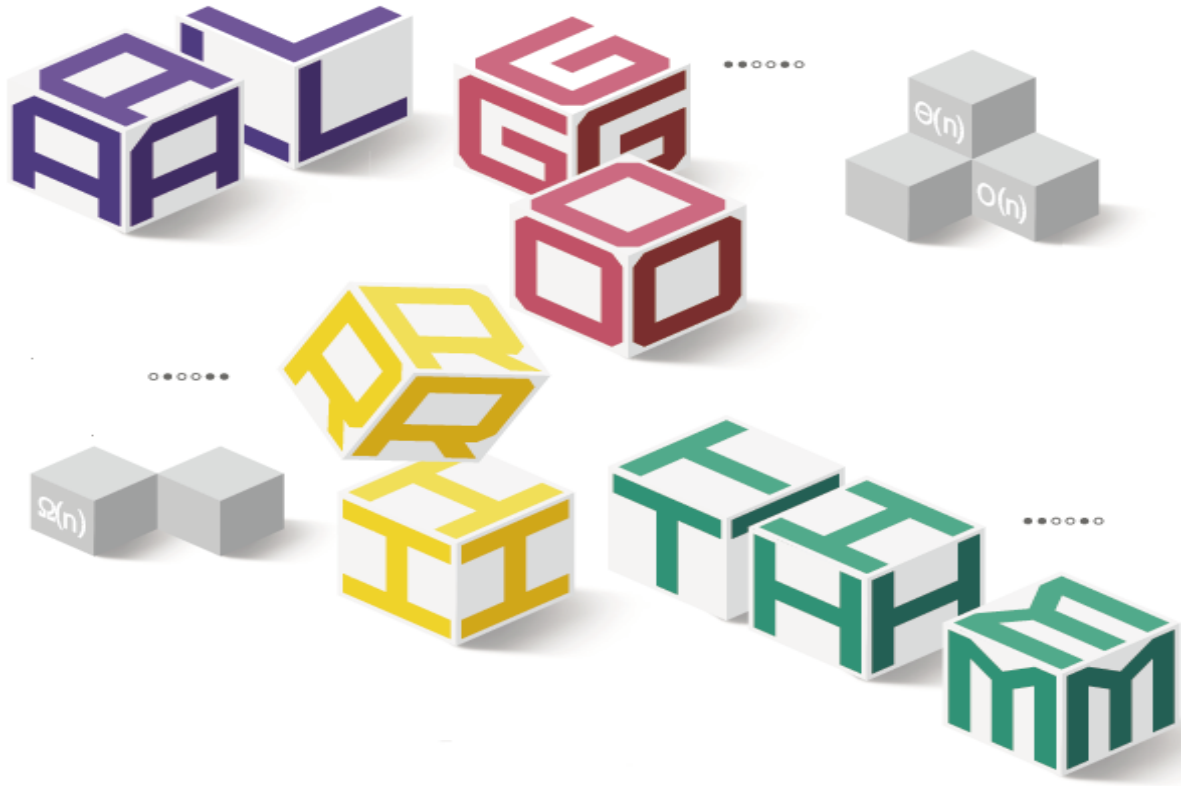


관계 중심의 사고법

쉽게 배우는 알고리즘

14주차: 그래프

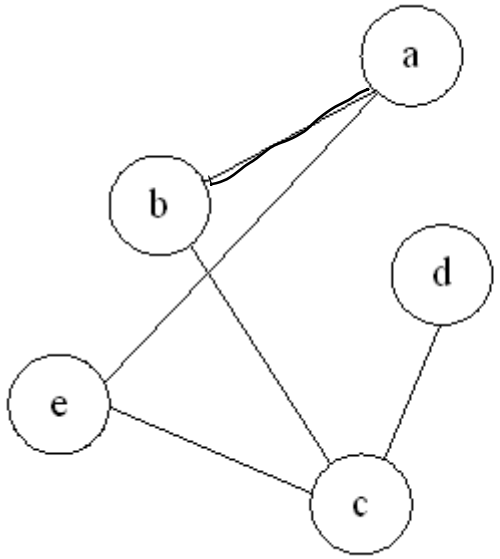

14주차
15주차
1. 그래프표현-인접리스트
4명
0명
9명
(0)
강의 보기
2. 그래프 방문-BFS
3명
0명
10명
(0)
강의 보기
3. 그래프 방문-DFS
3명
0명
10명
(0)
강의 보기
zoom 수업 연결
0명
1명
12명
(0)
강의 보기
4-최소신장트리-크루스칼
0명
0명
13명
(0)
강의 보기
1. 최소신장트리-Prim 알고리즘-1
0명
0명
13명
(0)
강의 보기
2-최소신장트리-Prim 알고리즘-2
0명
0명
13명
(0)
강의 보기
3-실습 Prim 알고리즘
0명
0명
13명
(0)
강의 보기
5. 최소신장트리 실습-2
0명
0명
13명

학습목표

- 그래프의 표현법을 익힌다
- **너비 우선 탐색과 깊이 우선 탐색의 원리를 충분히 이해하도록 한다**
- **신장 트리의 의미와 최소 신장 트리를 구하는 두 가지 알고리즘을 이해한다**

(지난시간)그래프 표현

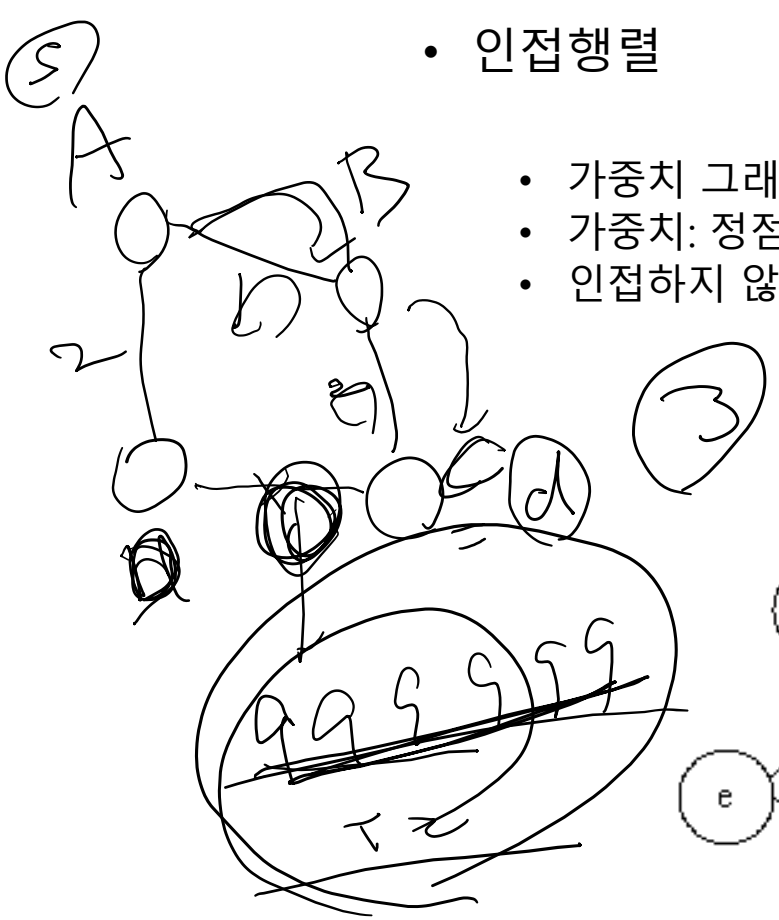
- 인접행렬(Adjacency Matrix)
  - 2차원 행렬 : 2차원 배열
  - 직접 연결된 간선이 있으면 해당 값을 1(True)
  - 대각선 요소는 무의미. 0 또는 1.

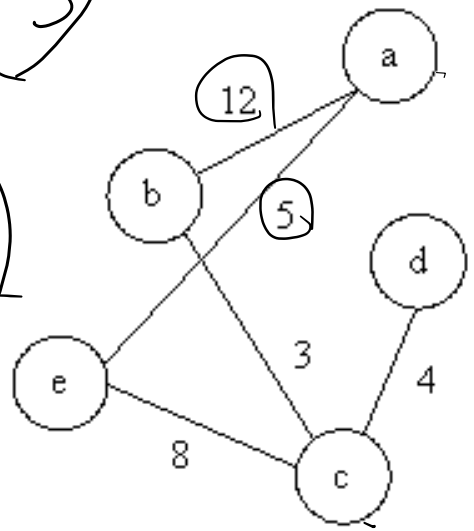


	a	b	c	d	e
a	0	1	0	0	1
b	1	0	1	0	0
c	0	1	0	1	1
d	0	0	1	0	0
e	1	0	1	0	0

(지난시간)그래프 표현

- 인접행렬
  - 가중치 그래프: 간선의 가중치
  - 가중치: 정점 사이를 이동하는데 필요한 비용이나 거리
  - 인접하지 않은 정점: 비용 무한대.





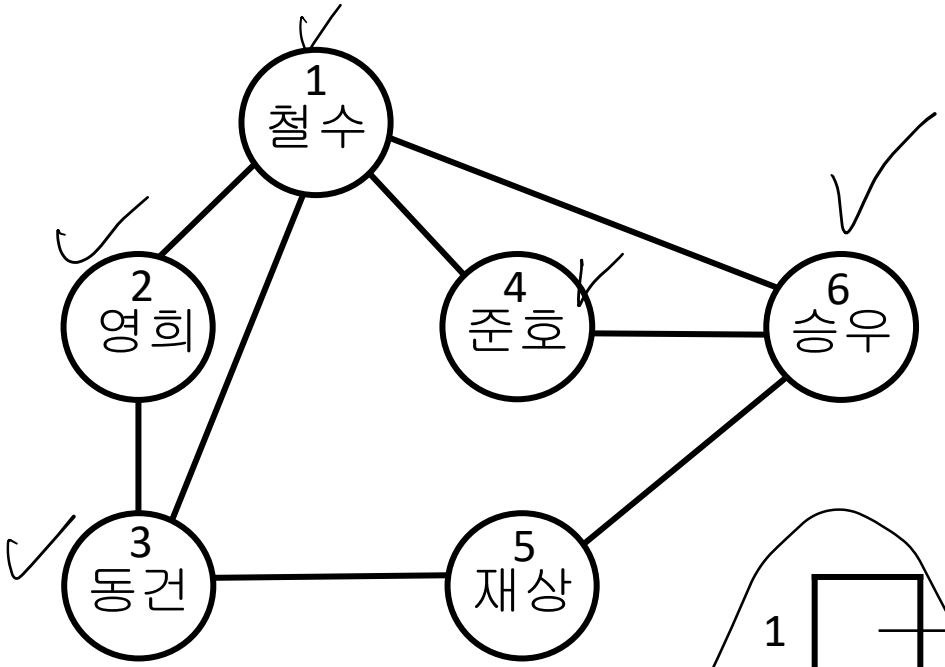
from \ to	a	b	c	d	e
a	0	12	inf	inf	5
b	12	0	3	inf	inf
c	inf	3	0	4	8
d	inf	inf	4	0	inf
e	5	inf	8	inf	0

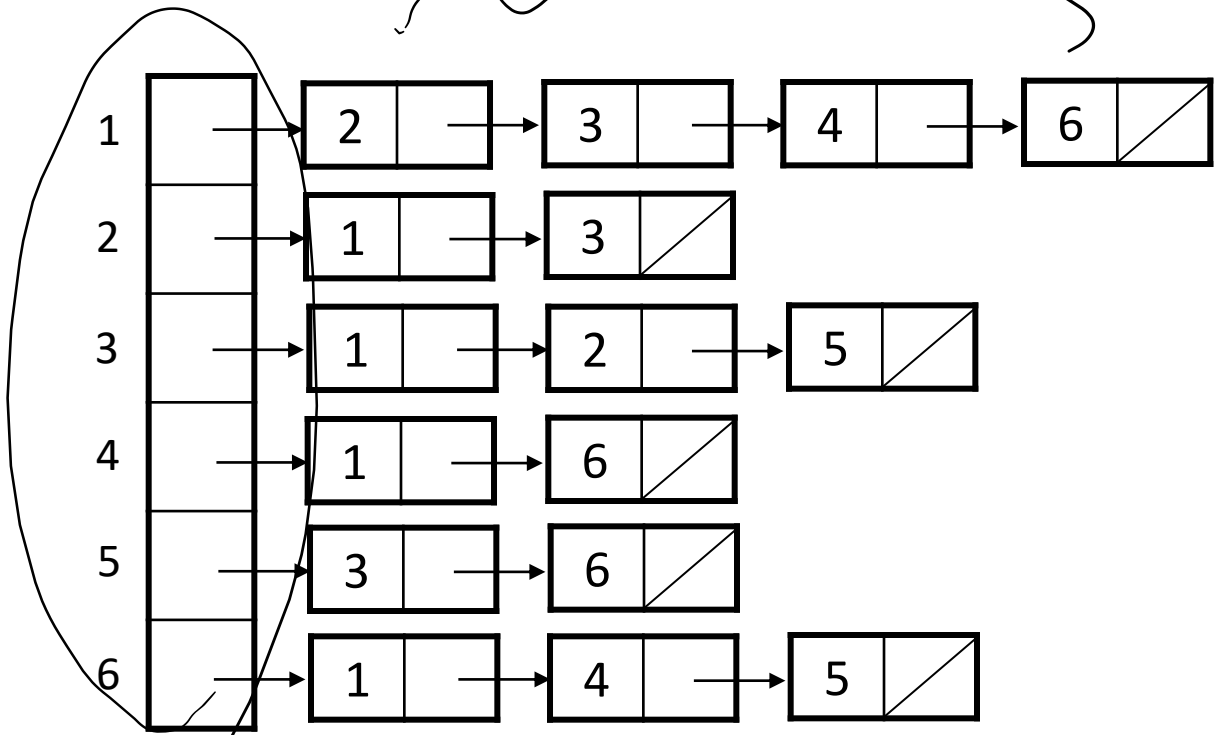
(지난시간)그래프 표현

- 배열를 사용한 인접행렬
  - 필요한 정점의 수를 예측. int AdjacencyMatrix[Max][Max]; 라고 선언
  - 고정 크기의 그래프
- 연결리스트를 사용한 인접리스트
  - 힙 메모리
  - 미리 배열 크기를 결정할 필요가 없음.

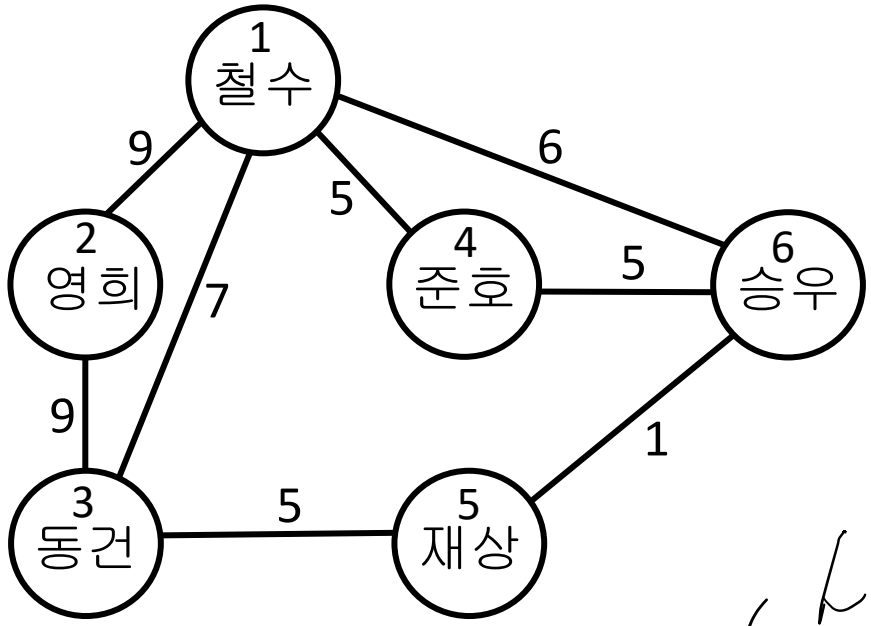
Graph의 표현 2: Adjacency List

- Adjacency list
  - N개의 연결 리스트로 표현
  - i번째 리스트는 정점 i에 인접한 정점들을 리스트로 연결해 놓음
  - 가중치 있는 그래프의 경우
    - 리스트는 가중치도 보관한다

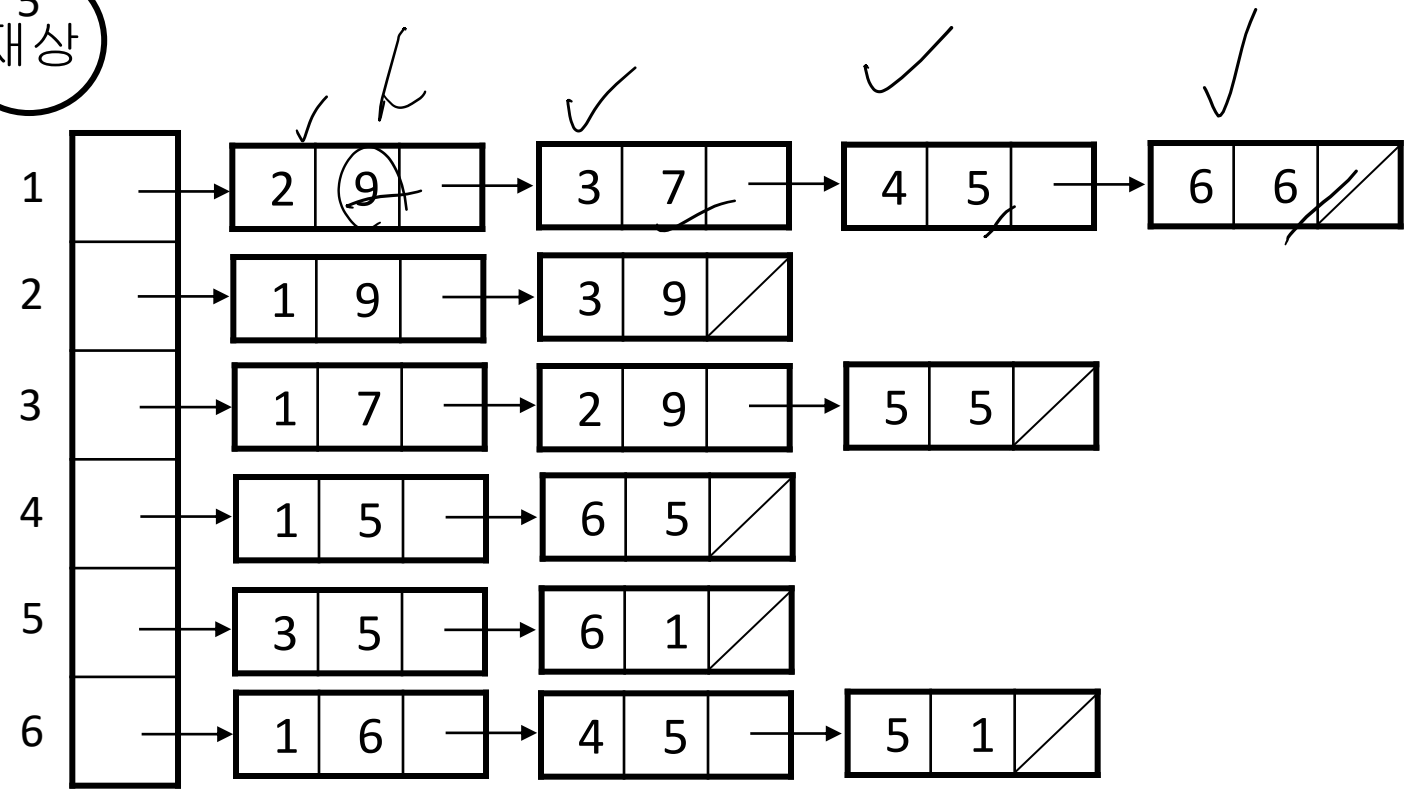




무향 그래프의 예


1
철수
2
영희
3
동건
4
준호
5
재상
6
승우
9
6
5
7
5
9
5
1

가중치 있는 그래프의 예


1
2 9
3 7
4 5
6 6
2
1 9
3 9
3
1 7
2 9
5 5
4
1 5
6 5
5
3 5
6 1
6
1 6
4 5
5 1

정리: 인접 행렬 vs 인접리스트

- 인접 리스트
  - 무방향 그래프: 하나의 간선에 대해 두 개의 노드가 나타남.
  - 인접 리스트의 노드 수는 간선 수 E의 2배

- 인접 리스트와 인접 행렬
  - 정점 i와 정점 j가 인접해 있는가: 인접행렬이 유리
  - 정점 i 에 인접한 모든 정점을 찾아라: 인접 리스트가 유리
  - 공간 면에서 인접행렬은 V^2 개의 공간, 인접 리스트는 2E 개의 공간이 필요

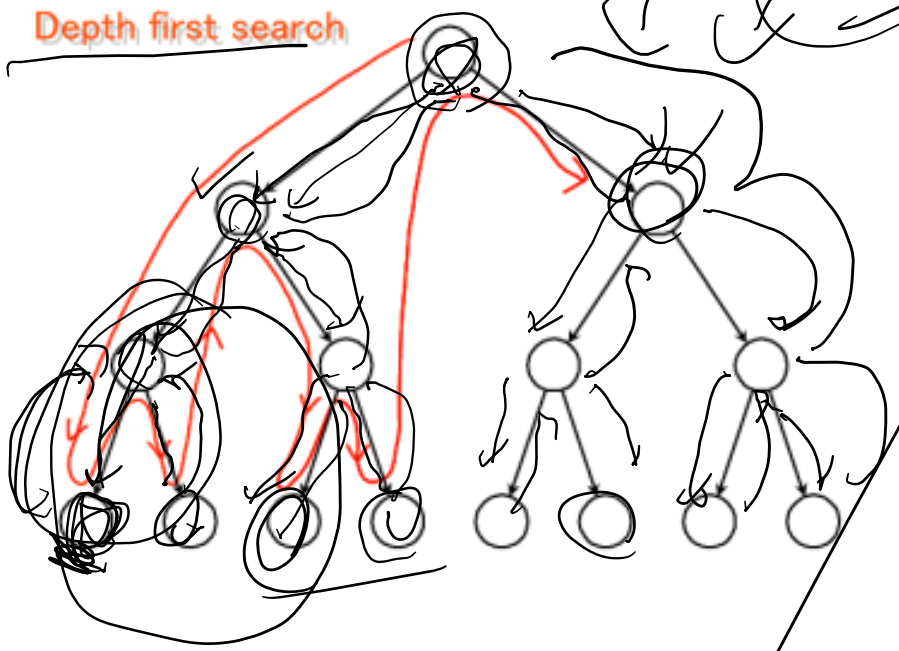
- 희소 그래프, 조밀 그래프

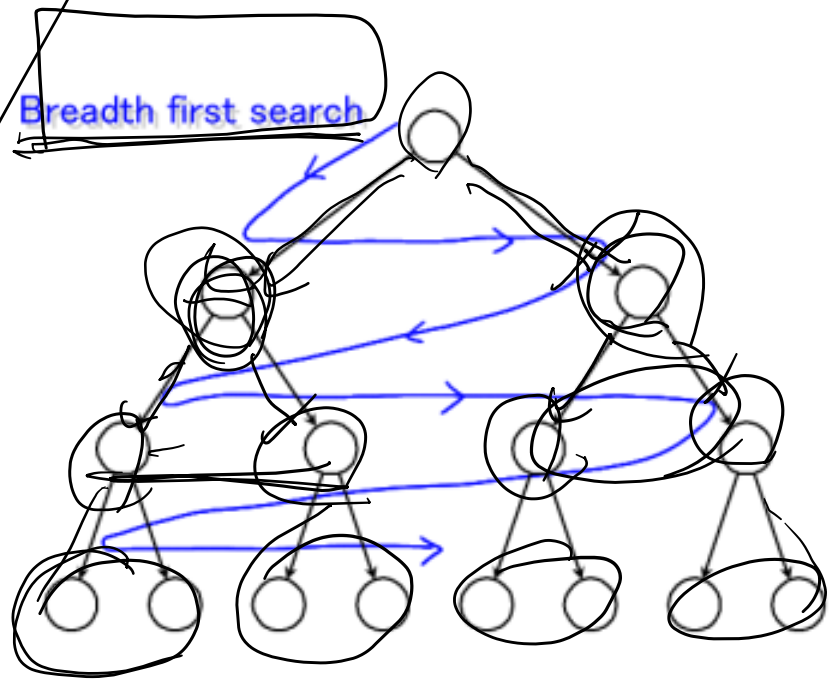
  - 간선 수가 적은 그래프를 희소 그래프(Sparse Graph)
  - 간선 수가 많은 그래프를 조밀 그래프(Dense Graph)
  - 조밀 그래프라면 인접행렬이 유리
  - 희소 그래프라면 인접 리스트가 유리

그래프에서 모든 정점 방문하기, Graph Traversal

- 대표적 두가지 방법
  - 너비우선탐색BFS (Breadth-First Search)
  - 깊이우선탐색DFS (Depth-First Search)

Graph Traversals





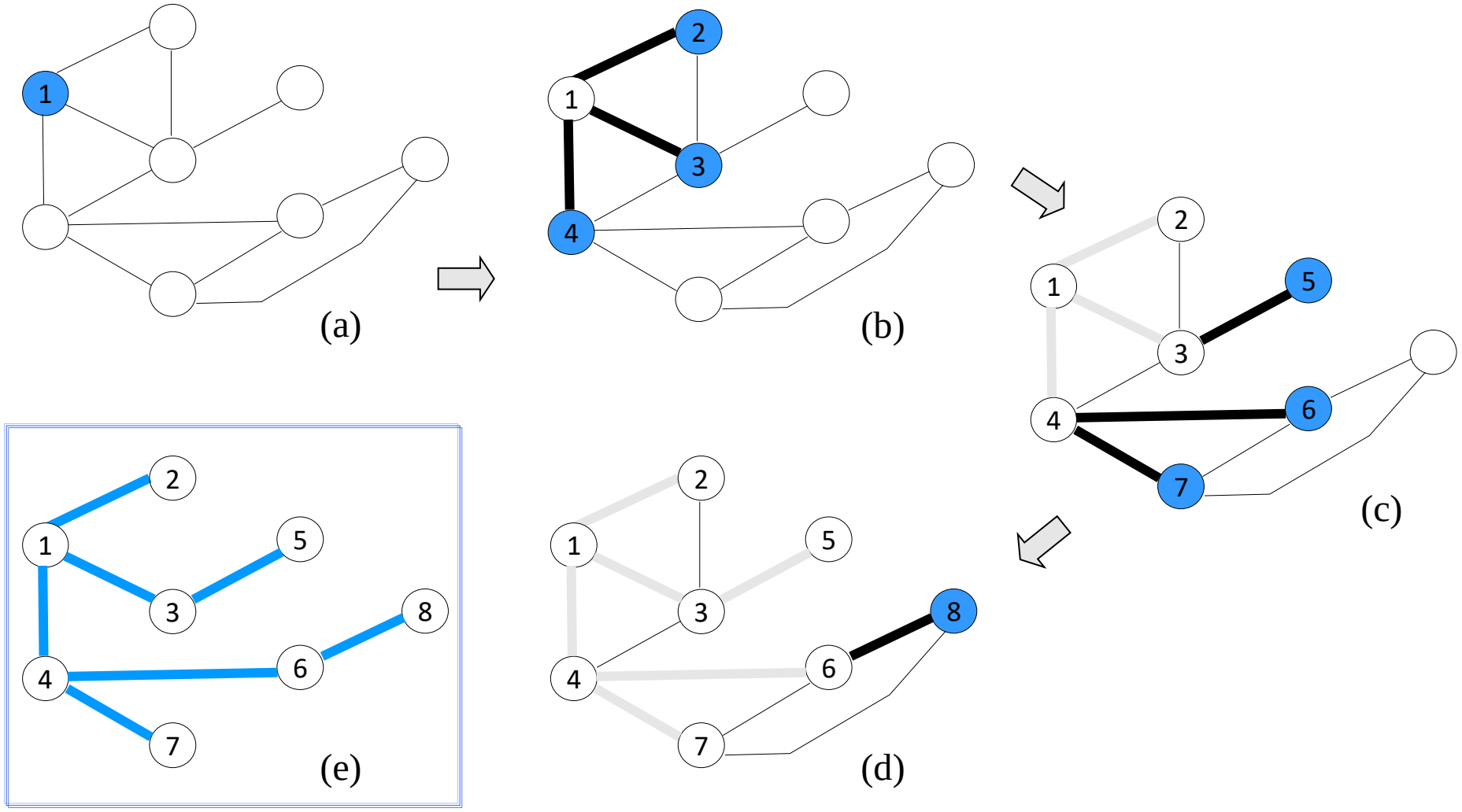
윤희 강이(가) 예약된 Zoom 회의에 귀하를 초대합니다.

주제: 윤희 강의 Zoom 회의
시간: 2020년 6월 19일 09:45 오전 서울

Zoom 회의 참가
https://us04web.zoom.us/j/7119694600?pwd=hKAfRiFB6ghUALM0v3U69hsfby8

회의 ID: 711 969 4600
비밀번호: 1234

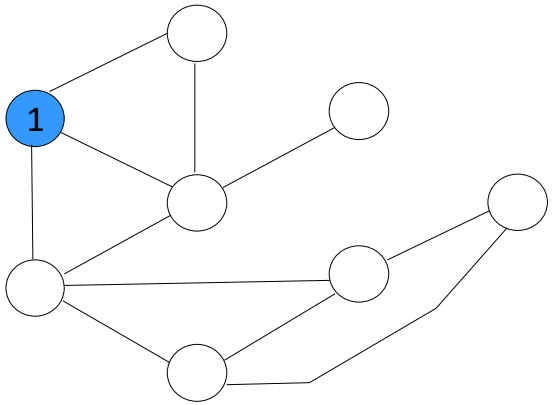
BFS의 작동 예



BFS너비우선탐색

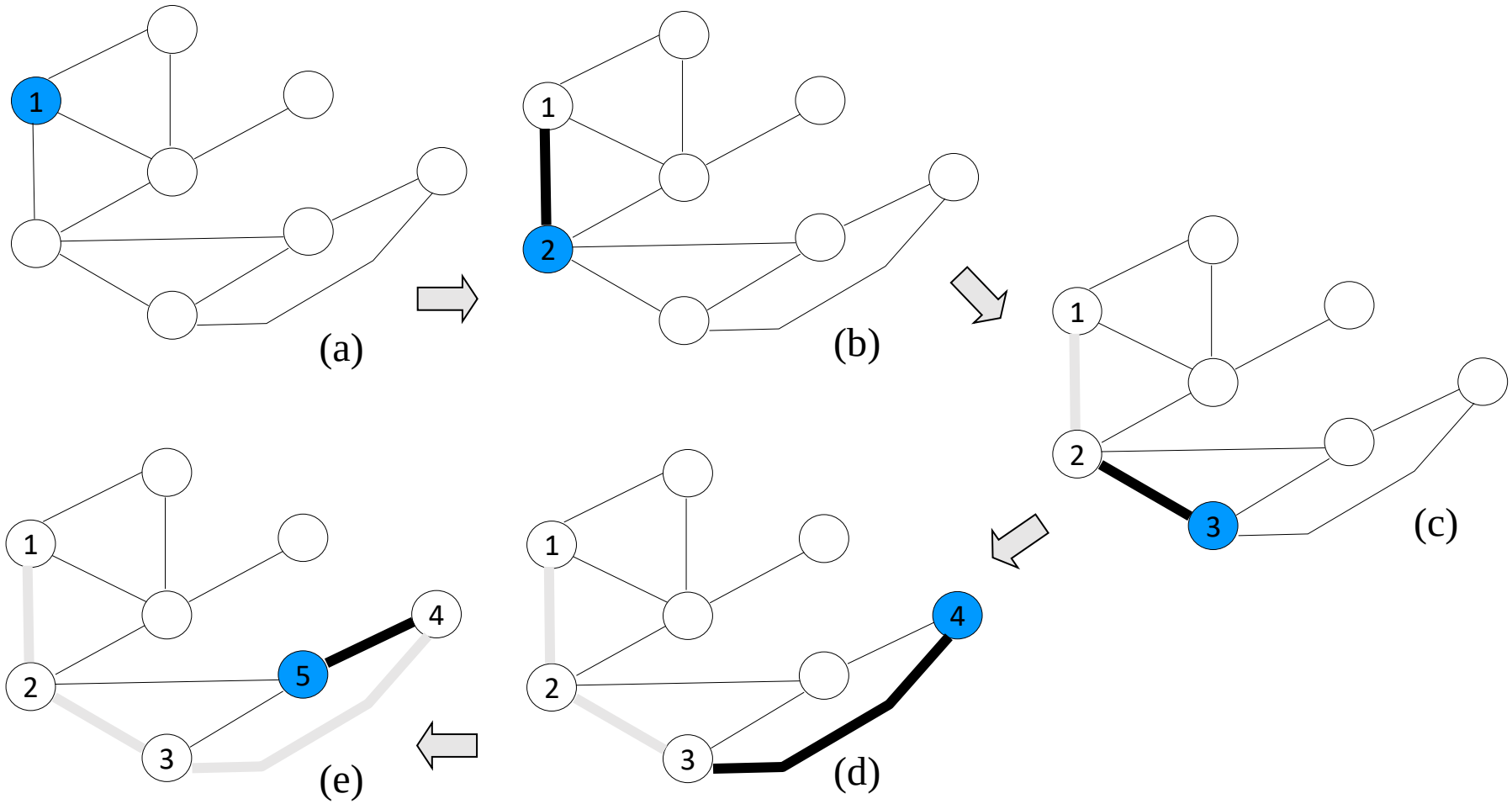
```
BFS(G, s)
{
        for each v ∈ V
                visited[v] ← NO;
        visited[s] ← YES;
        enqueue(Q, s);
        while (Q ≠Φ) {
                u ← dequeue(Q);
                for each v ∈ L(u)
                        if (visited[v] = NO) then
                                visited[v] ← YES;
                                enqueue(Q, v);
                        }

        }
}
```

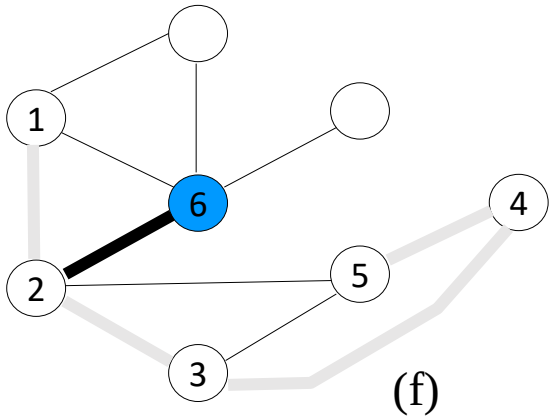


✓수행시간: $\Theta(|V|+|E|)$

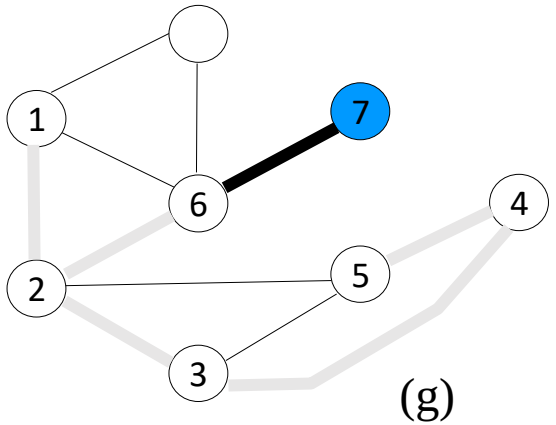
DFS의 작동 예



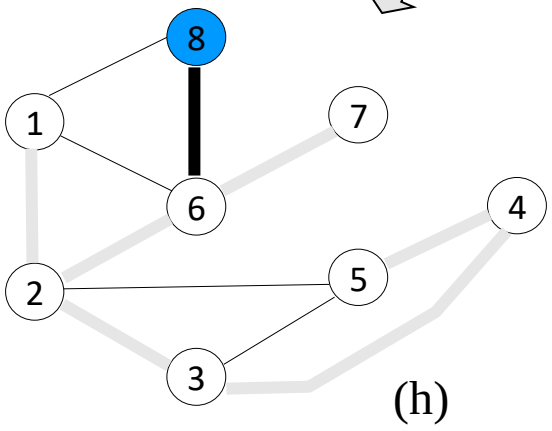
DFS의 작동 예 (계속)



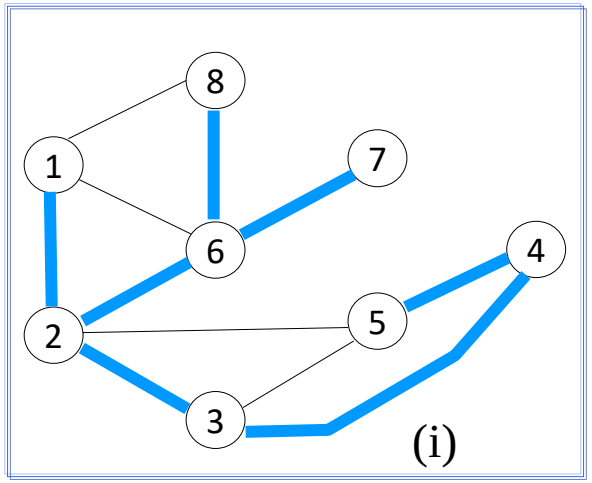
(f)



(g)

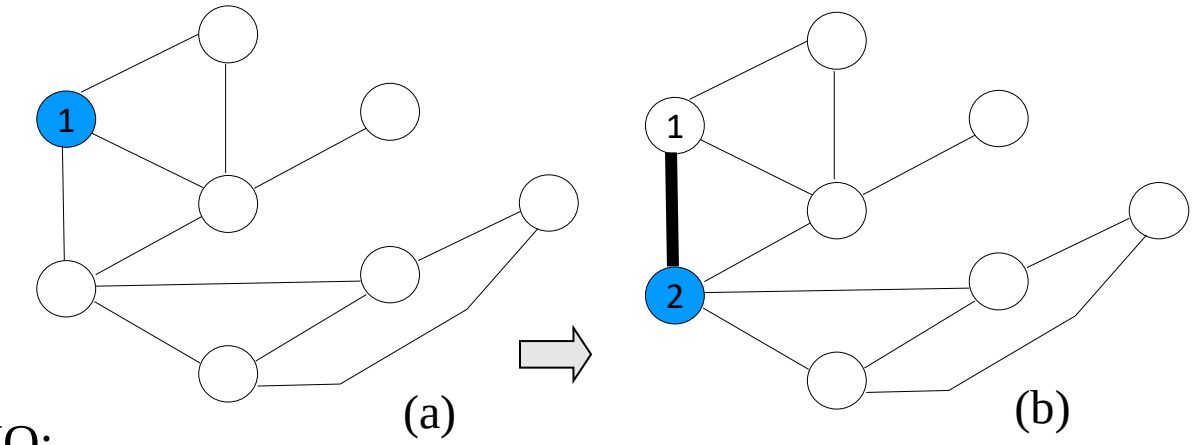


(h)



(i)

DFS깊이우선탐색




```
DFS(G)
{
        for each v ∈ V
                visited[v] ← NO;
        for each v ∈ V
                if (visited[v] = NO) then aDFS(v);
}
aDFS (v)
{
        visited[v] ← YES;
        for each x ∈ L(v)        ▷ L(v) : 정점 v의 인접 리스트
                if (visited[x] = NO) then aDFS(x);
}
```

방문여부 확인을 위한 자료구조가 필요
배열 값이 YES이면 방문된 상태
미 방문 상태로 초기화하기 위해 NO로 설정

✓수행시간: $\Theta(|V|+|E|)$

최소신장트리Minimum Spanning Trees

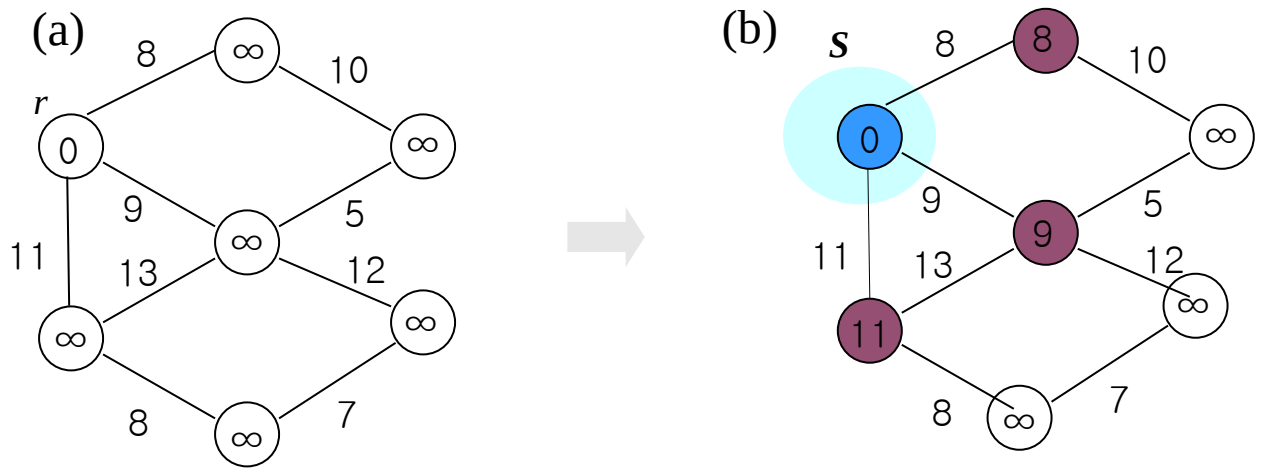
- 조건
  - 무향 연결 그래프
    - 연결 그래프connected graph : 모든 정점 간에 경로가 존재하는 그래프
- 트리
  - 싸이클이 없는 연결 그래프
  - *n* 개의 정점을 가진 트리는 항상 *n*-1개의 간선을 갖는다
- 그래프 *G*의 신장트리
  - *G*의 정점들과 간선들로만 구성된 트리 (BFS/DFS 로 구성)
- *G*의 최소신장트리
  - *G*의 신장트리들 중 간선의 합이 최소인 신장트리

프림Prim 알고리즘

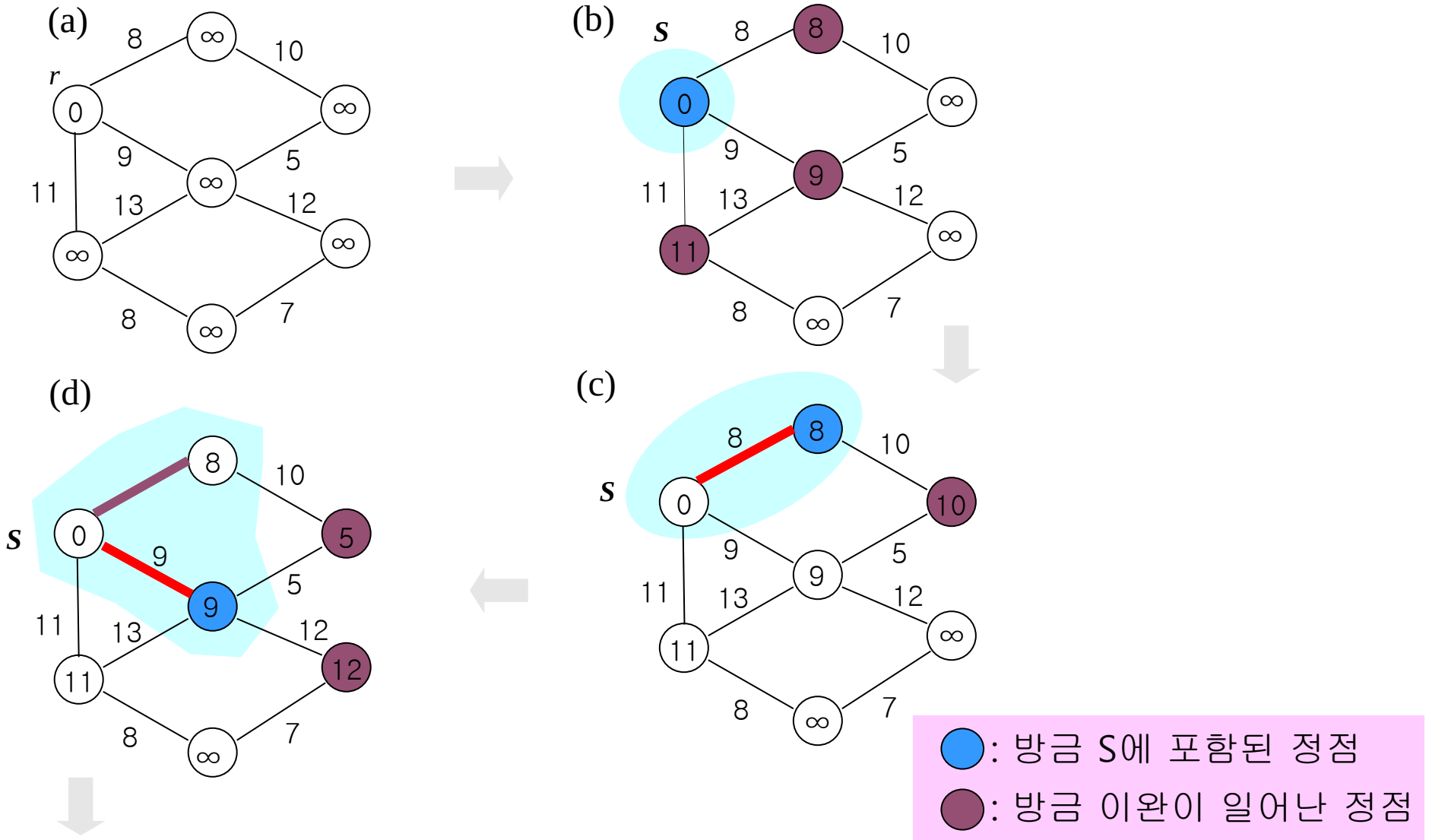
✓수행 시간: $O(|E|\log|V|)$

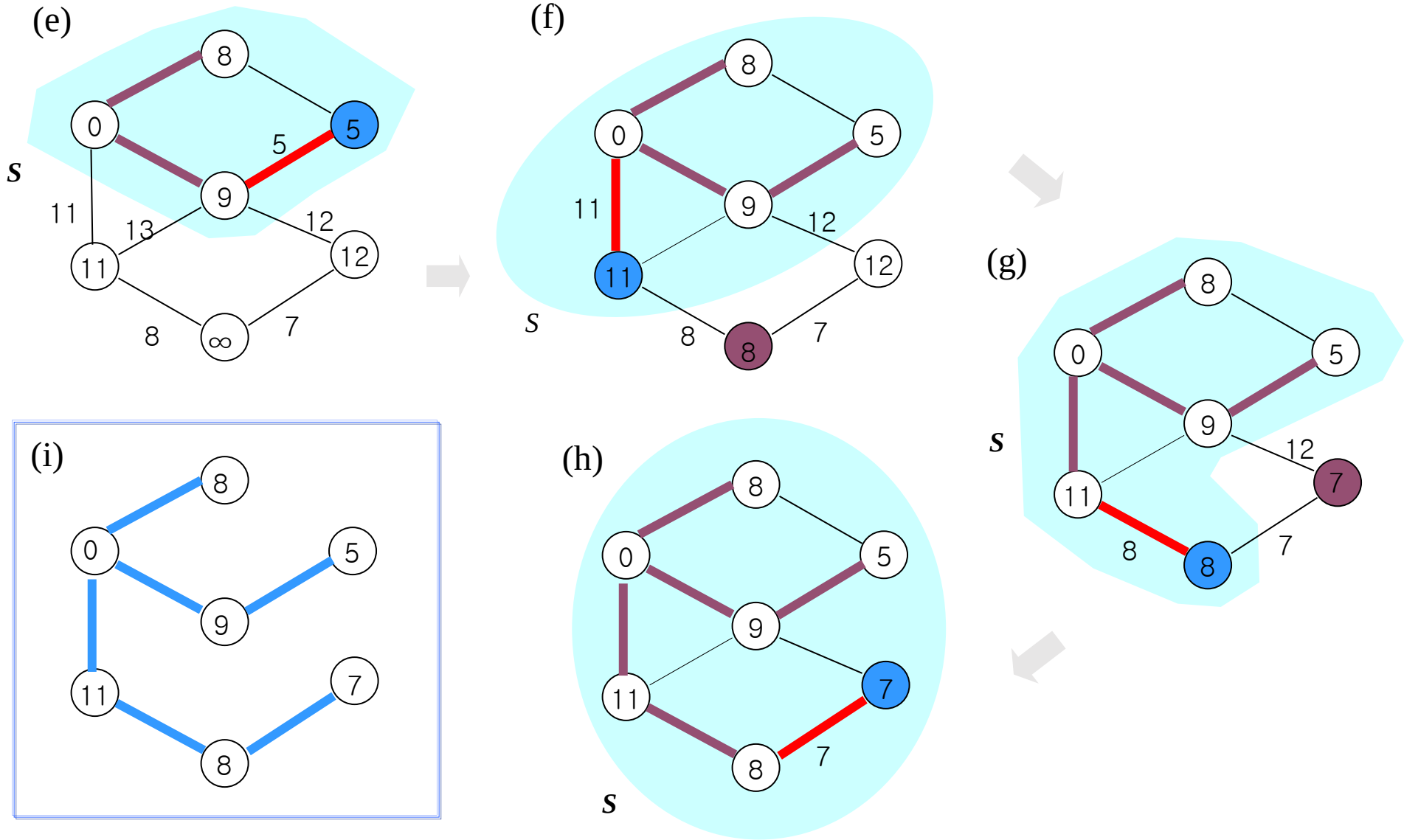
힙 이용

```
Prim (G, r)
{
    S ← Φ ;
    정점 r을 방문되었다고 표시하고, 집합 S에 포함시킨다;
    while (S≠ V) {
            S에서 V-S를 연결하는 간선들 중 최소길이의 간선 (x, y) 를 찾는다; ▷ (x∈S, y∈V-S)
            정점 y를 방문되었다고 표시하고, 집합 S에 포함시킨다;
    }
}
```

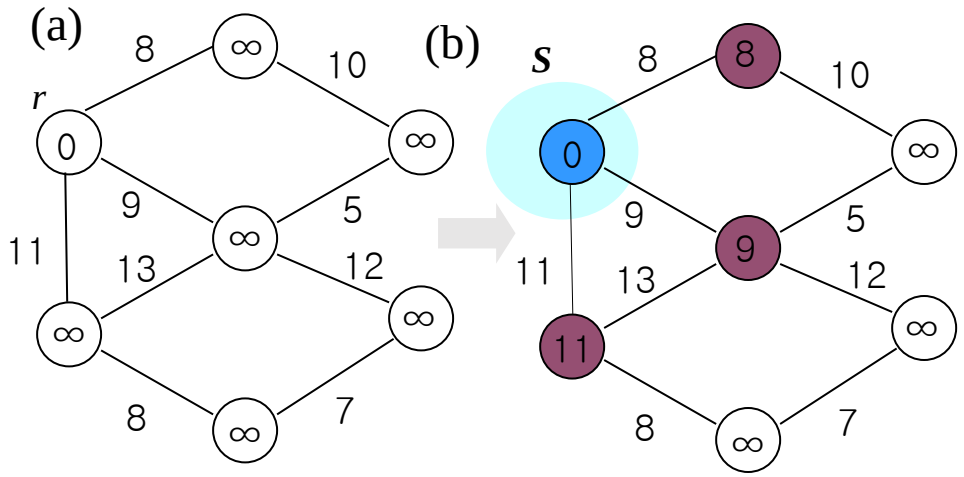


프림 알고리즘의 작동 예




(e)
8
0
5
5
s
9
11
13
12
12
11
8
∞
7
(f)
8
0
5
11
9
12
12
11
s
8
8
7
(g)
8
0
5
s
9
12
7
11
8
8
7
(h)
8
0
5
9
7
11
8
7
s
(i)
8
0
5
9
11
7
8

프림 알고리즘을 좀 더 구체적으로




```
Prim(G, r)
▷ G=(V, E): 주어진 그래프
▷ r: 시작으로 삼을 정점
{
    S ← Φ ;                    ▷ S : 정점 집합
    for each u∈V
        d_u ← ∞ ;
    d_r ← 0 ;
    while (S≠V){               ▷ n회 순환된다
        u ← extractMin(V-S, d) ;
        S ← S ∪ {u};
        for each v∈L(u) ▷ L(u) : u로부터 연결된 정점들의 집합
            if (v∈V-S and w_uv < d_v) then d_v ← w_uv ;
    }
}
```

이완(relaxation)

```
extractMin(Q, d)
{
    집합 Q에서 d값이 가장 작은 정점 u를 리턴한다 ;
}
```

✓수행시간: $O(|E|\log|V|)$

힙 이용