JavaScript 백트래킹 알고리즘

백트래킹 문제 풀이

백트래킹 문제 풀이 | 코딩 테스트에서 자주 등장하는 백트래킹 알고리즘 이해하기

강사 나동빈

JavaScript
백트래킹 알고리즘

백트래킹 문제 풀이

N-Queen: https://www.acmicpc.net/problem/9663

혼자 힘으로 풀어보기

JavaScript
백트래킹
백트래킹
문제 풀이

문제 제목: N-Queen

문제 난이도: ★★★☆☆

문제 유형: 백트래킹, 경우의 수, 완전 탐색

추천 풀이 시간: 50분

N-Queen: https://www.acmicpc.net/problem/9663

문제 해결 아이디어

- $N \times N$ 체스 보드 위에 퀸 N개가 서로 공격할 수 없게 놓는 문제다.
- 예를 들어 8×8에 하나의 퀸이 놓여져 있는 예시는 다음과 같다.

		Q					

N-Queen: https://www.acmicpc.net/problem/9663

문제 해결 아이디어

- 8×8에 8개의 퀸을 서로 공격할 수 없게 놓는 예시는 다음과 같다.

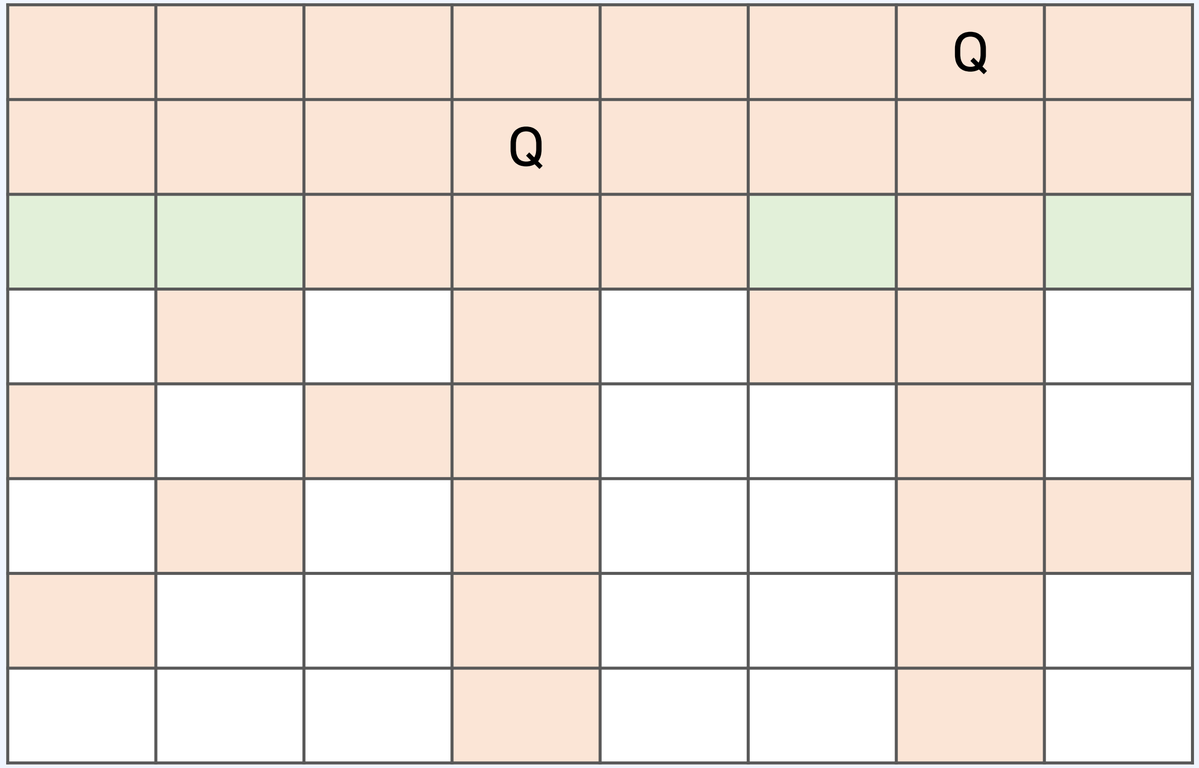
						Q	
			Q				
	Q						
							Q
					Q		
Q							
		Q					
				Q			

N-Queen: https://www.acmicpc.net/problem/9663

문제 해결 아이디어

- 하나의 퀸 A가 이미 존재하는 상태에서, 다른 퀸 B를 놓으려면 어떻게 해야 할까?
- 퀸 A의 "상하좌우 및 대각선" 위치가 아닌 위치에 퀸 B를 놓을 수 있다.

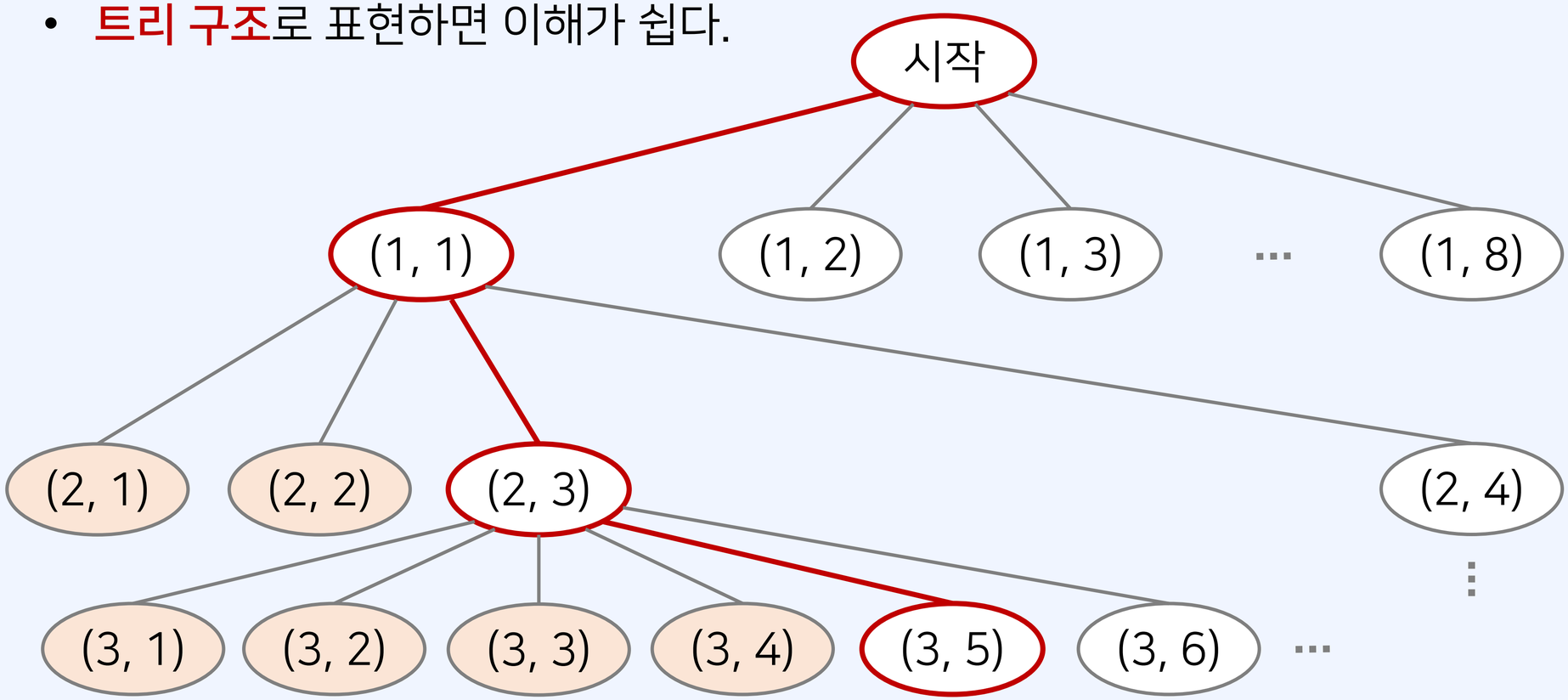
선택



N-Queen: https://www.acmicpc.net/problem/9663

문제 해결 아이디어

- 하나의 퀸 A가 이미 존재하는 상태에서, 다른 퀸 B를 놓으려면 어떻게 해야 할까?
- **트리 구조**로 표현하면 이해가 쉽다.



N-Queen: https://www.acmicpc.net/problem/9663

JavaScript
백트래킹
백트래킹
문제 풀이

문제 해결 아이디어

- 다만, 이 문제는 가능한 모든 조합의 수를 구하는 것과 같다.
- 매 재귀함수마다 실제로 $N \times N$ 모든 위치를 모두 볼 필요가 없다.
- 맨 처음 행(row)부터 차례대로 퀸을 놓는다고 생각하면 가짓수를 훨씬 줄일 수 있다.
- N-Queen 문제는 가능한 조합을 계산하는 것이므로, 현재 행의 위쪽 행은 보지 않아도 된다.

N-Queen: https://www.acmicpc.net/problem/9663

문제 해결 아이디어

JavaScript
백트래킹
백트래킹
문제 풀이

- 백트래킹은 기본적으로 가능한 노드에 대하여 계속해서 재귀적으로 함수를 호출한다.
- 백트래킹은 **모든 경우의 수**를 탐색하기에 적합하다.
- N-Queen 문제를 해결하기 위해서는 특정 위치(노드)의 가능 여부를 판단할 필요가 있다.
- 가능한 노드 여부는 다음의 두 가지를 보면 된다.

1) **같은 행에 있는지 체크**: $x_1 == x_2$ / **같은 열에 있는지 체크**: $y_1 == y_2$

2) **대각선에 있는지 체크**: $abs(x_1 - x_2) == abs(y_1 - y_2)$

N-Queen: https://www.acmicpc.net/problem/9663

JavaScript 백트래킹
백트래킹 문제 풀이

정답 코드 예시

```
let fs = require('fs');
let input = fs.readFileSync('/dev/stdin').toString().split('\n');

let n = Number(input[0]); // 전체 맵(map)의 크기
let queens = []; // 현재 체스판에 놓인 퀸(queen)의 위치 정보들

function possible(x, y) { // (x, y) 위치에 퀸을 놓을 수 있는지 확인
  for (let [a, b] of queens) { // 현재까지 놓았던 모든 퀸(queen)의 위치를 하나씩 확인하며
    if (a == x || b == y) return false; // 행이나 열이 같다면 놓을 수 없음
    if (Math.abs(a - x) == Math.abs(b - y)) return false; // 대각선에 위치한 경우 놓을 수 없음
  }
  return true;
}

let cnt = 0;
function dfs(row) {
  if (row == n) cnt += 1; // 퀸(queen)을 N개 배치할 수 있는 경우 카운트
  for (let i = 0; i < n; i++) { // 현재 행(row)에 존재하는 열을 하나씩 확인하며
    if (!possible(row, i)) continue; // 현재 위치에 놓을 수 없다면 무시
    queens.push([row, i]); // 현재 위치에 퀸을 놓기
    dfs(row + 1); // 재귀 함수 호출
    queens.pop(); // 현재 위치에서 퀸을 제거하기
  }
}
dfs(0);
console.log(cnt);
```

알파벳: https://www.acmicpc.net/problem/1987

JavaScript
백트래킹
백트래킹
문제 풀이

혼자 힘으로 풀어보기

문제 제목: 알파벳

문제 난이도: ★★★☆☆

문제 유형: 백트래킹, 경우의 수, 완전 탐색

추천 풀이 시간: 50분

알파벳: https://www.acmicpc.net/problem/1987

문제 해결 아이디어

- $R = 5, C = 5$인 예시를 확인해 보자.

I	E	F	C	J
F	H	F	K	C
F	F	A	L	F
H	F	G	C	F
H	M	C	H	H

알파벳: https://www.acmicpc.net/problem/1987

문제 해결 아이디어

- 가능한 경로: I → E → H → F → K → L → A → G → C → M

I	E	F	C	J
F	H	F	K	C
F	F	A	L	F
H	F	G	C	F
H	M	C	H	H

알파벳: https://www.acmicpc.net/problem/1987

JavaScript 백트래킹
백트래킹 문제 풀이

정답 코드 예시

```
let fs = require('fs');
let input = fs.readFileSync('/dev/stdin').toString().split('\n');

let [r, c] = input[0].split(' ').map(Number); // 맵의 크기 R X C 입력받기
let arr = [];
for (let i = 1; i <= r; i++) arr.push(input[i]);

let dx = [-1, 1, 0, 0]; // 상, 하, 좌, 우 방향
let dy = [0, 0, -1, 1];
let visited = new Set(); // 방문한 적 있는 알파벳 집합
let maxDepth = 0; // 최대 깊이

function dfs(depth, x, y) {
  maxDepth = Math.max(maxDepth, depth); // 최대 깊이(max depth) 계산
  for (let i = 0; i < 4; i++) {
    let nx = x + dx[i];
    let ny = y + dy[i];
    if (nx < 0 || nx >= r || ny < 0 || ny >= c) continue; // 맵을 벗어난다면 무시
    if (visited.has(arr[nx][ny])) continue;
    visited.add(arr[nx][ny]); // 방문 처리
    dfs(depth + 1, nx, ny); // 재귀 함수 호출
    visited.delete(arr[nx][ny]); // 방문 처리 해제
  }
}

visited.add(arr[0][0]); // 왼쪽 위에서 출발
dfs(1, 0, 0);
console.log(maxDepth);
```

부등호: https://www.acmicpc.net/problem/2529

혼자 힘으로 풀어보기

문제 제목: 부등호

문제 난이도: ★★★☆☆

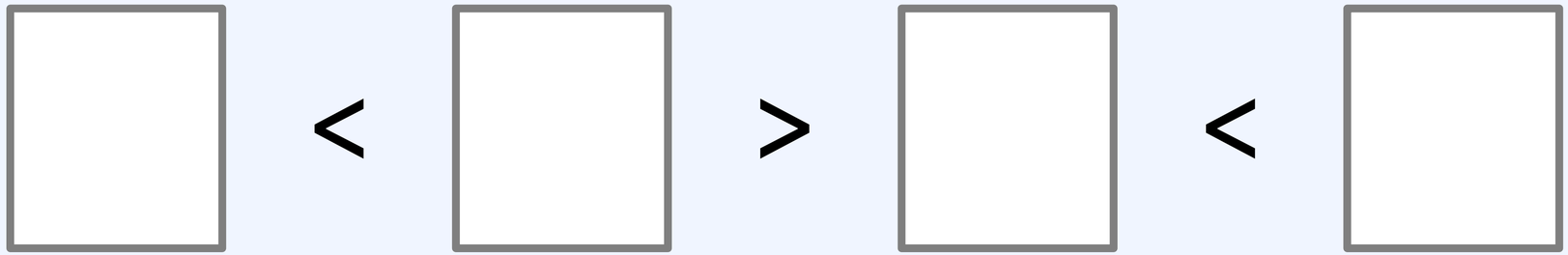
문제 유형: 백트래킹, 경우의 수, 완전 탐색

추천 풀이 시간: 50분

부등호: https://www.acmicpc.net/problem/2529

문제 해결 아이디어

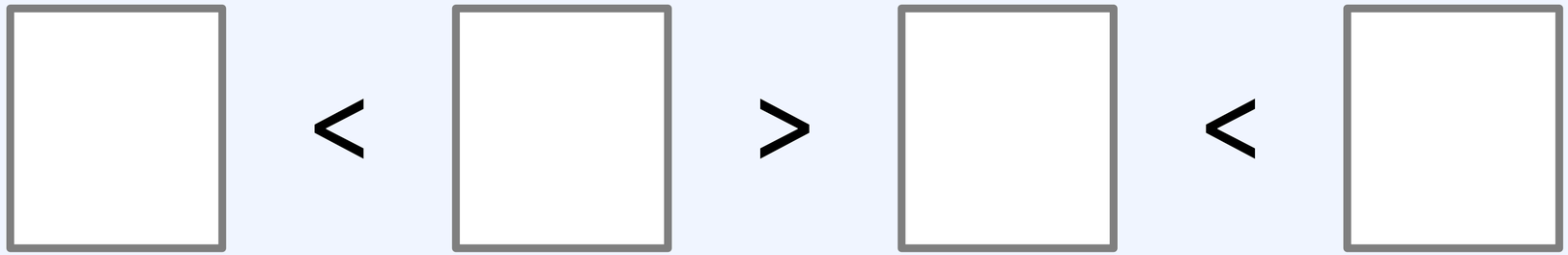
- 총 10개의 숫자 {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}중에서 **중복 없이 모든 순열을 선택**한다.
- 선택한 뒤에 K개의 부등호에 대한 순서를 만족하는지 확인한다.
- K는 최대 9까지 입력될 수 있다.



부등호: https://www.acmicpc.net/problem/2529

문제 해결 아이디어

- 0부터 9까지의 모든 **숫자(digit)** 중에 4개를 고르는 **모든 순열**을 고르면 다음과 같다.
- [0, 1, 2, 3], [0, 1, 2, 4], ⋯, [9, 8, 7, 6]



부등호: https://www.acmicpc.net/problem/2529

JavaScript 백트래킹
백트래킹 문제 풀이

정답 코드 예시

```
function dfs(depth) {
  if (depth == k + 1) { // 각 순열(permutation)에 대한 처리
    let check = true; // 부등식을 만족하는지 확인
    for (let i = 0; i < k; i++) {
      if (arr[i] == '<') {
        if (result[i] > result[i + 1]) check = false;
      }
      else if (arr[i] == '>') {
        if (result[i] < result[i + 1]) check = false;
      }
    }
    if (check) { // 부등식을 만족하는 경우에
      current = "";
      for (let x of result) current += x + "";
      if (first == "") first = current; // 첫째 문자열은 [가장 작은 수]
    }
    return;
  }
  for (let i = 0; i < 10; i++) { // 0부터 9까지의 숫자를 하나씩 고르는 순열(백트래킹)
    if (visited[i]) continue; // 이미 고른 숫자라면 무시하도록
    visited[i] = true; // 현재 선택한 숫자 방문 처리
    result.push(i);
    dfs(depth + 1); // 재귀 함수 호출
    visited[i] = false; // 현재 선택한 숫자 방문 처리 취소
    result.pop();
  }
}
```

부등호: https://www.acmicpc.net/problem/2529

JavaScript 백트래킹
백트래킹 문제 풀이

정답 코드 예시

```
let fs = require('fs');
let input = fs.readFileSync('/dev/stdin').toString().split('\n');

let k = Number(input[0]);
let arr = input[1].split(' ');
let visited = new Array(10).fill(false);
let result = [];
let current = "";
let first = "";

dfs(0);
console.log(current + "\n" + first); // 마지막에 남은 current 값은 [가장 큰 문자열]
```