Day1 정렬과 탐색

선린인터넷고등학교 소프트웨어과

30610 나정휘

https://JusticeHui.github.io

문제 목록

- BOJ2805 나무 자르기 (COCI 2011/2012 #5 2번)
- BOJ2512 예산 (KOI 2012 고등부 1번)
- BOJ11053 가장 긴 증가하는 부분 수열
- BOJ12015 가장 긴 증가하는 부분 수열 2
- BOJ 10775 공항
- Algospot WITHDRAWAL
- BOJ15977 조화로운 행렬 (KOI 2018 고등부 3번)

나무 자르기

- 다 알지?

- Parametric Search
- f(x) =높이를 x로 설정했을 때 M 이상 가져갈 수 있는가?

예산

- 이것도 다 알지?

- Parametric Search
- f(x) = 상한을 x로 설정했을 때 예산을 잘 배정할 수 있는가?

가장 긴 증가하는 부분 수열

- D(i) = i번째 원소로 끝나는 LIS의 길이

- D(i) = max{ D(j) } + 1 (j < i && Aj < Ai)
- O(N^2)

가장 긴 증가하는 부분 수열 2

- 1. O(N^2) DP 풀이를 세그먼트 트리로 최적화 -> O(N log N)
  - D(i) = max_query(1, Ai - 1) + 1
  - update(Ai, D(i))
- 2. 이분 탐색을 똑똑하게

가장 긴 증가하는 부분 수열 2

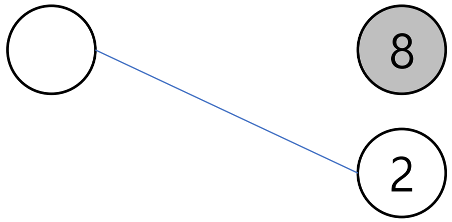
- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]

가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]

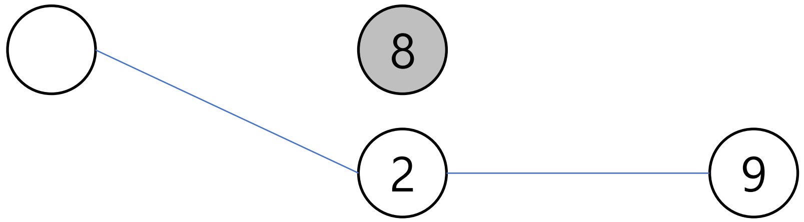
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



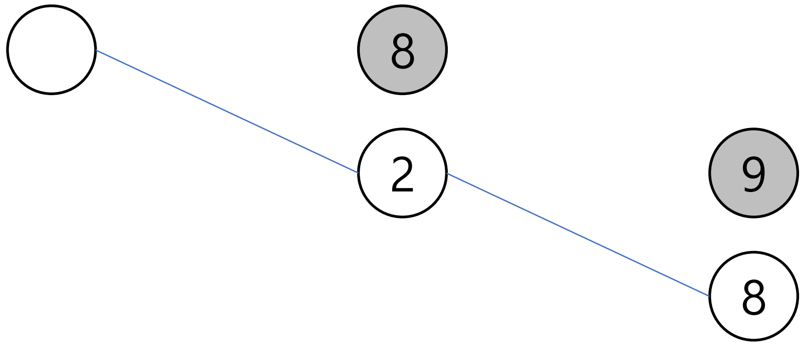
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



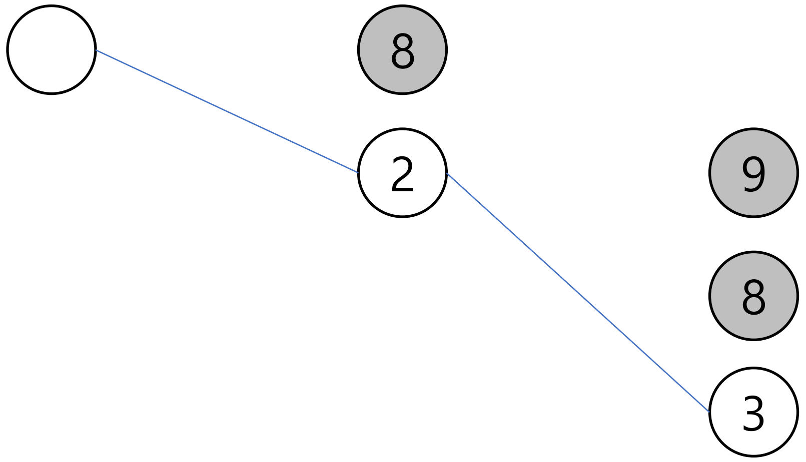
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



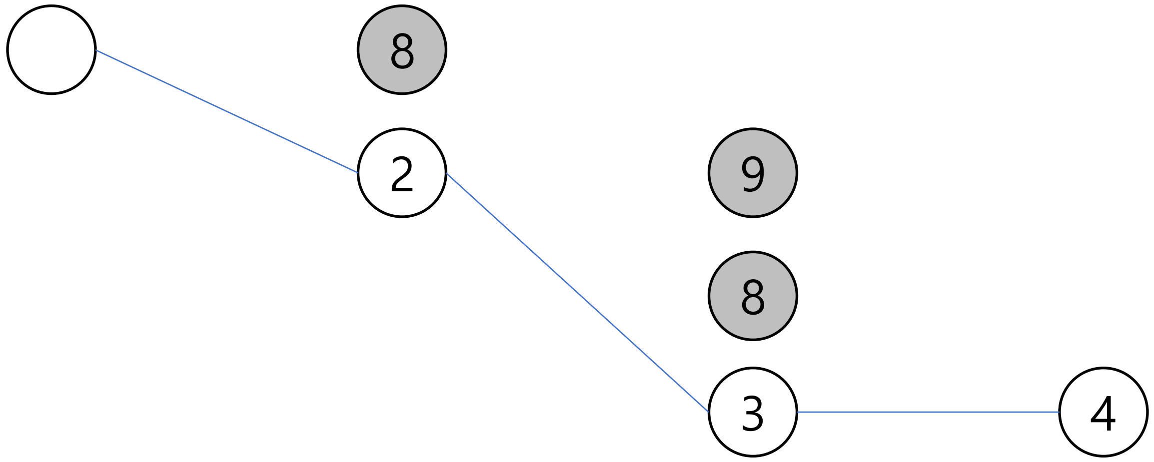
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



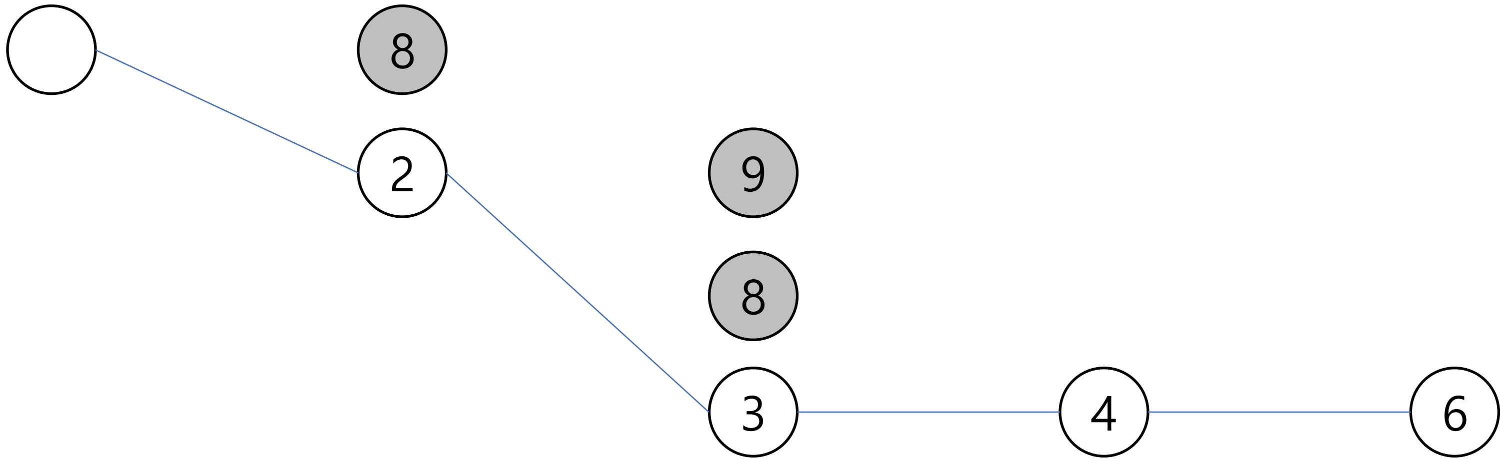
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



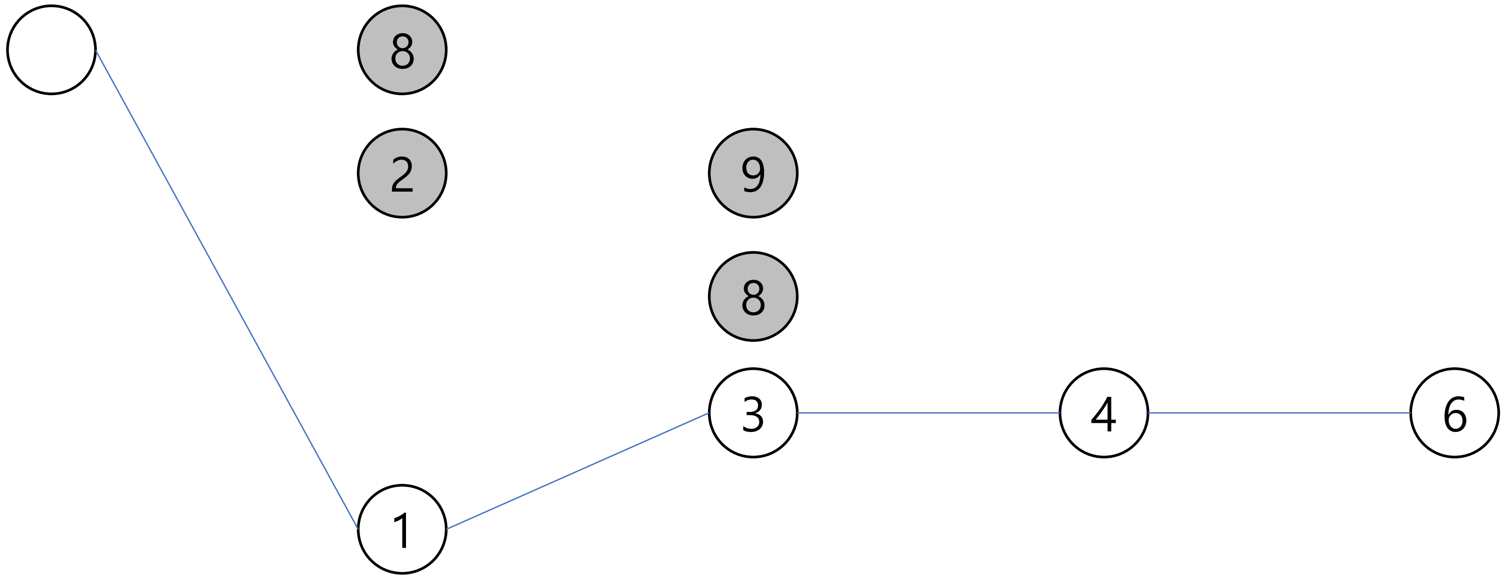
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



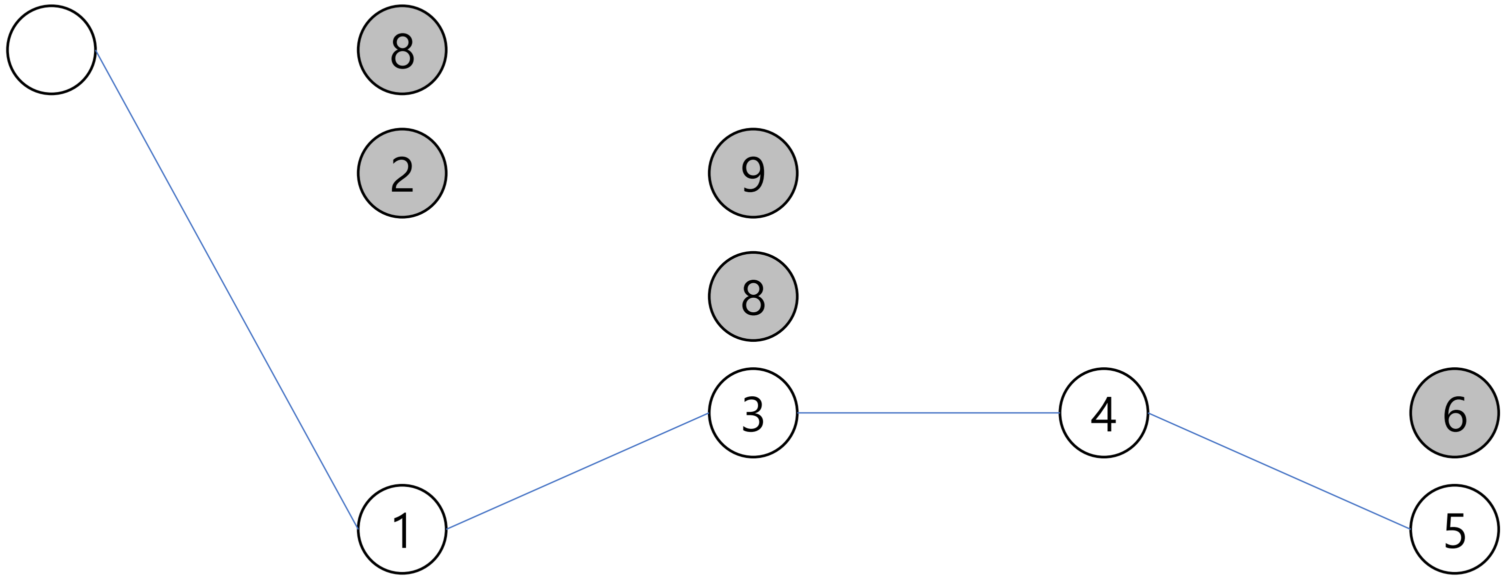
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



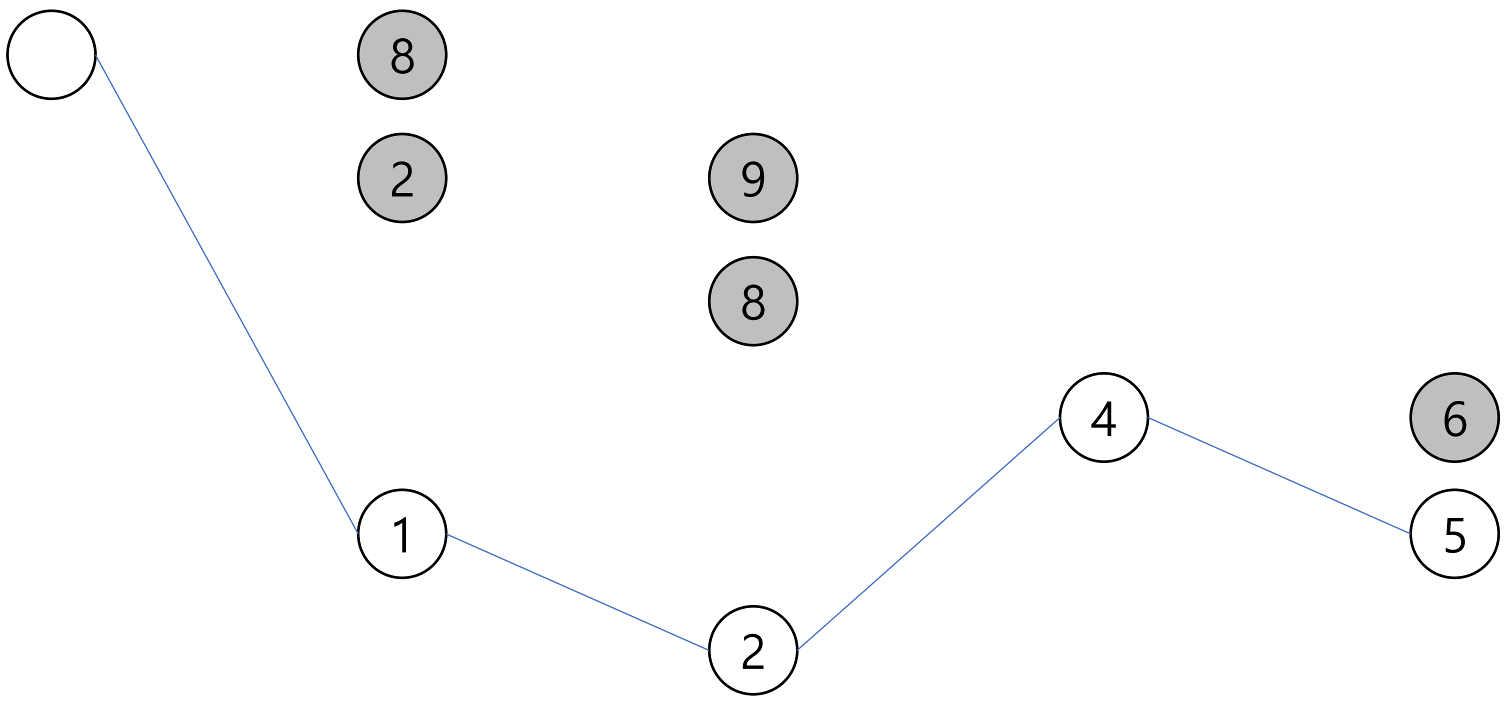
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



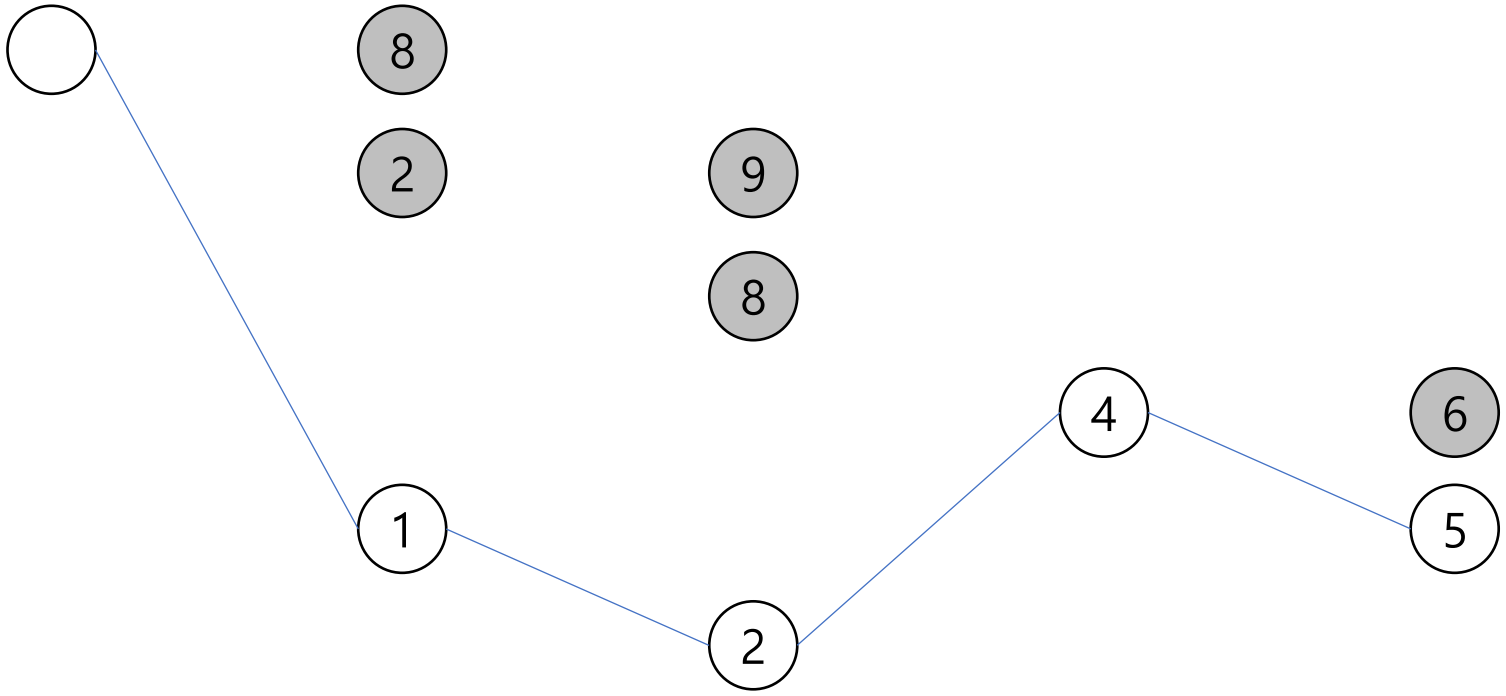
가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]



가장 긴 증가하는 부분 수열 2

- [8, 2, 9, 8, 3, 4, 6, 1, 5, 2]
- 이분 탐색 화이팅



공항

- 각 비행기를 최대한 큰 번호에 배치하면 됨

WITHDRAWAL

- k개를 잘 골라서 sum(Ri) / sum(Ci)를 최소화

WITHDRAWAL

- k개를 잘 골라서 sum(Ri) / sum(Ci)를 최소화

- 최대/최소 문제 -> Parametric Search
- f(x) = sum(Ri) / sum(Ci)를 x 이하로 만들 수 있는가?

WITHDRAWAL

- f(x) = sum(Ri) / sum(Ci)를 x 이하로 만들 수 있는가?

- sum(Ri) / sum(Ci) <= X
- sum(Ri) <= X * sum(Ci) = sum(X * Ci)
- sum(Ri - X * Ci) <= 0

- Ri - x * Ci 오름차순 정렬 후 K개 더해서 0보다 작은지 확인

조화로운 행렬

- 첫 번째 행 기준으로 정렬해도 답이 변하지 않음.

- m = 2: 단순 LIS
- m = 3: pair에 대한 LIS

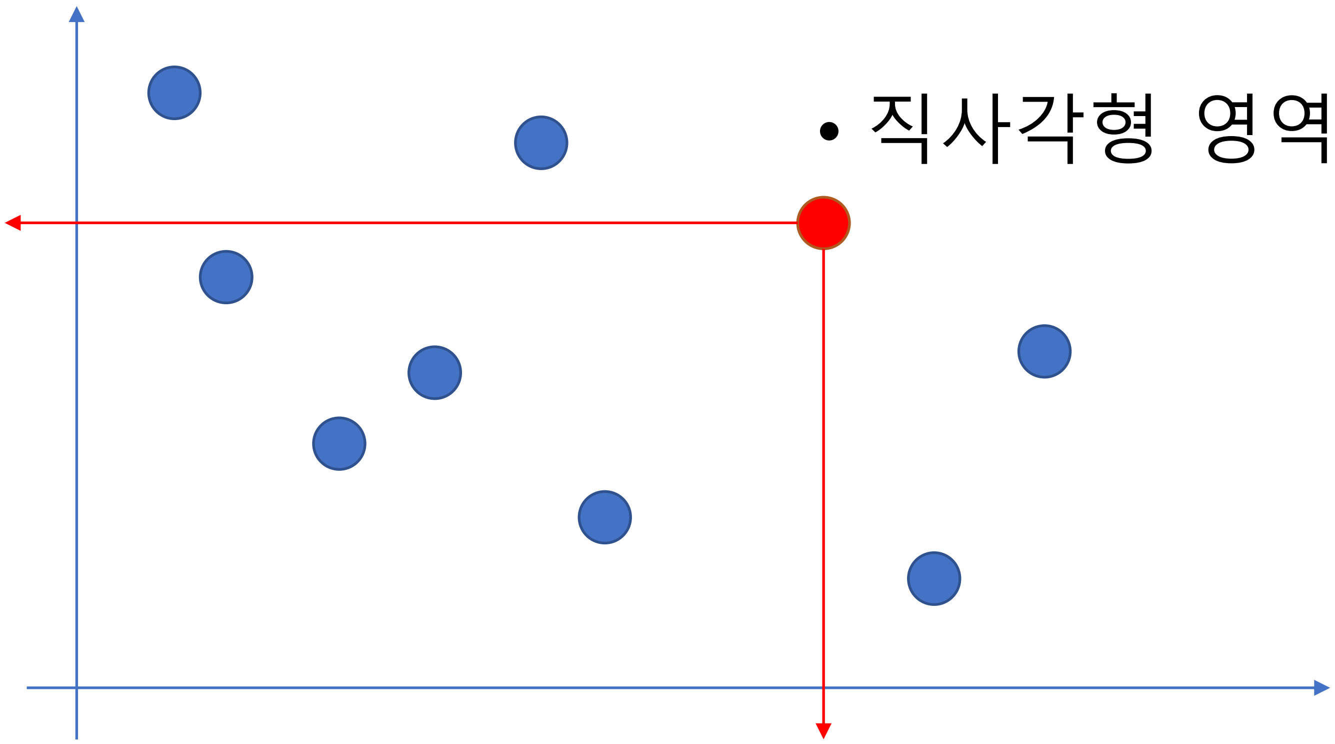
- m = 2인 경우를 m = 3으로 바꿀 수 있음.

조화로운 행렬

- 풀이1. LIS DP + 2D Segment Tree -> O(N log^2 N)
- 풀이2. 이분 탐색을 잘 하자.

조화로운 행렬

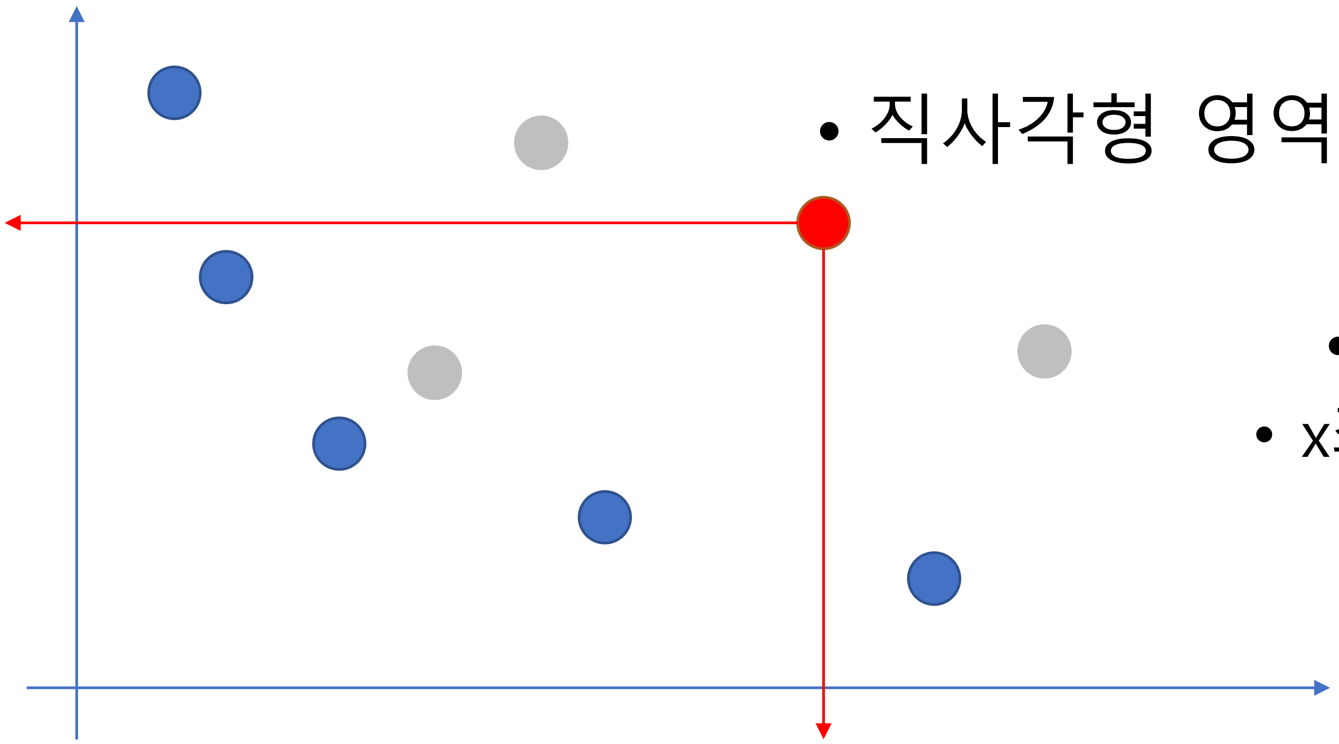
- $D(i) = \{ D(j) = k, X_j \le X_i, Y_j \le Y_i$인 j가 존재하는 최대 $k \} + 1$



- 직사각형 영역에 점이 하나라도 있는가?

조화로운 행렬

- $D(i) = \{ D(j) = k, X_j \le X_i, Y_j \le Y_i$인 j가 존재하는 최대 $k \} + 1$



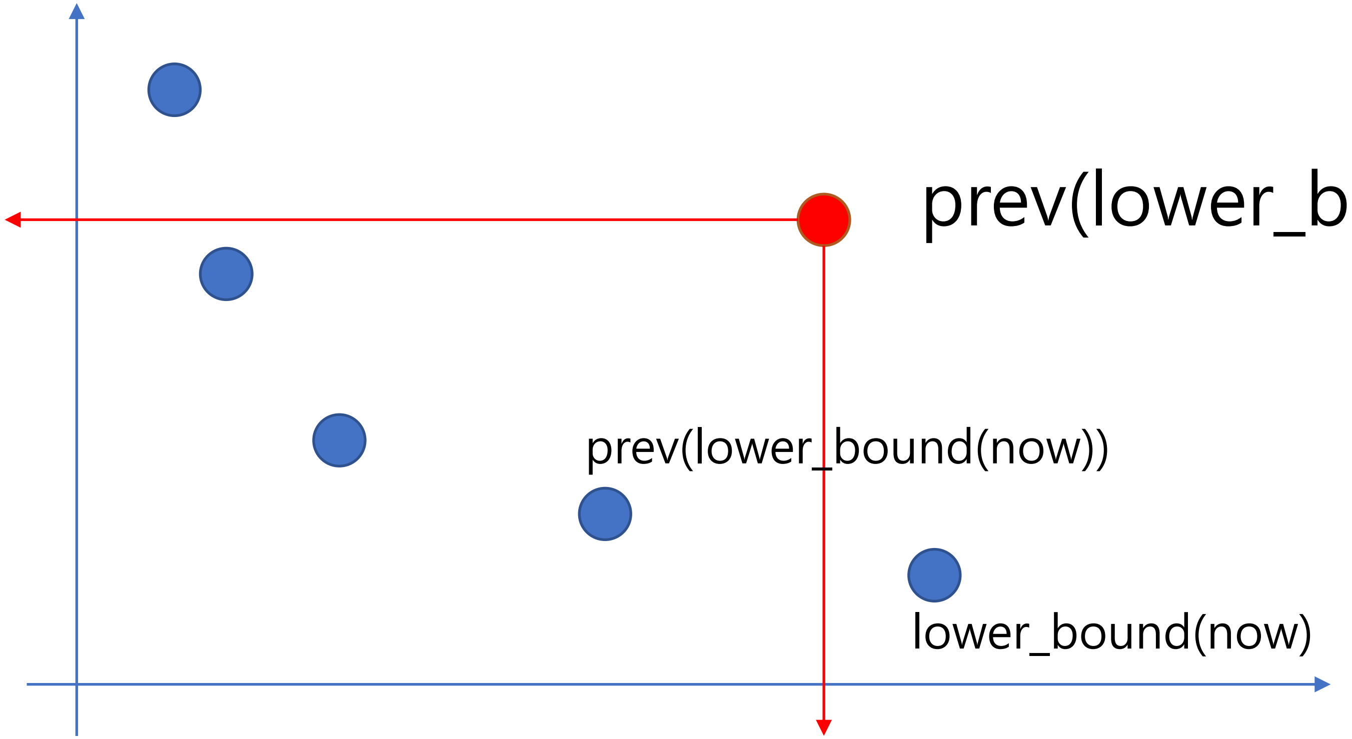
- 직사각형 영역에 점이 하나라도 있는가?
- 필요 없는 점을 제거하자
- x좌표가 증가하면 y좌표는 감소

조화로운 행렬

- D(i) = { D(j) = k, Xj ≤ Xi, Yj ≤ Yi인 j가 존재하는 최대 k } + 1
- K가 존재한다면 K-1도 존재하기 때문에 이분탐색 가능
- 영역에 점이 존재하는지는 K별로 set을 만들고 lower_bound로 확인 가능

조화로운 행렬

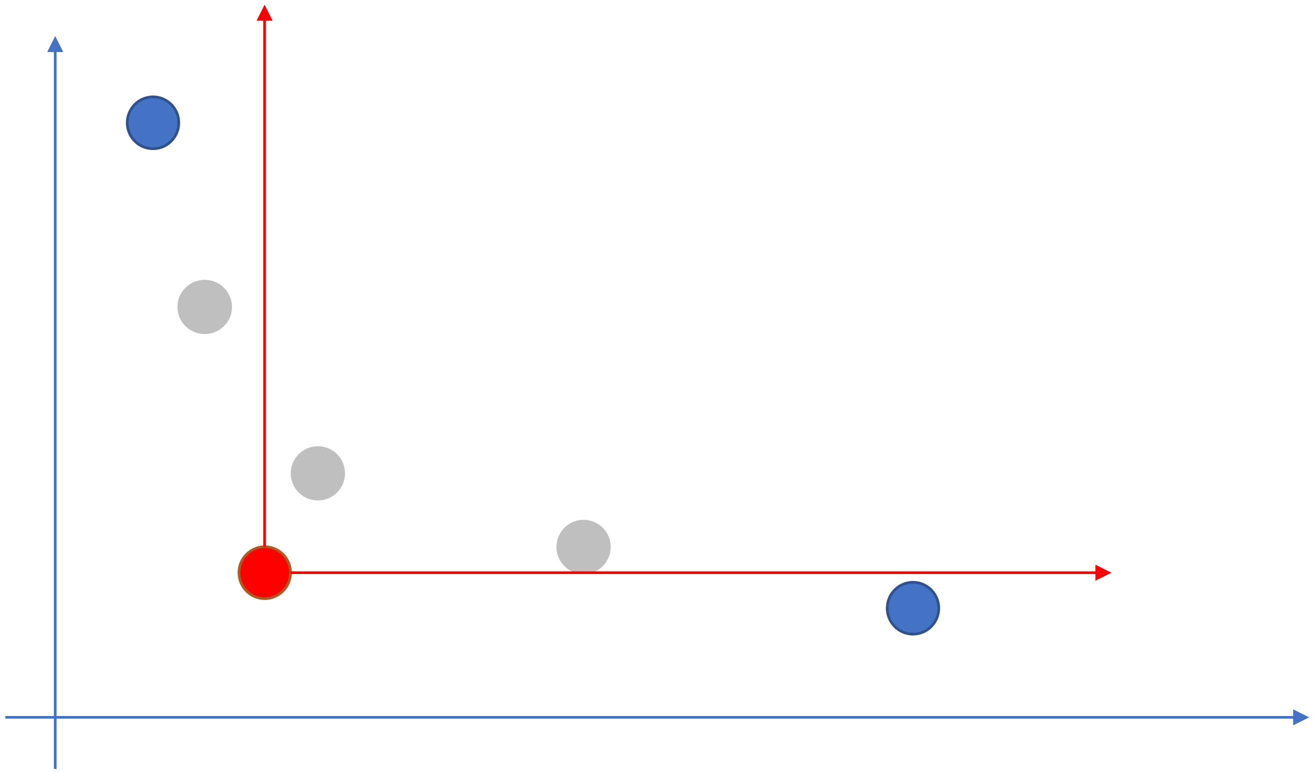
- $D(i) = \{ D(j) = k,\ X_j \le X_i,\ Y_j \le Y_i$인 j가 존재하는 최대 $k \} + 1$



prev(lower_bound())->y ≤ now.y

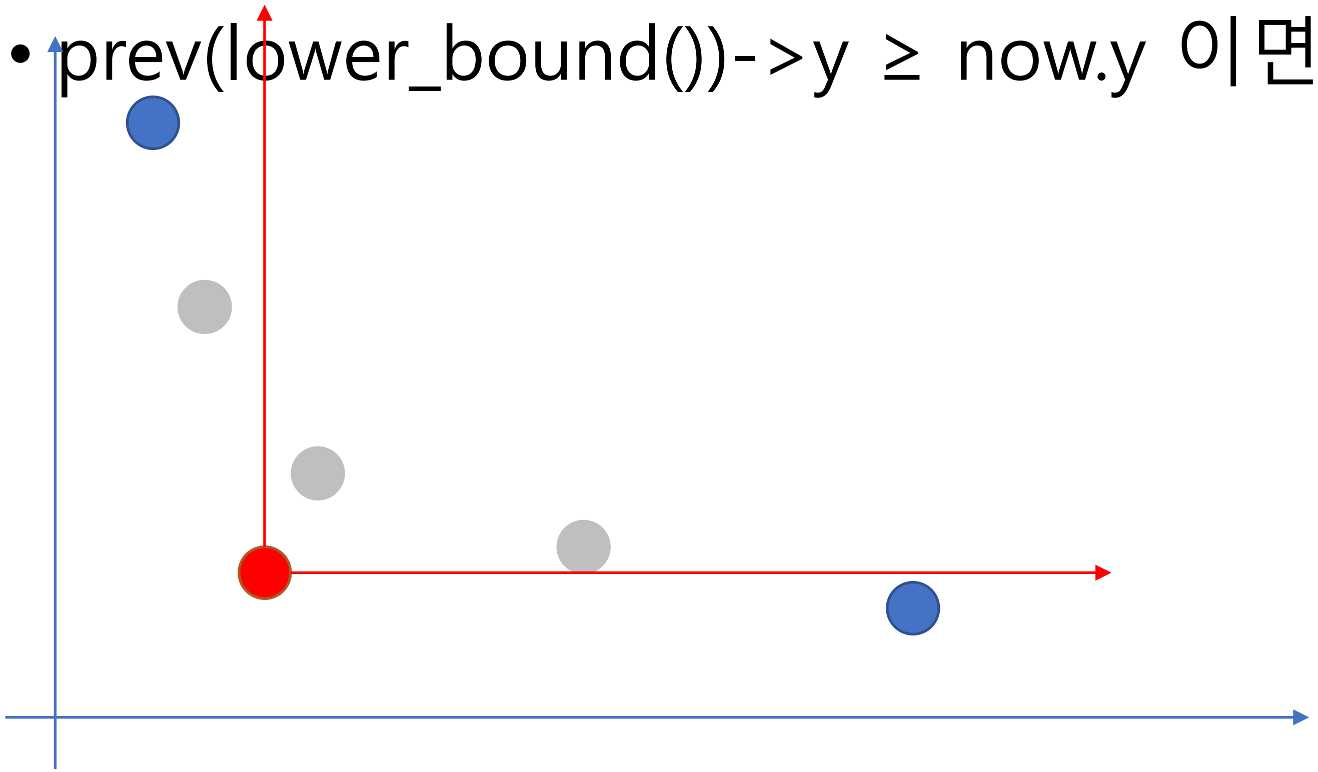
조화로운 행렬

- 점 삽입은 어떻게?



조화로운 행렬

- 점 삽입은 어떻게?
- prev(lower_bound())->y ≥ now.y 이면 삭제 반복



조화로운 행렬

- 각 점은 최대 O(N)번 삽입, O(N)번 삭제
- 이분 탐색 N번, 각 Decision에서 set 쓰니까 O(N log N)
  - 총 O(N log^2 N)