

BOJ 1058.

친구

---

소프트웨어학과 201921017 이지우

## Step0 – 문제

### 문제

---

지민이는 세계에서 가장 유명한 사람이 누구인지 궁금해졌다. 가장 유명한 사람을 구하는 방법은 각 사람의 2-친구를 구하면 된다. 어떤 사람 A가 또다른 사람 B의 2-친구가 되기 위해선, 두 사람이 친구이거나, A와 친구이고, B와 친구인 C가 존재해야 된다. 여기서 가장 유명한 사람은 2-친구의 수가 가장 많은 사람이다. 가장 유명한 사람의 2-친구의 수를 출력하는 프로그램을 작성하시오.

A와 B가 친구면, B와 A도 친구이고, A와 A는 친구가 아니다.

나는 나 자신과 친구가 아님!

### 입력

---

첫째 줄에 사람의 수 N이 주어진다. N은 50보다 작거나 같은 자연수이다. 둘째 줄부터 N개의 줄에 각 사람이 친구이면 Y, 아니면 N이 주어진다.

### 출력

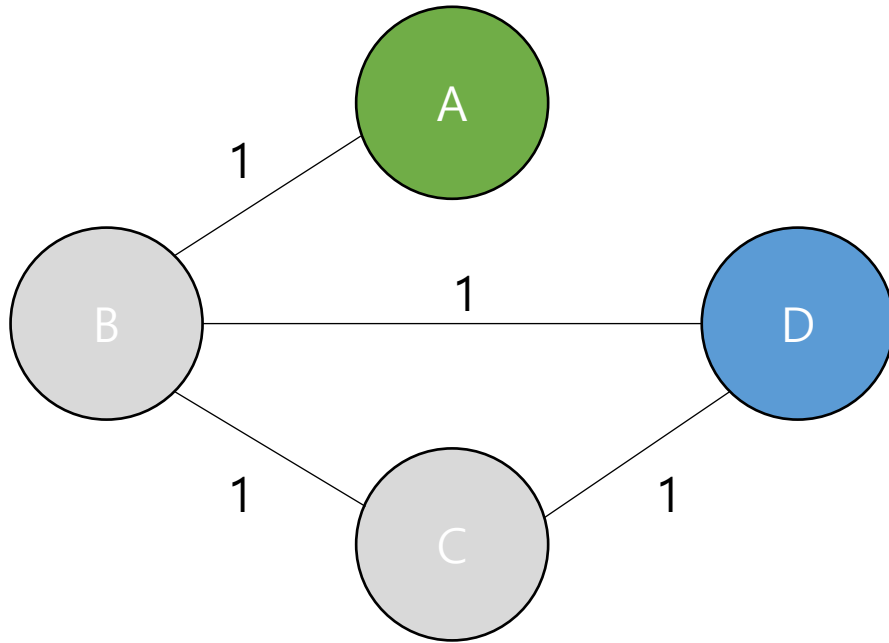
---

$1 \leq N \leq 50$

첫째 줄에 가장 유명한 사람의 2-친구의 수를 출력한다.

## Step1 – 접근

- 특정 한 사람으로부터 다른 모든 사람까지의 최단거리를 구하는 문제 → 다익스트라, 벨만-포드
- 모든 사람으로부터 다른 모든 사람까지의 최단거리를 구하는 문제 → 플로이드-와셜

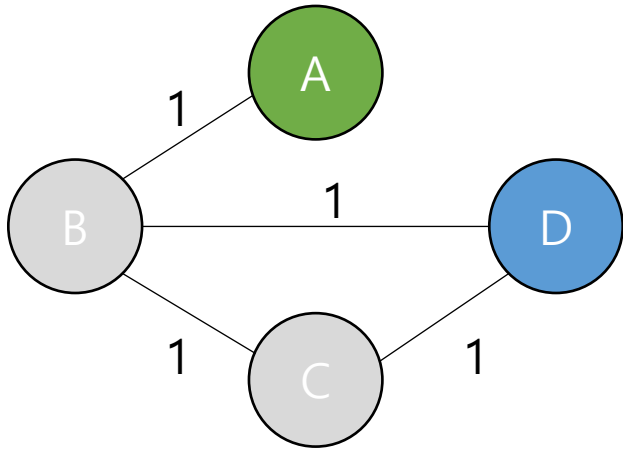


A와 D사이의 최단거리는  
 $A \rightarrow B \rightarrow D(2) \dots O$   
 $A \rightarrow B \rightarrow C \rightarrow D(3) \dots X$

## Step1 – 접근

### 플로이드-와셜(Floyd-Warshall) 알고리즘

- 모든 정점으로부터 다른 모든 정점으로까지의 최단거리를 구할 수 있는 알고리즘



INF	1	INF	INF
1	INF	1	1
INF	1	INF	1
INF	1	1	INF

- ~~$d(u, v) = \min(d(u, v), d(u, r) + w(r, v))$~~
- 음의 간선 허용
- $O(n^3)$ 의 시간복잡도를 가짐( $n$ 이 작은 경우 사용하는 게 좋음. 문제에서의  $n$ 은 50이하의 자연수)

50

## Step2 – 구현

```
int n;
cin >> n;

const int INF = 100;
vector<vector<char>> friends(n, vector<char>(n));
vector<vector<int>> two_friends(n, vector<int>(n));

for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        cin >> friends[i][j];
        two_friends[i][j] = (friends[i][j] == 'Y') ? 1 : INF;
    }
}
```

NTT  
↓  
char = h  
두 사람이 친구이면, 거리(가중치) 1로 저장

```
find_friend_network(n, two_friends);

int max_cnt = 0;
for (int i = 0; i < n; i++) {
    int cnt = 0;
    for (int j = 0; j < n; j++)
        if (two_friends[i][j] <= 2)
            cnt++;
    max_cnt = max(max_cnt, cnt);
}

cout << max_cnt << '\n';

return 0;
```

모든 사람 간의 최단 거리 계산  
2-친구 관계가 가장 많은 유명인의 친구 수 구하기  
2-친구

## Step2 - 구현

```
void find_friend_network(int n, vector<vector<int>> &v)
{
    for (int i = 0; i < n; i++) { // round #i
        for (int j = 0; j < n; j++) {


            if (i == j)
                continue;

            for (int k = 0; k < n; k++) {

                if (j == k)
                    continue;

                if (v[j][k] > v[j][i] + v[i][k])
                    v[j][k] = v[j][i] + v[i][k];

            }
        }
    }
}
```



THX~