그래프 2.5

최백준 choi@startlink.io

임계 경로

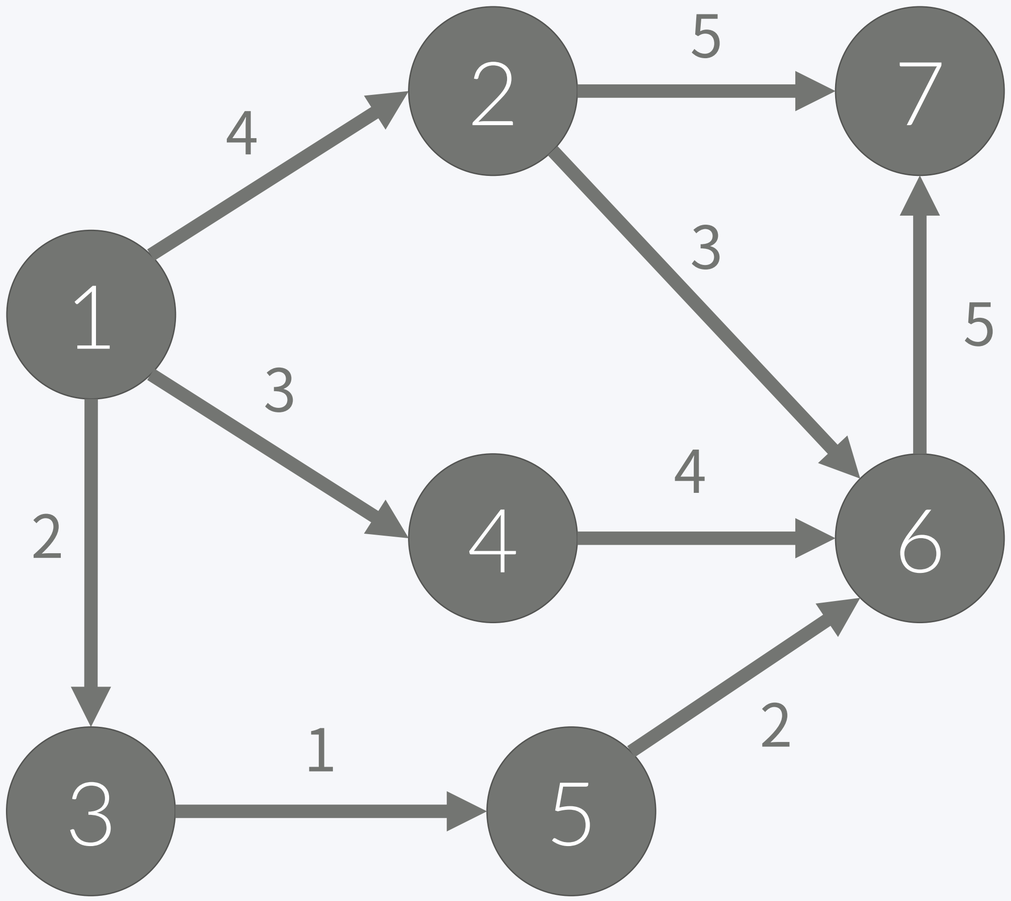
https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제
- 그리고 그 때, 가장 긴 경로에 포함된 간선의 개수를 구하는 문제

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

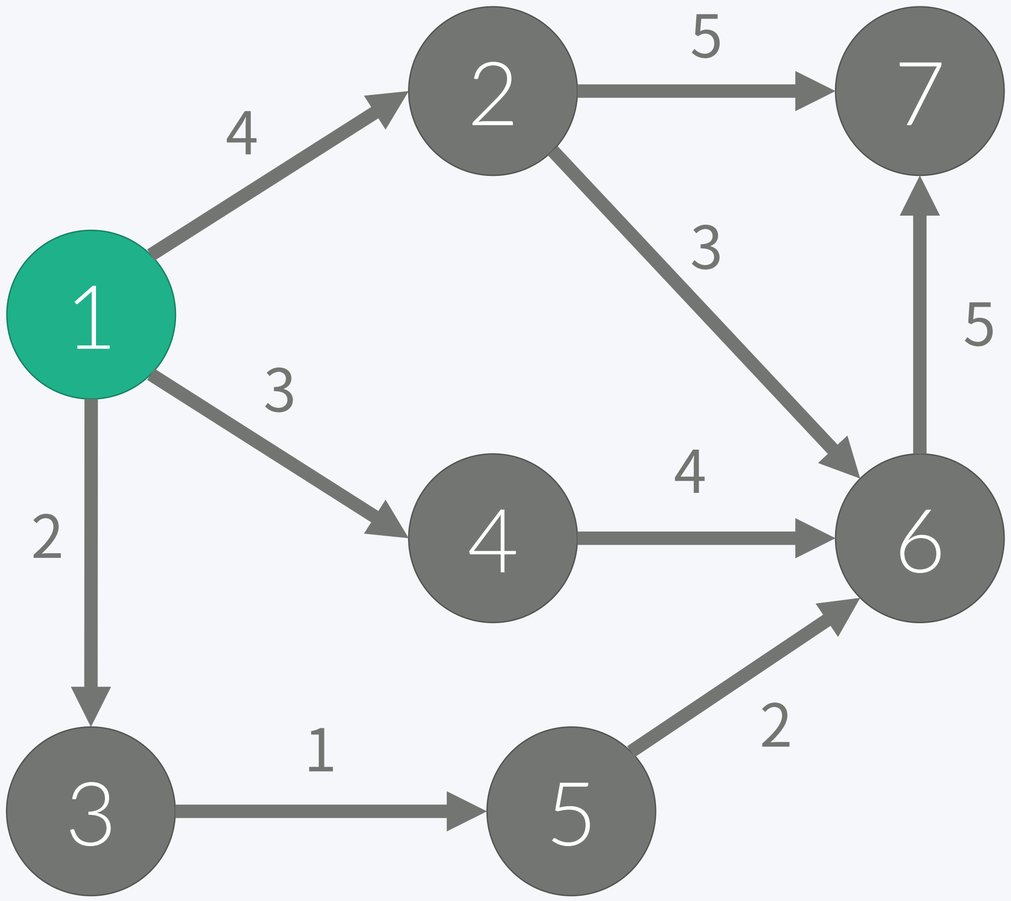


i	1	2	3	4	5	6	7
거리	0						

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

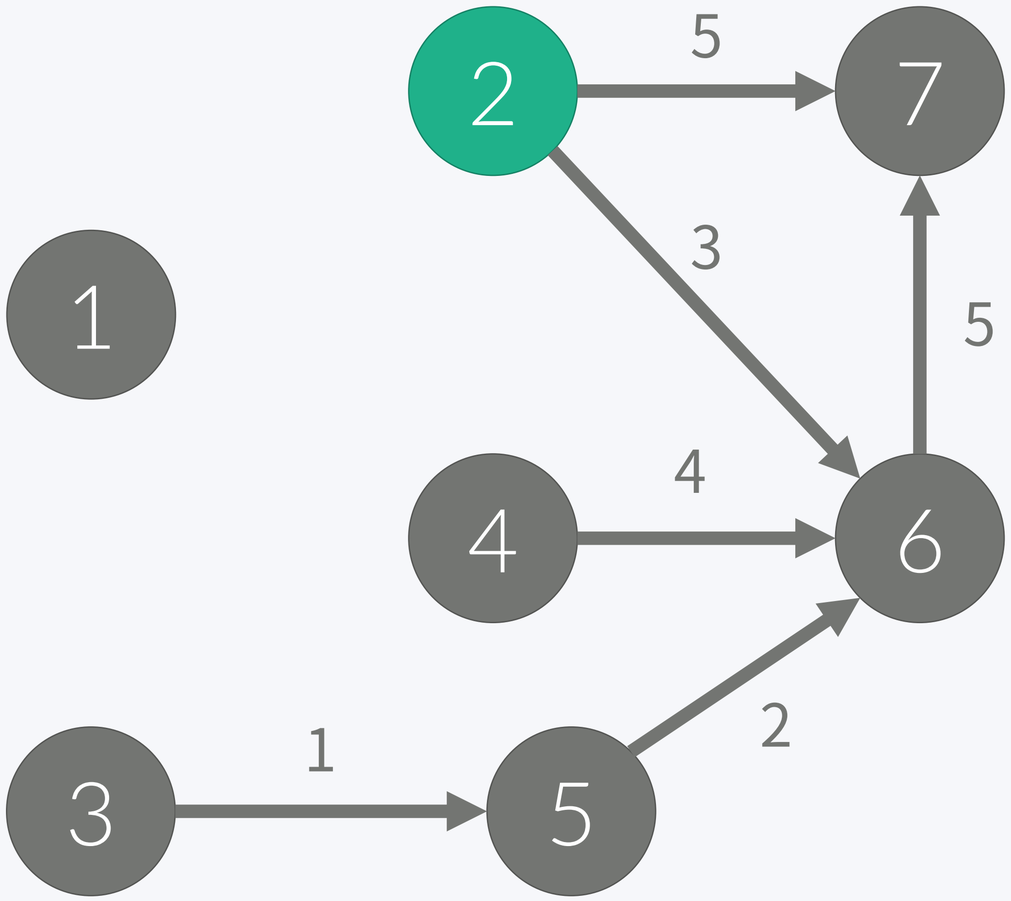


i	1	2	3	4	5	6	7
거리	0	4	2	3			

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

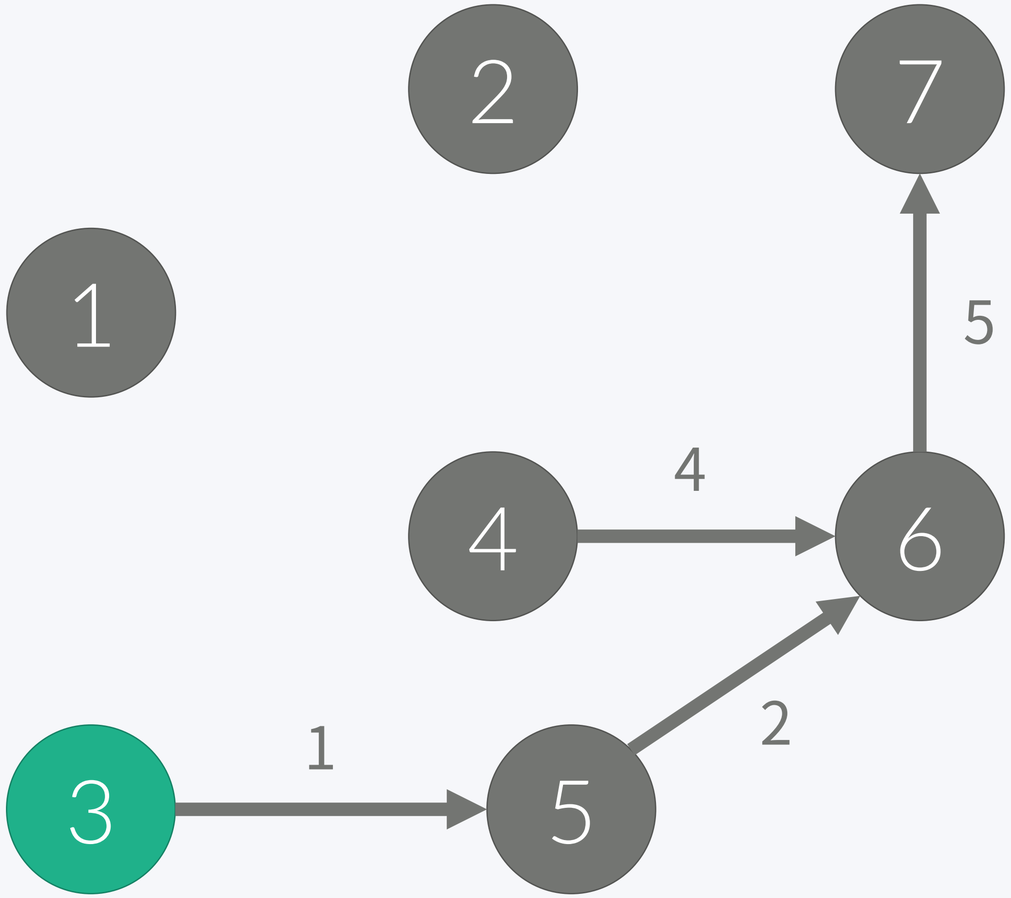


i	1	2	3	4	5	6	7
거리	0	4	2	3		7	9

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

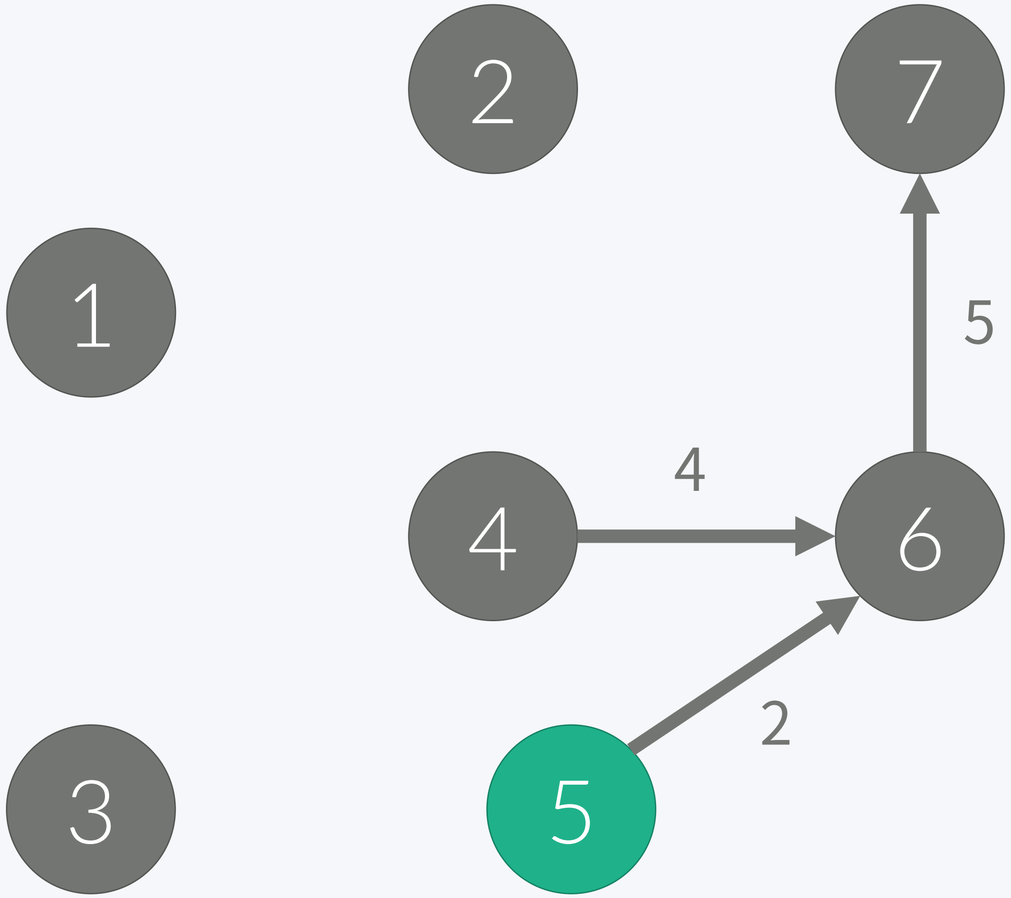


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	9

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

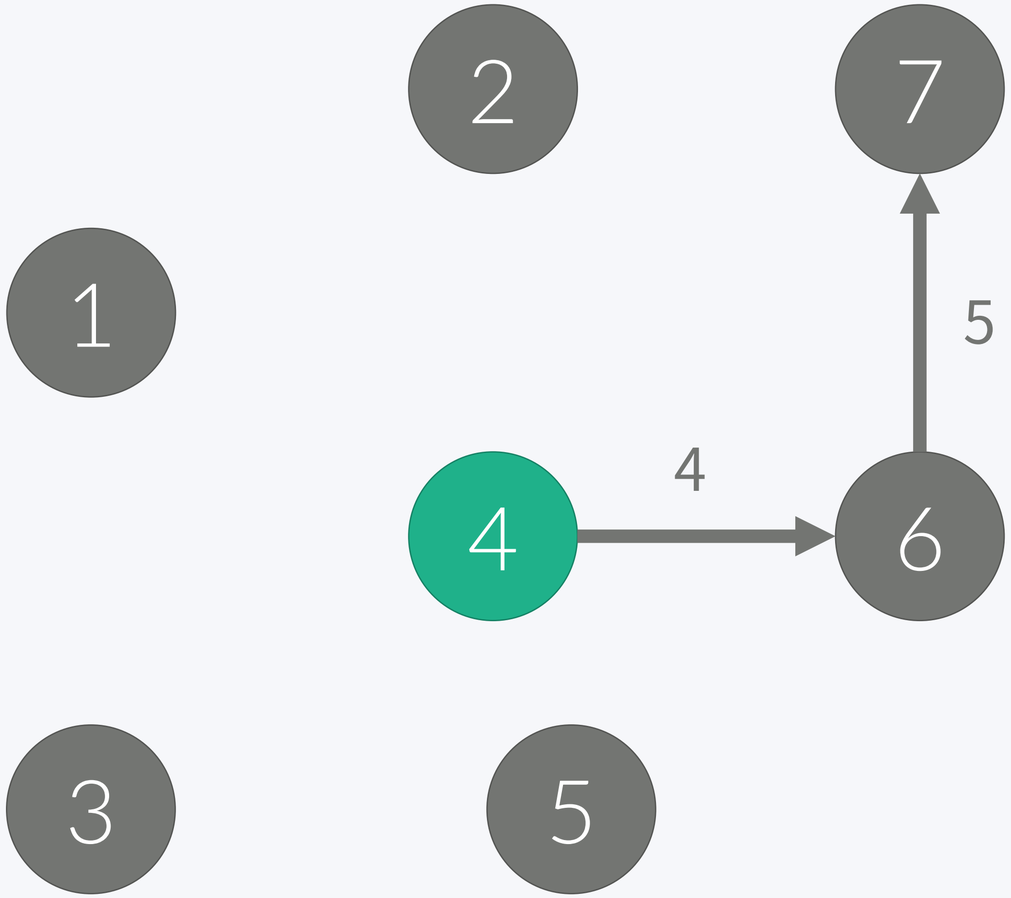


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	9

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

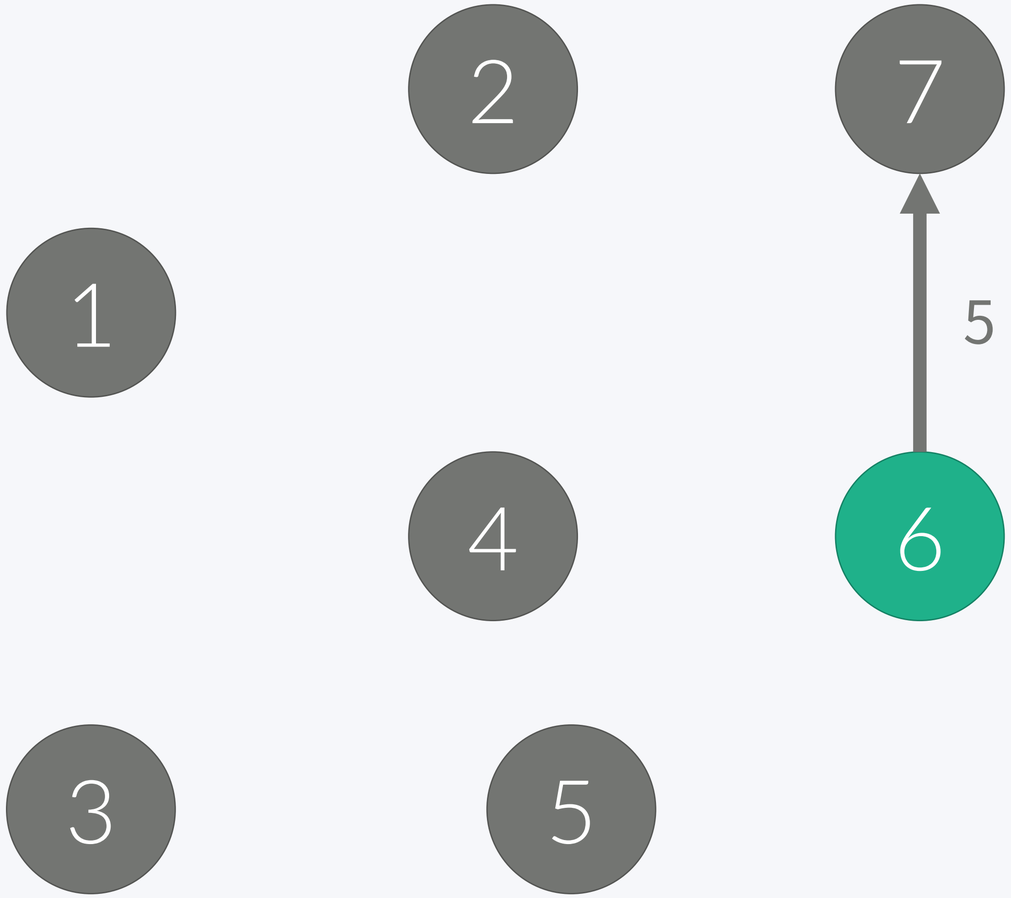


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	9

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

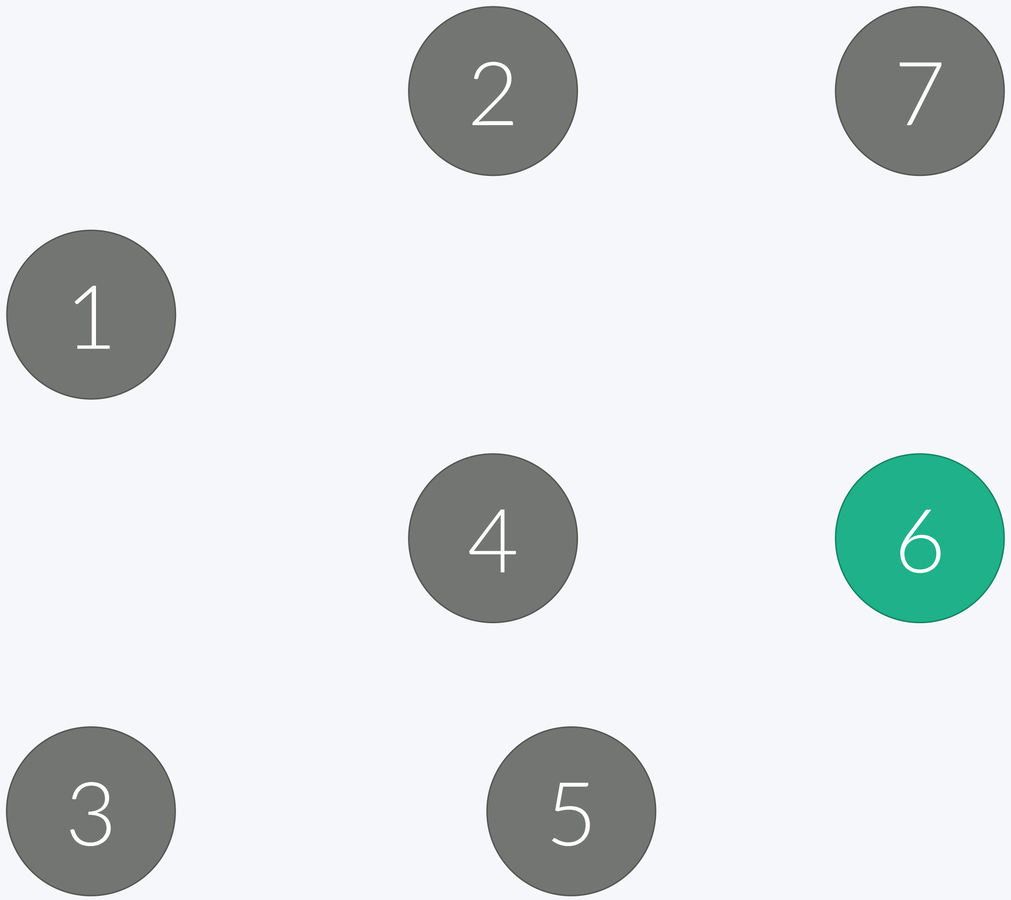


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

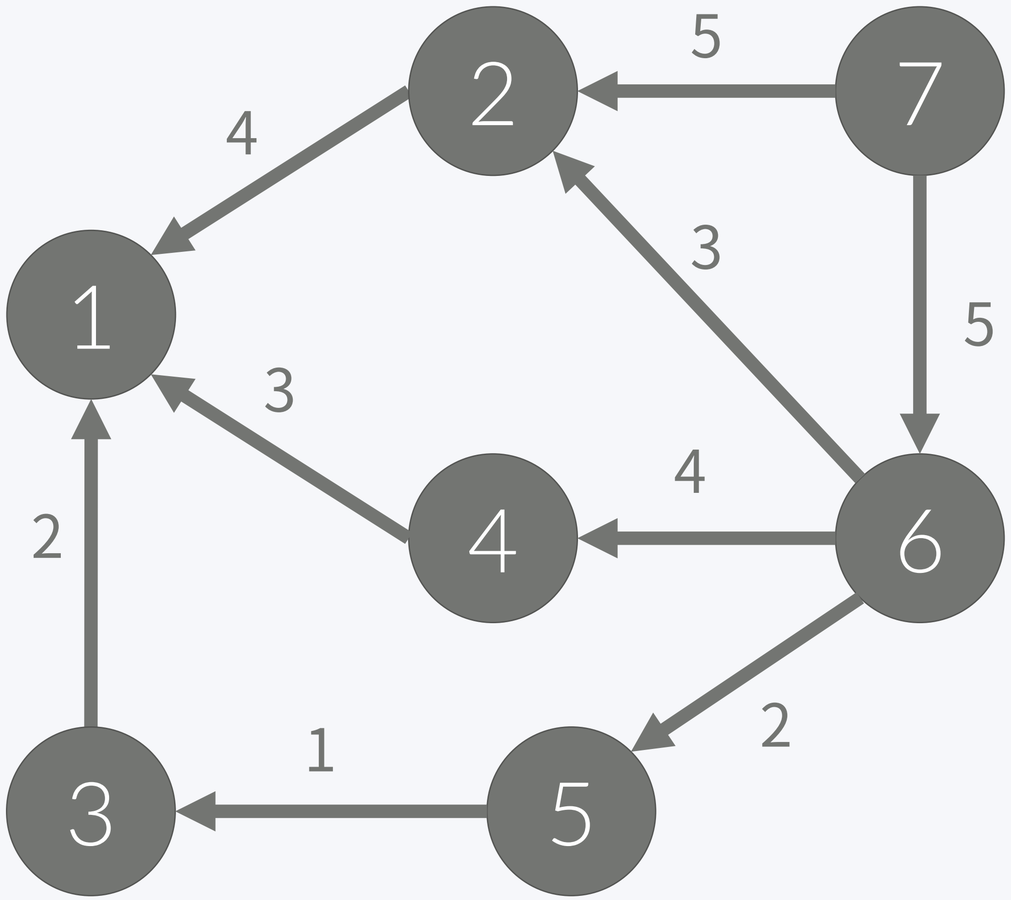


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

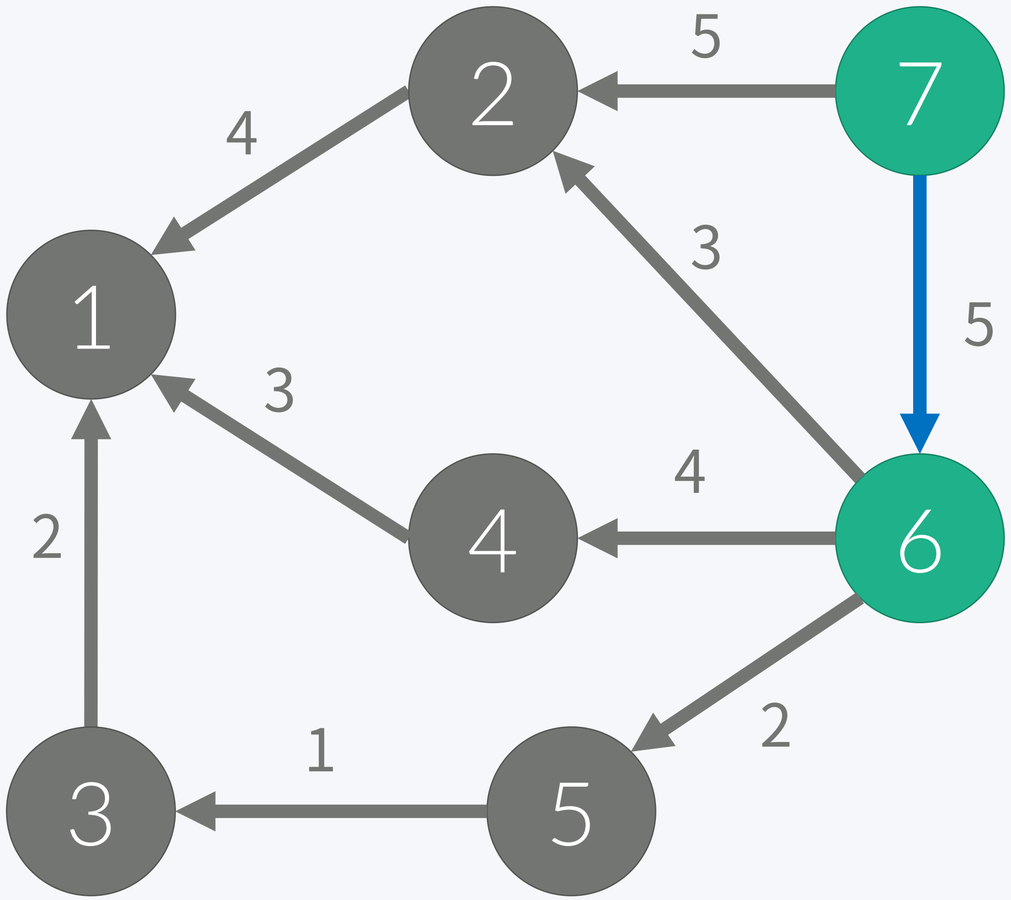


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

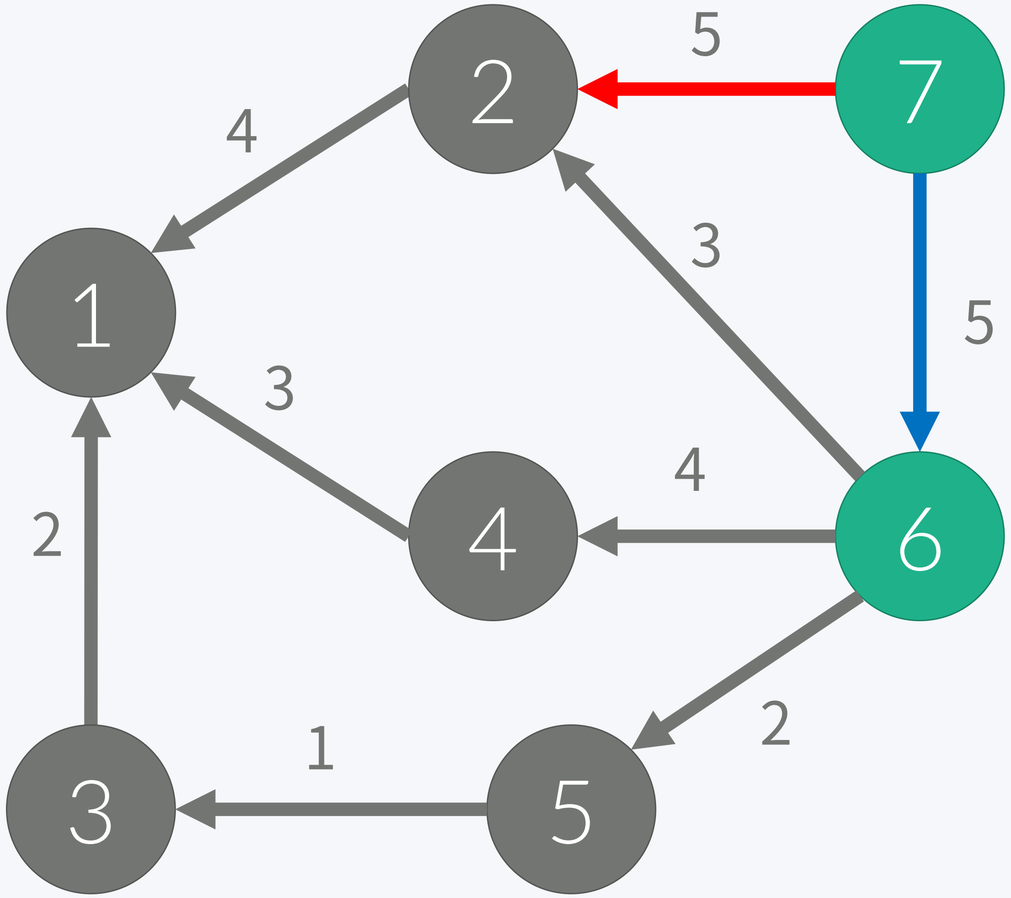


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

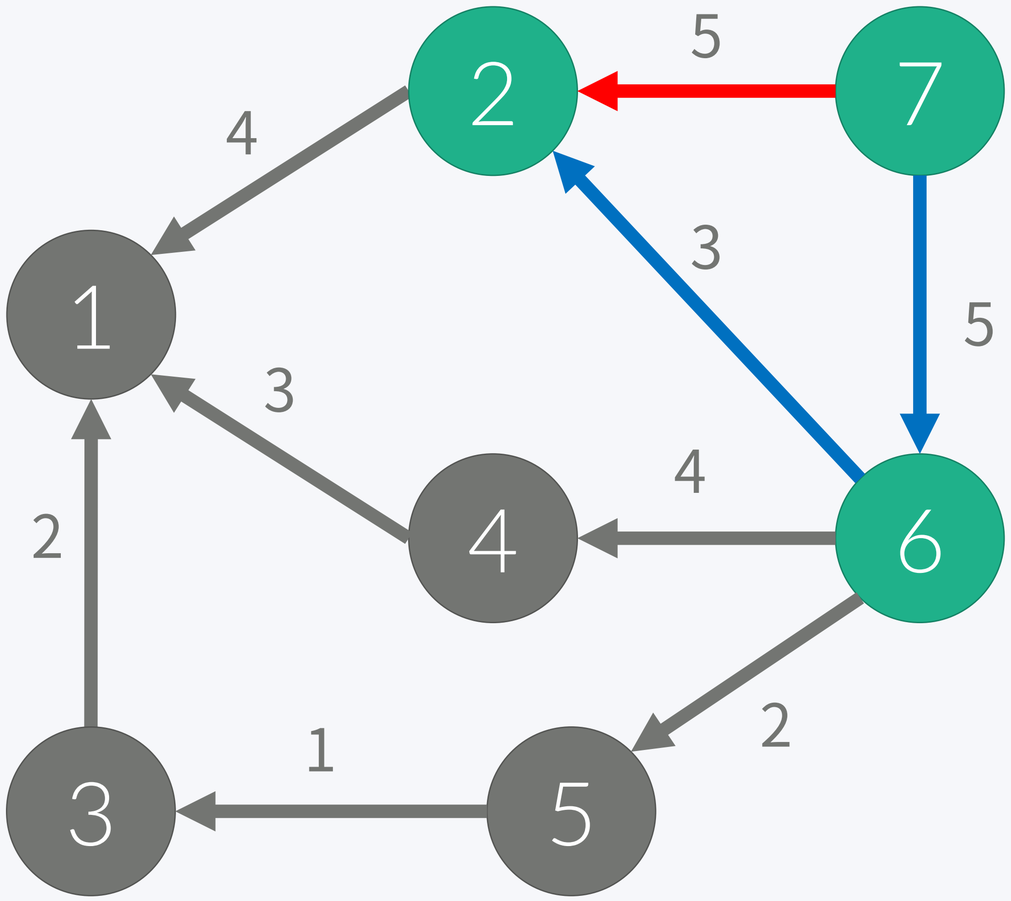


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

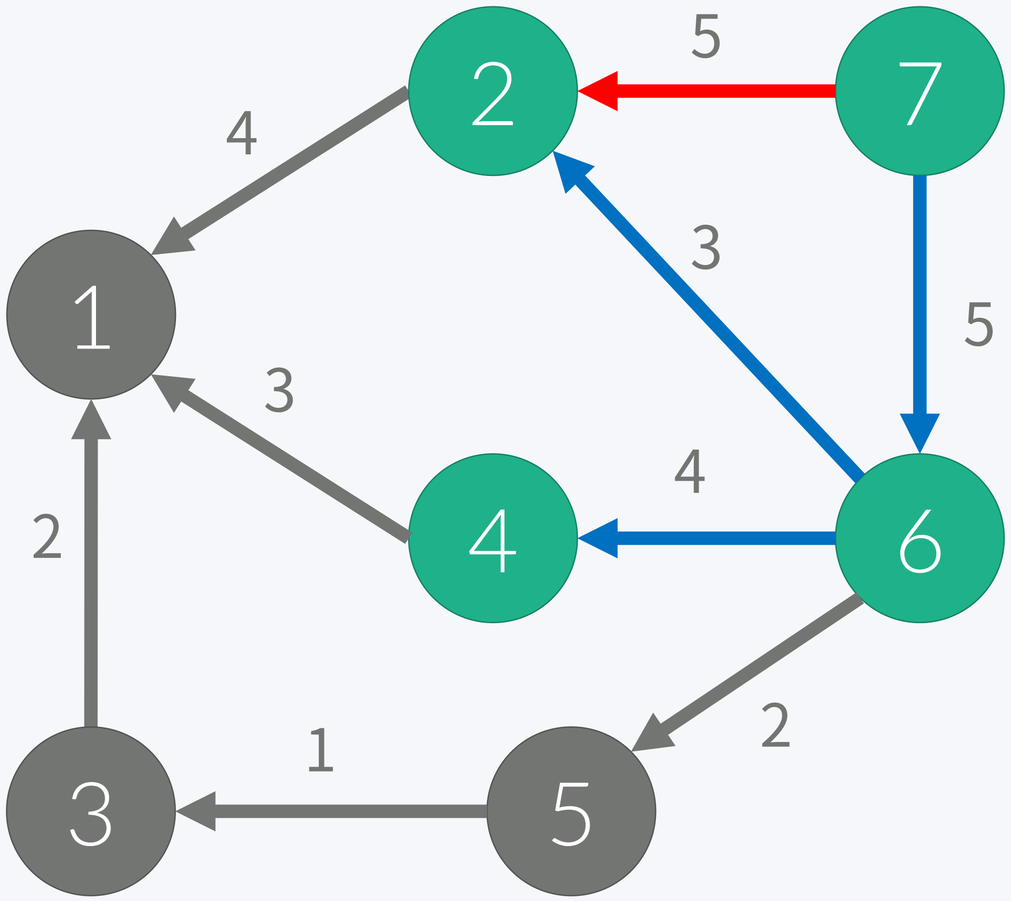


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

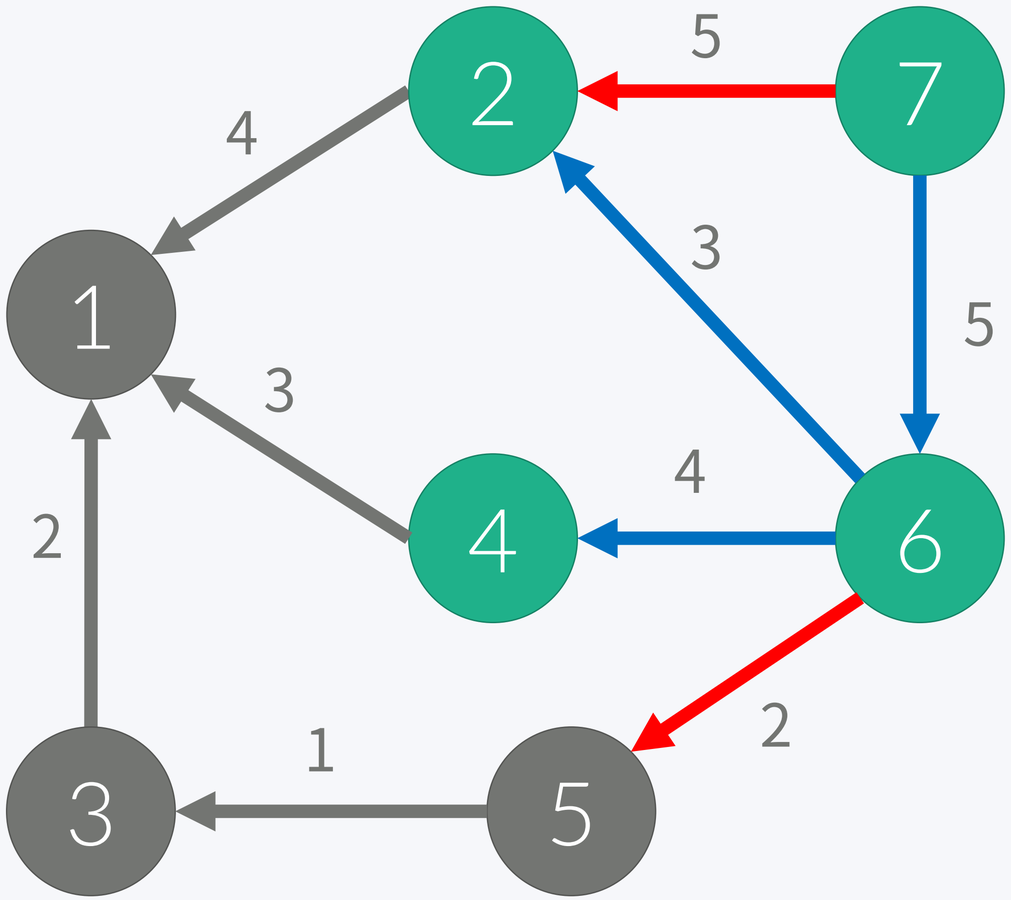


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

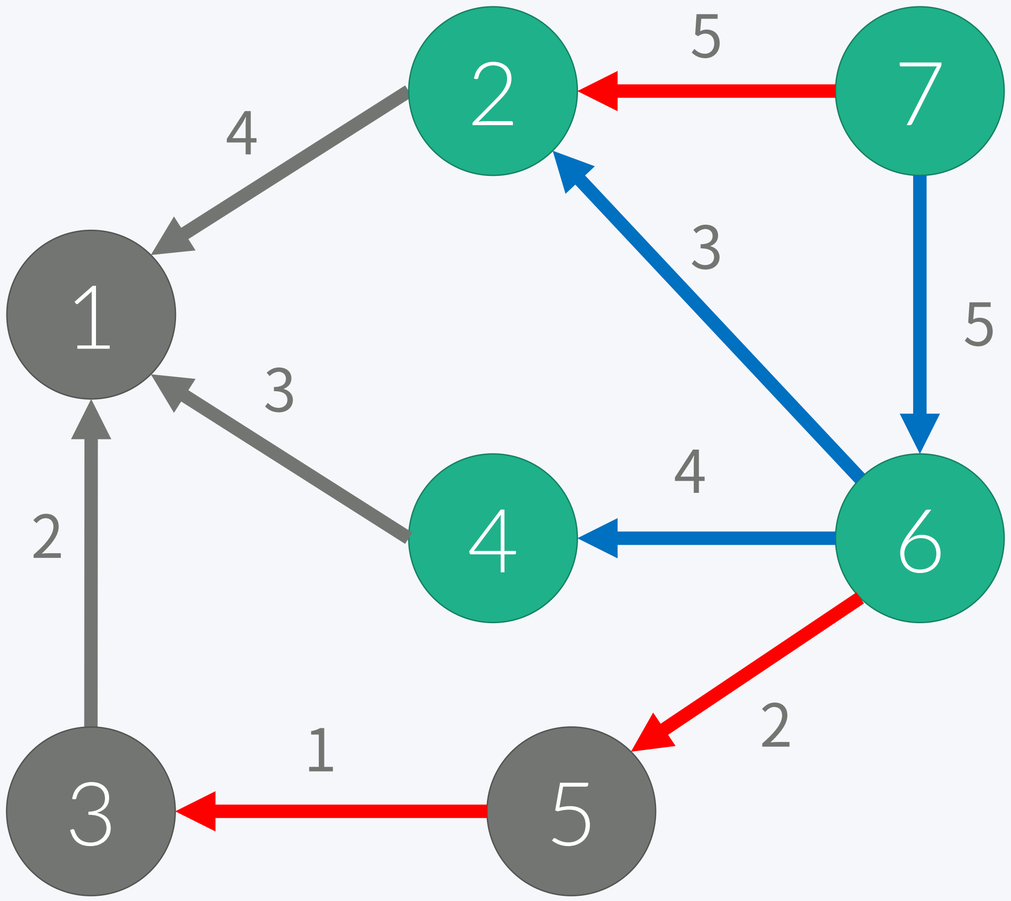


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

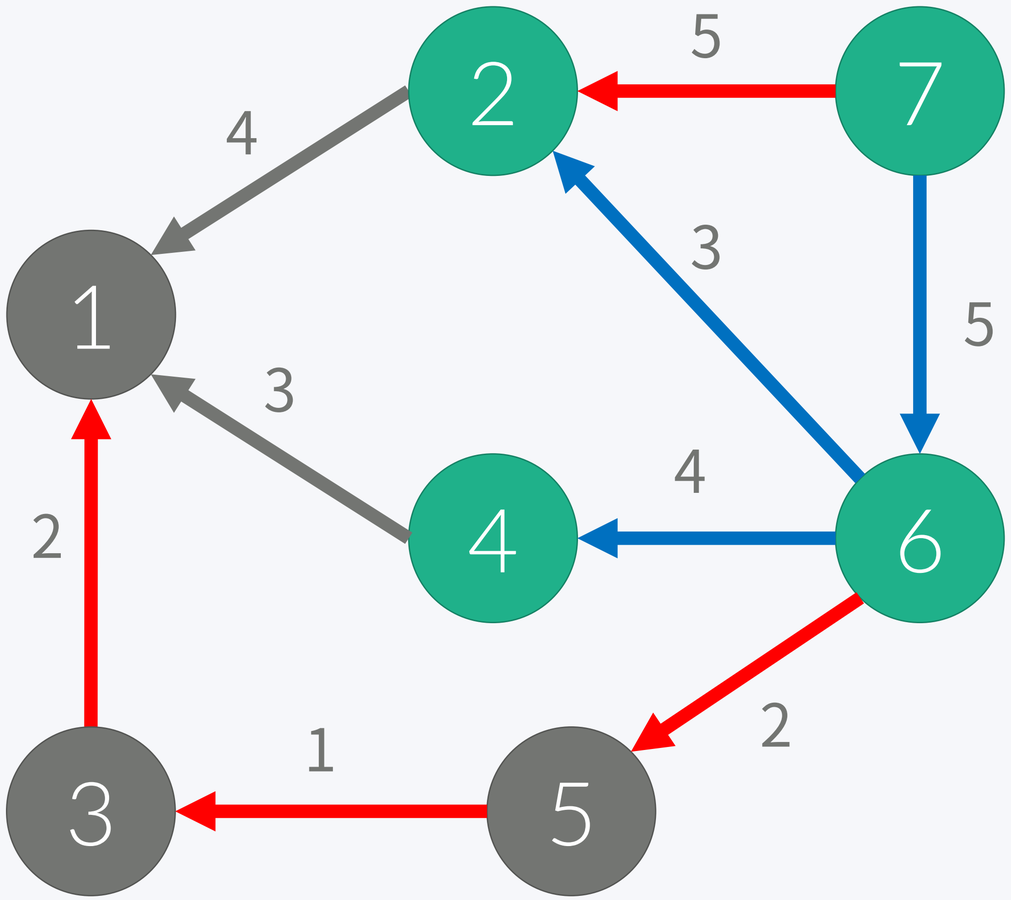


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

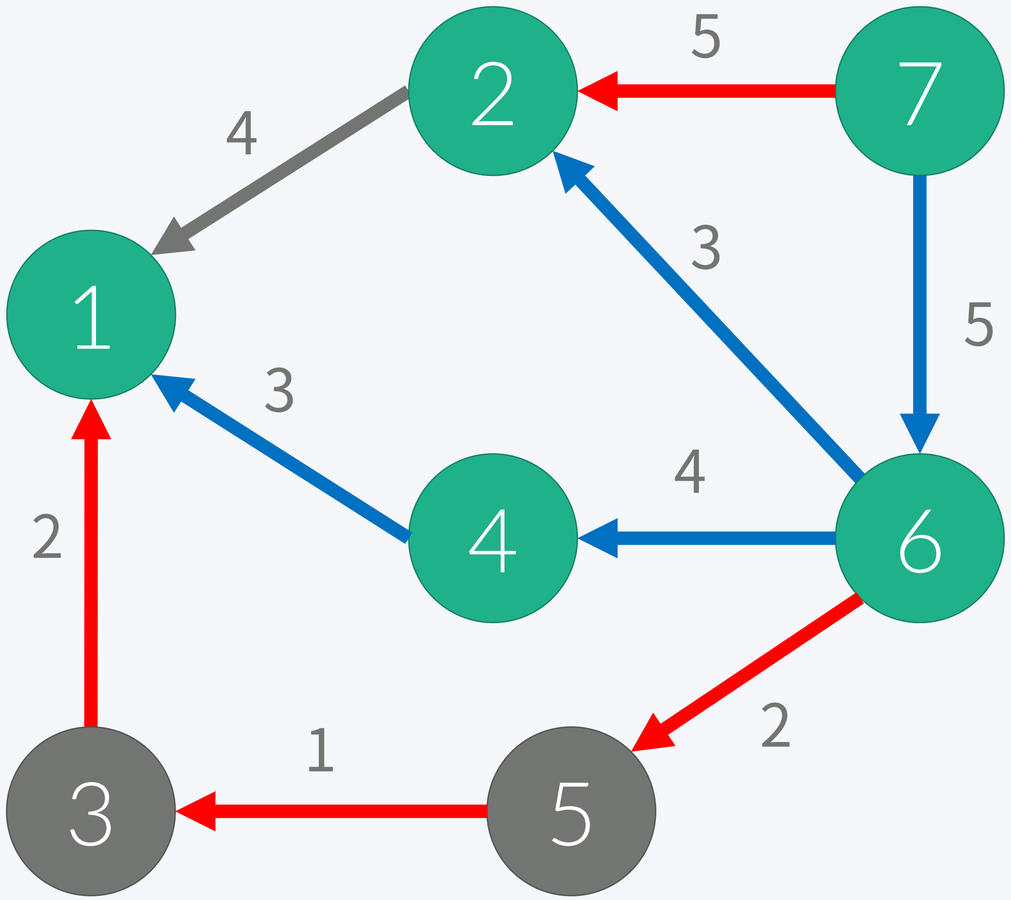


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제

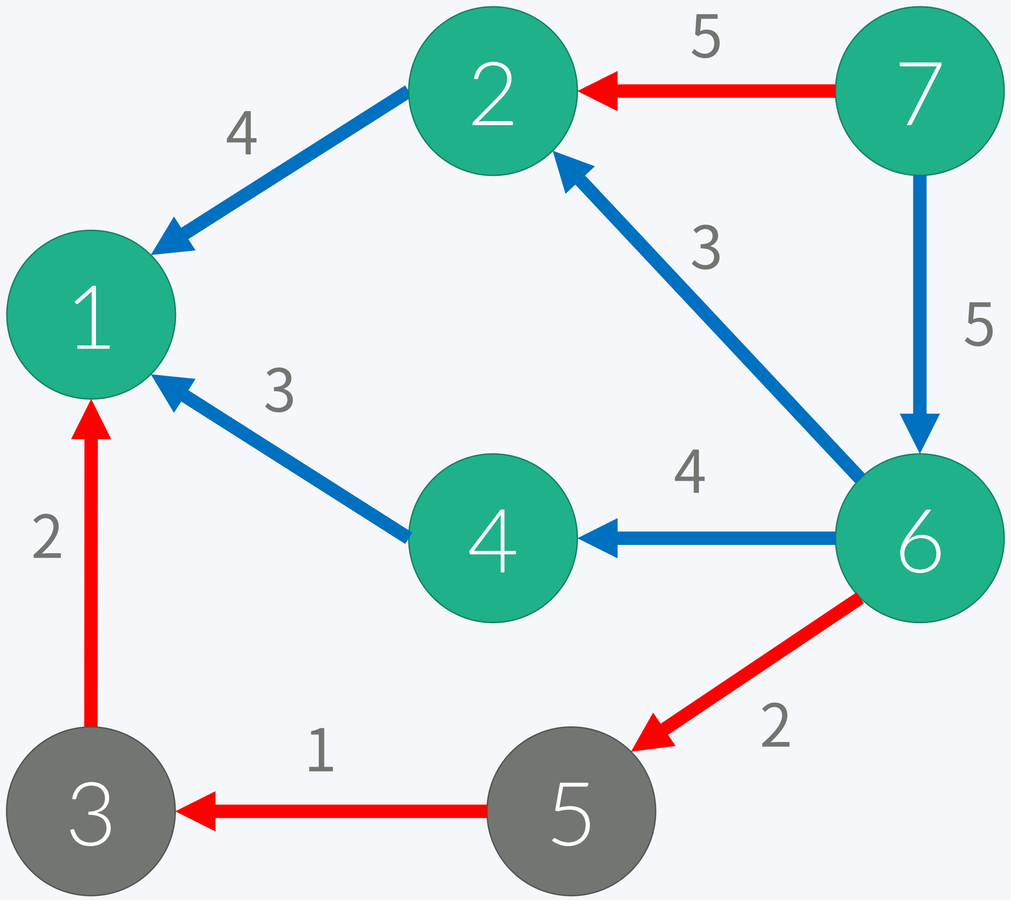


i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- DAG에서 가장 긴 경로를 구하는 문제



i	1	2	3	4	5	6	7
거리	0	4	2	3	3	7	12

임계 경로

https://www.acmicpc.net/problem/1948

- 소스: http://boj.kr/012f1ad05bd744878649bb6b56e4dc3f

도로 포장

https://www.acmicpc.net/problem/1162

- 도로를 포장하면 걸리는 시간이 0이다.
- 1에서 N으로 가는 최단 경로 중에서, K개 이하의 도로를 포장해서 최단 시간을 찾는 문제

도로 포장

https://www.acmicpc.net/problem/1162

- 그래프에서 최단 시간을 구하는 문제는 다익스트라 알고리즘으로 풀 수 있다

도로 포장

https://www.acmicpc.net/problem/1162

- 이 문제는 도로 포장 개념이 존재한다.
- 그럼, 어떤 정점 v에 도착했을 때, 1개를 포장하고 도착한 경우와 2개를 포장하고 도착한 경우는 서로 다른 경우이다.
- 정점을 k+1등분해서 다익스트라 알고리즘을 수행한다.
- (v, t) = 정점 v에 도착했는데, 포장한 도로는 t개 지나갔음

도로 포장

https://www.acmicpc.net/problem/1162

- 기존 다익스트라

```
if (dist[y] > dist[x] + a[x][i].cost) {
    dist[y] = dist[x] + a[x][i].cost;
    q.push(make_pair(-dist[y], y));
}
```

도로 포장

https://www.acmicpc.net/problem/1162

- 바뀐 다익스트라

```
if (step+1 <= k && dist[y][step+1] > dist[x][step]) {
    dist[y][step+1] = dist[x][step];
    q.push(make_tuple(-dist[y][step+1], y, step+1));
}
if (dist[y][step] > dist[x][step] + a[x][i].cost) {
    dist[y][step] = dist[x][step] + a[x][i].cost;
    q.push(make_tuple(-dist[y][step], y, step));
}
```

도로 포장

https://www.acmicpc.net/problem/1162

- 소스: http://boj.kr/acb1d5e8deb349d4bac1202ee6a87a9c

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- 1번 도시에서 다른 모든 도시로 가는 K번째 최단 경로를 찾는 문제

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- 최단 경로를 찾는 문제는 다익스트라를 이용해서 풀 수 있다
- 다익스트라를 응용해서 최단 경로 문제를 푼다.

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- 다익스트라의 dist 배열을 항상 K개의 최단 경로를 저장하게 구현한다.
- 이 때, 새로운 거리가 K개보다 작은 경우에는 그냥 추가하고
- K개와 같은 경우에는 dist에서 가장 큰 값과 비교한다.
- 비교했을 때, 가장 큰 값보다 작으면, 큰 값을 제거하고 추가한다.

K번째 최단경로 찾기

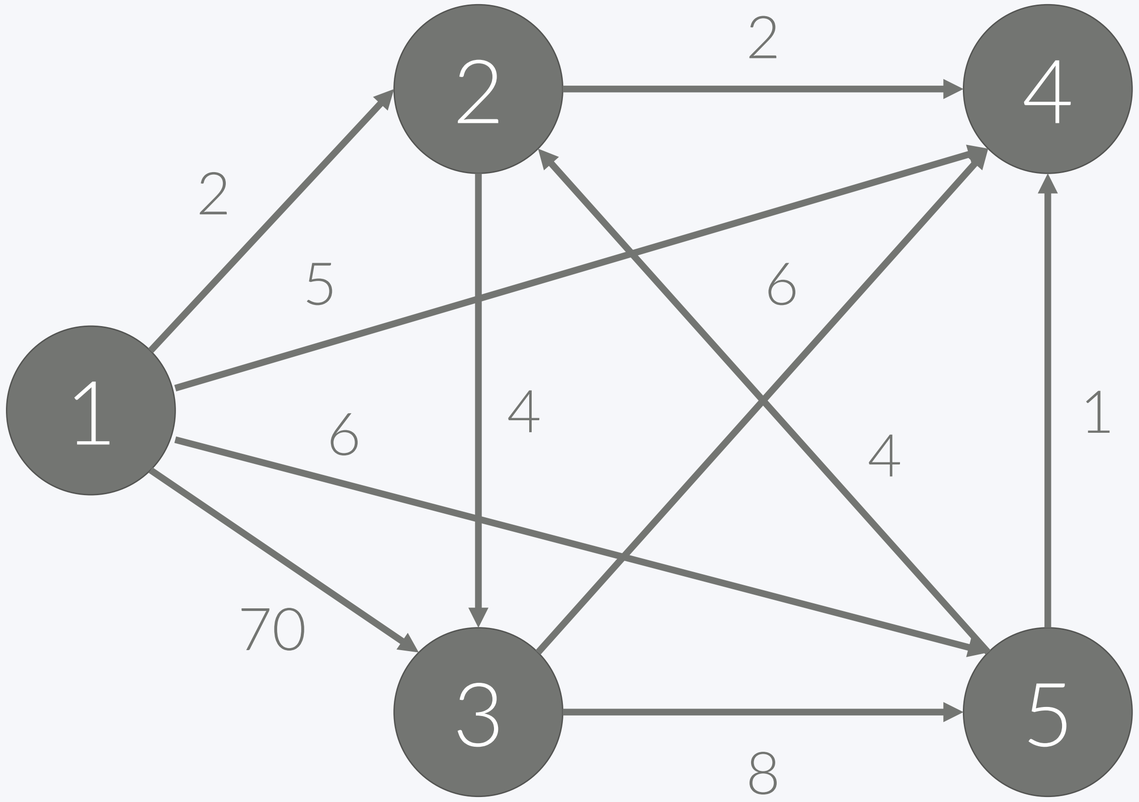
https://www.acmicpc.net/problem/1854

```
if (dist[y].size() < k || dist[y].top() > cur + a[x][i].cost) {
    if (dist[y].size() == k) {
        dist[y].pop();
    }
    dist[y].push(cur+a[x][i].cost);
    q.push(make_pair(-(cur+a[x][i].cost), y));
}
```

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- 시작



i	1	2	3	4	5
dist[1]	0				
dist[2]					

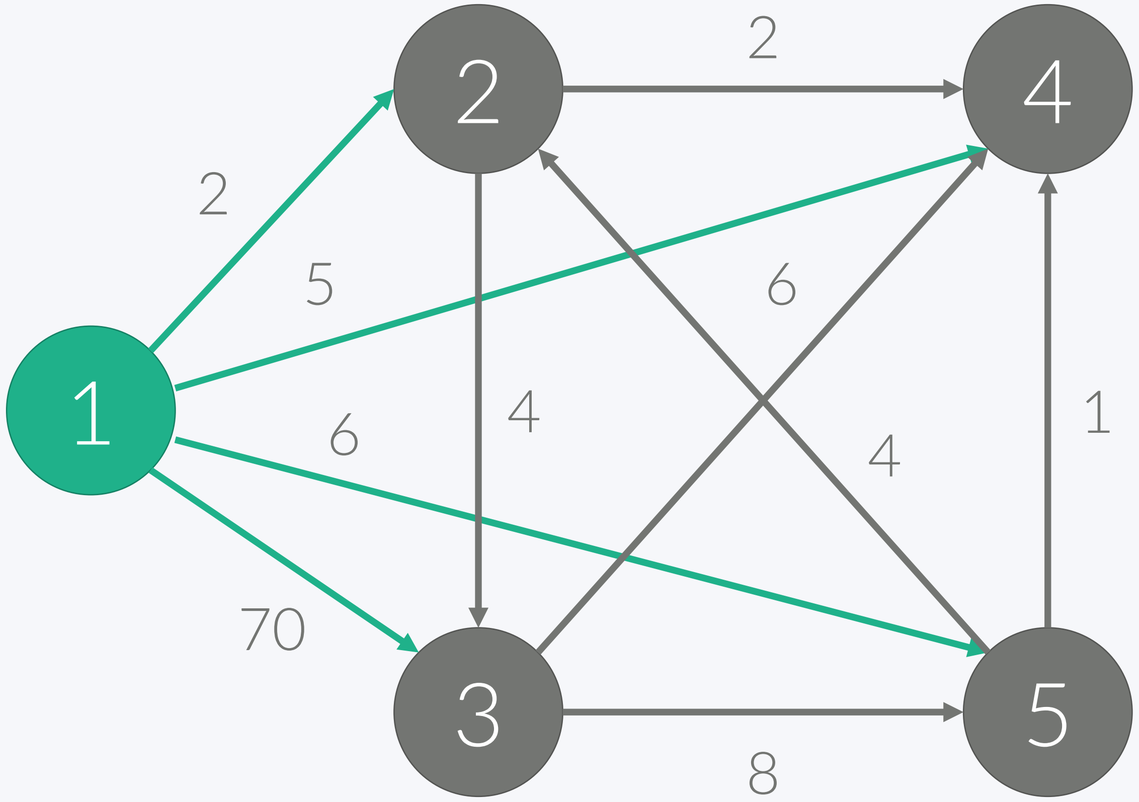
힙 (정점, 거리)

(1, 0)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (1, 0) 선택



i	1	2	3	4	5
dist[1]	0	2	70	5	6
dist[2]					

힙 (정점, 거리)

(2, 2)

(4, 5)

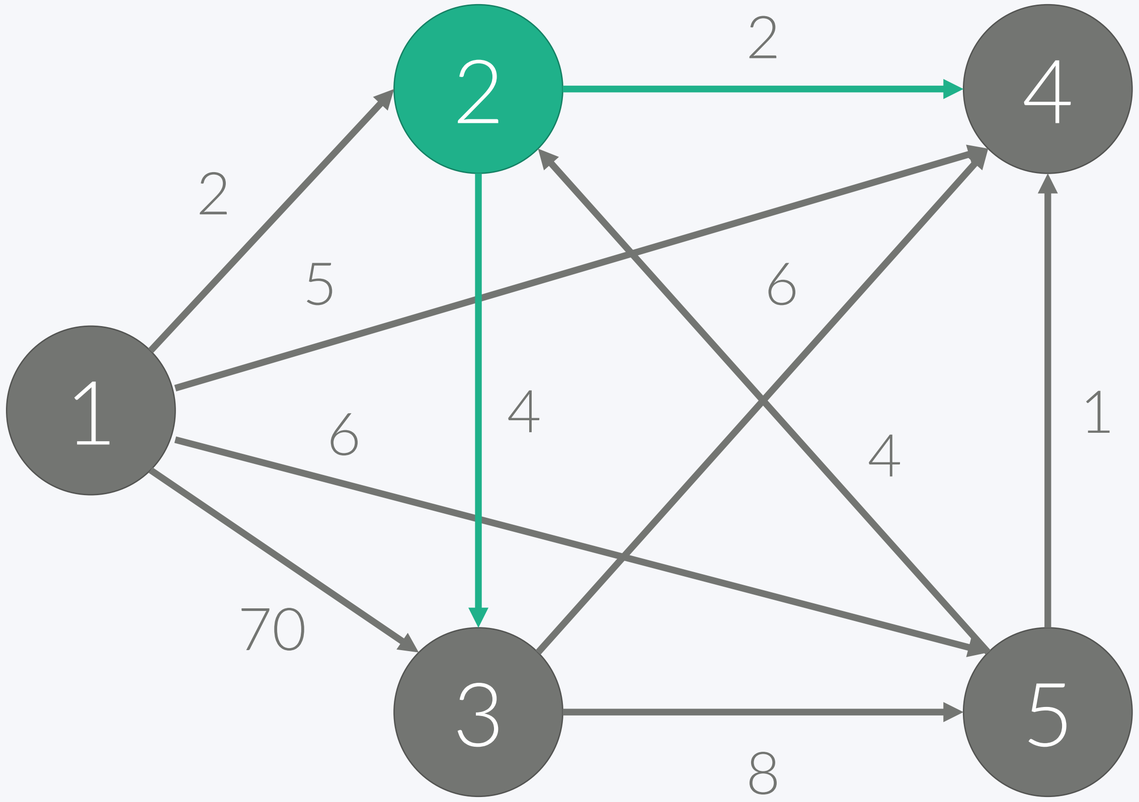
(5, 6)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (2, 2) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]			70	5	

힙 (정점, 거리)

(4, 4)

(4, 5)

(3, 6)

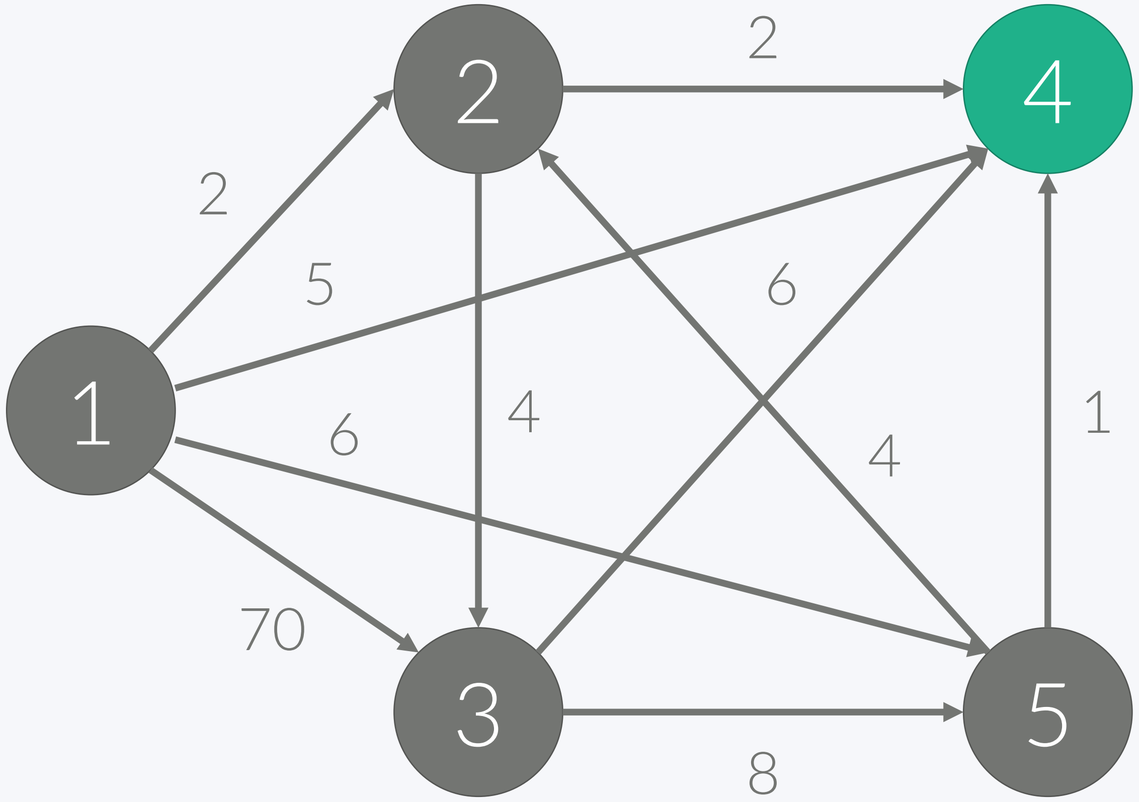
(5, 6)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (4, 4) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]			70	5	

힙 (정점, 거리)

(4, 5)

(3, 6)

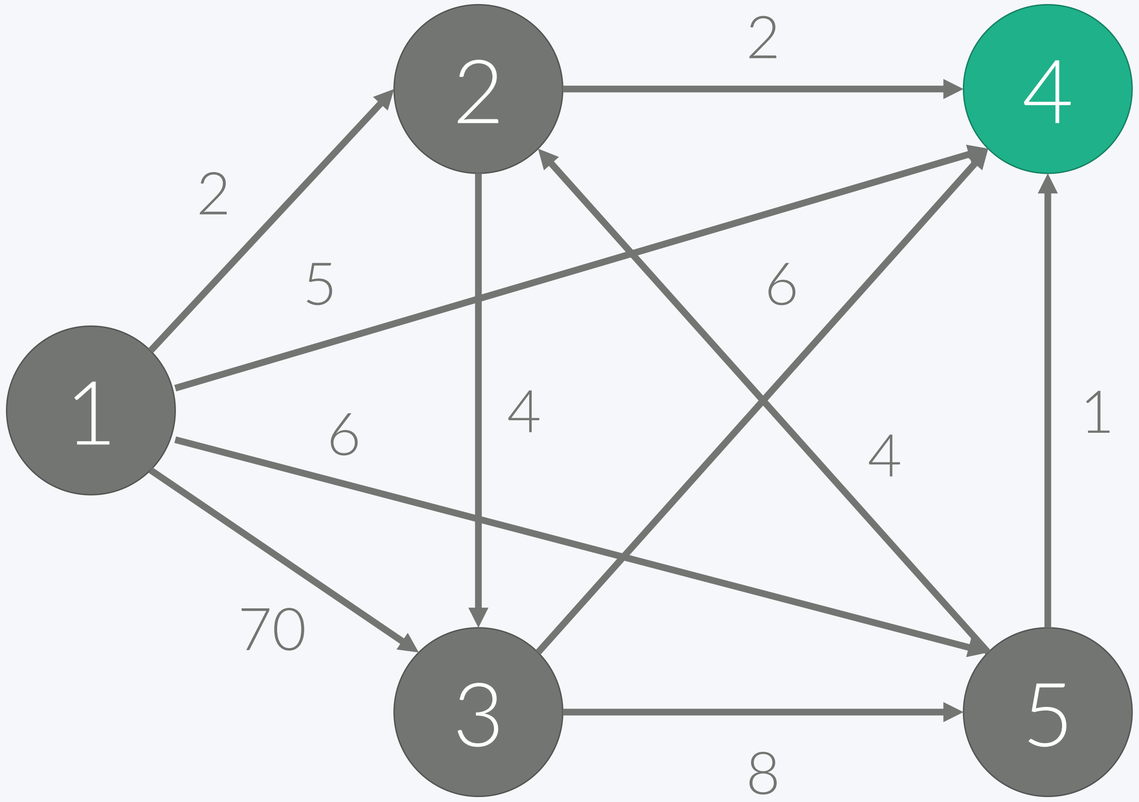
(5, 6)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (4, 5) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]			70	5	

힙 (정점, 거리)

(3, 6)

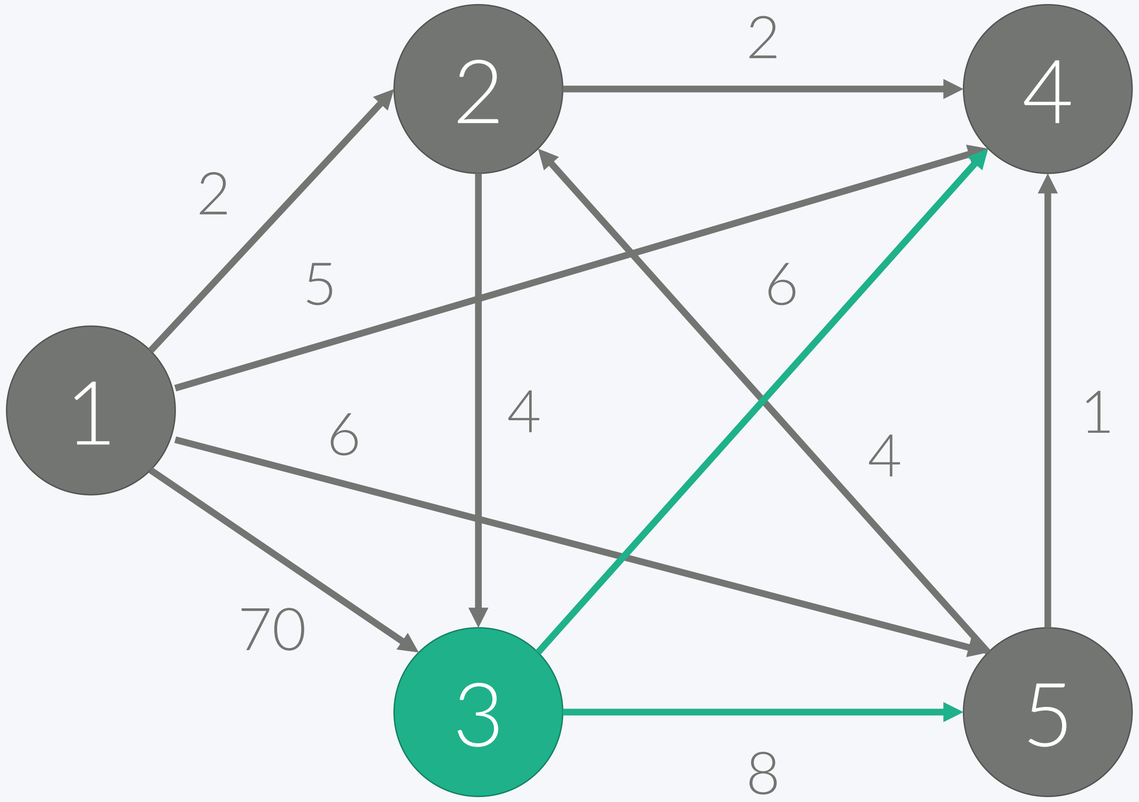
(5, 6)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (3, 6) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]			70	5	14

힙 (정점, 거리)

(5, 6)

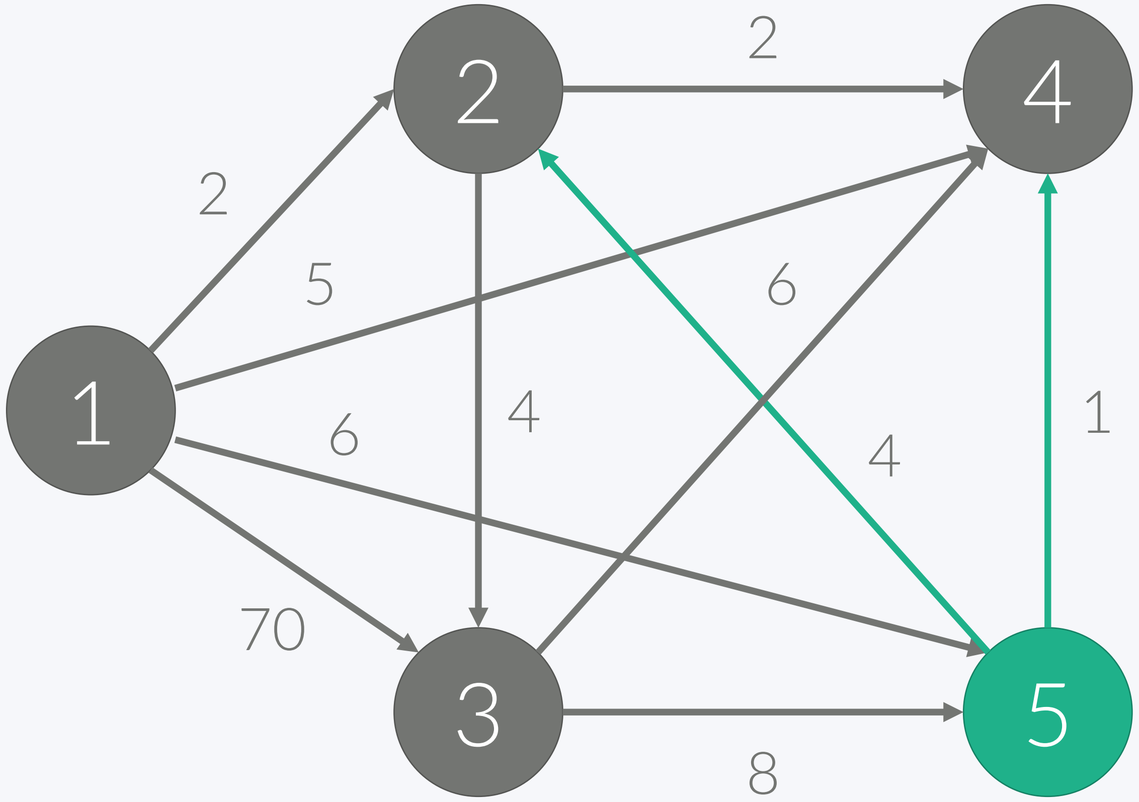
(5, 14)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (5, 6) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]		10	70	5	14

힙 (정점, 거리)

(2, 10)

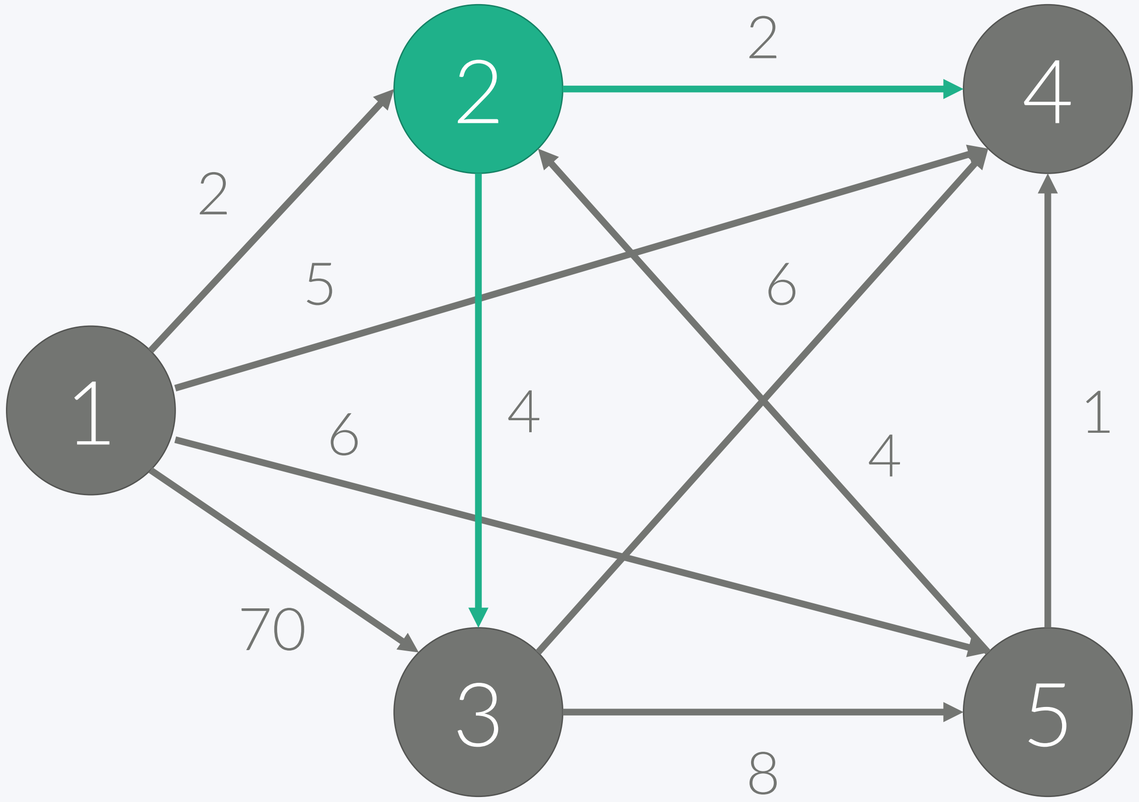
(5, 14)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (2, 10) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]		10	14	5	14

힙 (정점, 거리)

(5, 14)

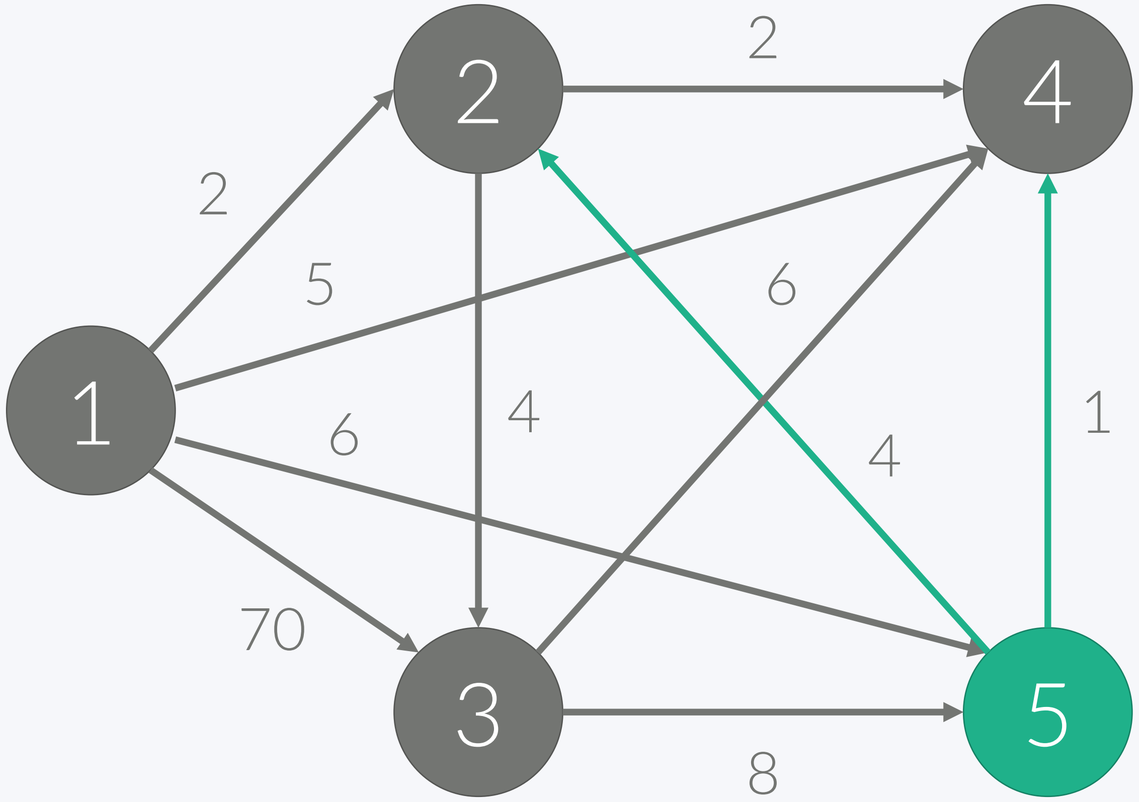
(3, 14)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (5, 14) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]		10	14	5	14

힙 (정점, 거리)

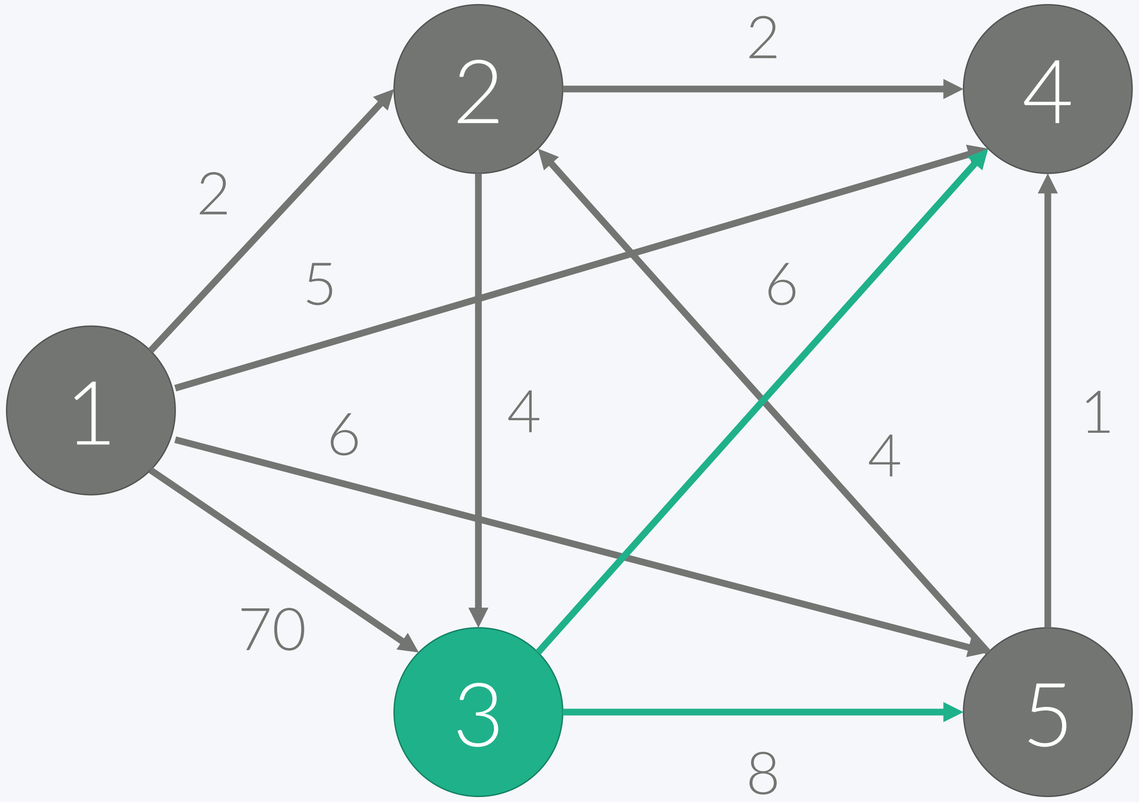
(3, 14)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (3, 14) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]		10	14	5	14

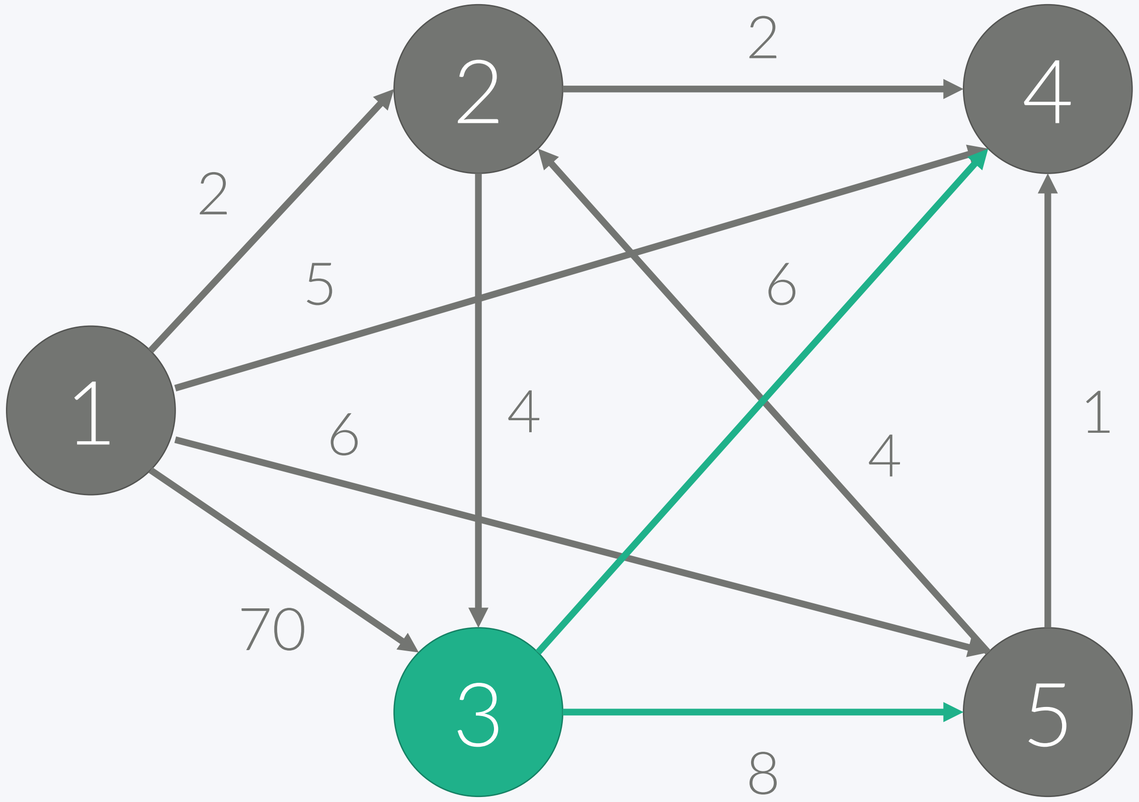
힙 (정점, 거리)

(3, 70)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- (3, 70) 선택



i	1	2	3	4	5
dist[1]	0	2	6	4	6
dist[2]		10	14	5	14

힙 (정점, 거리)

K번째 최단경로 찾기

https://www.acmicpc.net/problem/1854

- 소스: http://boj.kr/3c21834b4b8c4cda8bd6f4183823e652

길의 개수

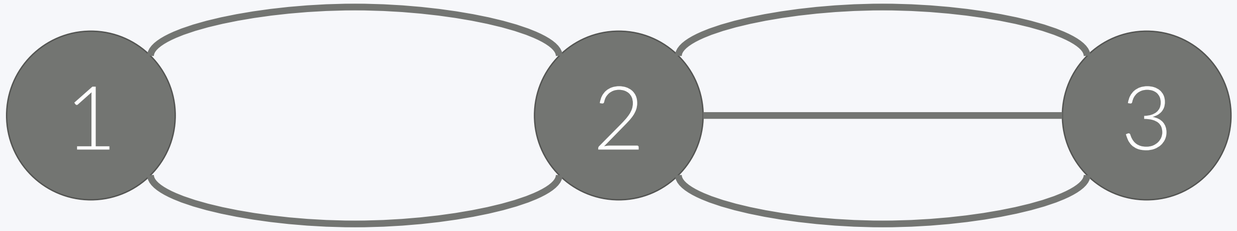
https://www.acmicpc.net/problem/1533

- 길의 정보가 인접 행렬로 주어졌을 때, S에서 E로 T분만에 가는 경로의 개수를 찾는 문제

길의 개수

https://www.acmicpc.net/problem/1533

- 인접 행렬 A[i][j]의 의미가 i에서 j로 가는 길의 개수라고 해보자.

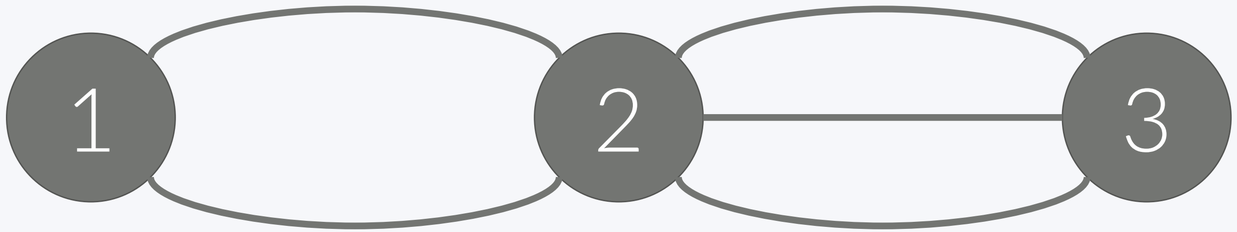


$$A = \begin{pmatrix} 0 & 2 & 0 \\ 2 & 0 & 3 \\ 0 & 3 & 0 \end{pmatrix}$$

길의 개수

https://www.acmicpc.net/problem/1533

- 인접 행렬 A[i][j]의 의미가 i에서 j로 가는 길의 개수라고 해보자.



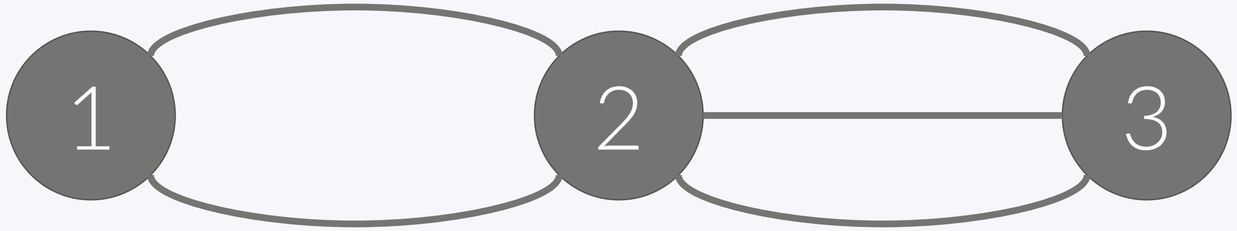
$$A = \begin{pmatrix} 0 & 2 & 0 \\ 2 & 0 & 3 \\ 0 & 3 & 0 \end{pmatrix}$$

$$A^2 = \begin{pmatrix} 4 & 0 & 6 \\ 0 & 13 & 0 \\ 6 & 0 & 9 \end{pmatrix}$$

길의 개수

https://www.acmicpc.net/problem/1533

- 인접 행렬 A[i][j]의 의미가 i에서 j로 가는 길의 개수라고 해보자.



$$A = \begin{pmatrix} 0 & 2 & 0 \\ 2 & 0 & 3 \\ 0 & 3 & 0 \end{pmatrix}$$

$$A^2 = \begin{pmatrix} 4 & 0 & 6 \\ 0 & 13 & 0 \\ 6 & 0 & 9 \end{pmatrix}$$

$$A^3 = \begin{pmatrix} 0 & 26 & 0 \\ 26 & 0 & 39 \\ 0 & 39 & 0 \end{pmatrix}$$

길의 개수

https://www.acmicpc.net/problem/1533

- 길의 정보가 인접 행렬로 주어졌을 때, S에서 E로 T분만에 가는 경로의 개수를 찾는 문제
- 인접 행렬의 의미는 A[i][j] = i에서 j로 가는 길의 소요 시간

길의 개수

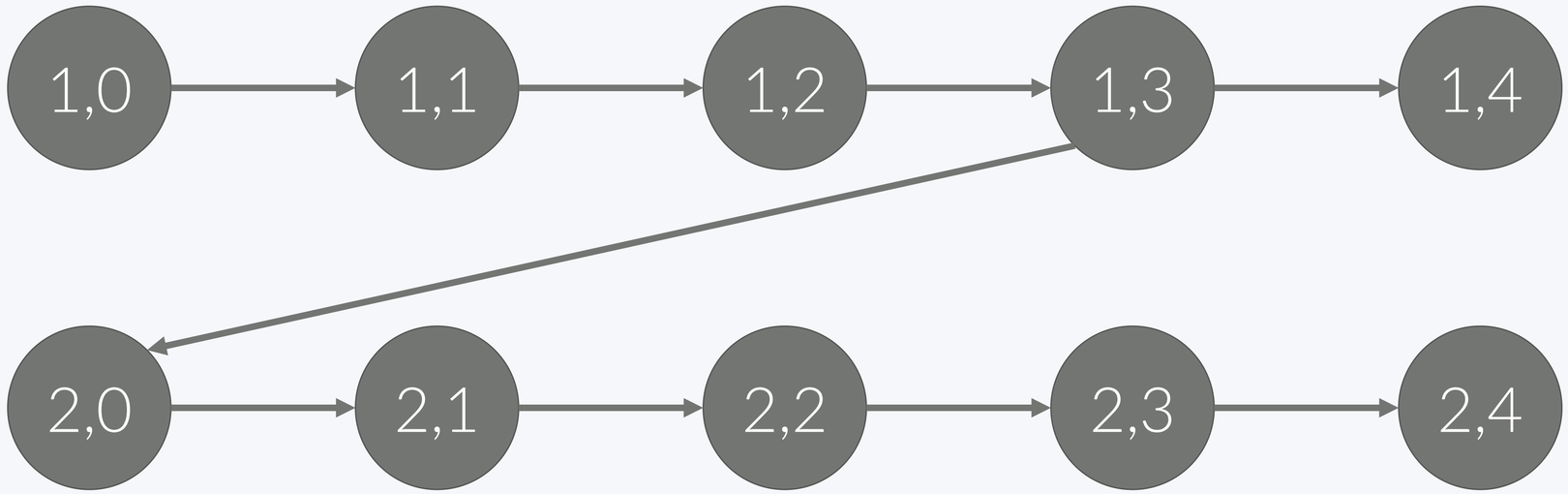
https://www.acmicpc.net/problem/1533

- 문제의 조건에 보면 A[i][j] ≤ 5 라는 조건이 있다.
- 이 조건을 이용해서 정점을 5등분해서 사용한다.
- (v, t) = t분 후 정점 v의 상태

길의 개수

https://www.acmicpc.net/problem/1533

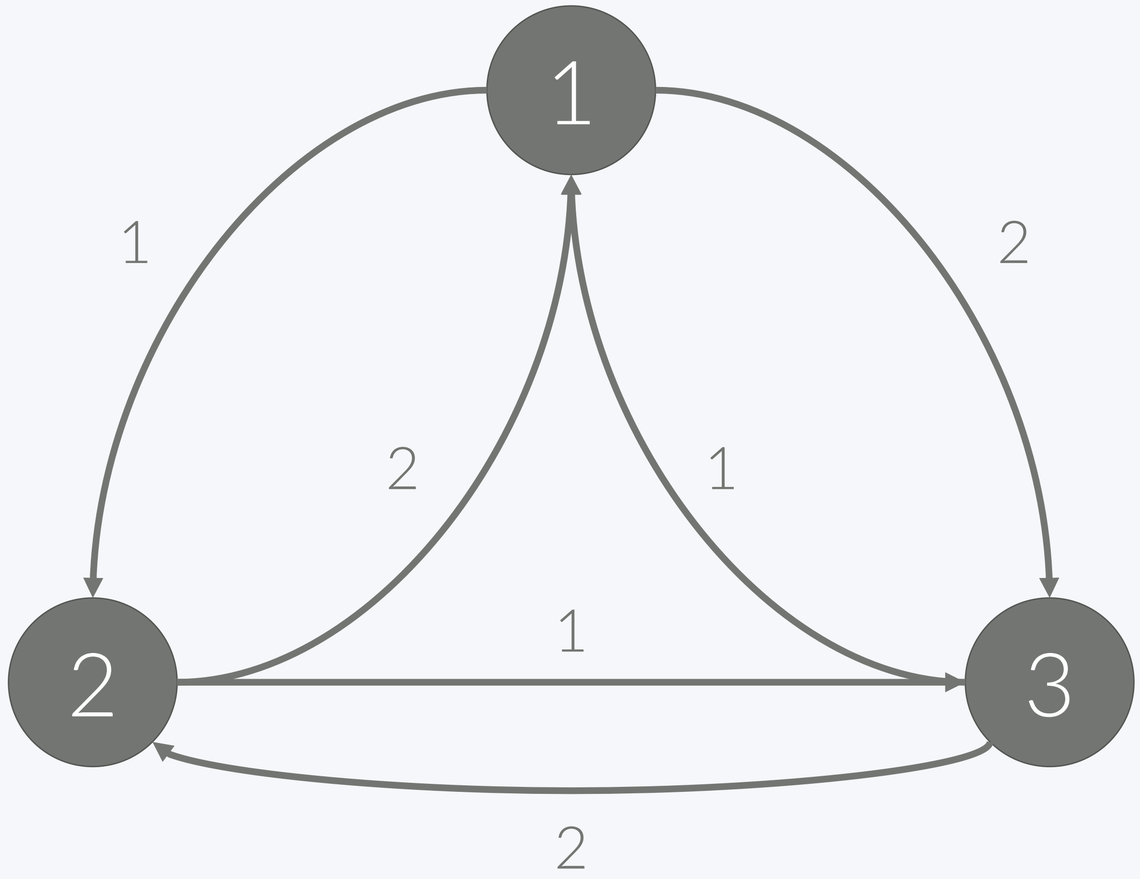
- A[1][2] = 4 라는 조건은 (1, 3)에서 (2, 0)으로 가는 간선으로 나타낼 수 있다.



길의 개수

https://www.acmicpc.net/problem/1533

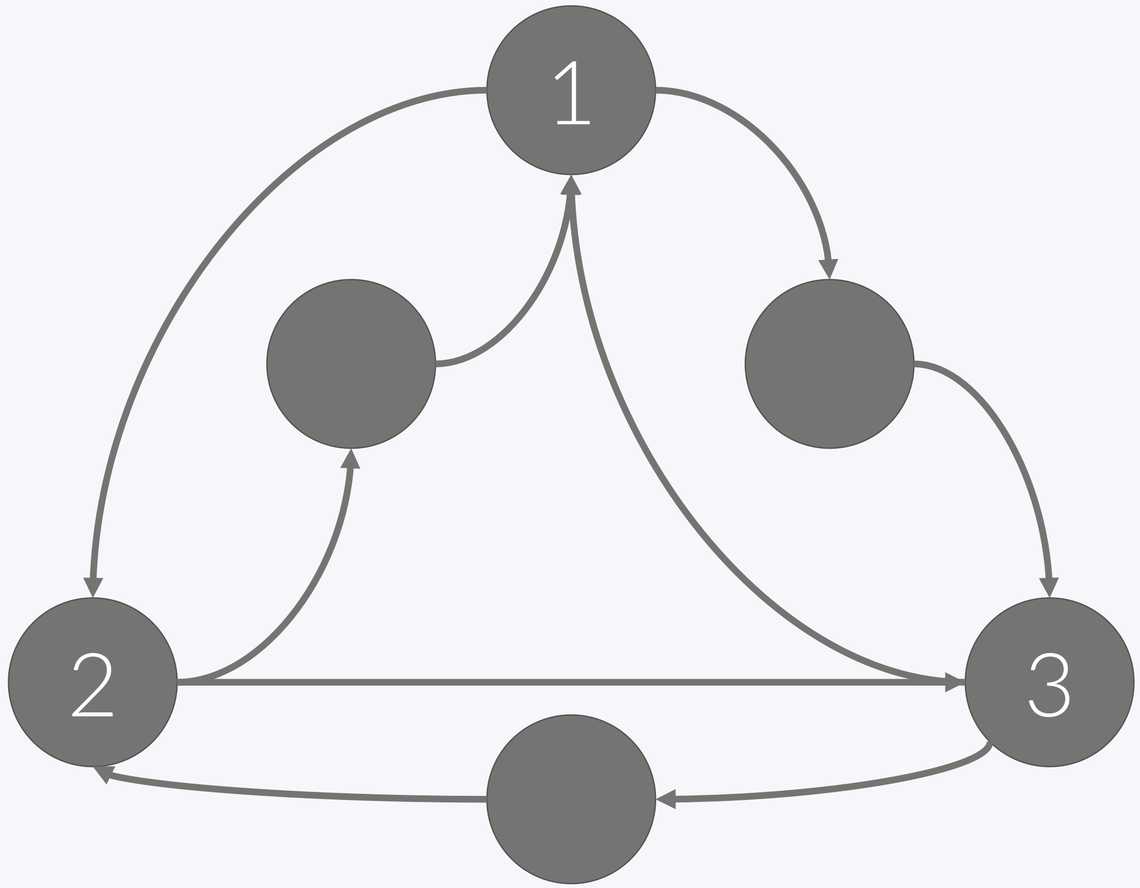
- 문제의 예제 그래프



길의 개수

https://www.acmicpc.net/problem/1533

- 문제의 예제 그래프



길의 개수

https://www.acmicpc.net/problem/1533

- 소스: http://boj.kr/d2fe8255543a44fe88483104e389eb58

두 가중치

https://www.acmicpc.net/problem/12930

- 그래프의 모든 간선은 두 개의 가중치를 가지고 있다.
- 경로의 비용은 가중치 1을 모두 더한 값인 W1과 가중치 2를 모두 더한 값인 W2를 곱해서 구할 수 있다.

두 가중치

https://www.acmicpc.net/problem/12930

- 최단 거리를 구하는 문제이다

두 가중치

https://www.acmicpc.net/problem/12930

- 다익스트라!

두 가중치

https://www.acmicpc.net/problem/12930

- 경로의 비용은 W1 * W2이다

두 가중치

https://www.acmicpc.net/problem/12930

- 경로의 비용은 W1 * W2이다
- 곱을 최소로 만들기 위해, W2를 고정시키고, W1을 감소시키는 방식을 생각해볼 수 있다.

두 가중치

https://www.acmicpc.net/problem/12930

- dist[i][j] = i번 정점에 도착했고, W2가 j일 때 W1의 최소 비용

두 가중치

https://www.acmicpc.net/problem/12930

- 소스: http://boj.kr/2500847db73d4086a2c120dfd2237811

일방통행

https://www.acmicpc.net/problem/1412

- 그래프가 주어졌을 때, 양방햔 간선에 방향을 결정한다.
- 이 때, 임의의 도시 x에서 출발해서 다시 그 도시 x로 돌아올 수 없게 만드는 것

일방통행

https://www.acmicpc.net/problem/1412

- 즉, 사이클이 없게 양방향 간선의 방향을 결정하는 문제이다

일방통행

https://www.acmicpc.net/problem/1412

- 원래 그래프에 무방향 그래프로 이루어진 사이클이 있으면 항상 NO이다

일방통행

https://www.acmicpc.net/problem/1412

- 완전 그래프가 있을 때
- u <-> v 간선이 있으면
- min(u,v) -> max(u, v)로 간선을 이어주면 사이클이 없게 된다

일방통행

https://www.acmicpc.net/problem/1412

- 원래 그래프에 방향 그래프로 이루어진 사이클이 없다면
- 양방향 그래프의 각 간선을 사이클이 없게 그래프에 추가한다

일방통행

https://www.acmicpc.net/problem/1412

- 원래 그래프에 방향 그래프로 이루어진 사이클이 없다면
- 양방향 그래프의 각 간선을 사이클이 없게 그래프에 추가한다
- 방향 그래프로만 이루어진 그래프를 위상 정렬해서 순서를 찾은 다음에
- 순서가 낮은것에서 큰 것으로 양방향 그래프의 간선을 이어주면 된다

일방통행

https://www.acmicpc.net/problem/1412

- 즉, 원래 그래프에 방향 간선으로 이루어진 사이클이 없으면 YES이다

일방통행

https://www.acmicpc.net/problem/1412

- 소스: http://boj.kr/48dac30194764be1ae6f7cbf7c7df02f

역사

https://www.acmicpc.net/problem/1613

- 역사적 사건의 전후 관계를 알고 있을 때
- 주어진 사건의 전후 관계를 알 수 있을까?

역사

https://www.acmicpc.net/problem/1613

- 입력으로 주어지는 사건에 모순인 관계가 없기 때문에
- 먼저 일어난 사건 -> 나중에 일어난 사건으로 간선을 연결할 수 있다

역사

https://www.acmicpc.net/problem/1613

- 두 사건 x, y가 있을 때,
- x에서 y로 가는 경로가 있으면 x가 먼저 일어난 것이고
- y에서 x로 가는 경로가 있으면 y가 먼저 일어난 것이다
- 두 경우가 아니면 알 수 없는 경우

역사

https://www.acmicpc.net/problem/1613

- 소스: http://boj.kr/6854b86f06924f19a5ae760b28bbd7da

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 마을은 N개의 집과 M개의 길로 이루어져 있다
- 길은 양방향이고, 유지비가 있다
- 마을을 두 개의 분리된 마을로 나누려고 한다
- 분리된 마을 안에 집이 서로 연결되어 있어야 한다
- 즉, 분리된 마을 안에 있는 임의의 두 집 사이에 경로가 존재해야 한다
- 두 마을 사이의 길은 필요가 없으니 없앨 수 있다
- 분리된 마을 안에서도 경로가 항상 존재하면, 길을 없앨 수 있다
- 길의 유지비의 합의 최소값을 구하는 문제

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 임의의 두 정점 사이의 경로가 존재해야 한다 =>

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 임의의 두 정점 사이의 경로가 존재해야 한다 => 트리를 만들어야 한다

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 그래프를 트리로 만들어야 한다 =>

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 그래프를 트리로 만들어야 한다 => MST를 구해야 한다

도시 분할 계획

https://www.acmicpc.net/problem/1647

- MST를 구한 다음, 가장 가중치가 큰 간선을 제거하면 두 마을로 나눌 수 있고
- 각 분리된 마을 사이에 연결된 경로가 존재하게 된다

도시 분할 계획

https://www.acmicpc.net/problem/1647

- 소스: http://boj.kr/ad14784e345c4fcf80cf177f5f78fad4

The game of death

https://www.acmicpc.net/problem/2099

- N명의 사람들이 원을 이루고 모여 앉아있다
- 사람들에게는 1번부터 N번까지 번호가 매겨져 있다
- 각 사람은 자신의 양 손을 이용해서 동시에 두 명의 사람을 가리킨다.
- 자기 자신은 가리킬 수 없으나 자신이 가리키는 두 사람이 꼭 다를 필요는 없다.
- 그리고 첫 사람이 자연수 K를 하나 정한 뒤에, 자신의 가리키고 있는 두 사람 중 한 사람을 지목한다.
- 첫 번째로 지목된 사람은 마찬가지로 자신이 가리키고 있는 두 사람 중 한 사람을 지목한다.
- 마찬가지로 지목된 사람이 같은 방법으로 사람들 지목해 나가며, K번째로 지목된 사람이 걸리게 되는 게임

The game of death

https://www.acmicpc.net/problem/2099

- N과 K, 그리고 각 사람이 지목한 두 사람에 대한 정보가 주어졌을 때, a번 사람이 시작했을 때 b번 사람이 걸리는 경우가 있는지 없는지를 알아내는 문제

The game of death

https://www.acmicpc.net/problem/2099

- 그래프에서 a에서 b까지 간선 K개를 거쳐서 갈 수 있는지 없는지 구하는 문제

The game of death

https://www.acmicpc.net/problem/2099

- 인접행렬의 제곱을 이용해서 해결할 수 있다
- A[i][j] = i에서 j까지 간선 1개를 거쳐서 갈 수 있는가?
- A^k[i][j] = i에서 j까지 간선 k개를 거쳐서 갈 수 있는가?

The game of death

https://www.acmicpc.net/problem/2099

- 소스: http://boj.kr/25cd82d5371441e585f1052daa38b2ed