* 순열 조합 부분집합 문제
* 순열
  + 뽑는 수가 정해져있다면, for문으로 해도 됨
  + 뽑는 수가 가변적이면, 직전의 수와 비교(반복 비교하지 않고, 수 개수만큼 boolean 배열 만들어서 T/F로 참고)해서 그 수를 제외하고 뽑는 메서드를 만들고(제외한 수가 뭔지 알려면 해당 수를 담는 배열을 만들어서 이를 공유한다), 해당 메소드를 r번 반복 – 재귀 이용

1. 문제 풀이법 생각 – 제약사항 확인
2. 시간 복잡도 생각
3. 수의 경계 확인(int로 충분한지, long 검토해야 하는지 등)

* 트리 (p.391) – 비선형자료구조
* 부모->자식 = 1대M : 부모에서부터 내려오기 때문에 사이클링이 생기지 않는다
* 어떤 노드를 봐도 트리의 성질을 만족해야 트리라고 한다.(하나라도 만족하지 않으면 그래프이다)
* 차수 : 부모에서 자식으로 가는 간선의 수 / 트리의 차수 : 모든 노드의 차수 중 가장 큰 값
* 레벨 : 루트(level0)에서 자신까지 오는데 거치는 간선의 수
* 이진트리 : 모든 노드의 차수가 2이하인 트리 + 이진(트리)탐색
  + 포화이진트리 : 노드의 개수 = 2(n+1)-1
  + 완전이진트리 : 왼쪽->오른쪽 마지막 레벨 제외하고 완전 채워진 트리 (Heap)
    - - Heap : 최대/최소를 빠르게 하기 위한 것. -> 왼쪽 트리의 내 부모랑만 비교하면 됨. (나머지 오른쪽 서브트리 값은 안 봄) = O(logN)
      * 1개의 값을 뺄 때 = O(logN) : 루트값을 다른 변수에 저장하고, 맨 마지막(리프노드)값을 루트에 채우고 리프노드 삭제 [완전이진트리 만들기] -> 그 다음 자식노드 중 최소값이랑 비교해서 스왑 혹은 그대로. [logN번 반복]
      * 정렬 값은 들어갈 때가 아니라 뺄 때 알 수 있다. 그래서 힙 정렬을 먼저 하고, 원하는 값을 얻으려면 빼봐야 한다.
    - 최소Heap : 부모노드값 <= 자식노드값 [기본]
    - 최대Heap : 부모노드값 >= 자식노드값
* 2\*\*20 = 100만 -> 루트부터 높이가 30정도 되면 노드가 100만개가 됨
* 원소 스스로가 comparable하거나, 우선순위q를 생성할 때 제 3자 comporator를 줘야 함
* 우선순위큐 = 이진트리Heap
* compareTo / Comparable

이진 탐색 (p.180)

* O(logN)
* 그냥 탐색하면 시간이 오바됨
* 데이터가 정렬된 상태에서만 사용할 수 있음
* 파라매트릭 써치

그래프

* 자료구조 8장
* Vertex(정점) / edge(간선) ./ 엣지에 있는 수치(가중치)
* 유향 = 단방향
* 무향 = 양방향
* 인접 행렬(정점 적을 때) < 인접리스트(정점 많을 때) : 시간 단축

BFS/DFS

* 재귀 or 스택자료구조
* 가중치가 없는 그래프에서 최단 : BFS가 유리

이클립스 단축키 : <https://yooniron.tistory.com/44>