Chapter. 02

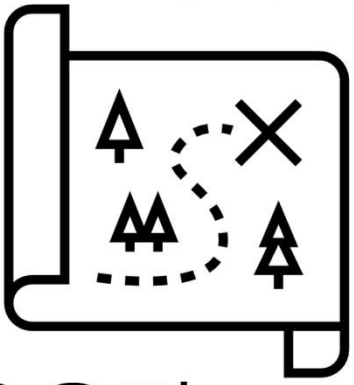
알고리즘

그래프와 탐색
Graph & Search – (응용편)

FAST CAMPUS
ONLINE
알고리즘 공채 대비반I

강사. 류호석

Life Changing Education



Chapter. 02

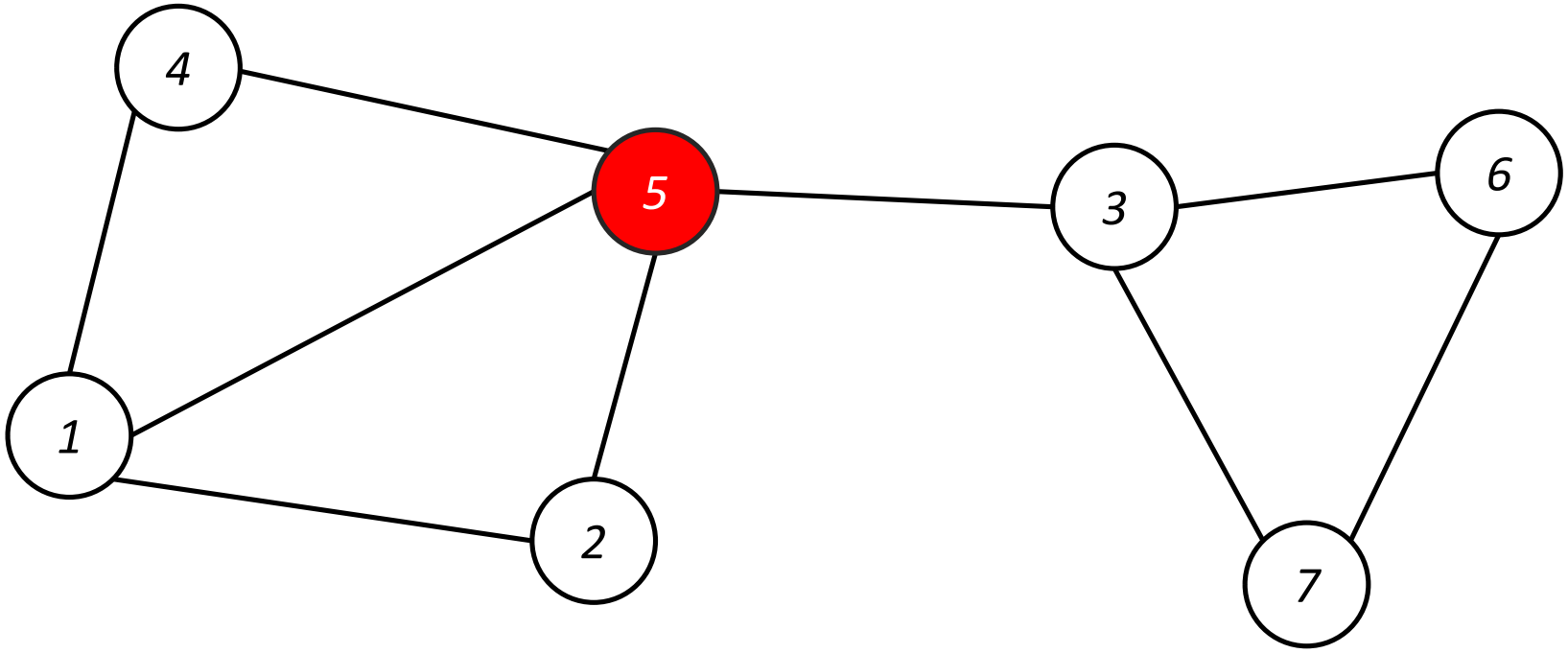
알고리즘
그래프와 탐색(Graph & Search) - 응용편

FAST CAMPUS
ONLINE
류호석 강사.

I BFS의 부가 효과

탐색(Search) = 시작점에서 갈 수 있는 정점들은?

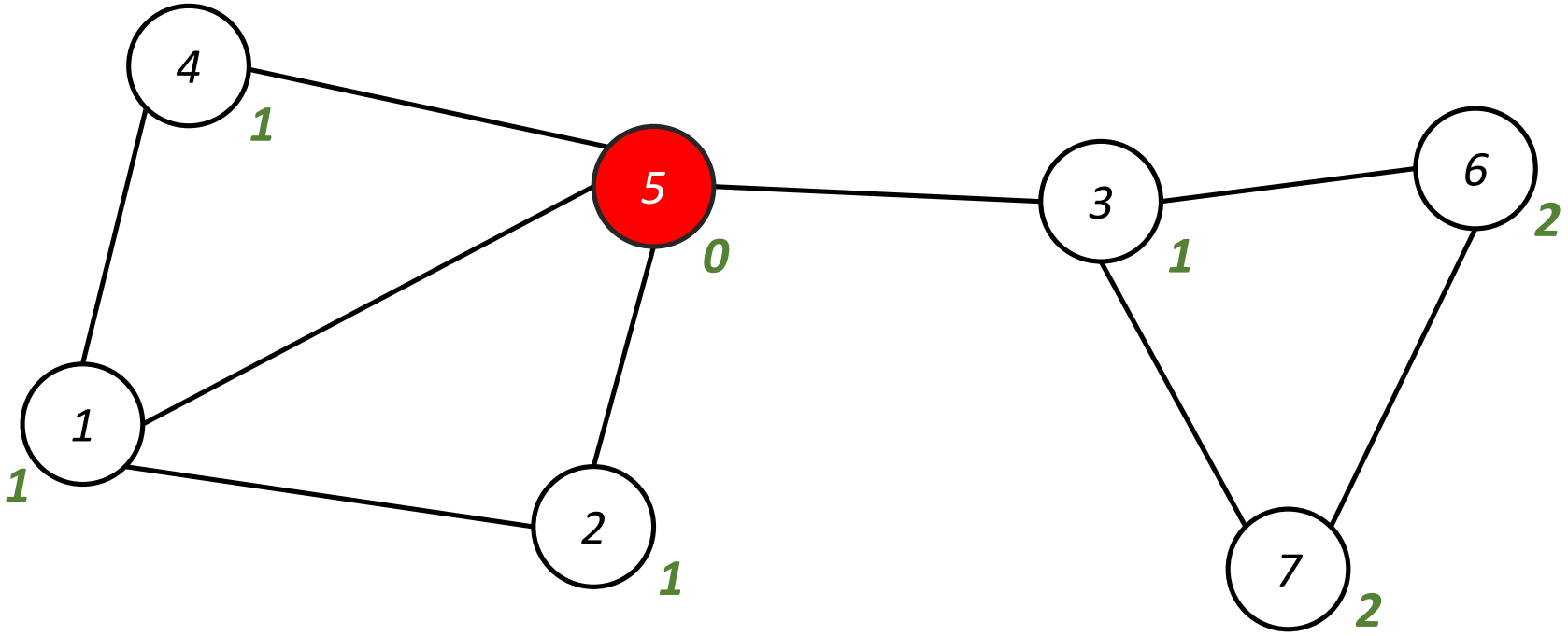
~~몇 번의 이동~~이 필요한가?



I BFS의 부가 효과

탐색(Search) = **시작점**에서 갈 수 있는 정점들은?

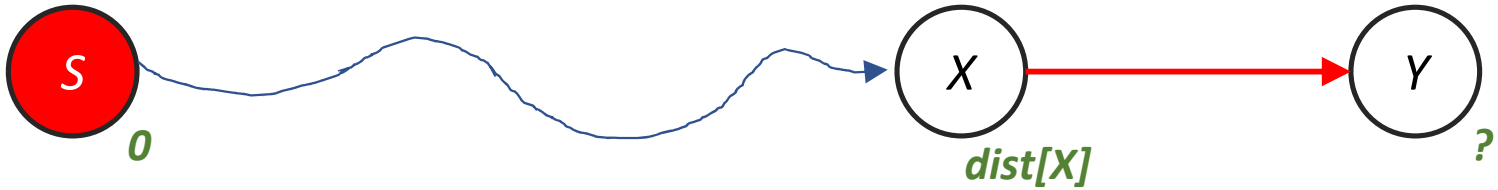
BFS ➔ 다른 정점까지 **최소 이동 횟수**도 계산 가능!



I BFS의 부가 효과

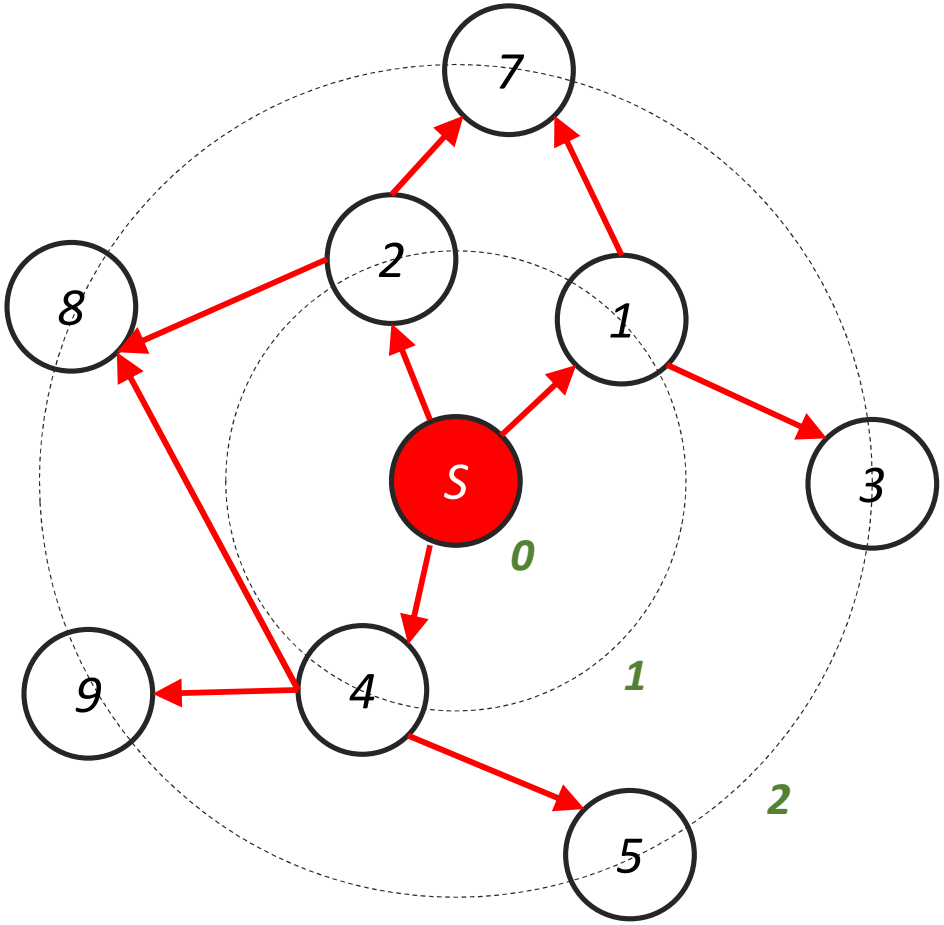
BFS ➔ 다른 정점까지 **최소 이동 횟수**도 계산 가능!

dist[i] : S에서 i 까지 갈 때 필요한 최소 간선 개수, 불가하면 -1




```
dist[Y] = dist[X] + 1
```

I 그림을 통한 이해



I "최소 이동 횟수", "최단 시간" 키워드

최소 "이동" 횟수와 관련된 것이기 때문에,
가중치에 대한 개념이 없는 문제에서만 생기는 부가 효과!

동생을 찾을 수 있는 가장 빠른 시간이 몇 초 후

이동할 때 지나야 하는 최소의 칸 수

고슴도치가 안전하게 비버의 굴로 이동하기 위해 필요한 최소 시간

때로는 그래프가 없는 문제에서 "정점"과 "간선"의 정의를 만들어서 그래프 문제로 접근해야 한다.

BOJ 2178 – 미로 탐색

난이도: 2

$2 \leq$ 지도의 크기, $N, M < 100$

$M = 6$

$N = 4$

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

정답: 15

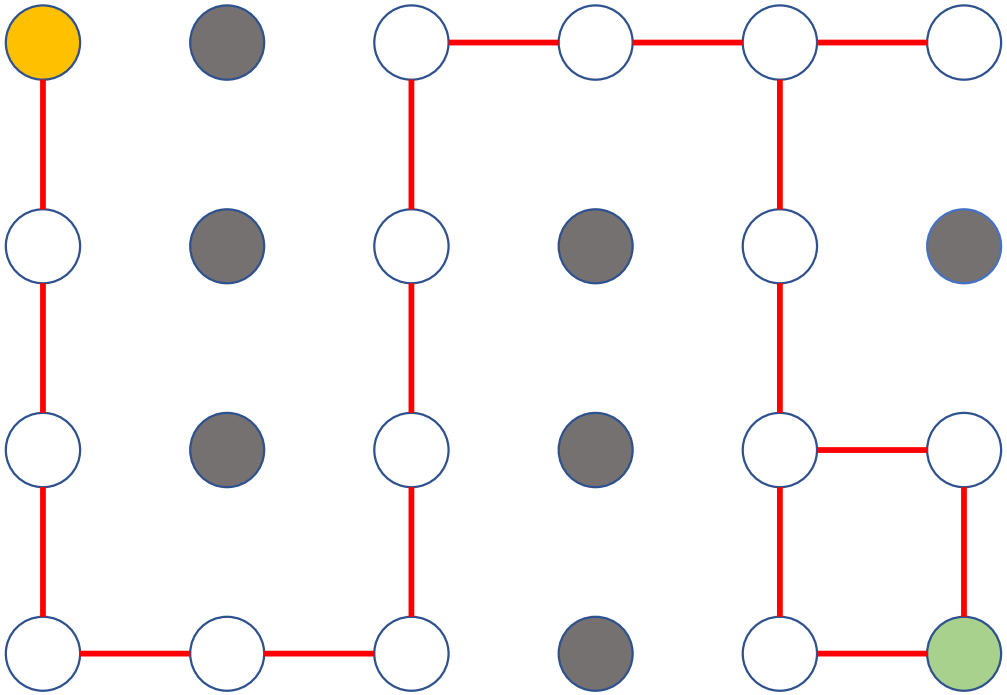
I 문제 파악하기 – 정답의 최대치

1	1	1	1	…	1	1
0	0	0	0	…	0	1
1	1	1	1	…	1	1
1	0	0	0	…	0	0
…	…	…	…	…	…	…
1	0	0	0	…	0	0
1	1	1	1	1	1	1

밟게 되는 최대 개수 = $O(N\text{^}2)$

I 접근 – 격자형 그래프 구성

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



BFS ➔ 다른 정점까지 **최소 이동 횟수**도 계산 가능!

I 시간, 공간 복잡도 계산하기

<격자형 그래프>

정점: $O(N^2)$

간선: $O(N^2 \times 4)$

BFS를 사용하므로 시간, 공간 복잡도는 모두 $O(N^2)$

I 구현

```
// x, y 를 갈 수 있다는 걸 알고 방문한 상태
static void bfs(int x, int y) {
    // dist 배열 초기화
    /* TODO */

    // (x, y)를 Q에 넣어주고, visit 표시와 dist 값 초기화
    /* TODO */

    // BFS 과정 시작
    /* TODO */
}

static void pro() {
    // 시작점이 (0, 0)인 탐색 시작
    /* TODO */

    // (N-1, M-1)까지 필요한 최소 이동 횟수 출력
    /* TODO */
}
```

I 연습 문제

- BOJ 7562 – 나이트의 이동
- BOJ 2644 – 촌수 계산
- BOJ 18404 – 현명한 나이트

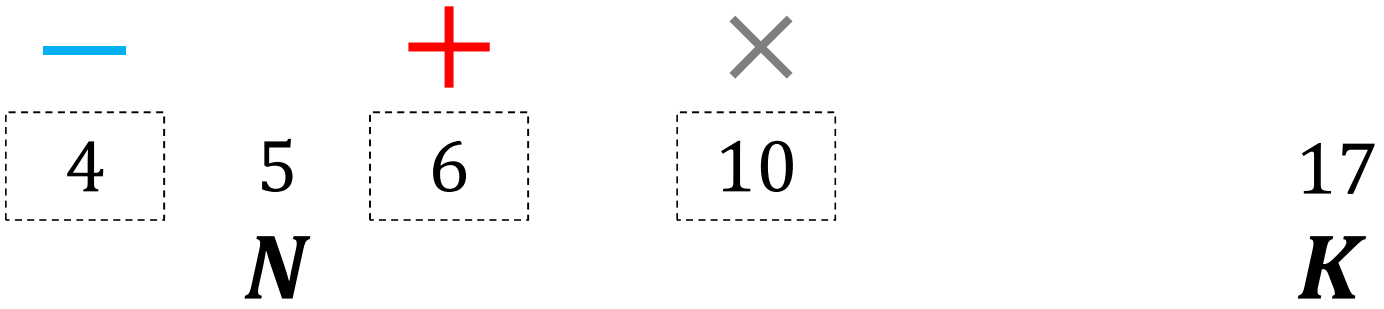
이외의 추천 문제가 추가되면 Github 자료에 코드 업로드

I BOJ 1697 – 숨바꼭질

난이도: 3

$0 \leq$ 수빈이의 위치, $N < 100{,}000$

$0 \leq$ 동생의 위치, $K < 100{,}000$



정답: 4

이유: 5 ➔ 10 ➔ 9 ➔ 18 ➔ 17

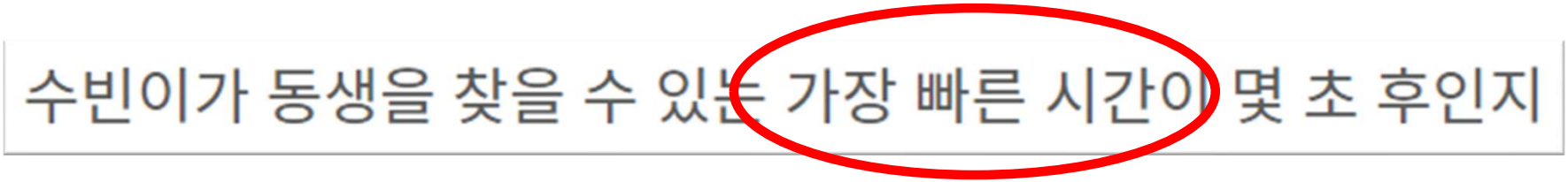
I 문제 파악하기 – 정답의 최대치

문제에 대한 관찰 연습! 언제 가장 오래 걸릴까?

N > K 이라면, 갈 수 있는 방법이 1 씩 감소하는 것 뿐이다.

즉, N=10만, K=0 인 경우가 10만 초로 제일 오래 걸린다!

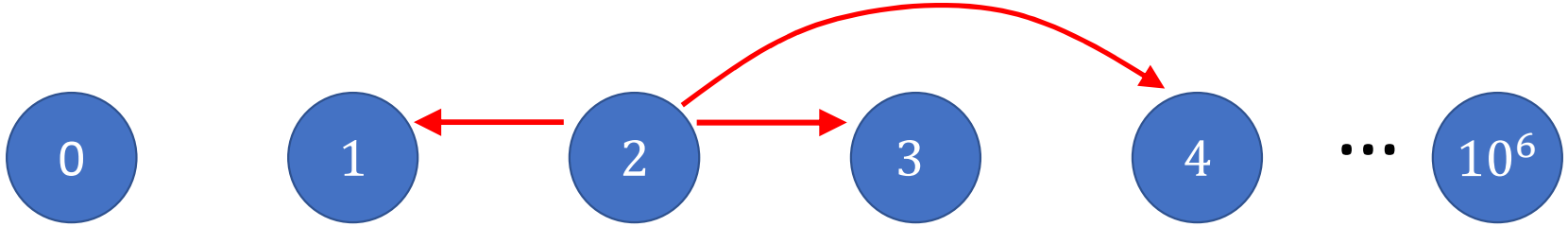
I 접근 – "**최소 연산 횟수**" 키워드



하지만!!! BFS를 하려고 해도, Graph 가 주어지지 않아서 불가능하다!

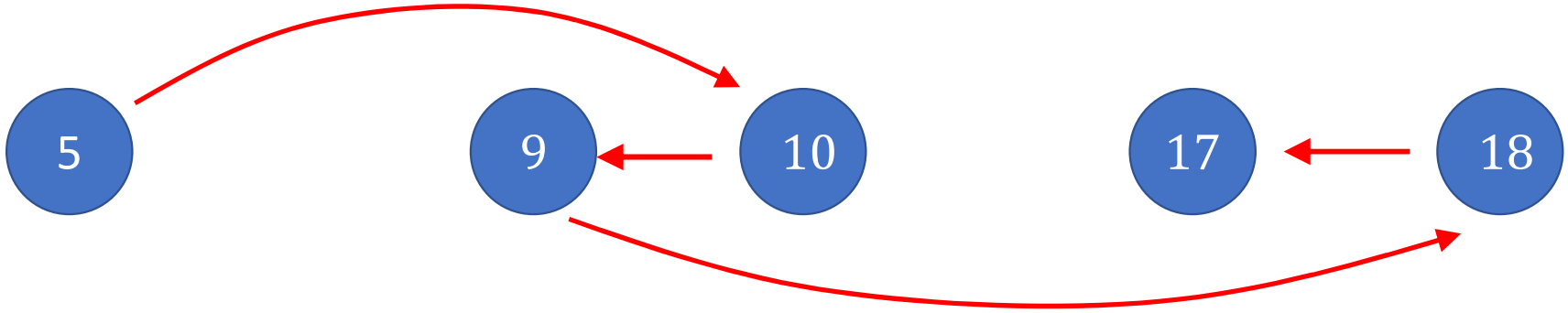
혹시 주어진 정보들을 가지고 정점과 간선을 정의할 수 있을까?

I 접근 – Graph 창조해 보기



- 정점
  - 보통 문제에서 표현하는 “하나의 상태”가 “하나의 정점”
  - 이 문제는 **“점의 번호”**가 곧 “정점의 번호”
- 간선
  - 이동을 의미하는 것을 간선으로 표현하기
  - 이 문제는 각 점마다 -1, +1, *2 인 점을 향하는 **“방향성 간선”**

I 접근 – Graph 창조해 보기



간선 하나를 타고 이동하는 행위 ➔ 1초 동안 이동하는 행위

가장 빨리 동생을 찾는 방법 ➔ 최소 개수로 간선 이동하는 방법

I 시간, 공간 복잡도 계산하기

<새로 만든 그래프>

정점: $O(10^5)$

간선: $O(10^5 \times 4)$

BFS를 사용하므로 시간, 공간 복잡도는 모두 $O(10^6)$

I 구현

```
// 숨바꼭질 시작~
static void bfs() {
    Queue<Integer> Q = new LinkedList<>();
    /* TODO */

    // BFS 과정 시작
    /* TODO */
}

static void pro() {
    bfs();
    System.out.println(dist[K]);
}
```

I 연습 문제

- BOJ 1389 – 케빈 베이컨의 6단계 법칙
- BOJ 5567 – 결혼식

이외의 추천 문제가 추가되면 Github 자료에 코드 업로드

BOJ 3055 – 탈출

난이도: 4

$1 \leq$ 지도의 크기, $N, M \leq 50$

$M = 6$

$N = 5$

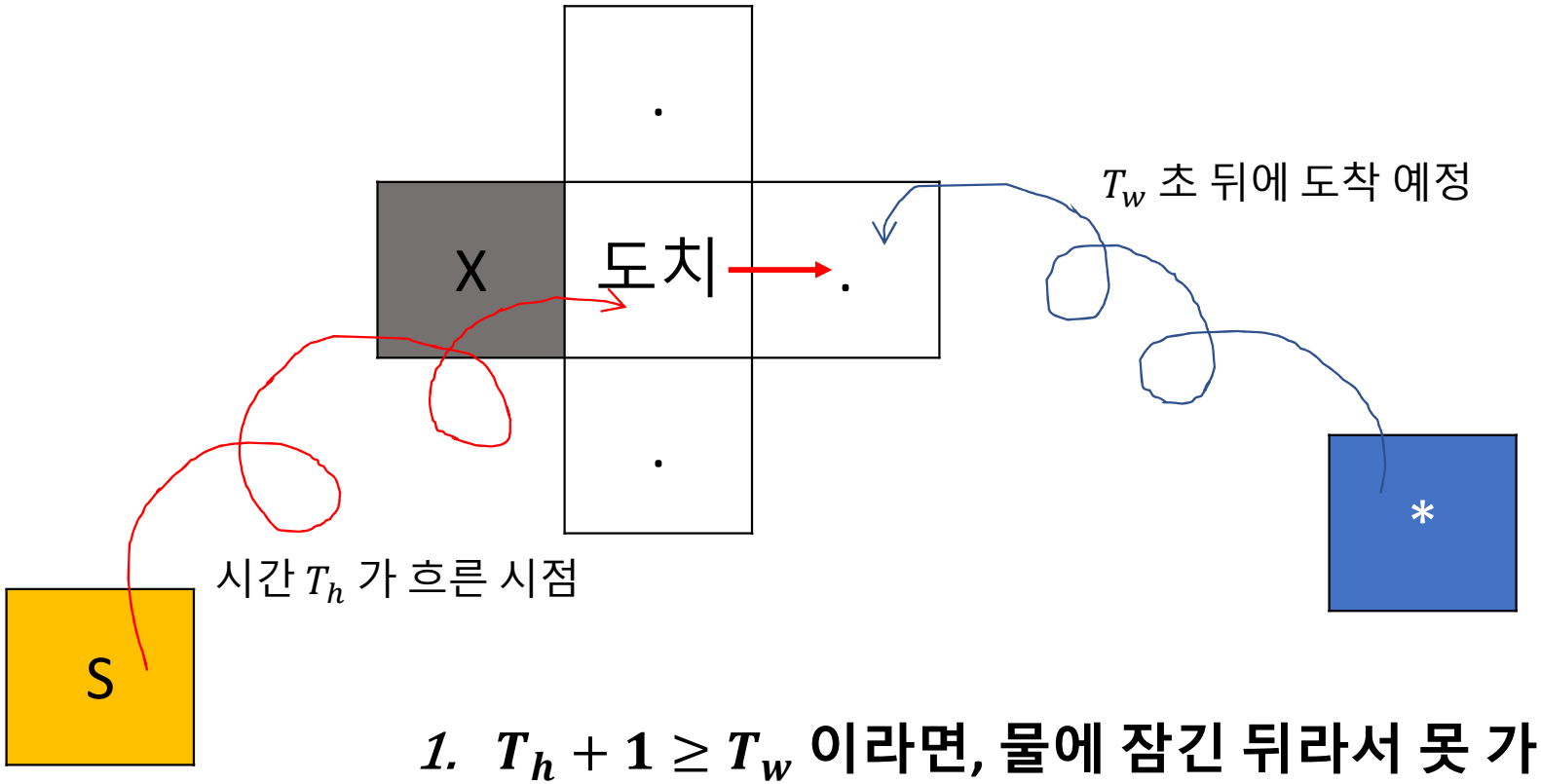
*	.	D	.	.	*
X	X	.	X	.	.
.	.	.	X	S	.
.	X	X	X	.	.
.	.	.	.	.	.

정답: 12

I 문제 파악하기 – 정답의 최대치

밟게 되는 최대 개수 = $O(N^2)$

I 접근 – 고슴도치의 행동 방법



1. $T_h + 1 \geq T_w$ 이라면, 물에 잠긴 뒤라서 못 가는 곳!
2. 벽도 못 가는 곳!

I 접근 – 각 칸이 물에 잠기는 시간 계산

*	1	D	2	1	*
X	X	14	X	2	1
11	12	13	X	S	2
10	X	X	X	4	3
9	8	7	6	5	4

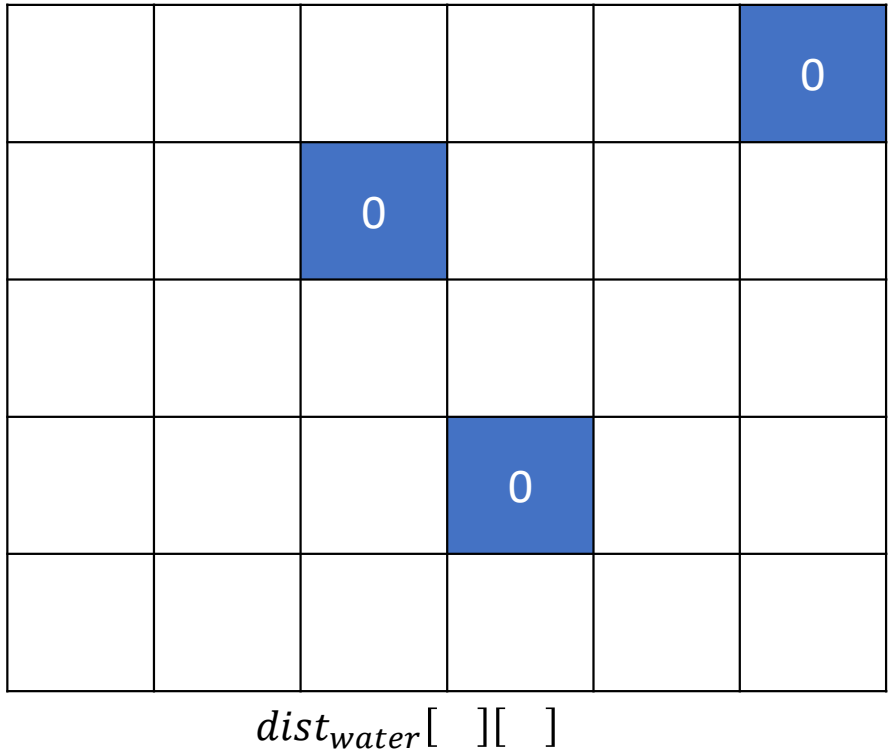
$dist_{water}[\ ][\ ]$

$dist_{water}[i][j]$
$:= (i,j)$가 물에 잠기는 시간

:= 물이 해당하는 칸에
도달하기까지 걸리는 시간

즉, 물에서 시작하는 BFS를 하자!

I 접근 – 각 칸이 물에 잠기는 시간 계산

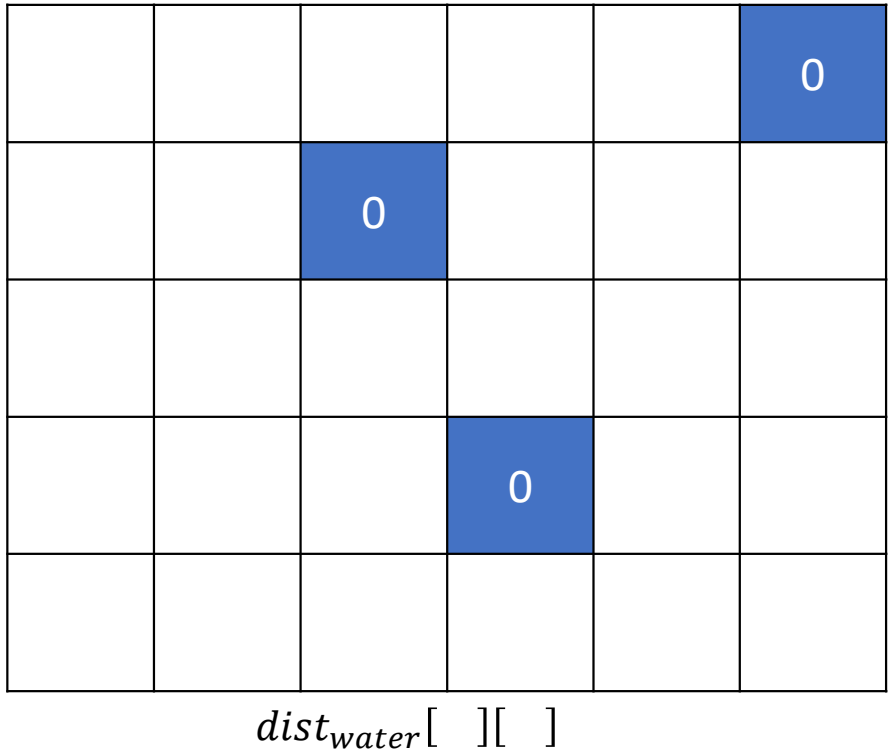


문제점! 시작점이 여러 개다!

모든 물마다 BFS를 각자 하면?

➔ $O(N^2 * N^2)$

I 접근 – 각 칸이 물에 잠기는 시간 계산



문제점! 시작점이 여러 개다!

모든 물을 동시에 Queue에 넣고 시작하면?

➔ 단 한 번의 $BFS, O(N^2)$

I 접근 – 각 칸이 물에 잠기는 시간 계산

*	1	D	2	1	*
X	X	14	X	2	1
11	12	13	X	S	2
10	X	X	X	4	3
9	8	7	6	5	4

$dist_{water}[\ \ ][\ \ ]$

$dist_{hedgehog}[i][j]$

:= 고슴도치가 물을 피해서 (i, j)에 도달하는 최소 시간

$dist_{water}$를 이미 구했으니 어떤 칸에 고슴도치가 갈 수 있는 지를 알 수 있다.

➔ 단 한 번의 BFS, $O(N\text{^}2)$

I 시간, 공간 복잡도 계산하기

1. $dist_{water}$를 $O(N^2)$에 계산

2. $dist_{hedgehog}$를 $O(N^2)$에 계산

총 2 번의 BFS를 이용

I 구현

```java
static void pro() {
    // 각 칸마다 물에 닿는 시간 계산하기
    bfs_water();

    // 고슴도치가 물을 피해 탐색할 수 있는 공간 찾기
    bfs_hedgehog();

    // 탈출구 'D' 에 대한 결과를 통해 정답 출력하기
    for (int i=0;i<N;i++){
        for (int j=0;j<M;j++){
            if (a[i].charAt(j) == 'D'){
                /* TODO */
            }
        }
    }
}
```

```java
// 모든 물들을 시작으로 동시에 BFS 시작!
static void bfs_water() {
    Queue<Integer> Q = new LinkedList<>();
    // 모든 물의 위치를 Q에 전부 넣어주기!
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            // dist_water 와 visit 배열 초기화
            /* TODO */
            if (a[i].charAt(j) == '*') {
                /* TODO */
            }
        }
    }

    // BFS 과정 시작
    /* TODO */
}

// 고슴도치를 시작으로 동시에 BFS 시작!
static void bfs_hedgehog() {
    Queue<Integer> Q = new LinkedList<>();
    // 고슴도치 위치를 Q에 넣어주기!
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            // dist_hedgehog 와 visit 배열 초기화
            /* TODO */
            if (a[i].charAt(j) == 'S') {
                /* TODO */
            }
        }
    }

    // BFS 과정 시작
    /* TODO */
}
```

I 연습 문제

- BOJ 7569 – 토마토
- BOJ 2644 – 촌수 계산

이외의 추천 문제가 추가되면 Github 자료에 코드 업로드