

DFS, BFS

그래프 1 (연습)

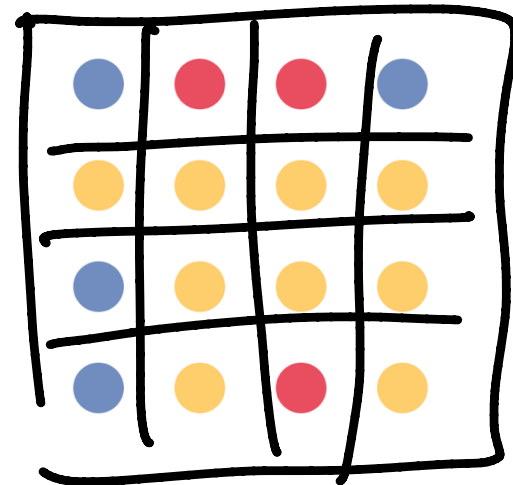
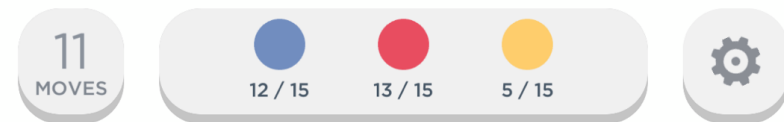
최백준 choi@startlink.io

Two Dots

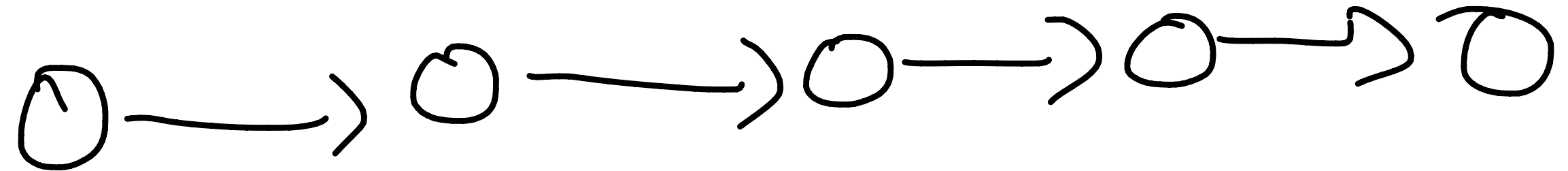
2

<https://www.acmicpc.net/problem/16929>

- Two Dots 게임에서 크기가 4이상인 사이클의 존재 여부를 구하는 문제



DFS



30
LV 1

Two Dots

<https://www.acmicpc.net/problem/16929>

- 존재 여부를 확인하는 것이기 때문에, 최소 4칸 이동할 수 있는지 검사하면 된다.

Two Dots

<https://www.acmicpc.net/problem/16929>

4

(x, y) 시작,

Color


방향을 2씩

$(x, y) \rightarrow (nx, ny)$

```
bool go(int x, int y, int cnt, char color) {  
    if (cnt == 4) return true;  
    for (int k=0; k<4; k++) {  
        int nx = x+dx[k], ny = y+dy[k];  
        if (0 <= nx && nx < n && 0 <= ny && ny < m) {  
            if (check[nx][ny] == false && a[nx][ny] == color) {  
                if (go(nx, ny, cnt+1, color)) {  
                    return true;  
                }  
            }  
        }  
    }  
    return false;  
}
```

Two Dots

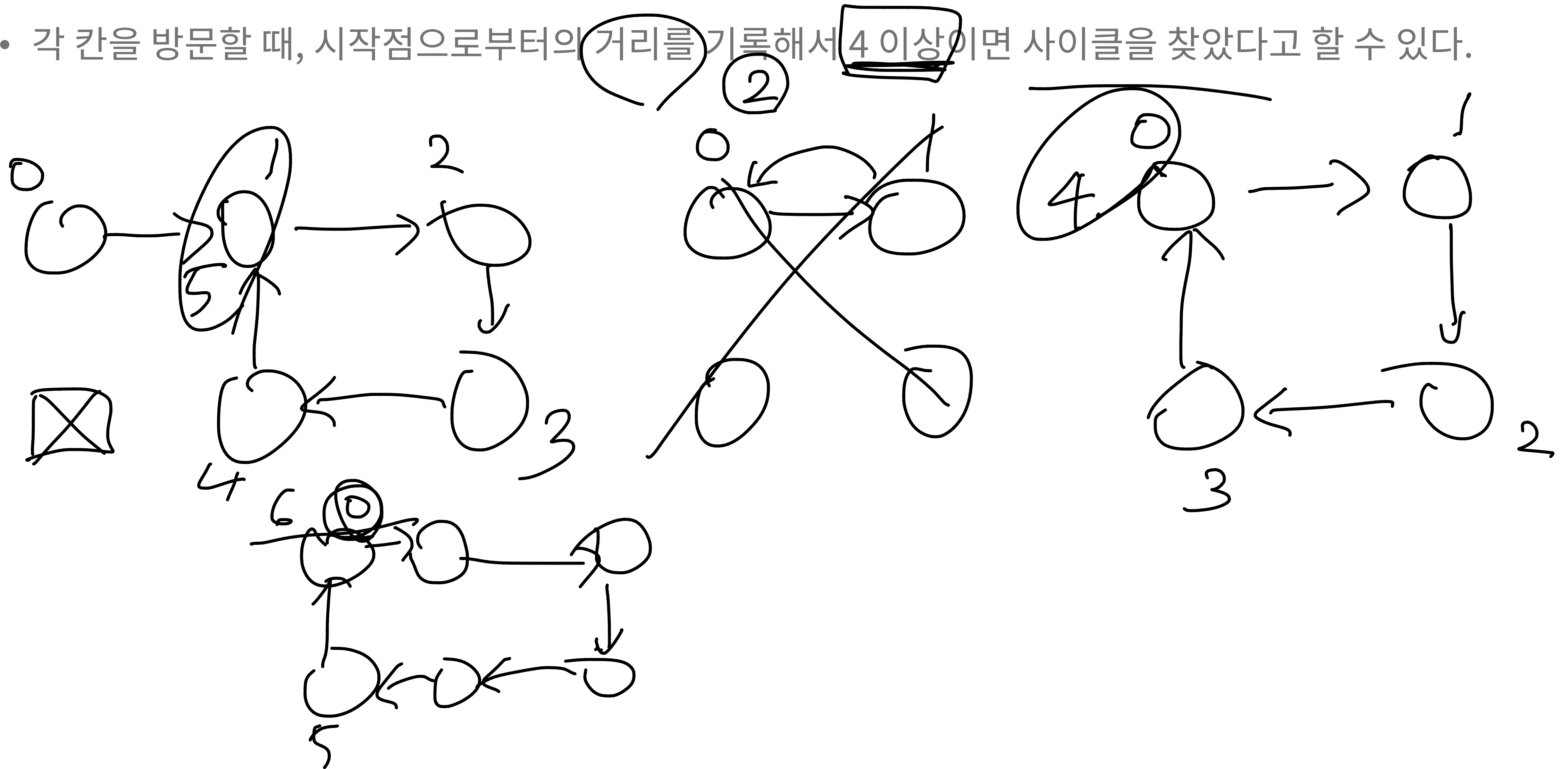
<https://www.acmicpc.net/problem/16929>

- 앞의 소스는 4칸 연속해서 이동할 수 있는지 알아보는 코드다.
 - 사이클을 찾는 코드는 아니다.
- 

Two Dots

<https://www.acmicpc.net/problem/16929>

- 각 칸을 방문할 때, 시작점으로부터의 거리를 기록해서 4 이상이면 사이클을 찾았다고 할 수 있다.



Two Dots

한칸씩 → 한칸씩

(x,y) 방문 횟수 cnt 증가

7

<https://www.acmicpc.net/problem/16929>

```
bool go(int x, int y, int cnt, char color) {  
    if (check[x][y]) return cnt-dist[x][y] >= 4;  
    check[x][y] = true; dist[x][y] = cnt;  
    for (int k=0; k<4; k++) {  
        int nx = x+dx[k], ny = y+dy[k];  
        if (0 <= nx && nx < n && 0 <= ny && ny < m) {  
            if (a[nx][ny] == color) {  
                if (go(nx, ny, cnt+1, color)) return true;  
            }  
        }  
    }  
    return false;  
}
```

(x,y) → (nx,ny)

Two Dots

<https://www.acmicpc.net/problem/16929>

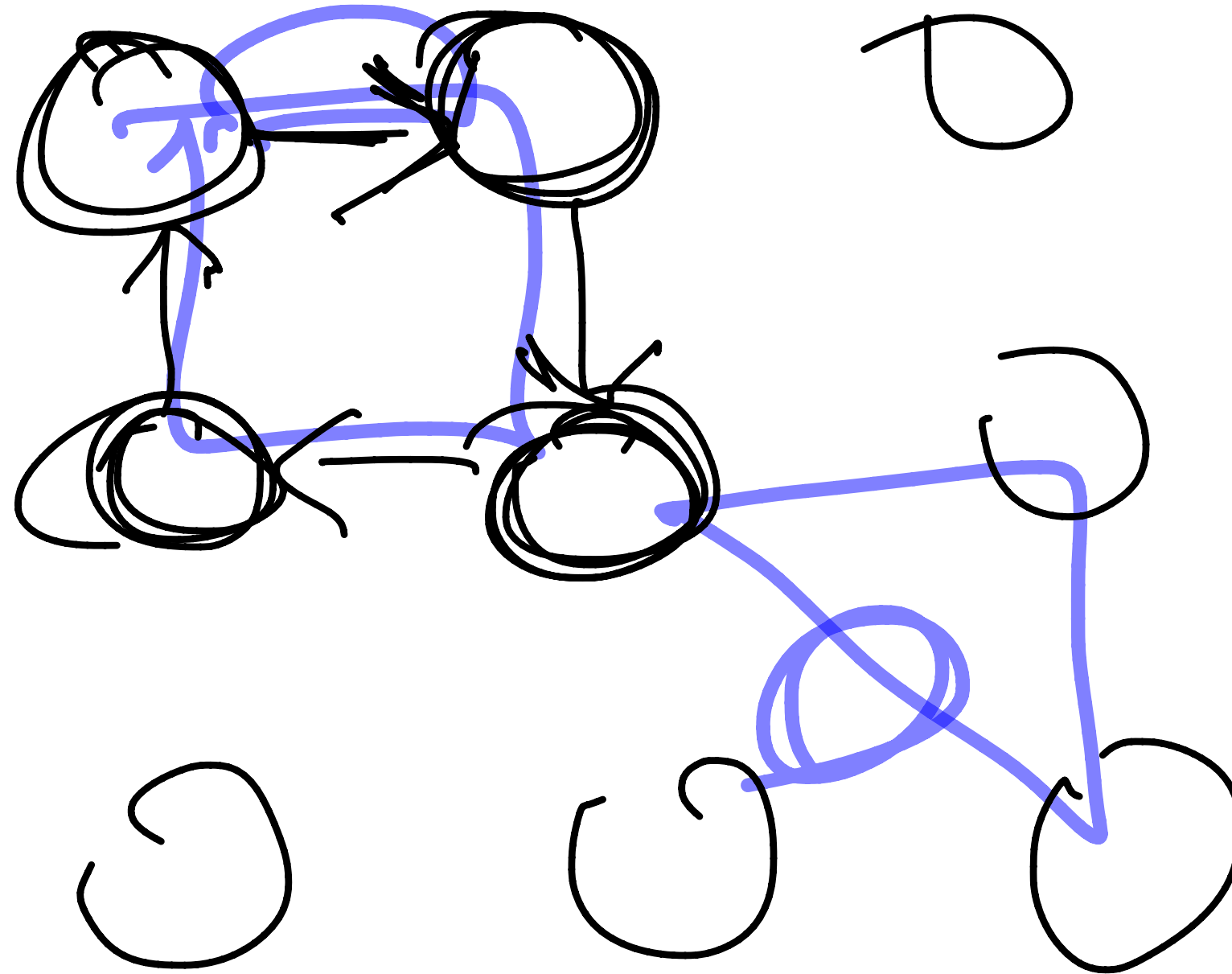
- 소스: <http://codeplus.codes/25890be9a91a4ed5ad85c9444a1ae84a>

Two Dots

<https://www.acmicpc.net/problem/16929>

9

- 이전 칸과 다른 칸으로 연속해서 이동했을 때, 이미 방문한 칸을 방문했으면 사이클이 존재한다고 볼 수 있다.

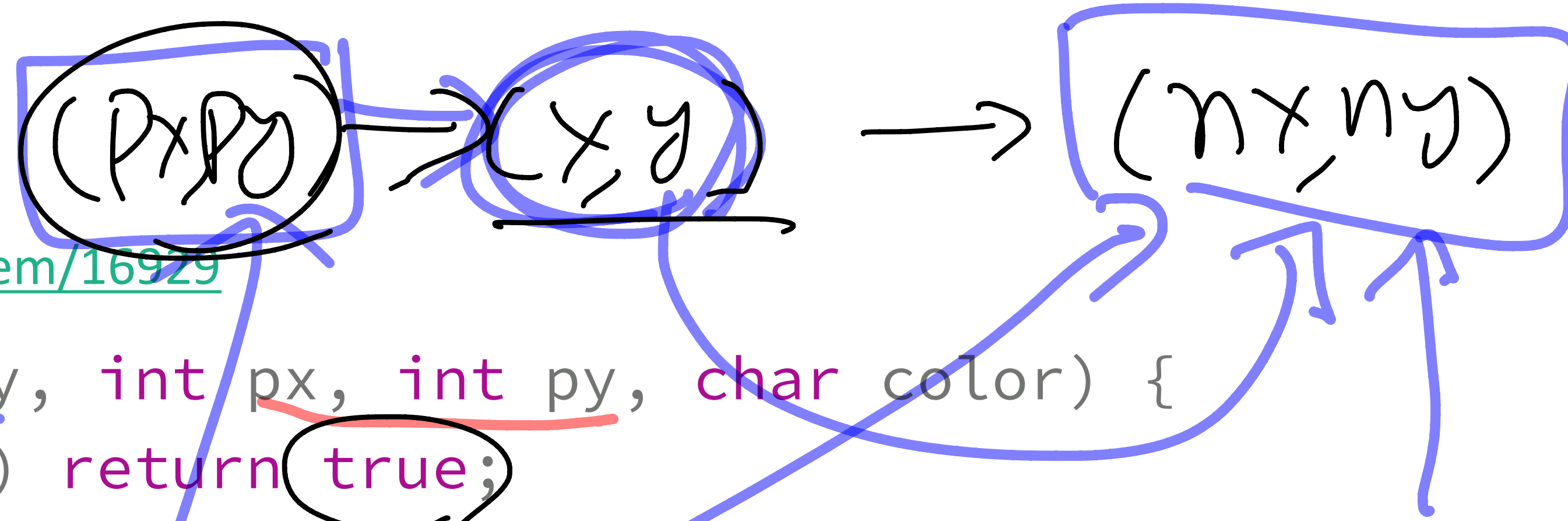


가이글 4미싱

Two Dots

<https://www.acmicpc.net/problem/16929>

10



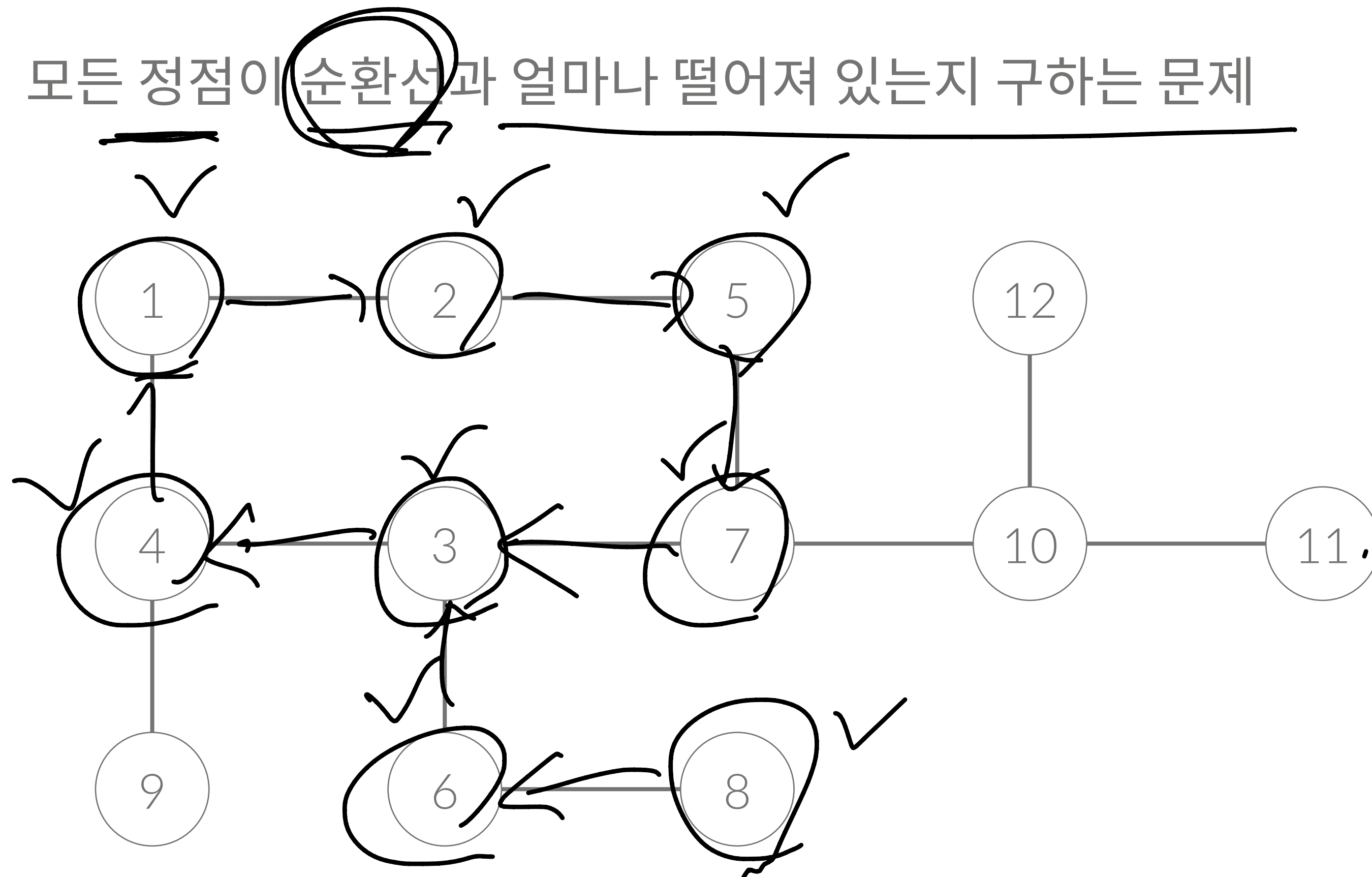
```
bool go(int x, int y, int px, int py, char color) {  
    if (check[x][y]) return true;  
    check[x][y] = true;  
    for (int k=0; k<4; k++) {  
        int nx = x+dx[k], ny = y+dy[k];  
        if (0 <= nx && nx < n && 0 <= ny && ny < m) {  
            if (!(nx == px && ny == py) && a[nx][ny] == color) {  
                if (go(nx, ny, x, y, color)) return true;  
            }  
        }  
    }  
    return false;  
}
```

Two Dots

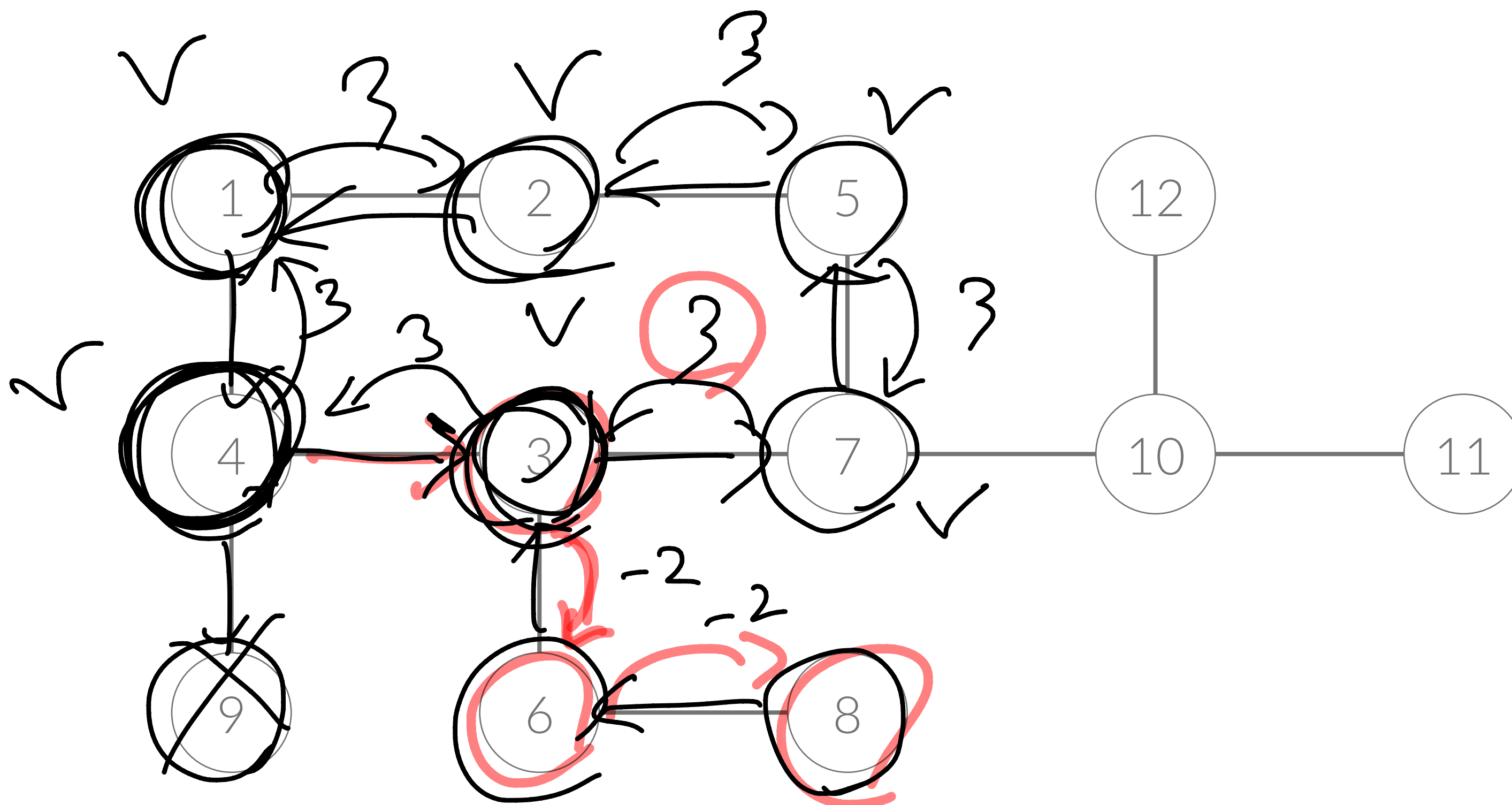
<https://www.acmicpc.net/problem/16929>

- 소스: <http://codeplus.codes/5037e35a4fdb4da88415afb1355bce8e>

- 지하철 2호선은 순환선 1개와 2개의 지선으로 이루어져 있다.
- N개의 정점과 N개의 간선으로 이루어져 있는 그래프로 나타낼 수 있다.
- 모든 정점이 순환선과 얼마나 떨어져 있는지 구하는 문제



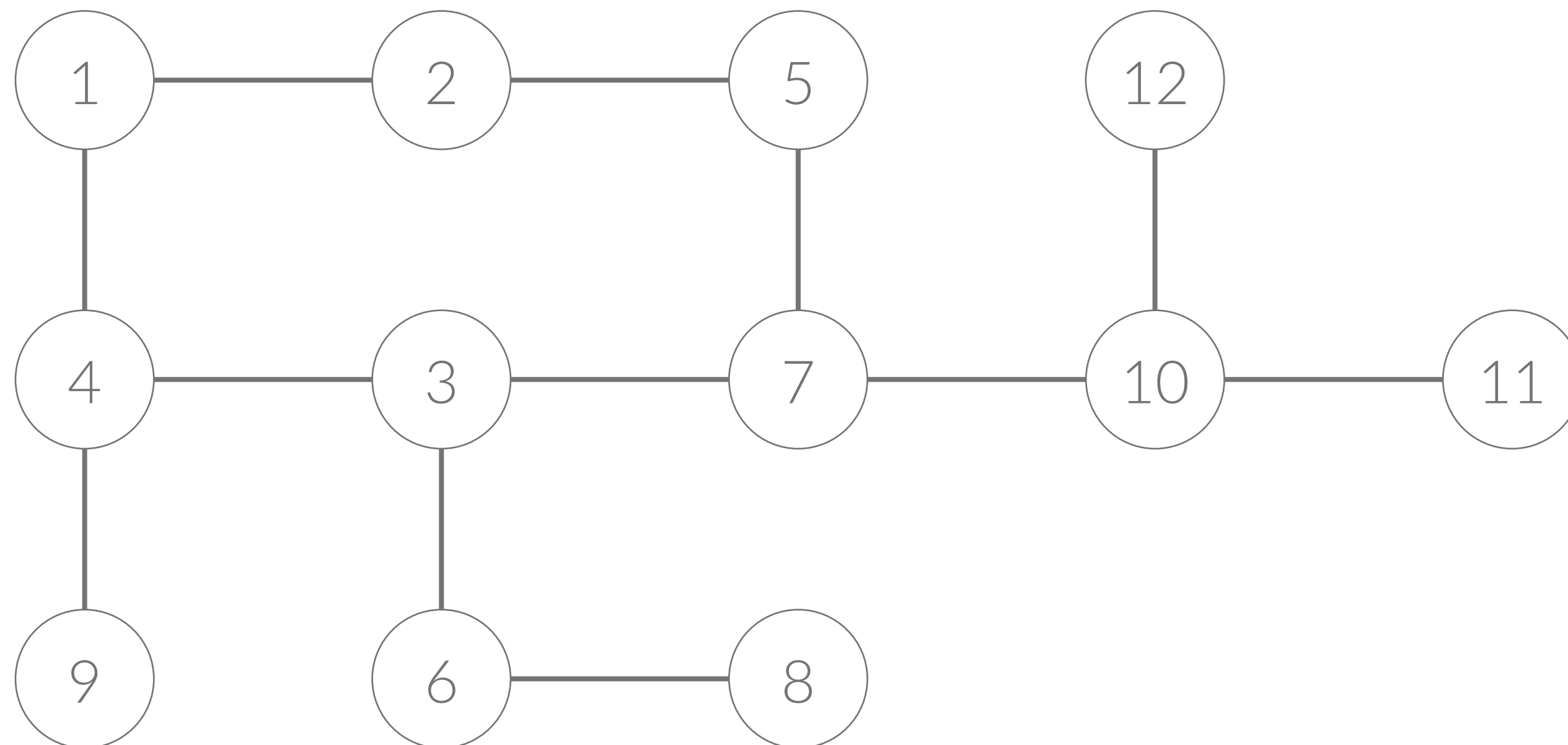
- N개의 정점과 N-1개의 간선으로 이루어져 있는 그래프는 트리이다.
- 여기서 간선 하나가 추가되면 사이클(순환선)은 반드시 하나만 있게 된다.



서울 지하철 2호선

<https://www.acmicpc.net/problem/16947>

- 먼저 순환선을 찾는 과정이 필요하다.



서울 지하철 2호선

<https://www.acmicpc.net/problem/16947>

```
int go(int x, int p) {  
    // -2: 사이클 찾음, 포함되지 않음, -1: 사이클 못 찾음, 0~n-1: 사이클 찾음, 시작 인덱스  
    if (check[x] == 1) return x;  
    check[x] = 1;  
    for (int y : a[x]) {  
        if (y == p) continue;  
        int res = go(y, x);  
        if (res == -2) return -2;  
        if (res >= 0) {  
            check[x] = 2;  
            if (x == res) return -2; else return res;  
        }  
    }  
    return -1;  
}
```

Handwritten diagram illustrating the cycle-finding process. It shows a directed graph with nodes x and y . Node x has a self-loop and an arrow pointing to node y . Node y has an arrow pointing back to node x , forming a cycle. A label p with an arrow points to node x . Another label p with an arrow points to node y . A large circle is drawn around the cycle. A line connects the text "0~n-1: 사이클 찾음, 시작 인덱스" to the cycle.

서울 지하철 2호선

<https://www.acmicpc.net/problem/16947>

- 순환선에 포함된 정점을 시작 정점으로
- BFS로 거리를 계산할 수 있다.

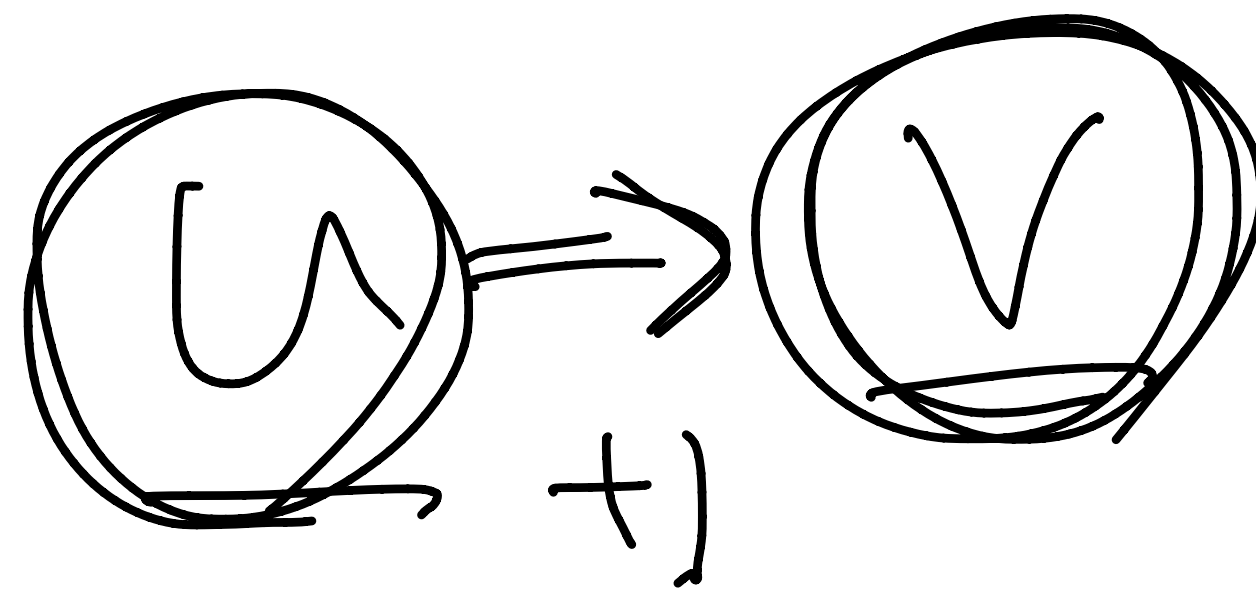
서울 지하철 2호선

17

<https://www.acmicpc.net/problem/16947>

- 소스: <http://codeplus.codes/b830798e0d4943f6bb60e89dc150b60b>

$$\text{dist}[i] = 1 \rightarrow \bar{i} \quad \text{X21}$$



$$\text{dist}[v] = \text{dist}[u] + 1$$

끝

코드 플러스

<https://code.plus>

- 슬라이드에 포함된 소스 코드를 보려면 "정보 수정 > 백준 온라인 저지 연동"을 통해 연동한 다음, "백준 온라인 저지"에 로그인해야 합니다.
- 강의 내용에 대한 질문은 코드 플러스의 "질문 게시판"에서 할 수 있습니다.
- 문제와 소스 코드는 슬라이드에 첨부된 링크를 통해서 볼 수 있으며, "백준 온라인 저지"에서 서비스됩니다.
- 슬라이드와 동영상 강의는 코드 플러스 사이트를 통해서만 볼 수 있으며, 동영상 강의의 녹화와 다운로드, 배포와 유통은 저작권법에 의해서 금지되어 있습니다.
- 다른 경로로 이 슬라이드나 동영상 강의를 본 경우에는 codeplus@startlink.io 로 이메일 보내주세요.
- 강의 내용, 동영상 강의, 슬라이드, 첨부되어 있는 소스 코드의 저작권은 스타트링크와 최백준에게 있습니다.