾는다.
   - 만약 a의 부모가 자기 자신이라면, a가 대표 노드이다.
   - 그렇지 않으면, a의 부모를 **재귀적으로 탐색**하여 대표 노드를 찾는다.
   - 이 과정에서 **경로 압축**을 적용하여, a와 경로 상의 모든 노드를 대표 노드와 바로 연결시킨다.
     → 이로 인해 추후에 find 연산의 시간이 거의 상수 시간으로 단축된다.

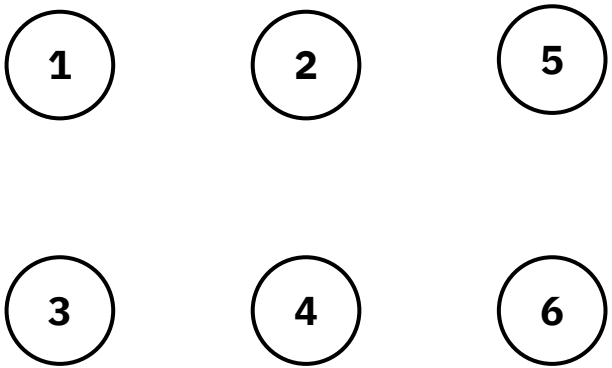
Hint

Step 2. 유니온 파인드 (Union-Find)

1. 초기화

모든 도시는 처음에 자기 자신이 대표 노드가 된다.

→ parent = [0, 1, 2, ... , n-1]



유니온 파인드 배열 (대표 노드 배열)

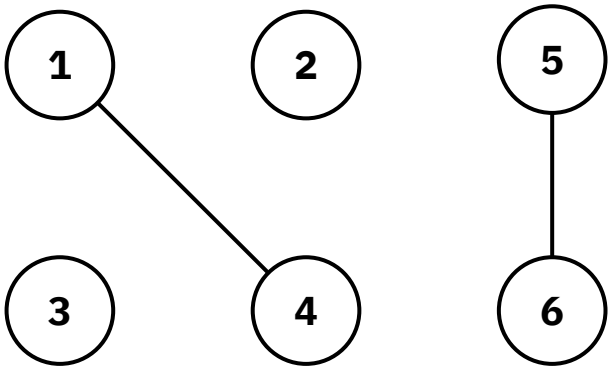
1	2	3	4	5	6
1	2	3	4	5	6

Hint

Step 2. 유니온 파인드 (Union-Find)

2. union 연산

(1) union(1, 4)
(2) union(5, 6)



유니온 파인드 배열 (대표 노드 배열)

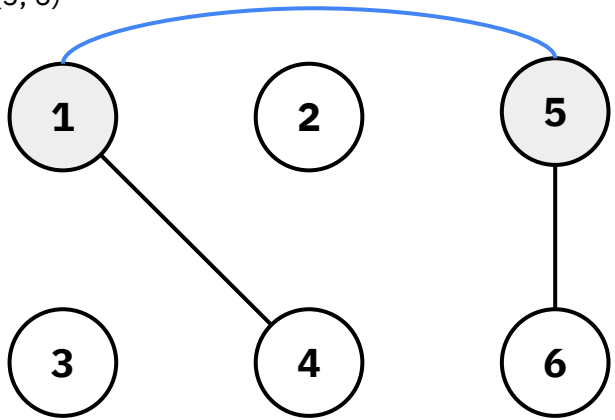
1	2	3	4	5	6
1	2	3	1	5	5

Hint

Step 2. 유니온 파인드 (Union-Find)

2. union 연산

(1) union(1, 4)
(2) union(5, 6)



유니온 파인드 배열 (대표 노드 배열)

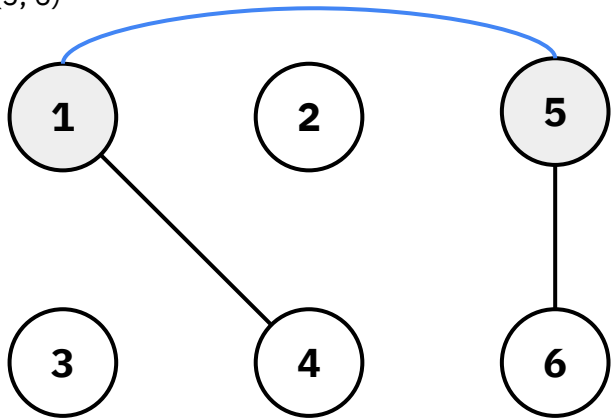
1	2	3	4	5	6
1	2	3	1	~~5~~ 1	5

Hint

Step 2. 유니온 파인드 (Union-Find)

2. union 연산

(1) union(1, 4)
(2) union(5, 6)



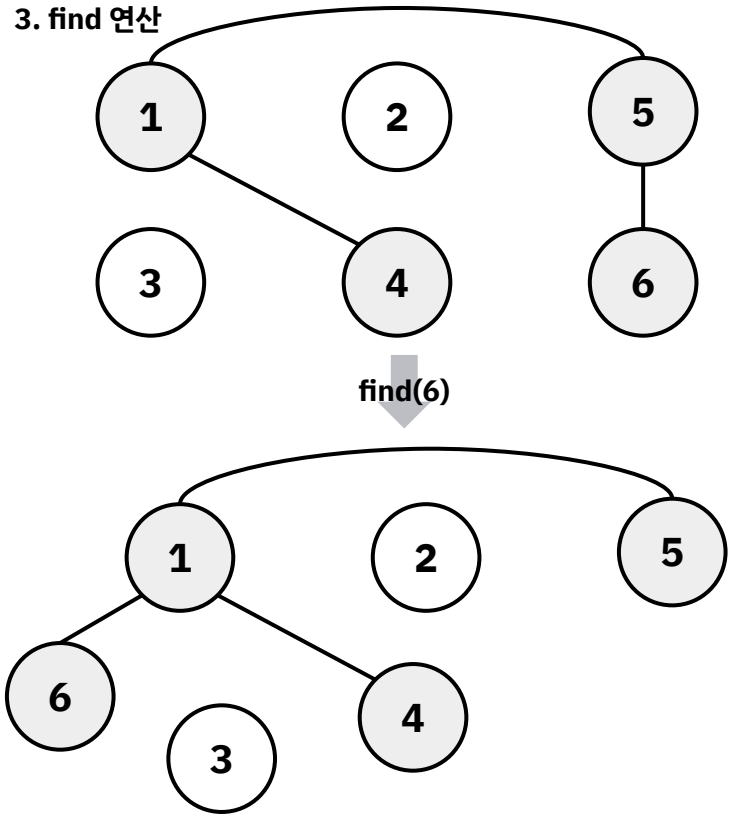
유니온 파인드 배열 (대표 노드 배열)

1	2	3	4	5	6
1	2	3	1	~~5~~ 1	5

Hint

Step 2. 유니온 파인드 (Union-Find)

3. find 연산



유니온 파인드 배열 (대표 노드 배열)

A[1] == 1 A[5] != 5 A[6] != 6

1	2	3	4	5	6
1	2	3	1	1	1

- index 값과 value 값 비교 (A[6] != 6 ➡️다름)
- value 값이 가리키는 index로 이동 후 다시 비교 (A[5] != 5 ➡️다름)
- value 값이 가리키는 index로 이동 후 다시 비교(A[1] == 1 ➡️같음)

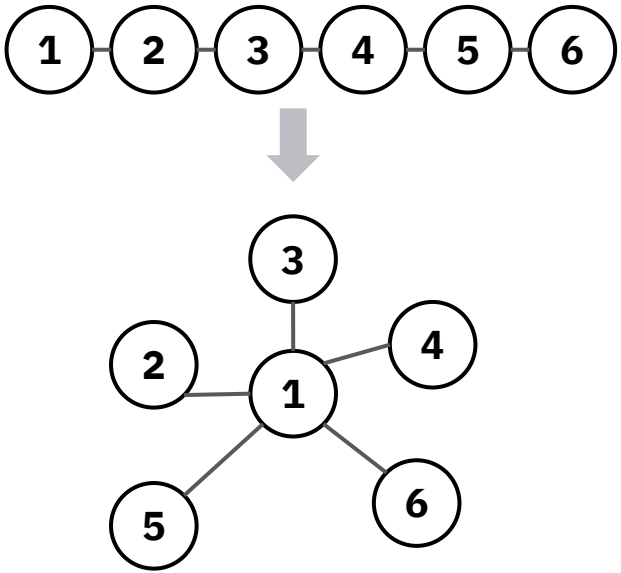
➡️1번 노드가 집합의 루트 노드

➡️재귀 함수를 나오면서 그동안 거친 노드의 value 값을 대표(루트) 노드인 1번 노드의 value 값으로 변경

Hint

Step 2. 유니온 파인드 (Union-Find)

경로 압축



find(6)

1	2	3	4	5	6
1	1	2	3	4	5

1	2	3	4	5	6
1	1	2	3	4	5

1	2	3	4	5	6
1	1	1	1	1	1

- find 연산은 단순히 대표 노드를 찾는 역할뿐만 아니라 그래프를 정돈하고 시간 복잡도를 향상시킨다.
- 연산을 할 때 거치는 노드들이 대표 노드와 바로 연결되는 형태로 변경되기 때문이다. 이렇게 되면 추후 노드와 관련된 find 연산 속도가 O(1)로 변경된다.

Hint

Step 3. 코드 플로우

1. **입력 및 초기화**
   - 도시의 개수 n과 여행 계획에 포함된 도시 수 m을 입력받는다.
   - 각 도시는 처음에 자기 자신을 대표 노드로 가지도록 parent 배열을 초기화한다.
2. **그래프 연결 정보 처리**
   - n개의 줄에 걸쳐 도시 간 연결 정보를 입력받는다.
   - 각 행에서 값이 1이면 두 도시가 연결되어 있으므로, union 연산을 수행하여 하나의 집합으로 합친다.
3. **유니온 파인드 연산 구현**
   - find(a) 함수: 재귀를 사용하여 a가 속한 집합의 대표 노드를 찾으며, 경로 압축을 적용한다.
   - union(a, b) 함수: 두 도시의 대표 노드를 찾아, 두 집합을 하나로 합친다.
4. **여행 계획 확인**
   - 여행 계획에 있는 첫 번째 도시의 대표 노드를 기준으로,
     나머지 도시들의 대표 노드와 비교하여 모두 동일하면 여행이 가능하다.
5. **최종 결과 출력**
   - 모든 도시가 같은 집합에 속하면 YES, 하나라도 다르면 NO를 출력한다.

✨Solution

https://eunjng.tistory.com/85

💡 유니온 파인드 풀이

```python
import sys
sys.setrecursionlimit(10 ** 8)
input = sys.stdin.readline
# 0 1 2
# 0 0 0
def union(a, b):
    p_a = find(a)
    p_b = find(b)

    if p_a > p_b:  # a의 대표보다 b의 대표가 더 작은 값을 가지면,
        parent[p_a] = p_b  # a가 속한 집합을 b의 집합에 합친다.
    else:
        parent[p_b] = p_a  # 그렇지 않으면 b의 집합을 a의 집합에 합친다.

def find(a):
    if a == parent[a]:  # a가 자기 자신의 부모이면 대표 노드이다.
        return a

    parent[a] = find(parent[a])  # 경로 압축을 통해 a의 부모를 대표 노드로 재설정한다.
    return parent[a]
```

```python
# 도시의 개수 n과 여행 계획에 포함된 도시의 수 m을 입력받는다.
n = int(input())
m = int(input())
# 각 도시는 처음에 자기 자신이 대표 노드이다.
parent = [i for i in range(n)]

# n개의 줄에 걸쳐 도시 간 연결 정보를 입력받고, 연결되어 있으면 union 연산을 수행한다.
for i in range(n):
    arr = list(map(int, input().split()))
    for j in range(n):
        if arr[j]:  # 1이면 i와 j가 연결되어 있으므로 union 연산을 수행한다.
            union(i, j)

# 여행 계획을 입력받는다.
plan = list(map(int, input().split()))
result = "YES"
# 여행 계획에 있는 모든 도시가 같은 집합(대표 노드)을 가지는지 확인한다.
for i in range(1, m):
    if parent[plan[i]-1] != parent[plan[0]-1]:
        result = "NO"
        break

print(result)
```

✨Solution

💡 BFS 풀이

https://eunjng.tistory.com/85

```
import sys
from collections import deque
input = sys.stdin.readline

def bfs(start):
    queue = deque([start])
    visited[start] = True

    while queue:
        now = queue.pop()
        for i in range(n):
            if arr[now][i] and not visited[i]:
                visited[i] = True
                queue.append(i)


n = int(input())
m = int(input())
arr = [list(map(int, input().split())) for _ in range(n)]
plan = list(map(int, input().split()))

visited = [False] * (n+1)
result = "YES"
bfs(plan[0] - 1)

for p in plan:
    if not visited[p-1]:
        result = "NO"
        break

print(result)
```

✨Solution

💡 결론

아이디	문제	결과		메모리	시간	언어	코드 길이	제출한 시간
merrong925	4 1976	맞았습니다!!	**BFS**	34924 KB	60 ms	Python 3 / 수정	607 B	12초 전
merrong925	4 1976	맞았습니다!!	**Union-Find**	32412 KB	40 ms	Python 3 / 수정	845 B	3분 전

유니온 파인드 방식으로 푼 풀이가 메모리/시간 측면에서 효율적인 것을 볼 수 있다.

💻 Assignment

백준 #1717. 집합의 표현 (골드5)

유니온 파인드(분리 집합)

발제 문제에서 사용한 **유니온 파인드 유형의 기본 문제**

1717번 제출 맞힌 사람 숏코딩 재채점 결과 채점 현황 내 제출 강의 질문 게시판

집합의 표현

스패셜 저지

골드 V

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	128 MB	117870	38351	23427	28.671%

문제

초기에 $n+1$개의 집합 $\{0\}, \{1\}, \{2\}, \dots, \{n\}$이 있다. 여기에 합집합 연산과, 두 원소가 같은 집합에 포함되어 있는지를 확인하는 연산을 수행하려고 한다.

집합을 표현하는 프로그램을 작성하시오.

입력

첫째 줄에 n, m이 주어진다. m은 입력으로 주어지는 연산의 개수이다. 다음 m개의 줄에는 각각의 연산이 주어진다. 합집합은 $0\ a\ b$의 형태로 입력이 주어진다. 이는 a가 포함되어 있는 집합과, b가 포함되어 있는 집합을 합친다는 의미이다. 두 원소가 같은 집합에 포함되어 있는지를 확인하는 연산은 $1\ a\ b$의 형태로 입력이 주어진다. 이는 a와 b가 같은 집합에 포함되어 있는지를 확인하는 연산이다.

출력

1로 시작하는 입력에 대해서 a와 b가 같은 집합에 포함되어 있으면 "YES" 또는 "yes"를, 그렇지 않다면 "NO" 또는 "no"를 한 줄에 하나씩 출력한다.