Dijkstra Algorithm

다익스트라 알고리즘 발표

Please Confirm Video and Speaker

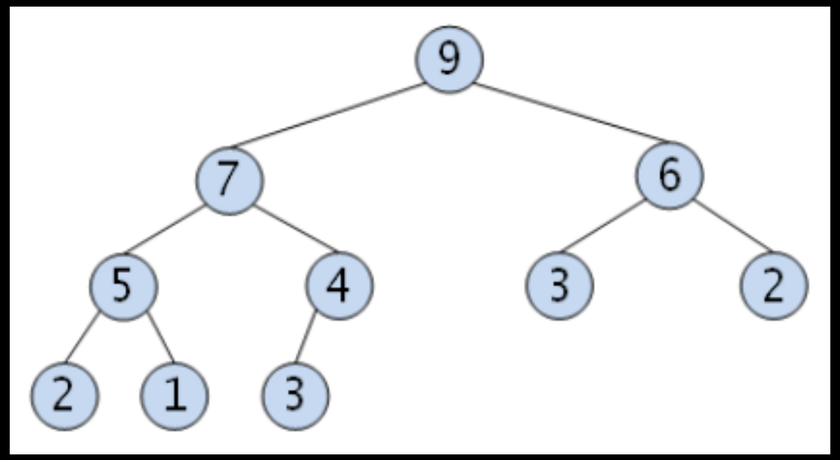
Dijkstra Algorithm

- **최단 경로 알고리즘** (가장 짧은 경로를 찾는 알고리즘)
- **가중치가 음수로 처리할 수 없음** (가중치의 합이 최소가 되는 방식) *Bellman-ford*
- 네비게이션 관련 문제나 미로 탐색 , *OSPF*에 활용되는 알고리즘

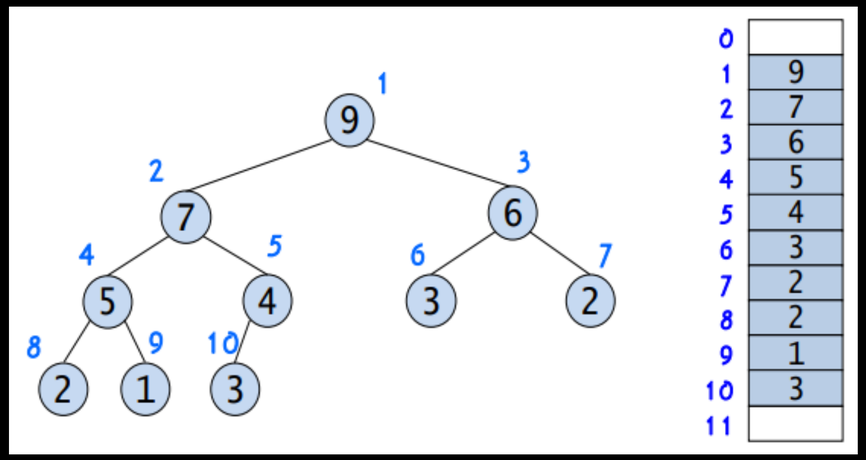
#Heap #Priority Queue #Graph #Greedy

Heap

- **최대값 또는 최소값을 빨리 찾아야할** 때 사용하며 이진 트리로 이루어진 자료구조
- O(logN)의 시간 복잡도



Max Heap



Priority Queue

Priority Queue

- **어떠한 특정에 조건**에 따라 우선순위가 높은 요소를 추출하는 자료구조

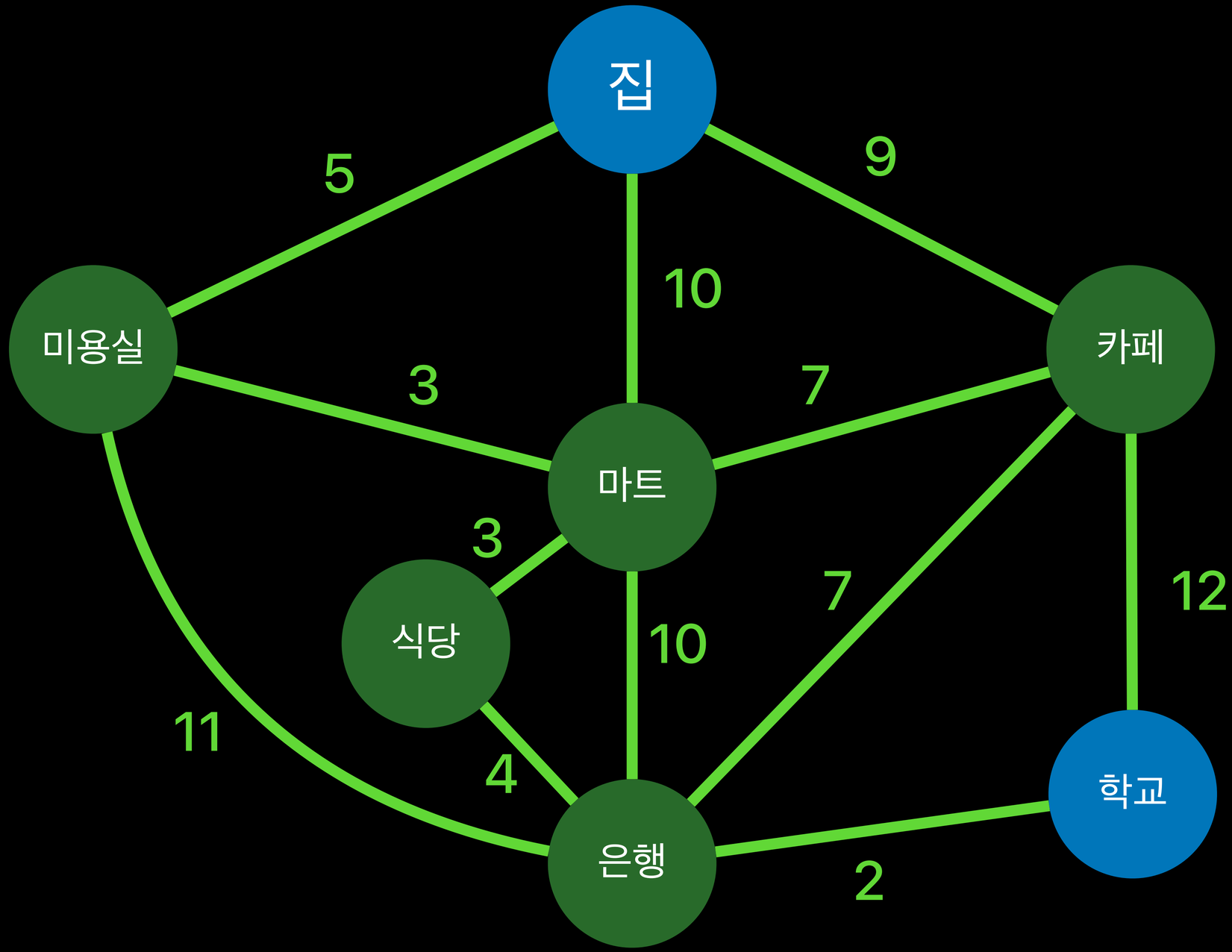
순서

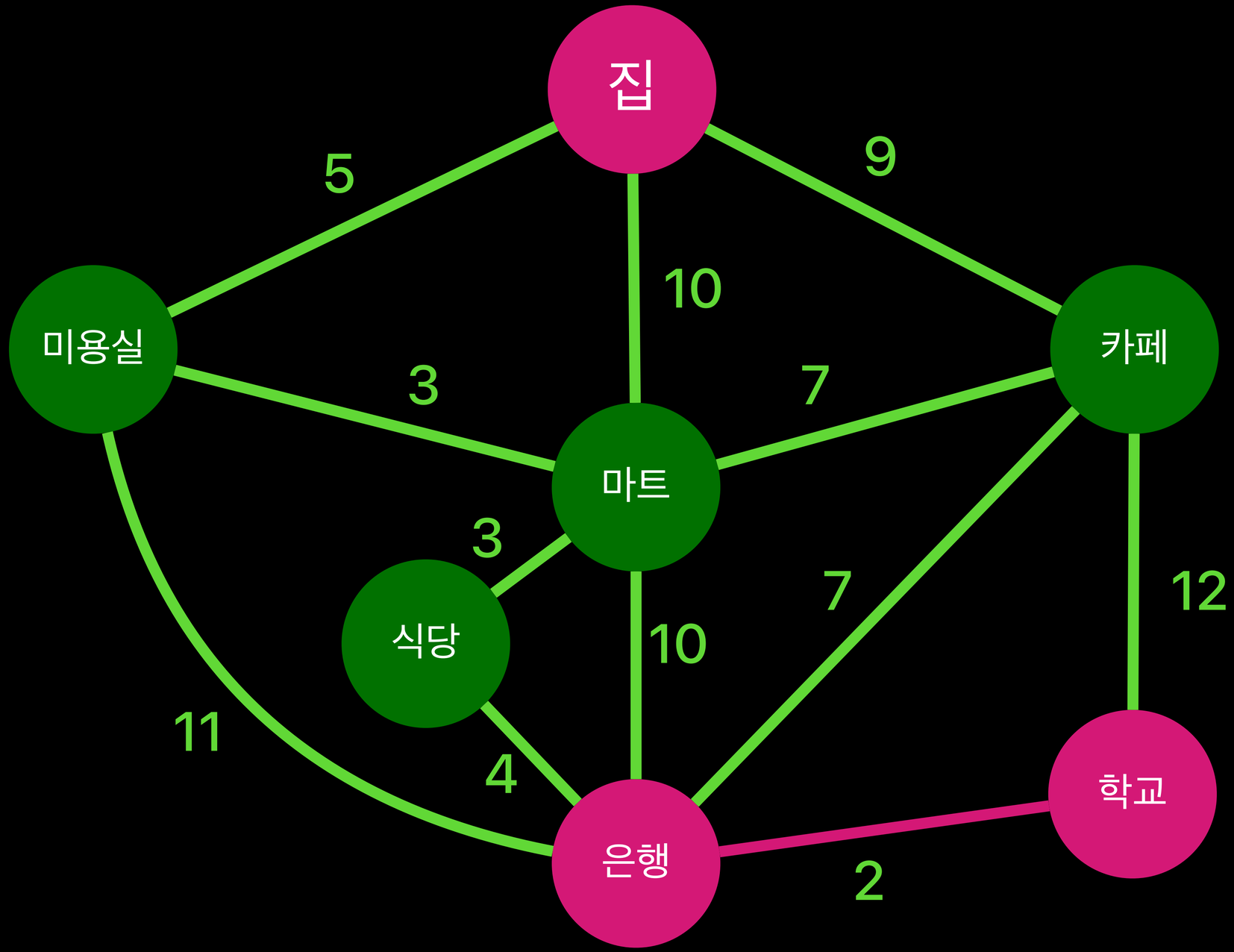
1. 출발 노드를 설정합니다.
2. 최단 거리를 저장하는 테이블을 초기화 합니다.
3. 방문하지 않은 노드**(인접 노드) 중 가장 짧은 노드를 선택**합니다.
4. 해당 노드를 거쳐 다른 노드로 가는 비용을 계산하여 테이블을 갱신합니다.

준비

- 인접리스트를 담을 수 있는 가중치 그래프
- 우선 순위 큐
- 최단 거리를 저장하는 배열

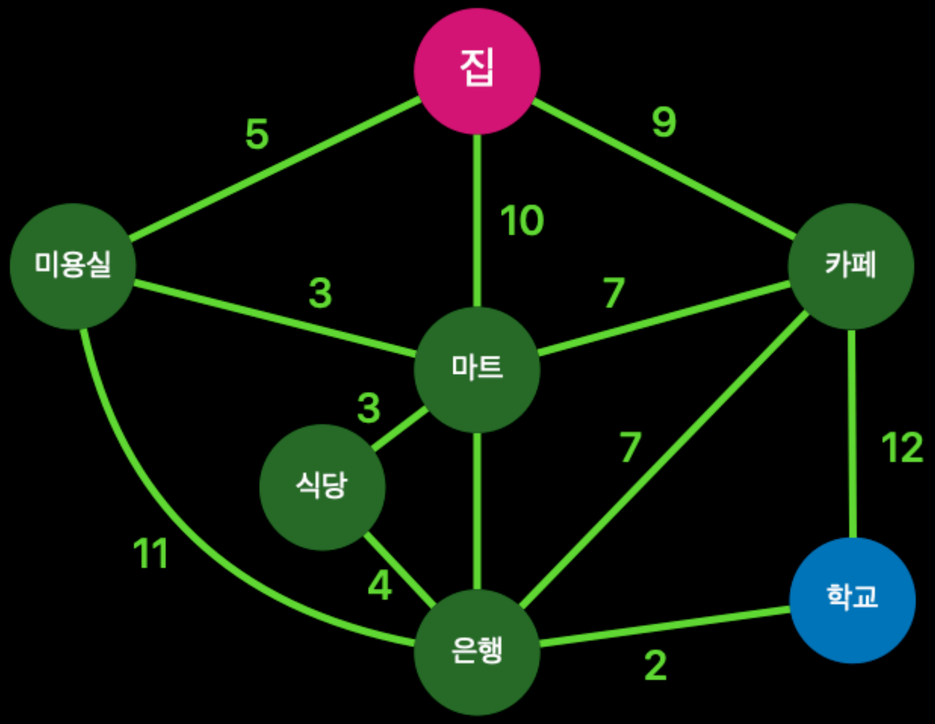

신천사회주택공영주차장
30
어린이보호구역
어린이 보호구역
30
시흥타임즈


집
미용실
카페
마트
식당
은행
학교
5
9
10
3
7
3
7
10
12
11
4
2


집
미용실
카페
마트
식당
은행
학교
5
9
10
3
7
3
7
12
10
11
4
2

1. 시작 노드 선정 & 테이블 초기화

- 출발노드를 지정하고 이에 관한 거리들을 대입합니다.



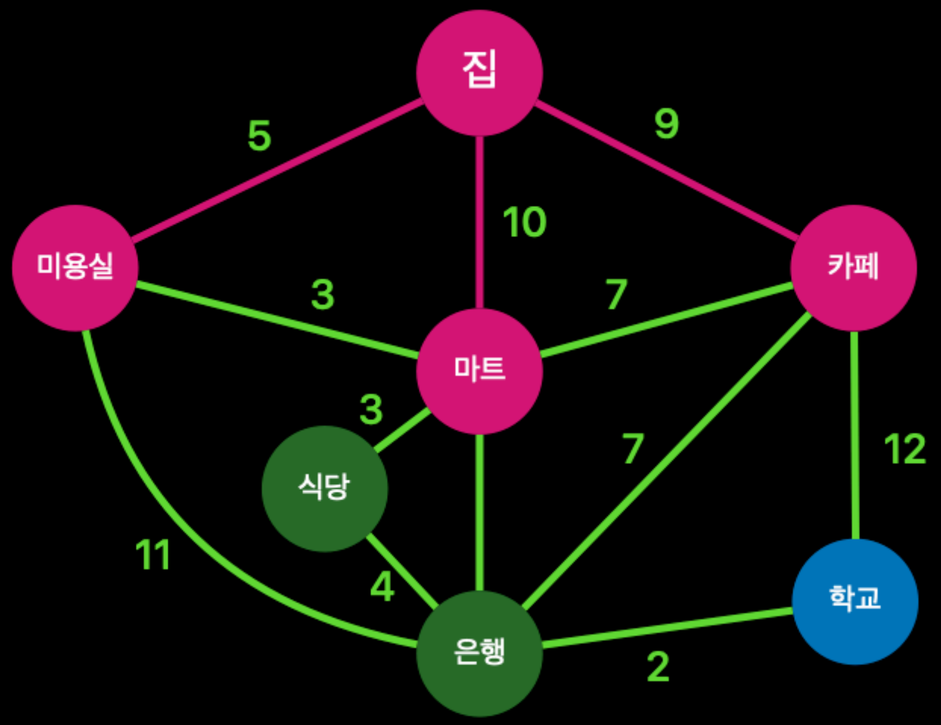
Q	(0, 집)						

건물	집	미용실	마트	카페	식당	은행	학교
거리	*INF*	*INF*	*INF*	*INF*	*INF*	*INF*	*INF*

- PriorityQueue
- heapq

2. 시작 노드를 거리 배열에 갱신

- 출발노드를 지정하고 이에 관한 거리들을 대입합니다.



(0,집)

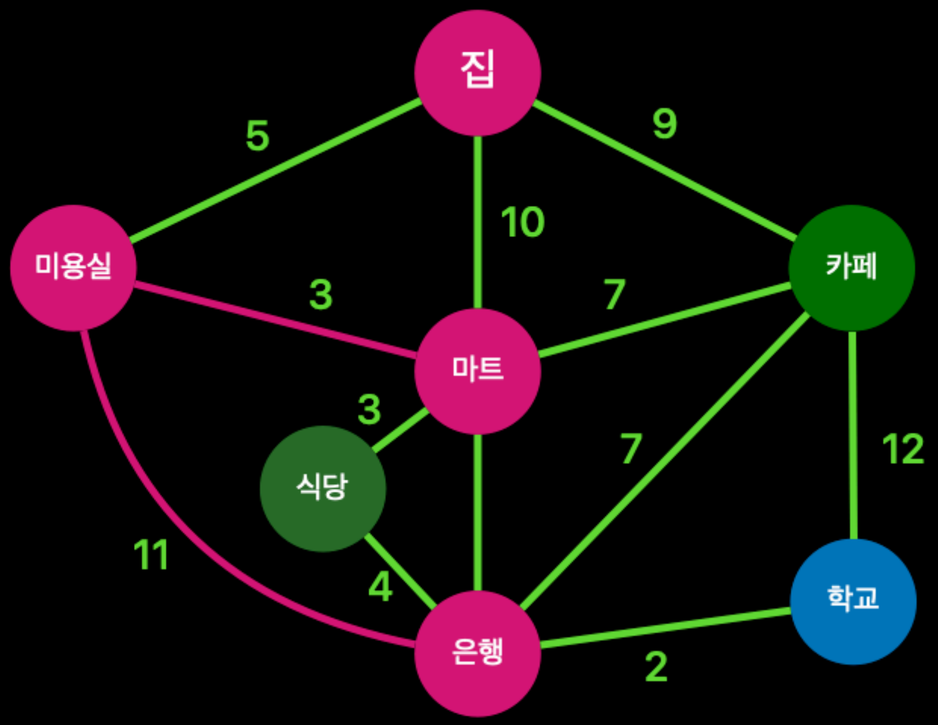
(5,미)	(9,카)	(10,마)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	10	9	*INF*	*INF*	*INF*

Q	(5,미)	(9,카)	(10,마)				

- PriorityQueue
- heapq

2. 우선순위 큐가 빌 때까지 반복



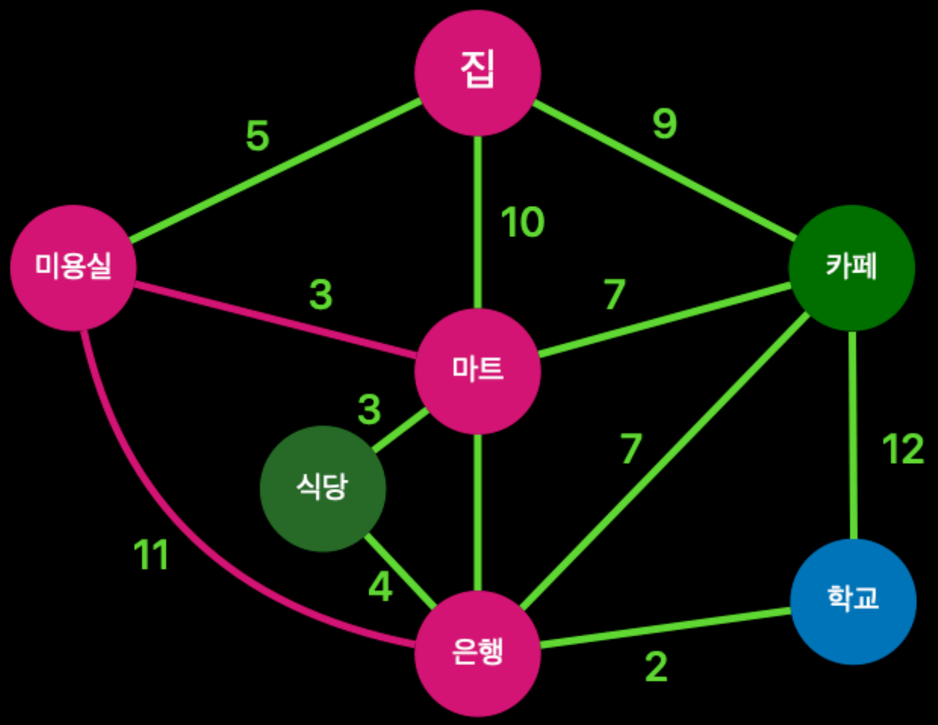
(5,미용실)	(3,마)	(11,은)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	10	9	*INF*	*INF*	*INF*

VS
5 + 3

VS
5 + 11

2. 우선순위 큐가 빌 때까지 반복

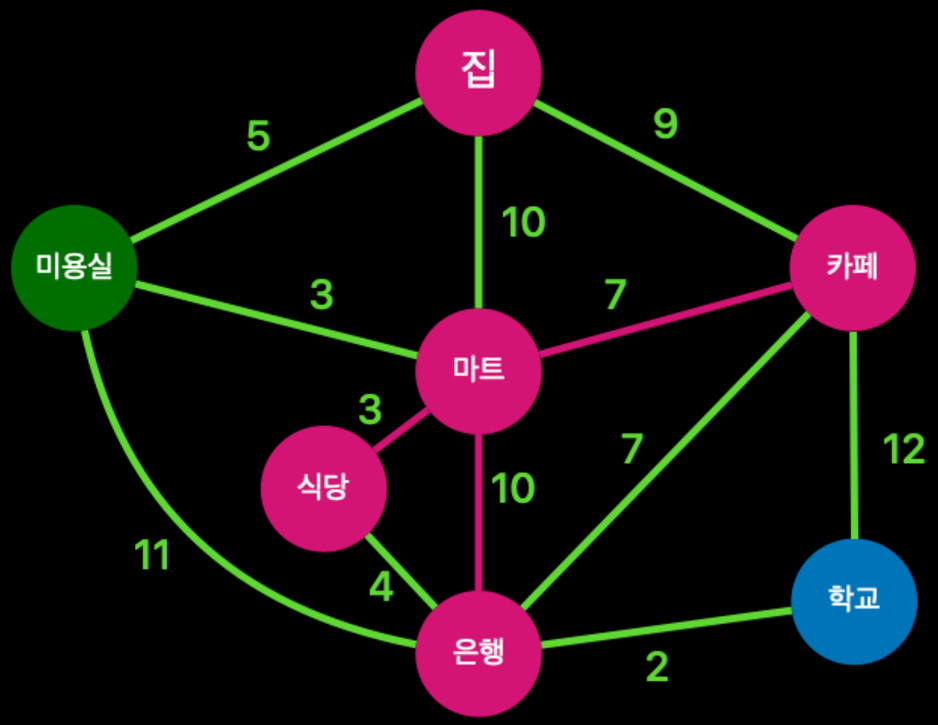


(5,미용실)	(3,마)	(11,은)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	*INF*	16	*INF*

Q	(8,마)	(9,카)	(10,마)	(16,은)			

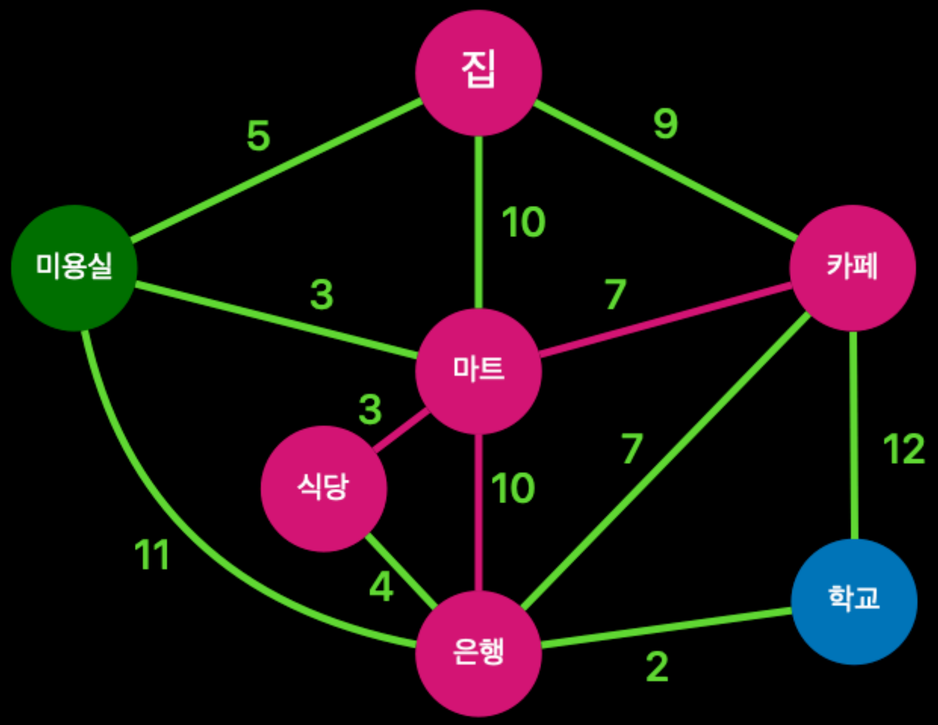
2. 우선순위 큐가 빌 때까지 반복



(8,마트)	(3,식)	(7,카)	(10,은)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	*INF*	16	*INF*
				VS 8+7	VS 8+3	VS 8+10	

2. 우선순위 큐가 빌 때까지 반복

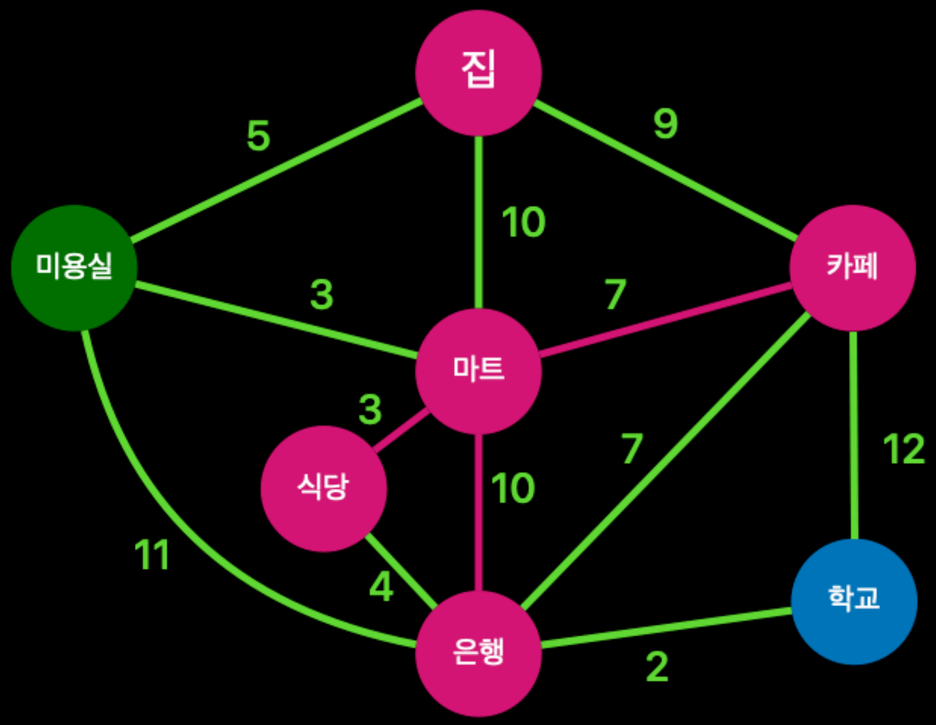


(8,마트)	(3,식)	(7,카)	(10,은)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	16	*INF*
				VS 8+7		VS 8+10	

Q	(9,카)	(11,식)	(10,마)	(16,은)			

2. 우선순위 큐가 빌 때까지 반복

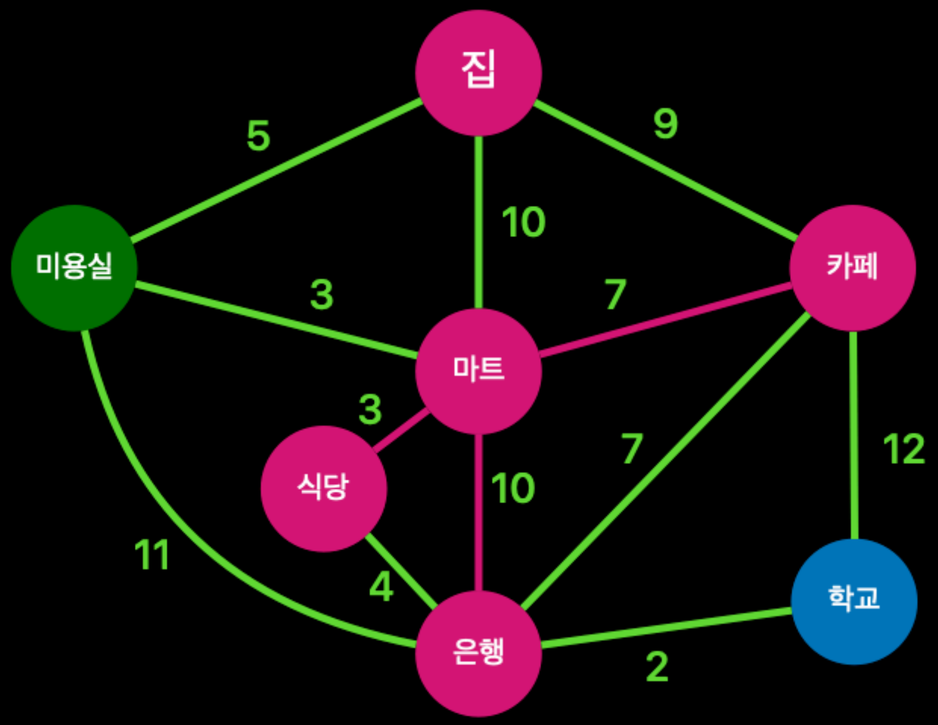


(9,카페)	(7,은행)	(12,학교)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	16	21
				VS 8+7		VS 9+7	VS 9+12

Q	(10,마)	(11,식)	(16,은)	(21,학)			

2. 우선순위 큐가 빌 때까지 반복

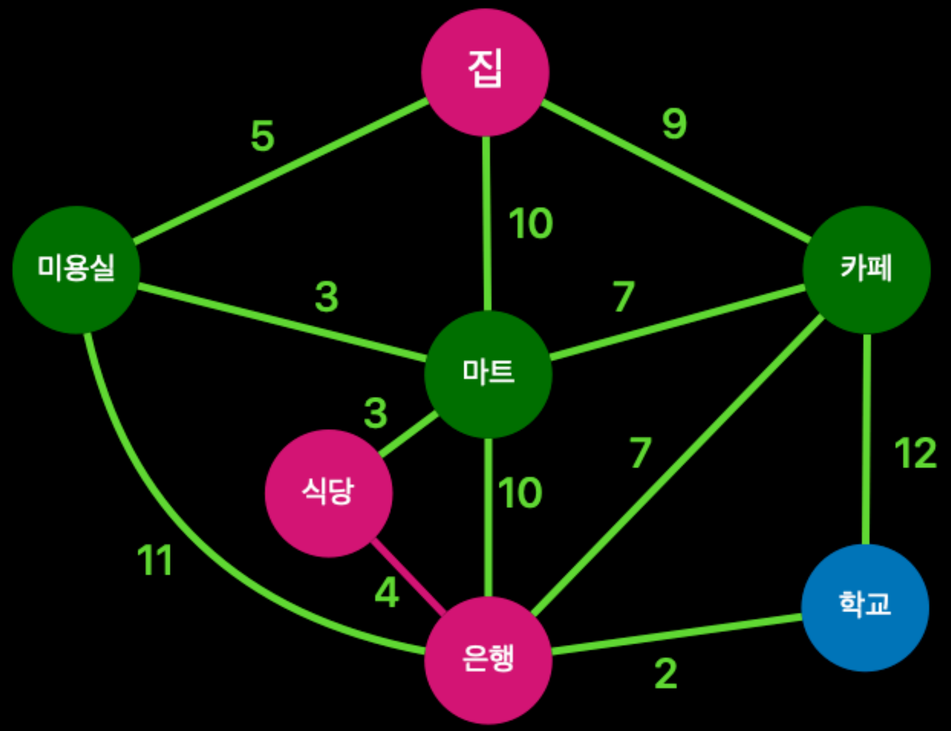


(10,마트)	(3,식)	(7,카)	(10,은)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	16	21
				VS 10+7	VS 10+3	VS 10+10	

Q	(11,식)	(16,은)	(21,학)				

2. 우선순위 큐가 빌 때까지 반복



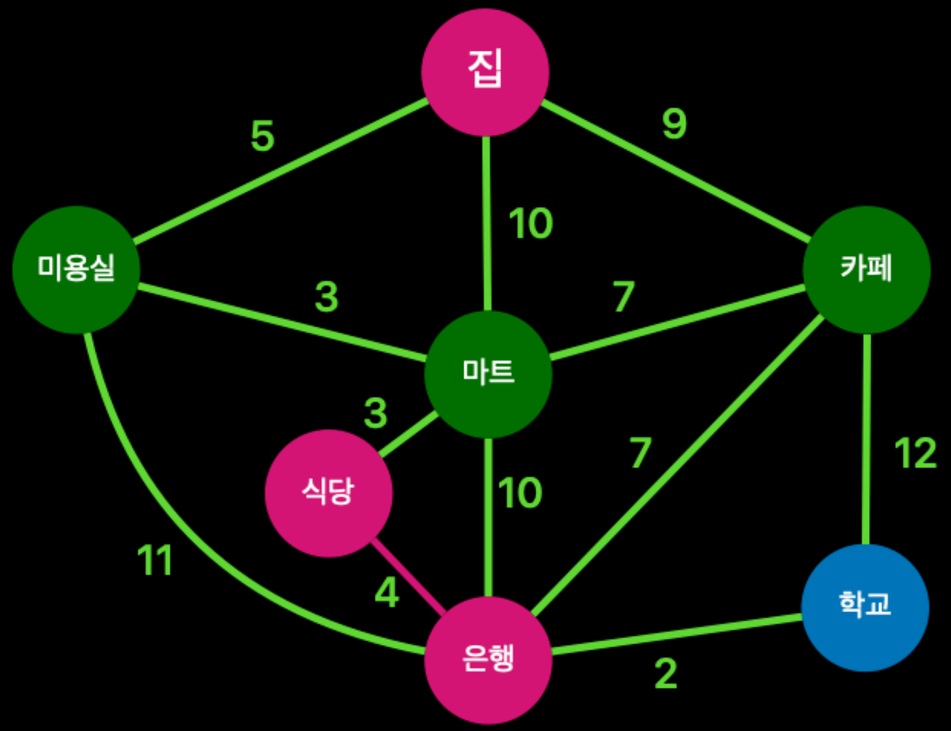
(3,식당)

(4,은행)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	16	21

VS
11+4

2. 우선순위 큐가 빌 때까지 반복

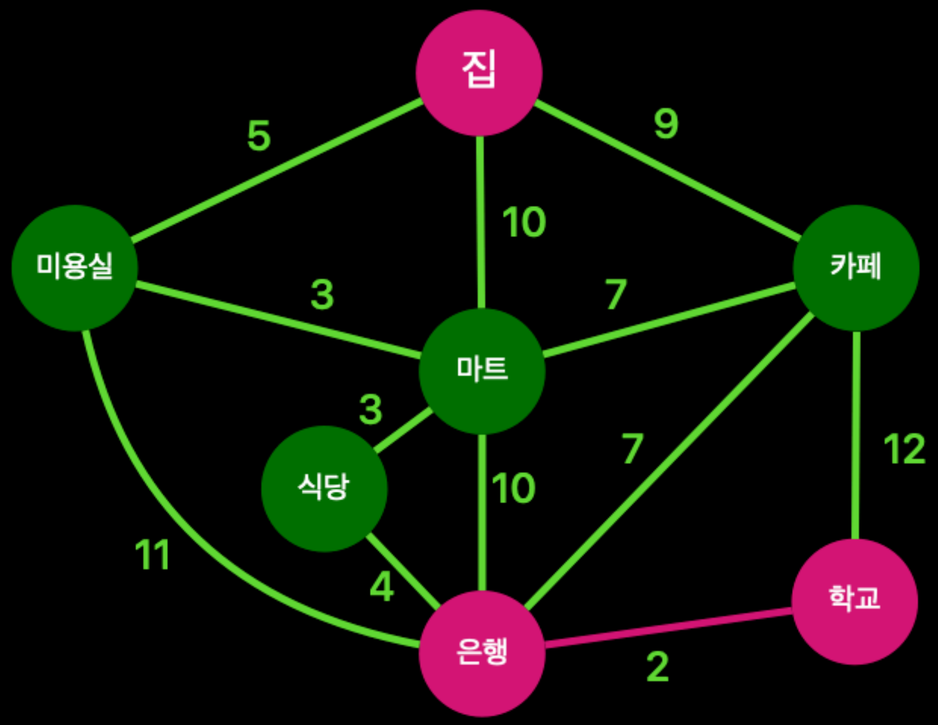


(3,식당)	(4,은행)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	21

Q	(15,은)	(16,은)	(21,학)				

2. 우선순위 큐가 빌 때까지 반복



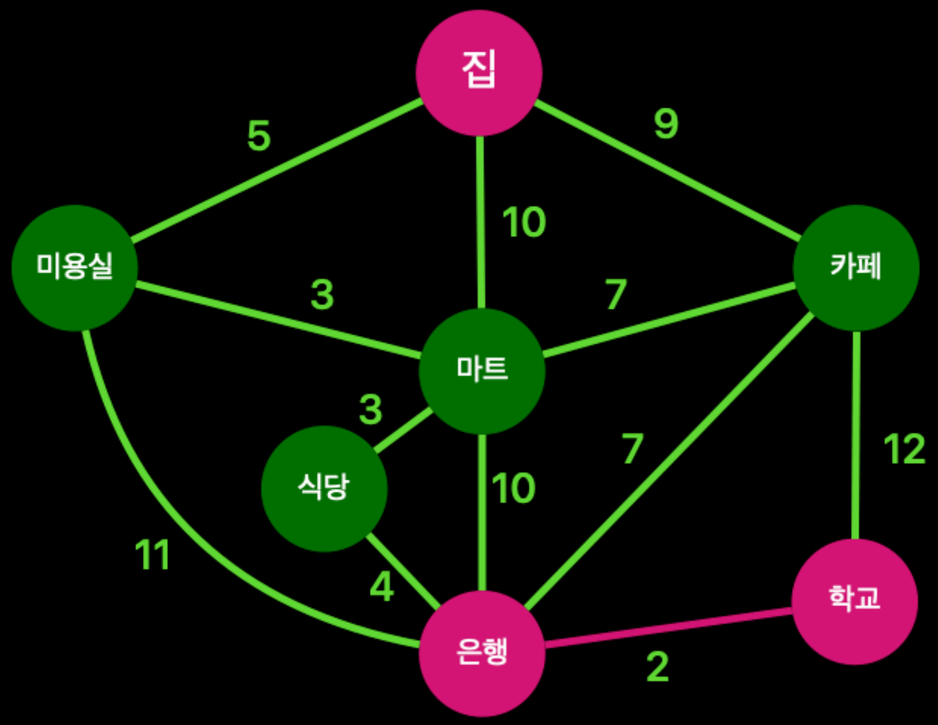
(15,은행)

(2,학교)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	21

VS
15+2

2. 우선순위 큐가 빌 때까지 반복

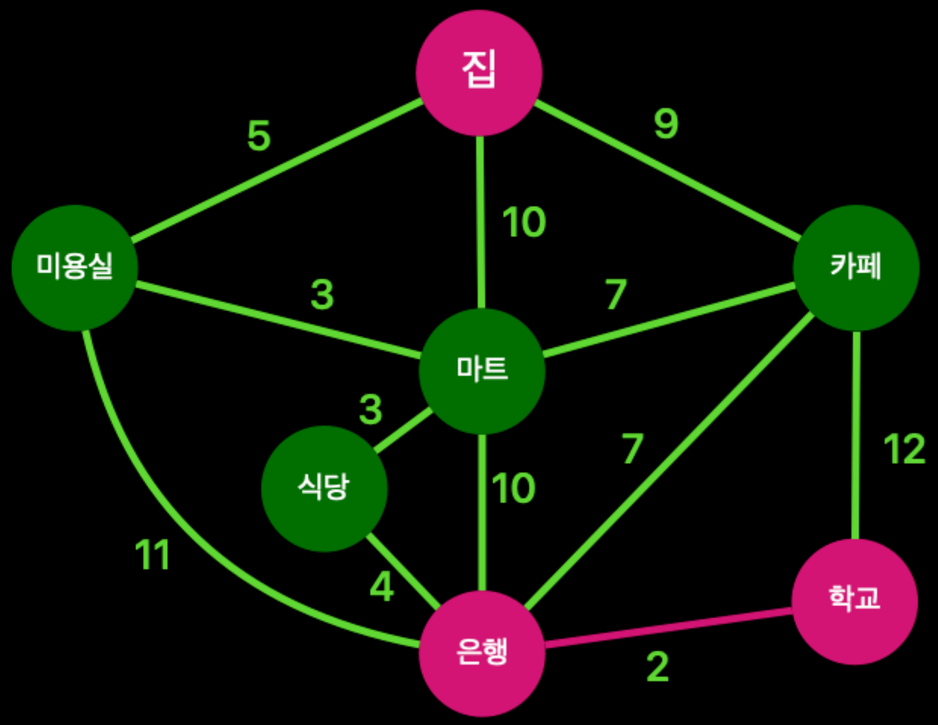


(16,은행)	(2,학교)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17

Q	(16,은)	(17,학)	(21,학)				

2. 우선순위 큐가 빌 때까지 반복



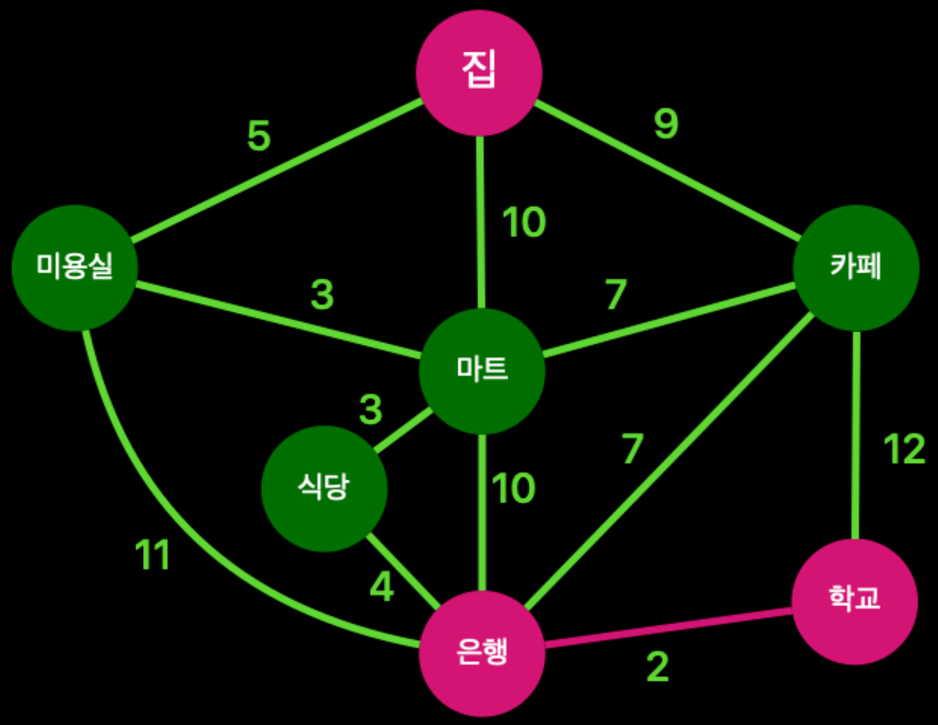
(15,은행) (2,힉교)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17

VS
16+2

Q	(17,학)	(21,학)					

2. 우선순위 큐가 빌 때까지 반복



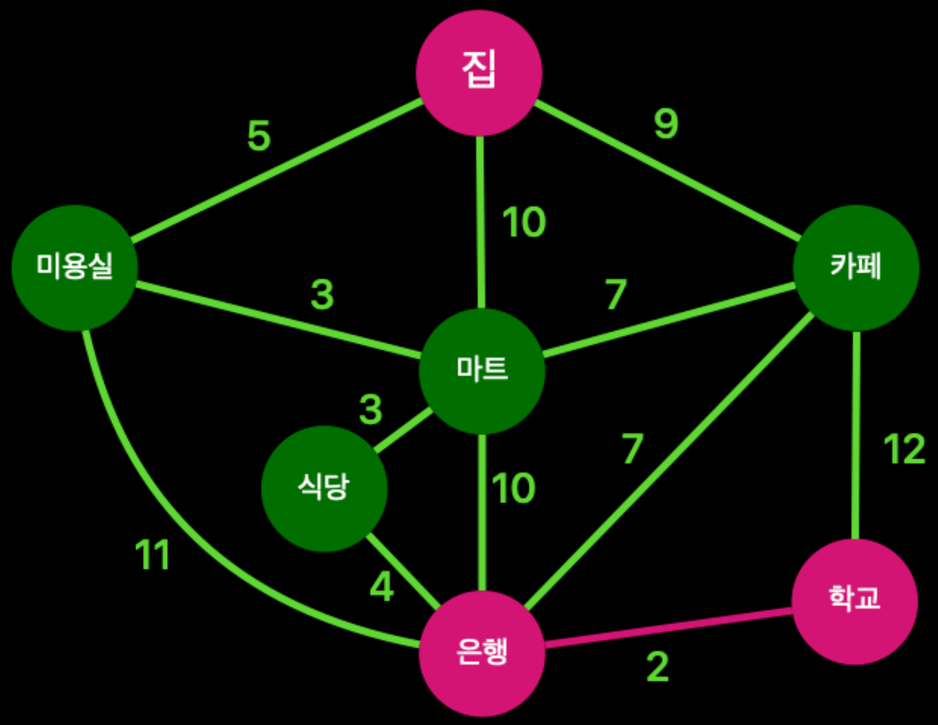
(17,학교)	(2,은행)	(12,카페)

건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17

VS 17+12　　VS 17+2

Q	(21,학)						

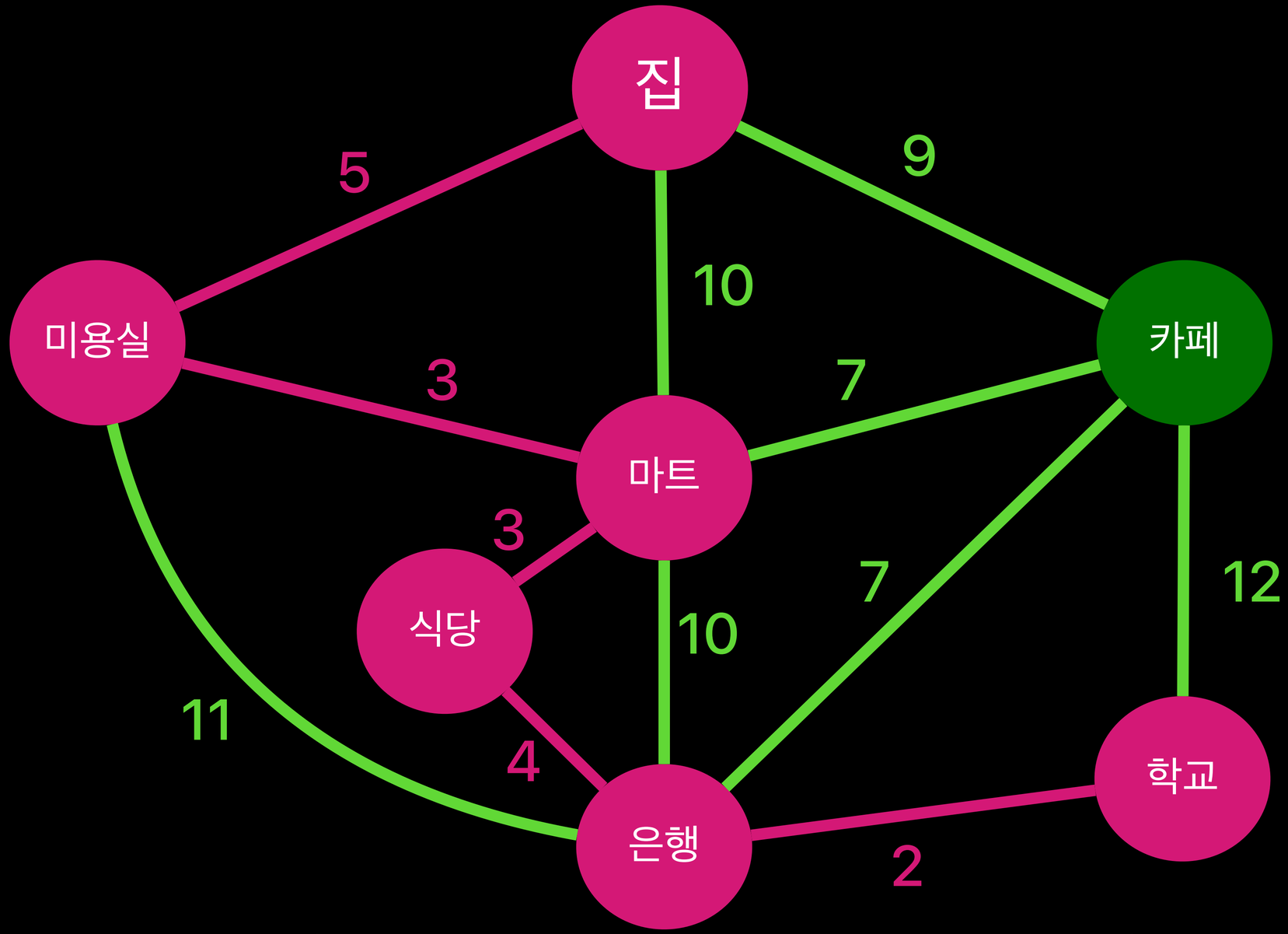
2. 우선순위 큐가 빌 때까지 반복



(21,학교)	(2,은행)	(12,카페)

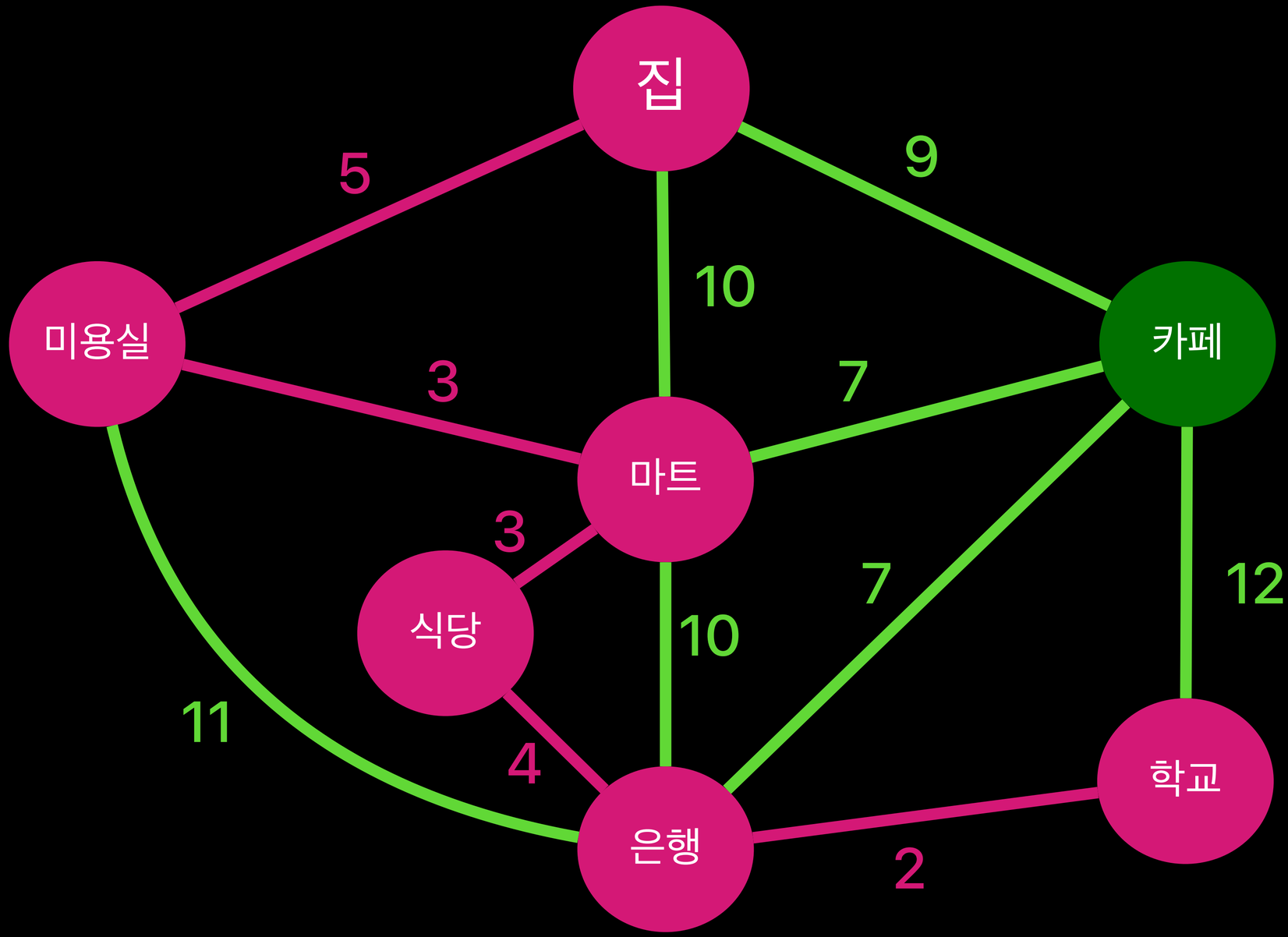
건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17
				VS 21+12		VS 21+2	

Q							



건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17

만약, 여기서
친구들을 만나기로 했는데
집부터 거리가 10 이내에서 만나야 한다면?



건물	집	미용실	마트	카페	식당	은행	학교
거리	0	5	8	9	11	15	17

만약, 여기서
친구들을 만나기로 했는데
집부터 거리가 10 이내에서 만나야 한다면?

만약, 여기서
배달을 하기로 했는데
해당 마을부터 거리가 K 이내에서 받을 수 있다면?

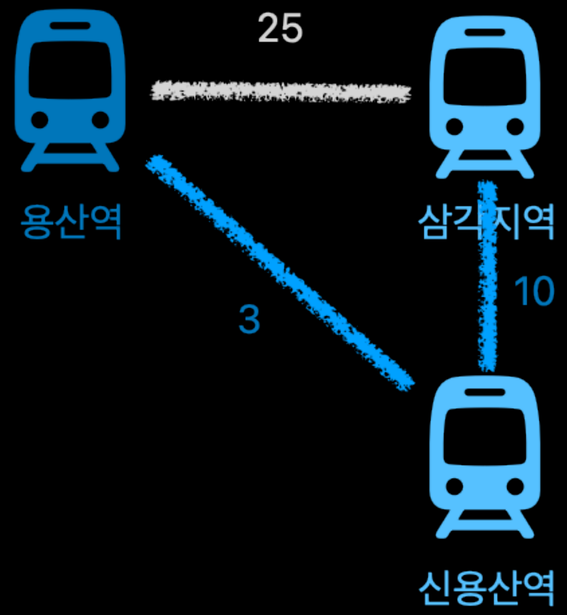
시간 복잡도

- 각 노드마다 방문하면서 **인접한 간선들을 모두 검사**합니다.
- 간선을 검사할 때마다 **노드 및 거리가 추가되는 과정**이 있습니다. (Heap)
  - 이때 E는 V^2보다 항상 작으므로 아무리 늦어도 logV!

$$O(E) + O(E \log V) = O(E \log V)$$

활용

- **특정 지점까지 가는 최단 거리 찾기**
  - 간선 방향을 뒤집고 다익스트라 알고리즘을 사용합니다.
- **플로이드 워셜 알고리즘**
  - 단일 노드가 아닌 모든 노드에 대한 최단 거리를 구하는 알고리즘
  - 3중 반복문을 사용
  - $D_{ab} = \min(D_{ab}, D_{ak} + D_{kb})$
  - 다이나믹 프로그래밍 활용
- **A* 알고리즘**
  - 평가함수를 통해 추정 거리를 계산하는 알고리즘
  - 평가함수 = $D_{\text{출발현재}} + D_{\text{현재도착}}$
  - DFS, BFS를 사용
- **크루스칼 알고리즘**
  - 최단 경로가 아니라 최소 비용을 찾는 알고리즘



관련 코드를 참고하세요.

- JS : https://velog.io/@jangws/22.-다익스트라Dijkstra-알고리즘
- Py : https://dustinlab.gitbook.io/python/algorithm/greedy/dijkstra
- Process Video : https://www.youtube.com/watch?v=XXzsUST5KSI
- Reference : https://www.youtube.com/watch?v=tZu4x5825LI