

백준 11404, 플로이드

키워드 : 플로이드-와샬 알고리즘

인접행렬

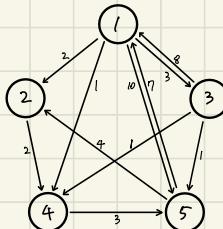
	1	2	3	4	5
1	0	2	3	1	10
2	INF	0	INF	2	INF
3	8	INF	0	1	1
4	INF	INF	INF	0	3
5	7	4	INF	INF	0

$d^{(k=0)}$

	1	2	3	4	5
1	0	2	3	1	10
2	INF	0	INF	2	INF
3	8	INF	0	1	1
4	INF	INF	INF	0	3
5	7	4	INF	INF	0

$\pi^{(k=0)}$

	1	2	3	4	5
1	NIL	1	1	1	1
2	NIL	NIL	N	2	NIL
3	3	NIL	NIL	3	3
4	NIL	NIL	NIL	NIL	4
5	5	5	NIL	NIL	NIL



$d_{ij}^{(k)}$: 경점 $i \rightarrow j$ 까지 최단경로
 $\{1, 2, 3, \dots, k\}$: $i \rightarrow j$ 까지의 경로의 집합

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k=0 \\ \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}) & \text{if } k \geq 1 \end{cases}$$

π : 직전 경계점 배열

① 경계 i 를 3 사이에 경로가 존재하지 않으면 $\pi_{ij}=NIL$
 ② π_{ij} 는 경점 j 의 직전 경점이다

$k=1$ 일 때 d, π 값 변화

	1	2	3	4	5
1	0	2	3	1	10
2	INF	0	INF	2	INF
3	8	INF	0	1	1
4	INF	INF	INF	0	3
5	7	4	INF	INF	0

	1	2	3	4	5
1	0	2	3	1	10
2	INF	0	INF	2	INF
3	8	10	0	1	1
4	INF	INF	INF	0	3
5	7	4	10	8	0

	1	2	3	4	5
1	NIL	1	1	1	1
2	NIL	NIL	N	2	NIL
3	3	1	NIL	3	3
4	NIL	NIL	NIL	NIL	4
5	5	5	1	1	NIL

$$d_{ij}^{(k=1)} = \min(d_{ij}^{(k=0)}, d_{ik}^{(k=0)} + d_{kj}^{(k=0)})$$

$k=1$

$$\begin{aligned} d_{11}^{(k=1)} &= \min(d_{11}^{(k=0)}, d_{11}^{(k=0)} + d_{11}^{(k=0)}) \\ &= \min(0, 0+0)=0 \end{aligned}$$

$d_{12}^{(k=1)}$

$$\begin{aligned} d_{12}^{(k=1)} &= \min(d_{12}^{(k=0)}, d_{11}^{(k=0)} + d_{12}^{(k=0)}) \\ &= \min(2, 0+2)=2 \end{aligned}$$

$d_{13}^{(k=1)}$

$$\begin{aligned} d_{13}^{(k=1)} &= \min(d_{13}^{(k=0)}, d_{11}^{(k=0)} + d_{13}^{(k=0)}) \\ &= \min(3, 0+3)=3 \end{aligned}$$

$d_{14}^{(k=1)}$

$$\begin{aligned} d_{14}^{(k=1)} &= \min(d_{14}^{(k=0)}, d_{11}^{(k=0)} + d_{14}^{(k=0)}) \\ &= \min(1, 0+1)=1 \end{aligned}$$

$d_{15}^{(k=1)}$

$$\begin{aligned} d_{15}^{(k=1)} &= \min(d_{15}^{(k=0)}, d_{11}^{(k=0)} + d_{15}^{(k=0)}) \\ &= \min(10, 0+10)=10 \end{aligned}$$

$$\begin{aligned} d_{21}^{(k=1)} &= \min(d_{21}^{(k=0)}, d_{21}^{(k=0)} + d_{11}^{(k=0)}) \\ &= \min(INF, INF+0)=INF \end{aligned}$$

$d_{22}^{(k=1)}$

$$\begin{aligned} d_{22}^{(k=1)} &= \min(d_{22}^{(k=0)}, d_{21}^{(k=0)} + d_{22}^{(k=0)}) \\ &= \min(0, INF+2)=INF \end{aligned}$$

$d_{23}^{(k=1)}$

$$\begin{aligned} d_{23}^{(k=1)} &= \min(d_{23}^{(k=0)}, d_{21}^{(k=0)} + d_{23}^{(k=0)}) \\ &= \min(INF, INF+2)=INF \end{aligned}$$

$d_{24}^{(k=1)}$

$$\begin{aligned} d_{24}^{(k=1)} &= \min(d_{24}^{(k=0)}, d_{21}^{(k=0)} + d_{24}^{(k=0)}) \\ &= \min(2, INF+2)=INF \end{aligned}$$

$d_{25}^{(k=1)}$

$$\begin{aligned} d_{25}^{(k=1)} &= \min(d_{25}^{(k=0)}, d_{21}^{(k=0)} + d_{25}^{(k=0)}) \\ &= \min(INF, INF+10)=INF \end{aligned}$$

$$\begin{aligned} d_{31}^{(k=1)} &= \min(d_{31}^{(k=0)}, d_{31}^{(k=0)} + d_{11}^{(k=0)}) \\ &= \min(8, 8+0)=8 \end{aligned}$$

$d_{32}^{(k=1)}$

$$\begin{aligned} d_{32}^{(k=1)} &= \min(d_{32}^{(k=0)}, d_{31}^{(k=0)} + d_{32}^{(k=0)}) \\ &= \min(INF, 8+2)=10 \end{aligned}$$

$d_{33}^{(k=1)}$

$$\begin{aligned} d_{33}^{(k=1)} &= \min(d_{33}^{(k=0)}, d_{31}^{(k=0)} + d_{33}^{(k=0)}) \\ &= \min(0, 8+3)=3 \end{aligned}$$

$d_{34}^{(k=1)}$

$$\begin{aligned} d_{34}^{(k=1)} &= \min(d_{34}^{(k=0)}, d_{31}^{(k=0)} + d_{34}^{(k=0)}) \\ &= \min(1, 8+1)=1 \end{aligned}$$

$d_{35}^{(k=1)}$

$$\begin{aligned} d_{35}^{(k=1)} &= \min(d_{35}^{(k=0)}, d_{31}^{(k=0)} + d_{35}^{(k=0)}) \\ &= \min(1, 8+10)=11 \end{aligned}$$

$$\begin{aligned} d_{41}^{(k=1)} &= \min(d_{41}^{(k=0)}, d_{41}^{(k=0)} + d_{11}^{(k=0)}) \\ &= \min(INF, INF+0)=INF \end{aligned}$$

$d_{42}^{(k=1)}$

$$\begin{aligned} d_{42}^{(k=1)} &= \min(d_{42}^{(k=0)}, d_{41}^{(k=0)} + d_{42}^{(k=0)}) \\ &= \min(INF, INF+2)=INF \end{aligned}$$

$d_{43}^{(k=1)}$

$$\begin{aligned} d_{43}^{(k=1)} &= \min(d_{43}^{(k=0)}, d_{41}^{(k=0)} + d_{43}^{(k=0)}) \\ &= \min(INF, INF+3)=INF \end{aligned}$$

$d_{44}^{(k=1)}$

$$\begin{aligned} d_{44}^{(k=1)} &= \min(d_{44}^{(k=0)}, d_{41}^{(k=0)} + d_{44}^{(k=0)}) \\ &= \min(0, INF+4)=0 \end{aligned}$$

$d_{45}^{(k=1)}$

$$\begin{aligned} d_{45}^{(k=1)} &= \min(d_{45}^{(k=0)}, d_{41}^{(k=0)} + d_{45}^{(k=0)}) \\ &= \min(1, INF+10)=11 \end{aligned}$$

$$\begin{aligned} d_{51}^{(k=1)} &= \min(d_{51}^{(k=0)}, d_{51}^{(k=0)} + d_{11}^{(k=0)}) \\ &= \min(7, 7+0)=7 \end{aligned}$$

$d_{52}^{(k=1)}$

$$\begin{aligned} d_{52}^{(k=1)} &= \min(d_{52}^{(k=0)}, d_{51}^{(k=0)} + d_{52}^{(k=0)}) \\ &= \min(4, 7+2)=9 \end{aligned}$$

$d_{53}^{(k=1)}$

$$\begin{aligned} d_{53}^{(k=1)} &= \min(d_{53}^{(k=0)}, d_{51}^{(k=0)} + d_{53}^{(k=0)}) \\ &= \min(INF, 7+3)=10 \end{aligned}$$

$d_{54}^{(k=1)}$

$$\begin{aligned} d_{54}^{(k=1)} &= \min(d_{54}^{(k=0)}, d_{51}^{(k=0)} + d_{54}^{(k=0)}) \\ &= \min(1, 7+10)=8 \end{aligned}$$

$d_{55}^{(k=1)}$

$$\begin{aligned} d_{55}^{(k=1)} &= \min(d_{55}^{(k=0)}, d_{51}^{(k=0)} + d_{55}^{(k=0)}) \\ &= \min(0, 7+10)=0 \end{aligned}$$

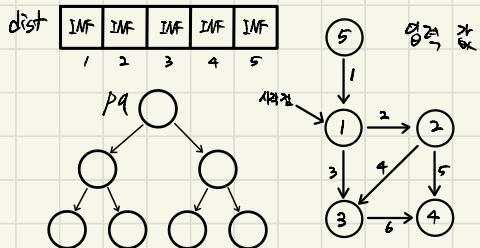
백준 1753, 최단 경로

키워드 : 다익스트라 알고리즘

Step 1. 변수 초기화

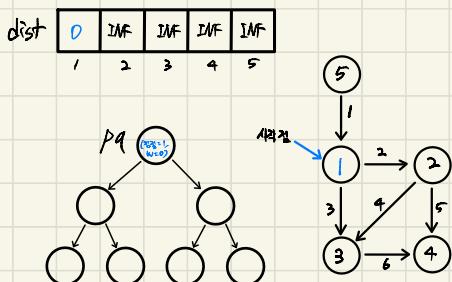
dist: 시작점부터 최단 거리를 저장하는 배열

PQ: 현재 경유지에서 최단 거리로 갈 수 있는 간선을 저장하는 priority-queue (가중치 힙구조)



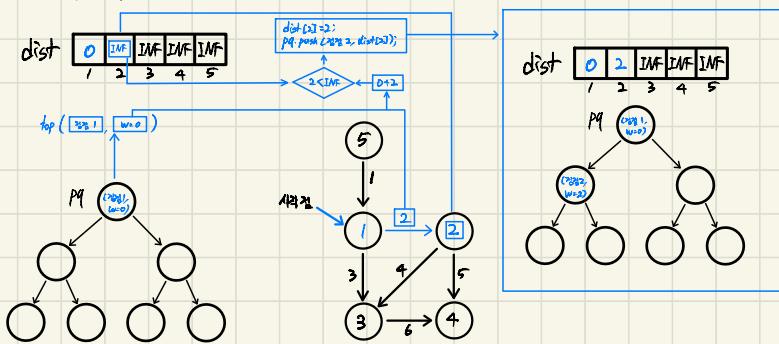
Step 2. 시작점 1의 dist를 0으로 초기화

PQ에 간선 정보 (경유지, w=0) push



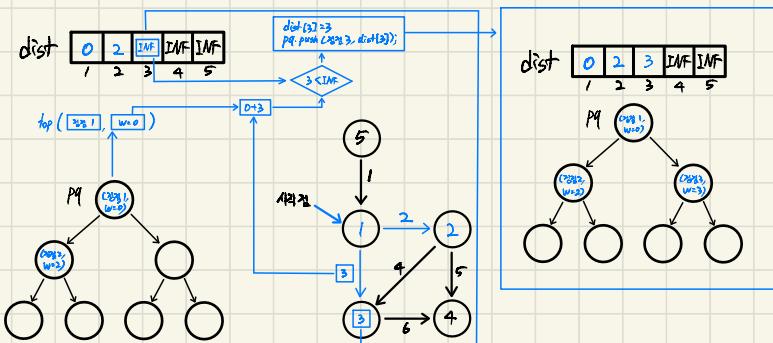
Step 3-1. PQ의 top ([경유지], [w=0])의 경유지에서 조속히는 경유지 2를 끌어

미래와 같이 dist를 갱신한다



Step 3-2. PQ의 top ([경유지], [w=0])의 경유지에서 조속히는 경유지 3을 끌어

미래와 같이 dist를 갱신한다



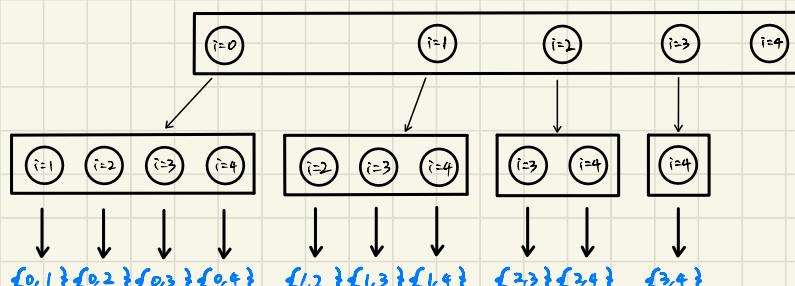
Step 4. PQ가 모두 비어질 때까지 step3 반복

Result: dist [0 2 3 7 INF]

백준 15686, 치킨 배달

키워드 : 조합 구하기, DFS

모든 조합 구하기
DFS ($s=0, L=0$), $nCr (n=5, r=2)$



$L=0$
 $arr[L]=i$
 $s=i+1$
 $L=L+1$
 $L=1$
 $arr[L]=i$
 $s=i+1$
 $L=L+1$
 $L=2$
 $if(L==r)$
return;

step 1 변수 초기화

h : 집의 위치를 담은 vector

C : 치킨집의 위치를 담은 vector

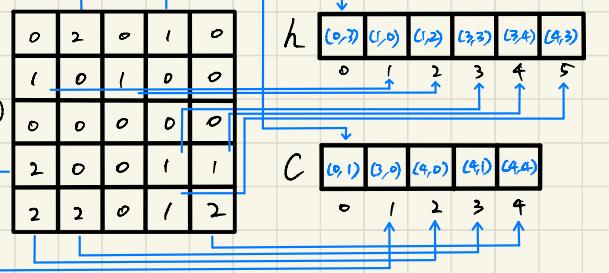
ARR : C 의 모든 조합의 원소로 배열

(지금은 경비를 위해 예상 형태로 바꾸었지만,
실제는 다른 형태로 DFS에서 원형으로 계산)

DATA: 5시작 치킨집의 좌표값 (총 4개)

입력
(0:본관, 1:점, 2:치킨점)
(0:본관, 1:점, 2:치킨점)

ARR [0,1] [0,2] [0,3] [0,4] [1,2] [1,3] [1,4] [2,3] [2,4] [3,4]



step 2 ARR 전체를 돌면서 최소값을 찾는다.

pseudo code

```

int out = INT_MAX; // 출력값 초기화
for(int i=0; i<arr.size(); ) // 전체 arr 순회
{
    int sum=0; // 현재 조합의 거리
    // 경비 4
    for(int j=0; j<h.size(); )
    {
        int min_dist = INT_MAX; // 각 집마다 최초거리 값을 변수
        // 예상지역의 양수 카운트 4
        for(int k=0; k<ARR[i].size(); )
        {
            pair<int, int> h-point = h[j]; // 집 위치
            pair<int, int> c-point = C[ARR[i][k]]; // 치킨점 위치

            int dist = abs(h-point.first - c-point.first) +
                      abs(h-point.second - c-point.second); // 거리 계산
            min_dist = min(min_dist, dist); // 최초거리 갱신
        }
        sum += min_dist;
        j++;
    }
    out = min(out, sum); // 출력 값 갱신
}
    
```

ARR [0,1] [0,2] [0,3] [0,4] [1,2] [1,3] [1,4] [2,3] [2,4] [3,4]

h
(0,1) (1,0) (1,2) (2,3) (3,4) (4,3)
0 1 2 3 4 5

C
(0,1) (3,0) (4,0) (4,1) (4,4)
0 1 2 3 4

백준 2178, 미로 탐색

키워드 : BFS

Step1 맵과 초기화

ATM : 현재 칸과 상대를 같은 배열(이동할 가능한 칸) (:이동 가능한 칸)

dist : 시작 위치에서 출발지까지 거리로 초기화하는 배열(:이동 경로)

q : 찾을 경로 좌표를 담은 Queue

ATM	1	0	1	1	1	1	1
	1	0	1	0	1	0	
	1	0	1	0	1	1	
	1	1	1	0	1	1	

dist

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

9							
---	--	--	--	--	--	--	--

Step 2 시작 점 (1,1)을 큐에 push 한다
출발점에서 목적점으로 이동한다

ATM	1	0	1	1	1	1
	1	0	1	0	1	0
	1	0	1	0	1	1
	1	1	1	0	1	1

dist

1	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

9	(1,1)						
---	-------	--	--	--	--	--	--

Step 3 큐에서 꺼낸 다음 그 경우 찾는 경위 좌표를 큐에 push 한다.
여기서 dist[4][4] = dist[4][front-1] + 1로 업데이트
다음은 front를 pop한다.

ATM	0	0	1	1	1	1
	1	0	1	0	1	0
	1	0	1	0	1	1
	1	1	1	0	1	1

dist

1	0	0	0	0	0	0
2	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

9	(1,1)	(1,2)					
---	-------	-------	--	--	--	--	--

Step 4 큐가 모두 비어있을 때까지 Step 3를 계속 한다

ATM	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0
	0	0	0	0	0	0

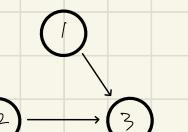
dist

1	0	9	10	11	12
2	0	8	0	12	0
3	0	7	0	13	14
4	5	6	0	14	15

9							
---	--	--	--	--	--	--	--

백준 2252, 줄 세우기

키워드 : 위상 정렬



Step1 degree 값이 0인 정점을 q에 push



Step2 q의 front 정점(1)에서 친밀하게 한 정점(3)의
degree를 1만큼 빼고, q.front.pop 한다



Step3 q의 front 정점(2)에서 친밀하게 한 정점(3)의
degree를 1만큼 빼고, q.front.pop 한다
이때 정점(3)의 degree가 0이므로 push 한다



Step4 q의 front 정점(3)에서 친밀하게 한 정점이
없으므로 q.front.pop 한다



Step 5 q의 size가 0이면 종료

백준 11675, 탐색 알고리즘

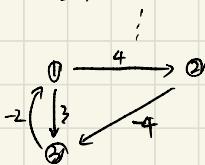
가우드

벨만-포드 알고리즘

정렬

3(정점 S) 4(전선 A)

1(정점 A) 2(정점 B) 4(가중치)



step 1 dist 배열 (1부터 각 정점 까지 걸리는 시간)을 무한 대로 초기화

INF	INF	INF
1	2	3

step 2 dist[1(정점 A), 1]은 0으로 초기화

0	INF	INF
1	2	3

step 3 간선 수 - 1 × 정점 수 만큼 거리를 초기화 가중치를 더한 값을 차운 것으로 계속 업데이트
문, dist가 INF는 몇 수 없으므로 계산이 끝나기

0	INF	INF
1	2	3

 \Rightarrow

0	4	INF
1	2	3

 \Rightarrow

0	4	3
1	2	3

step 4 마지막으로 정점 수 만큼 업데이트 하는데 기존값보다 같아져면 사이클이 있는지

0	4	3
1	2	3

 \Rightarrow

0	0	3
1	2	3

같이 출력되므로 사이클이 있는지