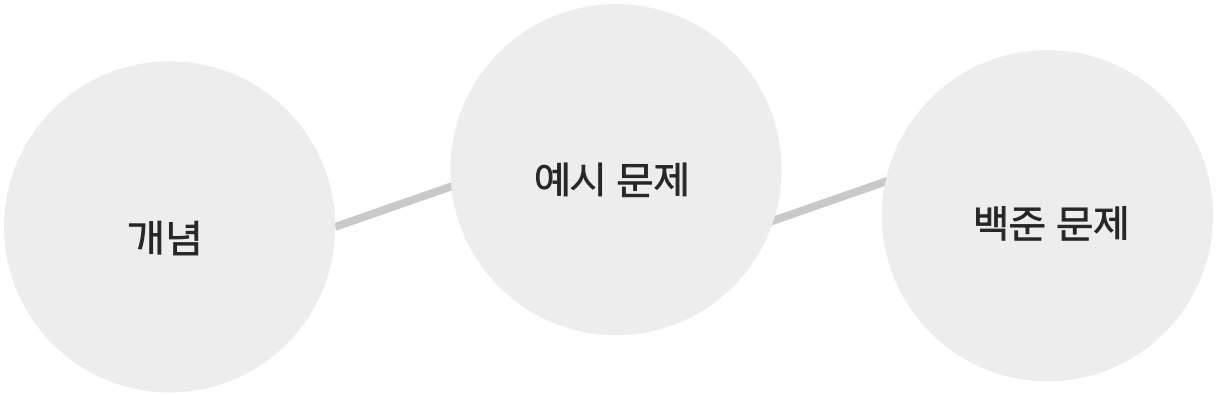
알고리즘 – BFS

너비 우선 탐색(BFS, Breadth-First Search)

목 차



BFS 설명

BFS란?

<그래프 탐색이란>

하나의 정점으로부터 시작하여 차례대로 모든 정점들을 한 번씩 방문하는 것

Ex) 특정 도시에서 다른 도시로 갈 수 있는지 없는지, 전자 회로에서 특정 단자와 단자가 서로 연결되어 있는지

BFS란?

〈BFS〉

시작 으로부터 가까운 정점을 먼저 방문하고 멀리 떨어져 있는 정점을 나중에 방문하는 순회 방법. 즉, 넓게(wide) 탐색하는 것

사용하는 경우: 두 노드 사이의 최단 경로 혹은 임의의 경로를 찾고 싶을 때 이 방법을 선택

Ex) 지구상에 존재하는 모든 친구 관계를 그래프로 표현한 후 Ash와 Vanessa 사이에 존재하는 경로를 찾는 경우

깊이 우선 탐색(dfs)의 경우 - 모든 친구 관계를 다 살펴봐야 할지도 모른다.

너비 우선 탐색(bfs)의 경우 - Ash와 가까운 관계부터 탐색

너비 우선 탐색(BFS)이 깊이 우선 탐색(DFS)보다 좀 더 복잡하다.

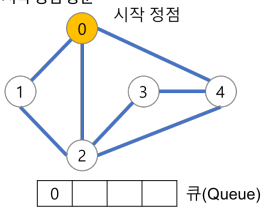
BFS 특징

어떤 노드를 방문 했었는지 여부를 반드시 검사 → 검사하지 않을 경우 무한루프에 빠질 위험

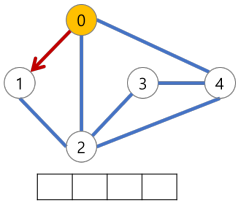
BFS는 방문한 노드들을 차례로 저장한 후 꺼낼 수 있는 자료 구조인 큐(Queue) 사용 → 선입선출(FIFO) 원칙으로 탐색

BFS 과정

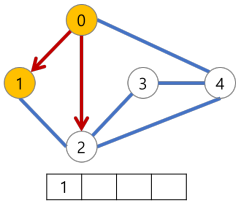
(1) 시작 정점 방문



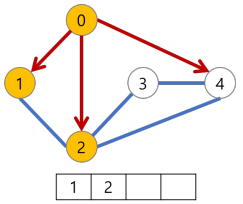
(2)



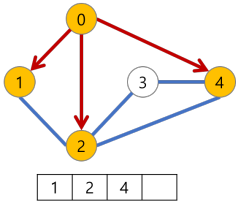
(3)



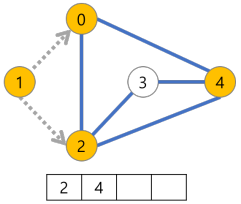
(4)



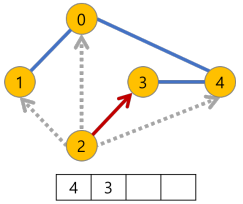
(5)



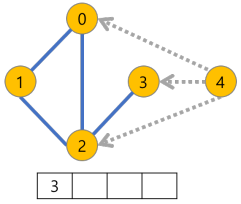
(6)



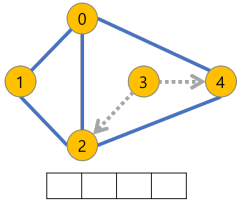
(7)



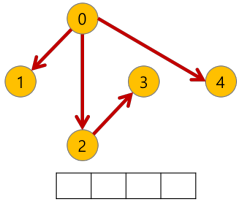
(8)



(9)



(10) 탐색 결과(방문 순서: 0,1,2,4,3)



BFS 과정

0(시작 root)에서 시작

0과 이어진 곳들 queue에 집어넣기 : 순서 상관 없음

더 이상 이어진 곳이 없으면 첫 번째 queue에 들어있는 것 꺼내기 – 기준 노드로 설정

1(기준 node)에서 시작

1과 이어져 있는 것 탐색 : 0은 이미 지나왔고, 2는 queue에 이미 담겨 있음 – pass

Queue에서 새로운 것 다시 꺼내기 – 기준 노드 설정(2)

2(기준 node)에서 시작

2과 이어져 있는 것 탐색 : 0, 1은 이미 지나왔고, 3은 queue에도 없고 지나온 것도 아님, 4는 이미 queue에 담겨 있음

Queue에서 새로운 것 다시 꺼내기 – 기준 노드 설정(4)

4(기준 node)에서 시작

4과 이어져 있는 것 탐색 : 0, 2은 이미 지나왔고, 3은 이미 queue에 담겨 있음

Queue에서 새로운 것 다시 꺼내기 – 기준 노드 설정(3)

3(기준 node)에서 시작

3과 이어져 있는 것 탐색 : 0, 2은 모두 이미 방문한 곳, queue는 다 비워졌음

끝!

BFS 예제 문제

BFS 예제 문제(https://www.acmicpc.net/problem/2667)

<그림 1>과 같이 정사각형 모양의 지도가 있다.

1은 집이 있는 곳을, 0은 집이 없는 곳을 나타낸다.

철수는 이 지도를 가지고 연결된 집들의 모임인 단지를 정의하고, 단지에 번호를 붙이려 한다.

여기서 연결되었다는 것은 어떤 집이 좌우, 혹은 아래위로 다른 집이 있는 경우를 말한다. (대각선 상에 집이 있는 경우는 연결된 것이 아니다.)

<그림 2>는 <그림 1>을 단지별로 번호를 붙인 것이다.

지도를 입력하여 단지 수를 출력하고, 각 단지에 속하는 집의 수를 오름차순으로 정렬하여 출력하는 프로그램을 작성하시오.

0	1	1	0	1	0	0
0	1	1	0	1	0	1
1	1	1	0	1	0	1
0	0	0	0	1	1	1
0	1	0	0	0	0	0
0	1	1	1	1	1	0
0	1	1	1	0	0	0

<그림 1>

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

<그림 2>

BFS 예제 문제

〈입력〉

첫 번째 줄에는 지도의 크기 N(정사각형이므로 가로와 세로의 크기는 같으며 5≤N≤25)이 입력되고, 그 다음 N줄에는 각각 N개의 자료(0혹은 1)가 입력된다.

〈출력〉

첫 번째 줄에는 총 단지 수를 출력 하시오. 그리고 각 단지내 집의 수를 오름차순으로 정렬하여 한 줄에 하나씩 출력 하시오.

〈예제 입력〉

```
7
0 1 1 0 1 0 0
0 1 1 0 1 0 1
1 1 1 0 1 0 1
0 0 0 0 1 1 1
0 1 0 0 0 0 0
0 1 1 1 1 1 0
0 1 1 1 0 0 0
```

〈예제 출력〉

```
3
7
8
9
```

BFS 예제 문제

접근법

지도에서 하나하나 조사하면서 집을 발견하면 단지로 정의하고 집의 수를 세는 방법으로 풀이가 가능 → 하나하나 조사를 해야하기 때문에 완전탐색

단지 안에서 집의 수를 세는 건 주변으로 뻗어 나가는 방법을 사용 → BFS나 DFS를 통해서 4방향으로 탐색을 진행

풀이법

1. 전체의 지도를 돌면서 집이 있는지 확인
2. 집(배열의 값 == 1)을 발견하면 단지로 정의하고 ID를 부여
3. BFS 또는 DFS 탐색을 통해서 해당 단지의 집의 수를 구하기
4. 해당 단지의 집의 수를 다 구하게 되면 해당 집의 ID를 인덱스로 하는 배열에 추가
5. 전체 지도의 탐색이 다 끝나면 배열을 오름차순으로 정렬하고 출력

BFS 예제 해설

```
// BFS
void bfs(int x, int y){

    queue< pair<int,int> > q; // 이용할 큐, (x,y) -> (행, 열)
    q.push(make_pair(x,y)); // pair를 만들어서 queue에 넣습니다.

    // 처음 x,y를 방문 했기때문에
    visited[x][y] = true;
    groups[group_id]++;

    while(!q.empty()){

        // 큐의 현재 원소를 꺼내기
        x = q.front().first;
        y = q.front().second;
        q.pop();

        // 해당 위치의 주변을 확인
        for(int i = 0; i < 4; i++){
            int nx = x + dx[i];
            int ny = y + dy[i];
```

BFS 예제 해설

```
            // 지도를 벗어나지 않고
            if(0 <= nx && nx < n && 0 <= ny && ny < n){
              // 집이면서 방문하지 않았다면 -> 방문
              if(map[nx][ny] == 1 && visited[nx][ny] == false){
                visited[nx][ny]=true;

                // 해당 단지의 집의 수를 증가시킴
                groups[group_id]++;

                // 큐에 현재 nx,ny를 추가
                q.push(make_pair(nx,ny));
              }
            }
          }
        }
    }
```

BFS 문제

BFS 문제(양)(https://www.acmicpc.net/problem/3184)

양

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	2272	1358	1071	61.729%

문제

미키의 뒷마당에는 특정 수의 양이 있다. 그가 푹 잠든 사이에 배고픈 늑대는 마당에 들어와 양을 공격했다.

마당은 행과 열로 이루어진 직사각형 모양이다. 글자 '.' (점)은 빈 필드를 의미하며, 글자 '#'는 울타리를, 'o'는 양, 'v'는 늑대를 의미한다.

한 칸에서 수평, 수직만으로 이동하며 울타리를 지나지 않고 다른 칸으로 이동할 수 있다면, 두 칸은 같은 영역 안에 속해 있다고 한다. 마당에서 "탈출"할 수 있는 칸은 어떤 영역에도 속하지 않는다고 간주한다.

다행히 우리의 양은 늑대에게 싸움을 걸 수 있고 영역 안의 양의 수가 늑대의 수보다 많다면 이기게 된다. 그렇지 않다면 늑대가 그 지역 안의 모든 양을 먹는다.

맨 처음 모든 양과 늑대는 마당 안 영역에 존재한다.

아침이 도달했을 때 살아남은 양과 늑대의 수를 출력하는 프로그램을 작성하라.

BFS 문제(적록색약)(https://www.acmicpc.net/problem/10026)

적록색약

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	9266	5224	4174	57.636%

문제

적록색약은 빨간색과 초록색의 차이를 거의 느끼지 못한다. 따라서, 적록색약인 사람이 보는 그림은 아닌 사람이 보는 그림과는 좀 다를 수 있다.

크기가 N×N인 그리드의 각 칸에 R(빨강), G(초록), B(파랑) 중 하나를 색칠한 그림이 있다. 그림은 몇 개의 구역으로 나뉘어져 있는데, 구역은 같은 색으로 이루어져 있다. 또, 같은 색상이 상하좌우로 인접해 있는 경우에 두 글자는 같은 구역에 속한다. (색상의 차이를 거의 느끼지 못하는 경우도 같은 색상이라 한다)

예를 들어, 그림이 아래와 같은 경우에

```
RRRBB
GGBBB
BBBRR
BBRRR
RRRRR
```

적록색약이 아닌 사람이 봤을 때 구역의 수는 총 4개이다. (빨강 2, 파랑 1, 초록 1) 하지만, 적록색약인 사람은 구역을 3개 볼 수 있다. (빨강-초록 2, 파랑 1)

그림이 입력으로 주어졌을 때, 적록색약인 사람이 봤을 때와 아닌 사람이 봤을 때 구역의 수를 구하는 프로그램을 작성하시오.

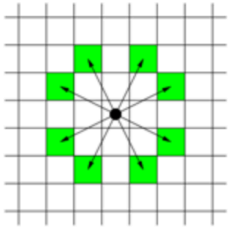
BFS 문제(나이트의 이동)(https://www.acmicpc.net/problem/7562)

나이트의 이동

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	256 MB	13600	6027	4567	43.825%

문제

체스판 위에 한 나이트가 놓여져 있다. 나이트가 한 번에 이동할 수 있는 칸은 아래 그림에 나와있다. 나이트가 이동하려고 하는 칸이 주어진다. 나이트는 몇 번 움직이면 이 칸으로 이동할 수 있을까?



끝!