

let, let 스코프 — 이분검색, 투포인터, 슬라이드 윈도우 등
(다른 Idx는 중복적으로 바뀌어 가는 것)

DFS 스코프 — 가지각 구하기, 그룹핑, 전위중위 후위 순회
그룹은 이어 나가기 위해서 누적합을 사용해야 하는 경우

(순열, 조합) ~ 두 그룹 간의 비교 \Rightarrow 무엇을 누적합하지?

BFS — 최단거리, 시간 등, 최단 cost + 되는 문제

트리의 Depth 관련 문제: DFS, BFS 등 트리를 가는 방식


좌표 관련 같이, 순회 하는 경우가 생기는 문제: 순회 방식에 따른 check Array 쓰이기

Stack — 괄호, 문자열 합성, 분해

구간 — 특정한 배치할 때, 가장 많이 배치할 수 있는지 / 대충: 스케줄링
(구간 - 2번 질문하는 문제) (정렬한 후)

무선호문(문) - 반복문 최소와 최대 값을 통해 연산을 해본다

DP, 그리디 - 배치한 요소를 (변환사항) => 두(부)러 배열

이분탐색 -  function(lt, rt)

중간을 기점으로 좌우의 값을 동시에 고려해야 하는 경우 (ex 기차표, ^{이론적인} ^{바라볼} ^{수업})
=> dist [] 2개

최소화문제 - Union & find + greedy

DP - 이전 상태를 불러 현재 상태의 계산값이 주어지거나,
현재에서 최적의 방향으로 전개되어야 하는 경우
→ 재귀로 가는 dist [] 필요

해킹 - Map 사용, key의 순서가 중요하게 작용 object 사용됨