컴퓨터데이터구조

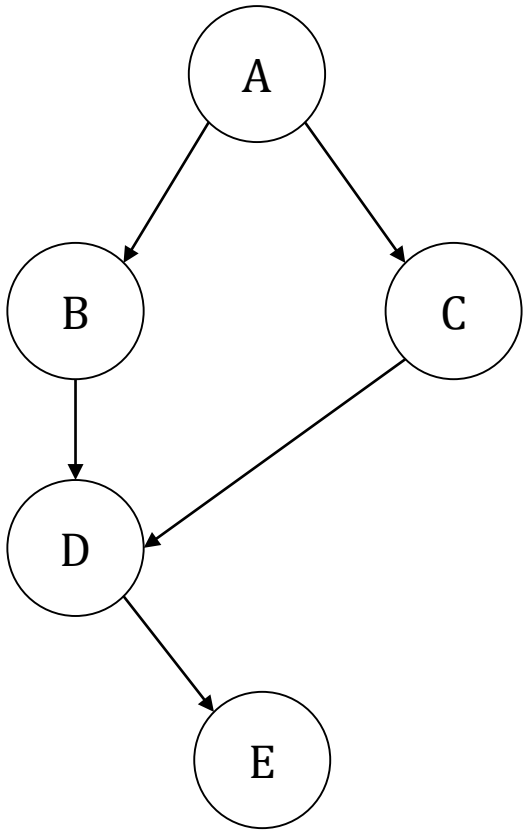
김응희

ehkim@sunmoon.ac.kr

05.28 – 05.31

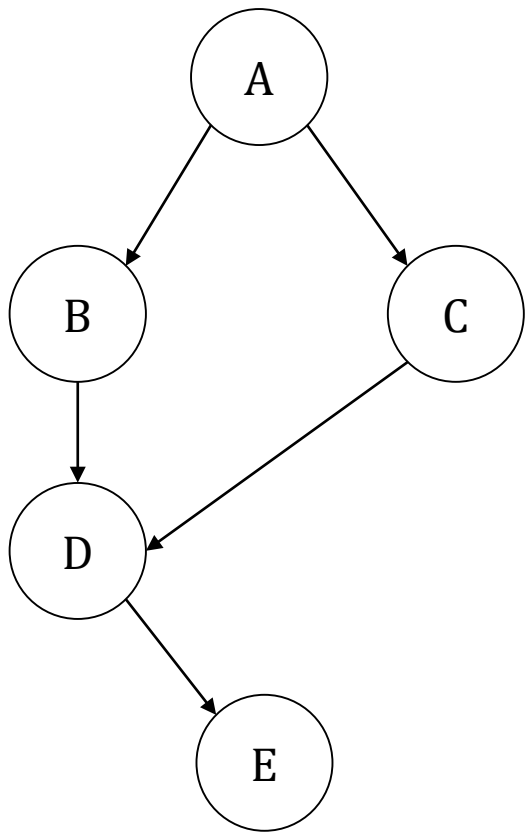
위상 정렬
Topological sort

- **싸이클이 없는 유방향 그래프 Directed Acyclic Graph (DAG)의 정점들을 일렬로 나열**



위상 정렬
Topological sort

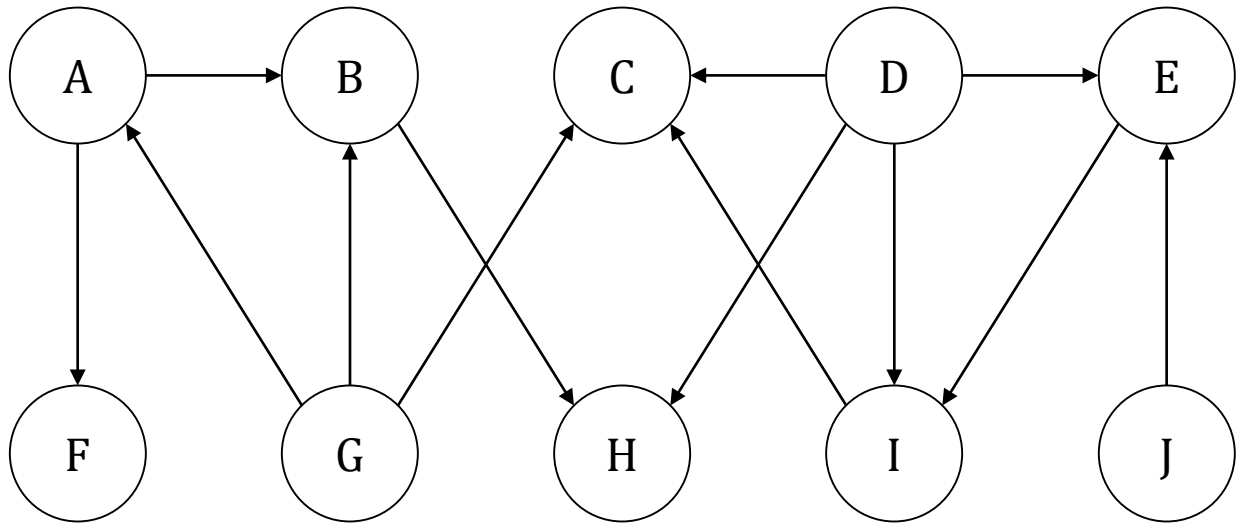
- **싸이클이 없는 유방향 그래프 Directed Acyclic Graph (DAG)의 정점들을 일렬로 나열**



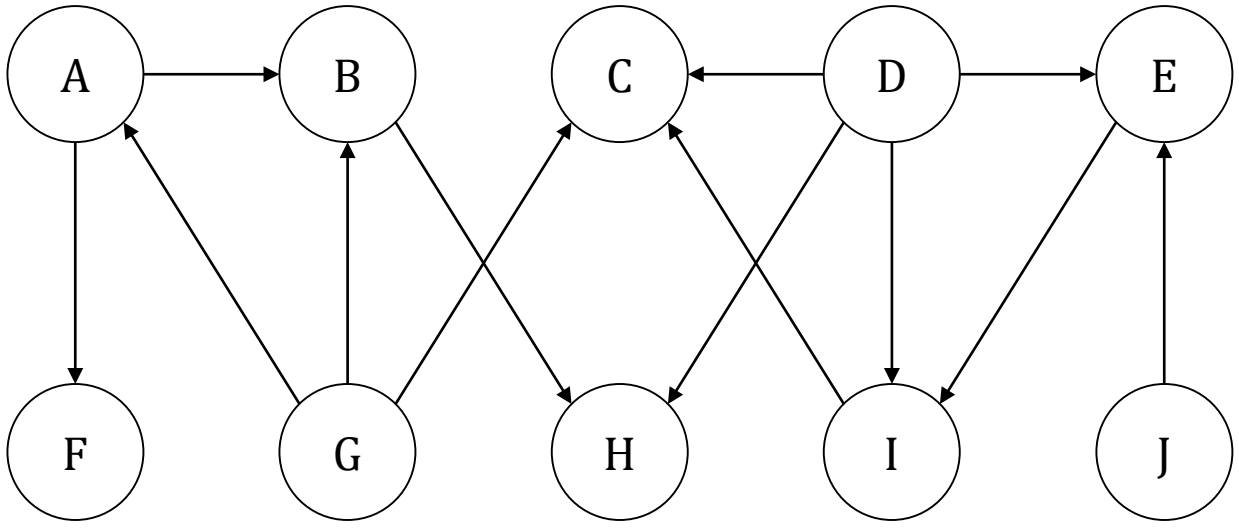
위상 정렬

- A B C D E
- A C B D E

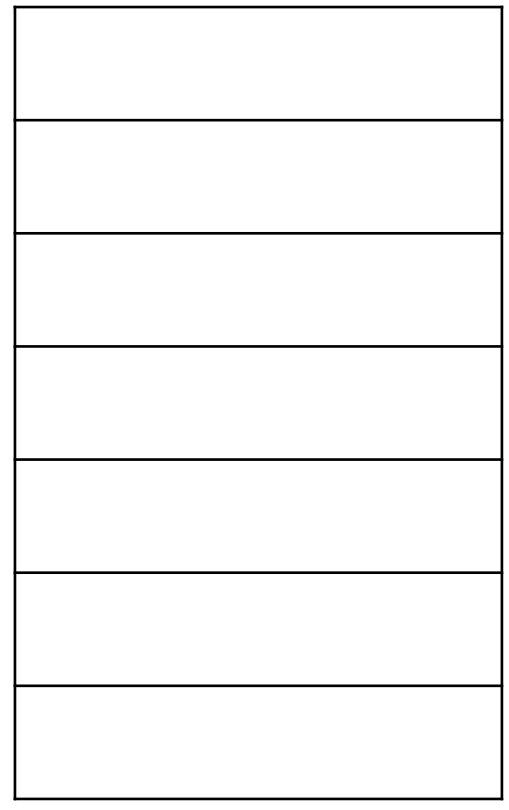
위상 정렬
Topological sort



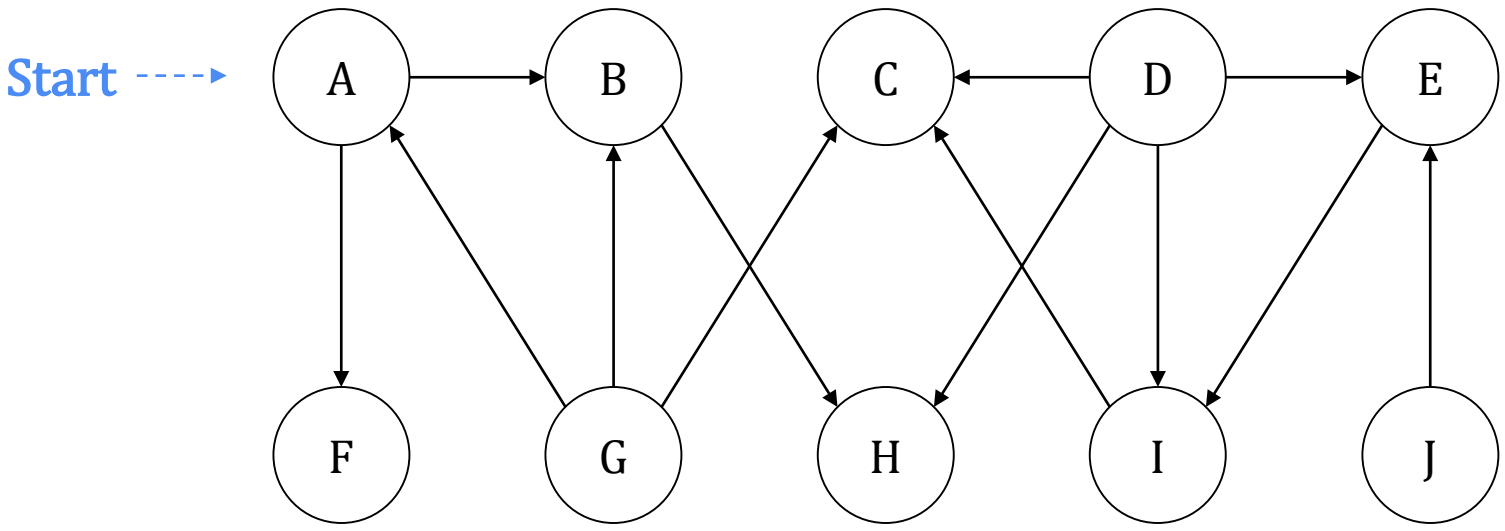
위상 정렬
Topological sort

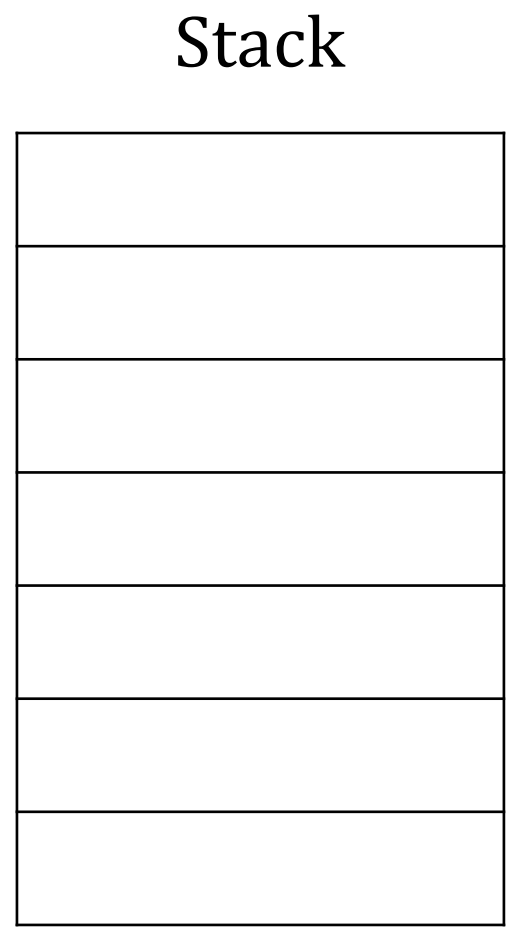


Stack

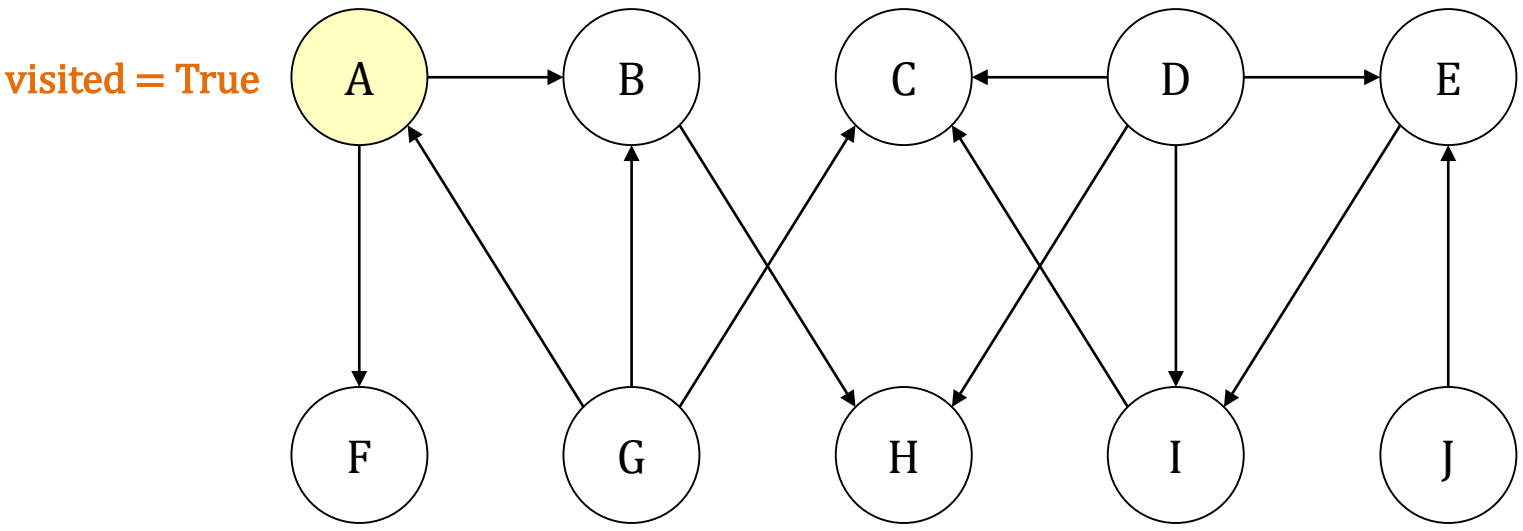


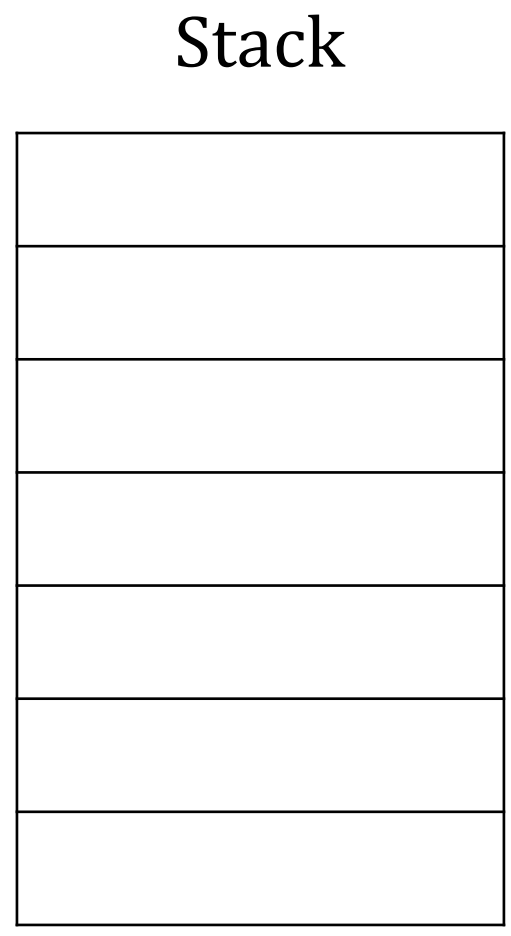
위상 정렬
Topological sort



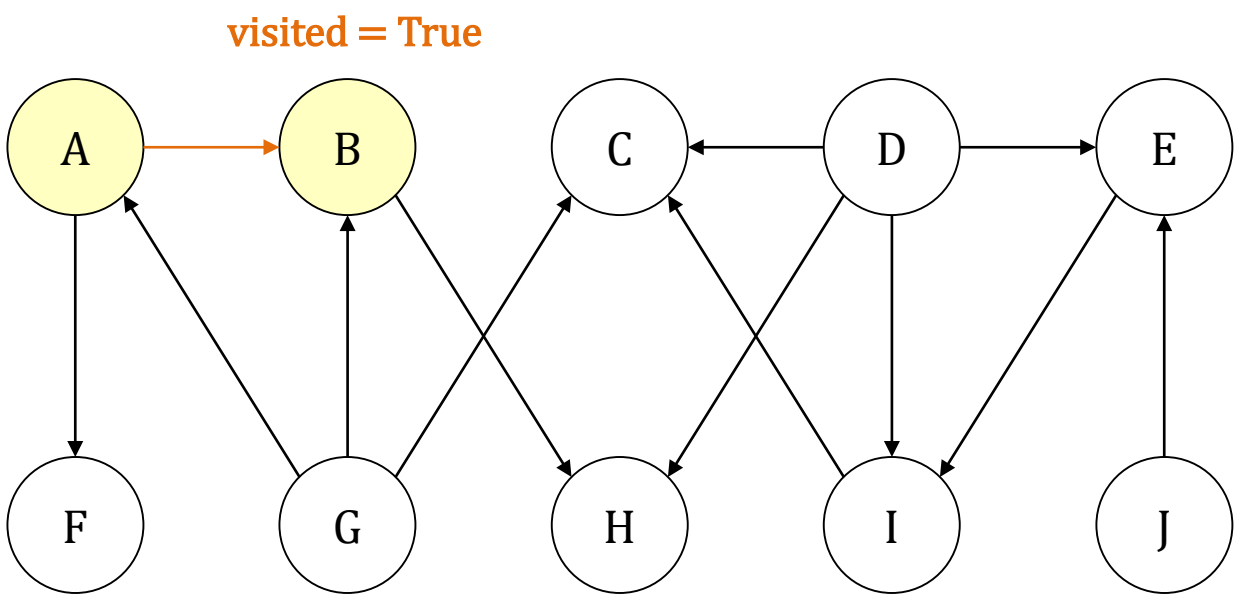


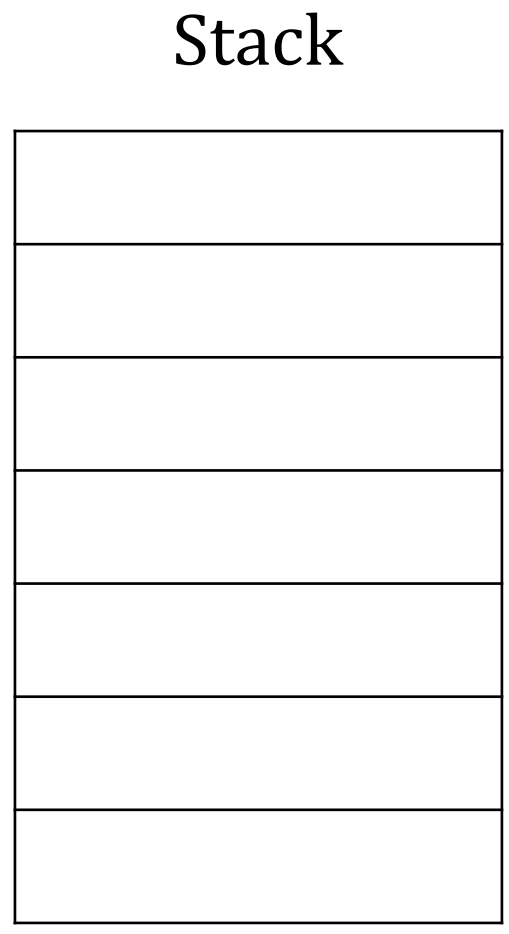
위상 정렬
Topological sort



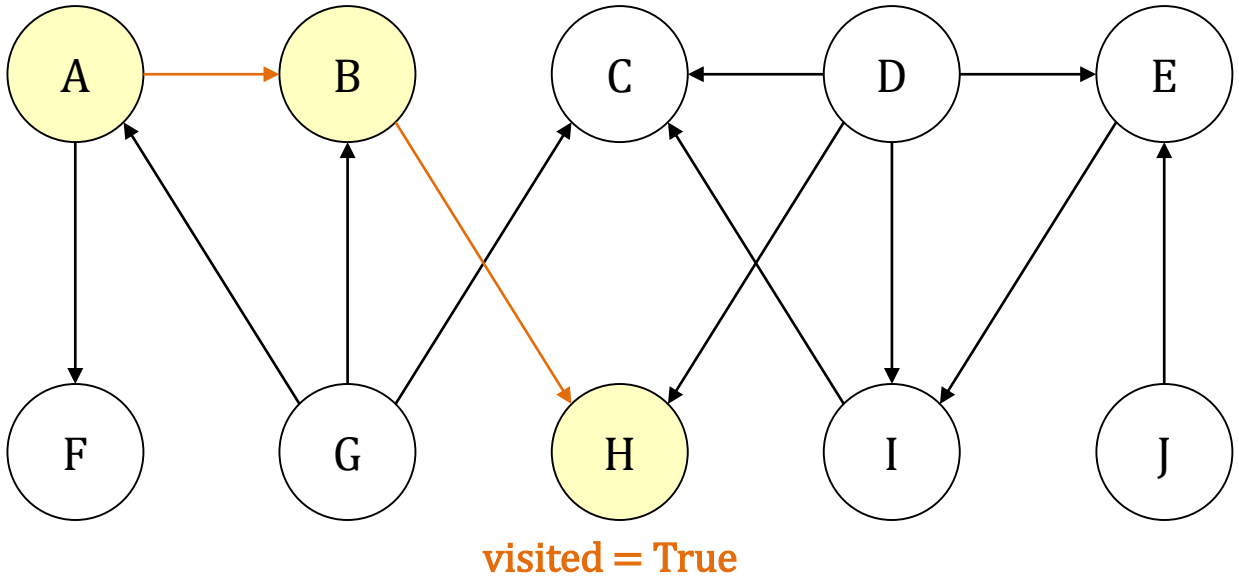


위상 정렬
Topological sort

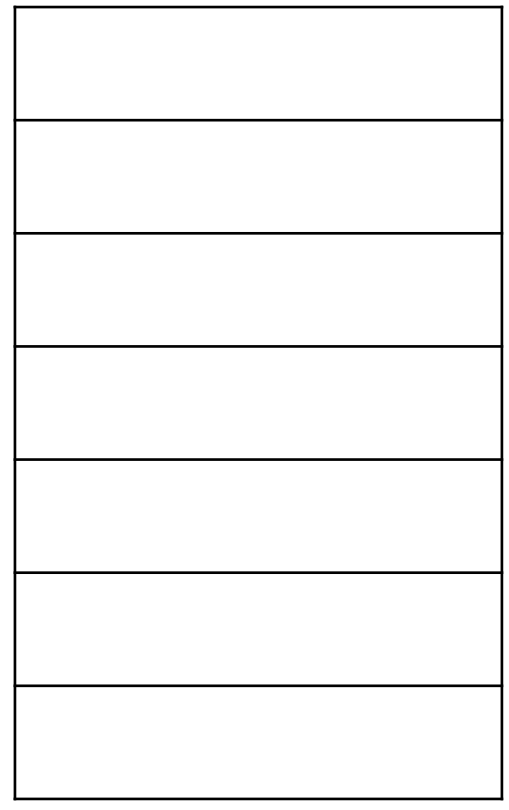




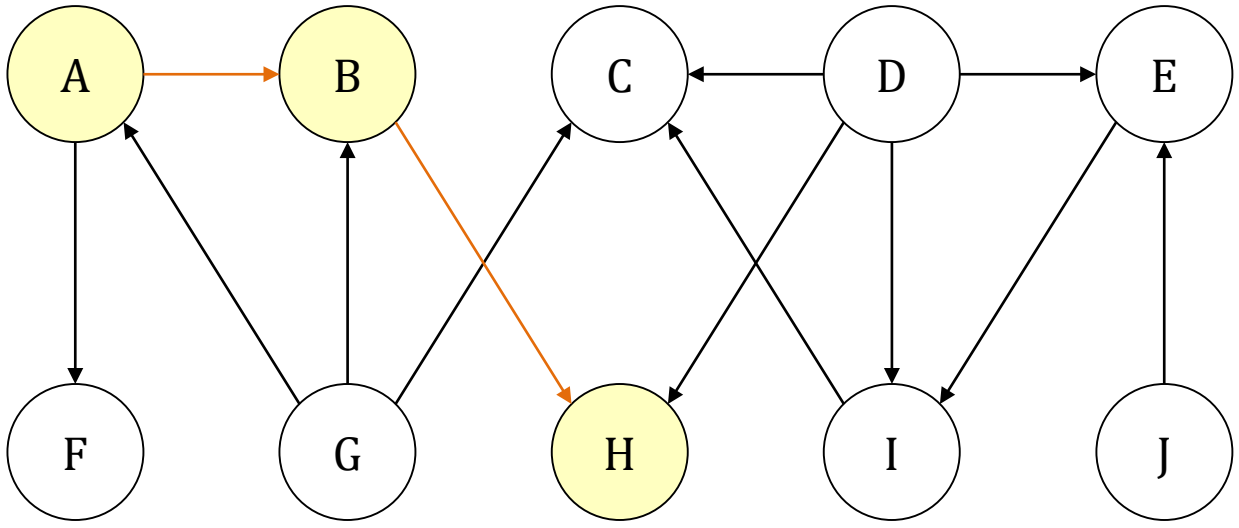
위상 정렬
Topological sort



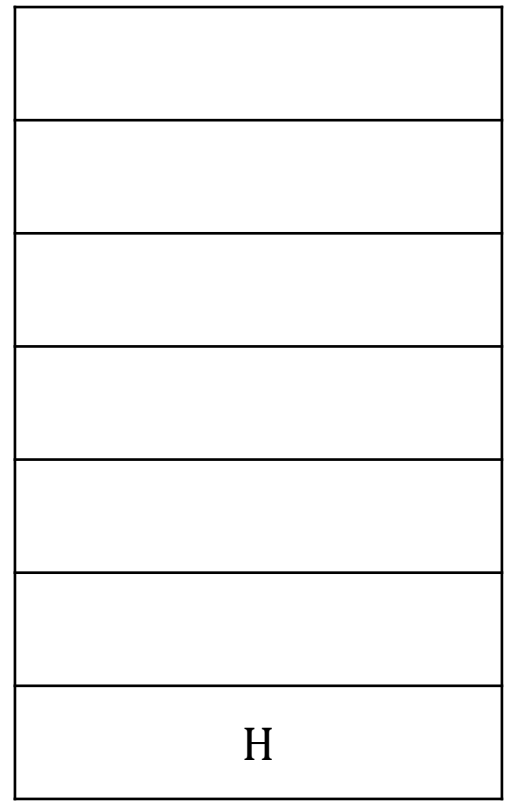
Stack



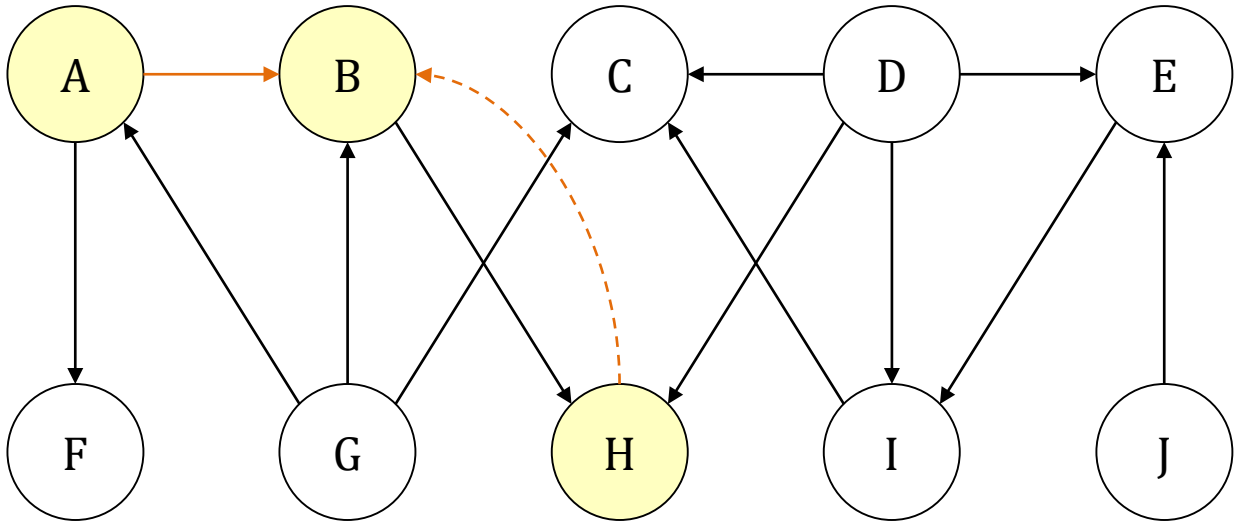
위상 정렬
Topological sort



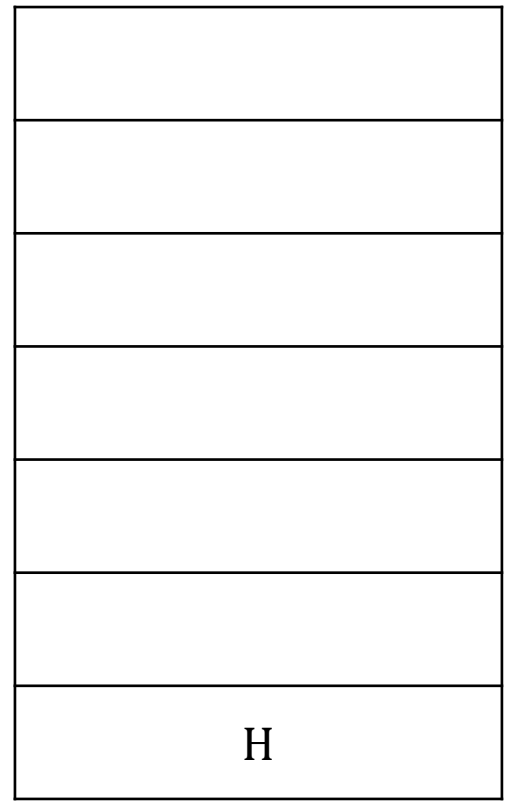
Stack



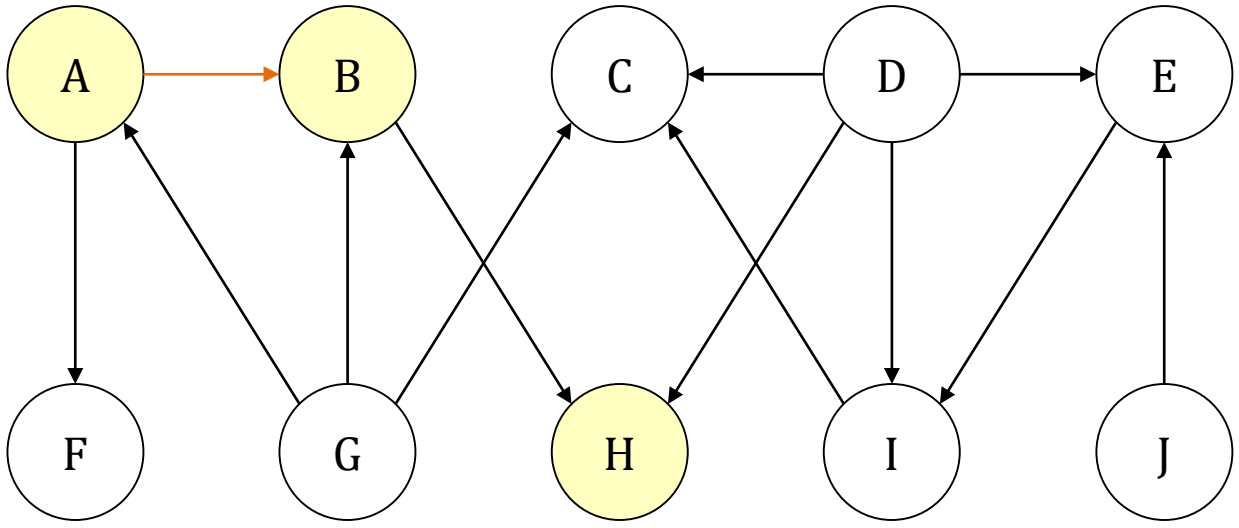
위상 정렬
Topological sort



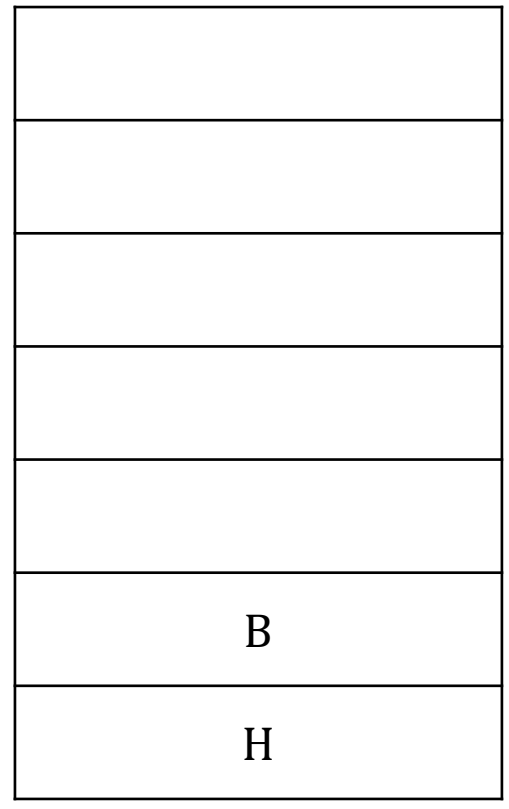
Stack



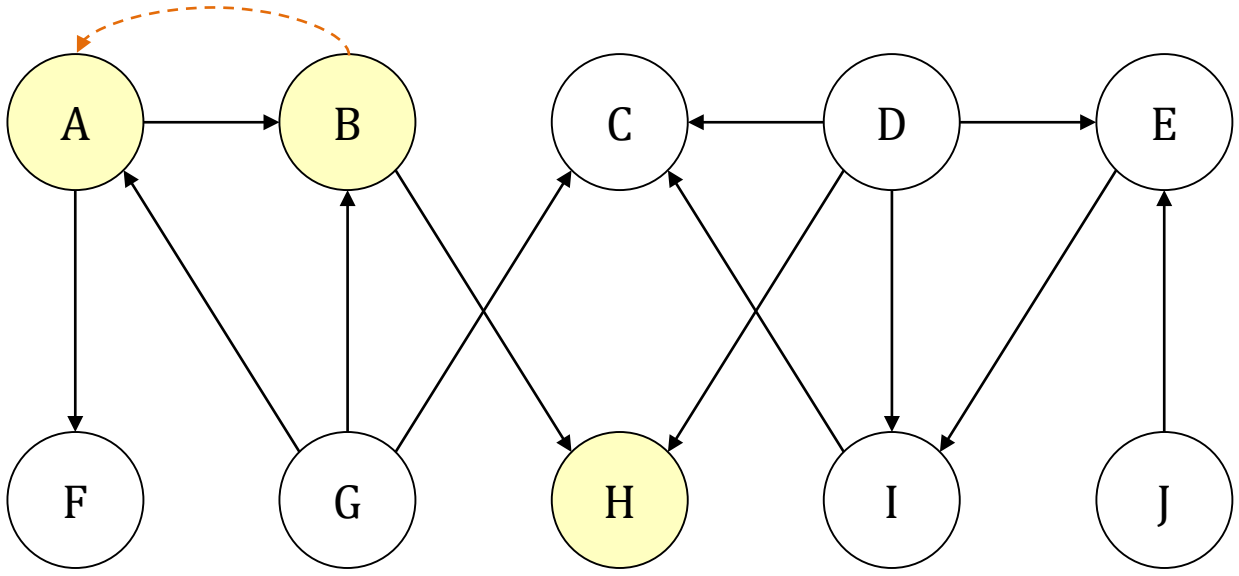
위상 정렬
Topological sort



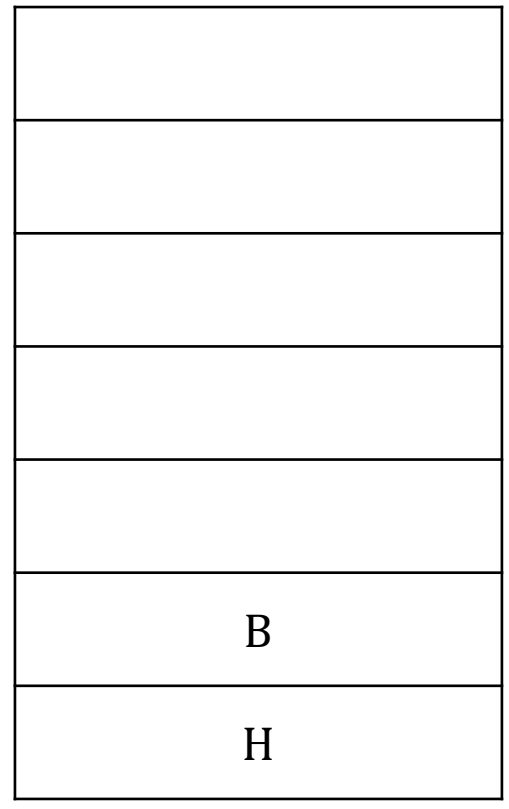
Stack



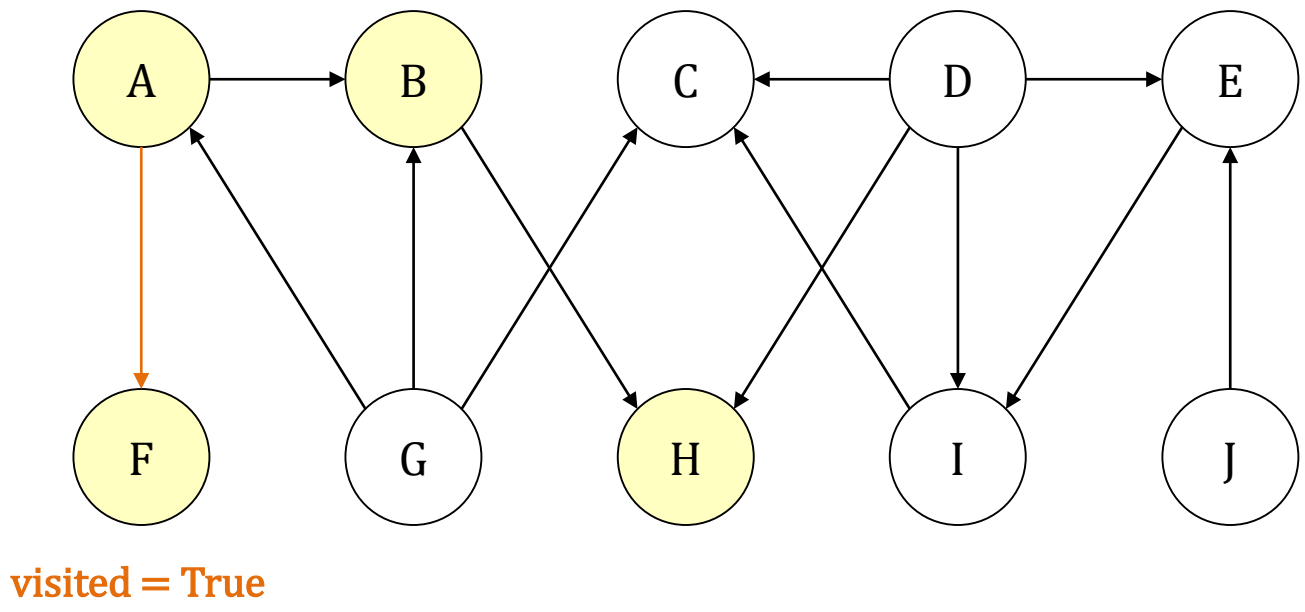
위상 정렬
Topological sort



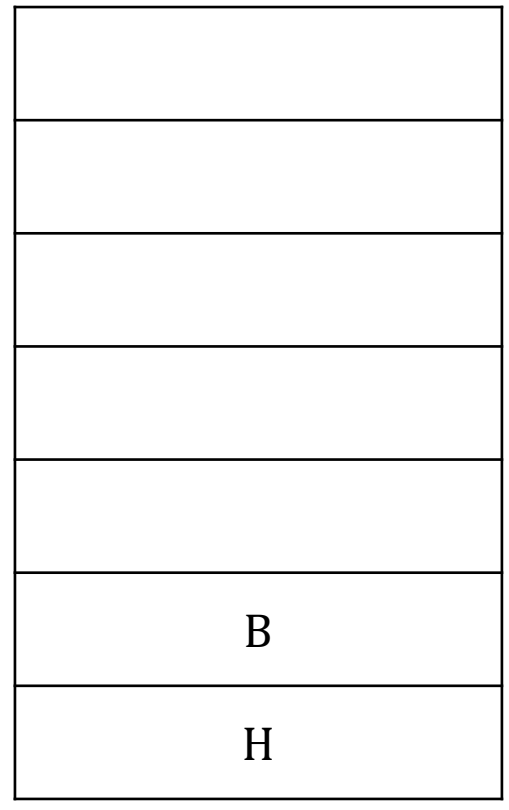
Stack



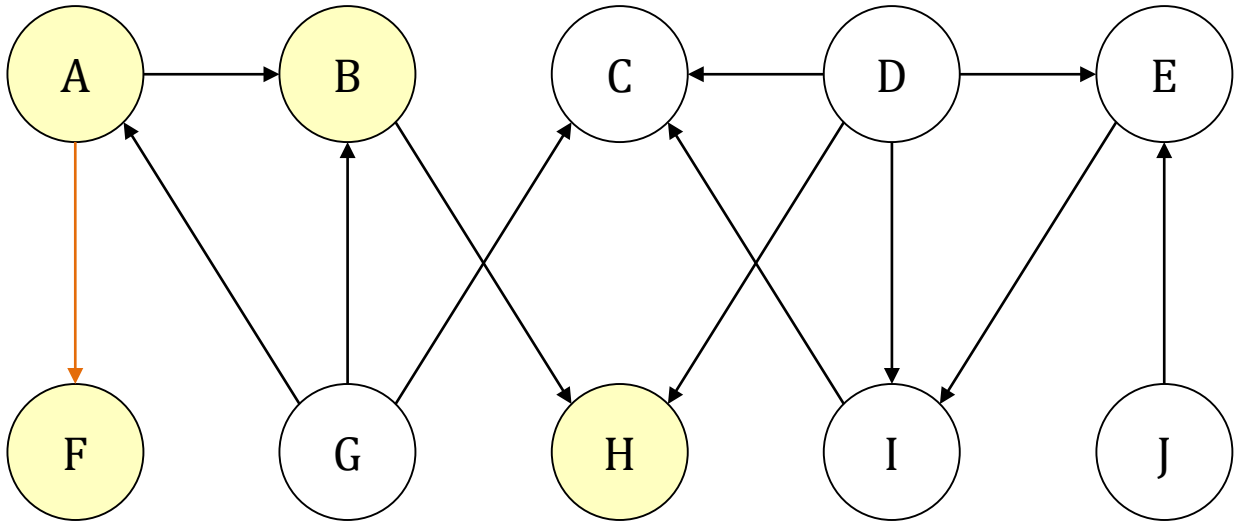
위상 정렬
Topological sort



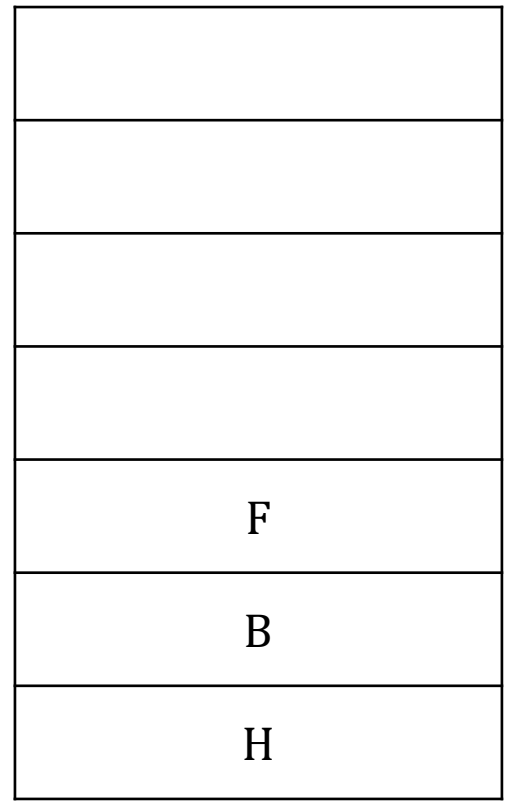
Stack



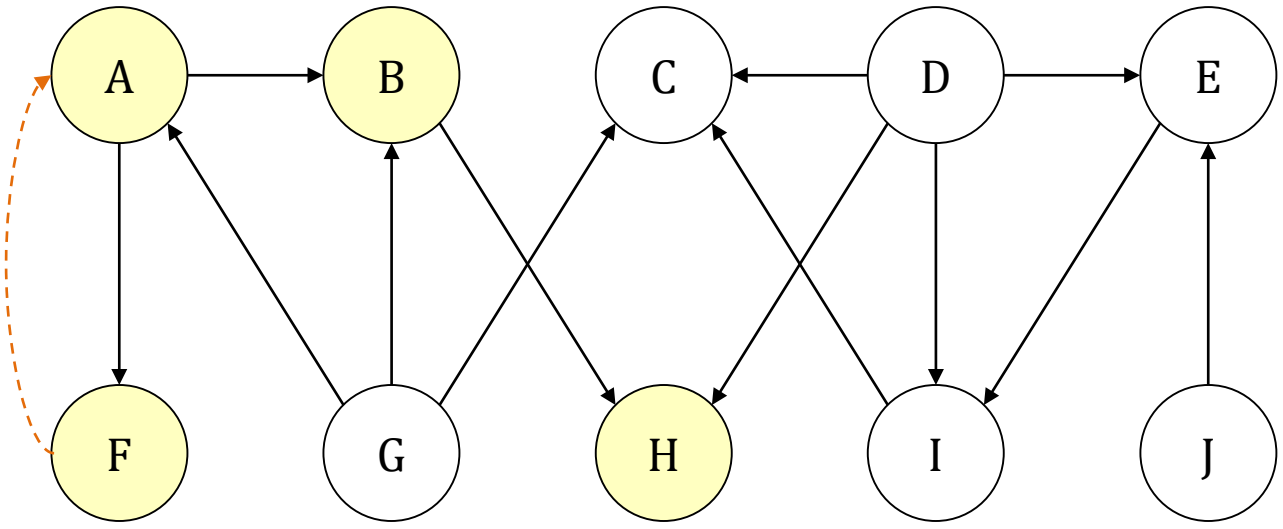
위상 정렬
Topological sort



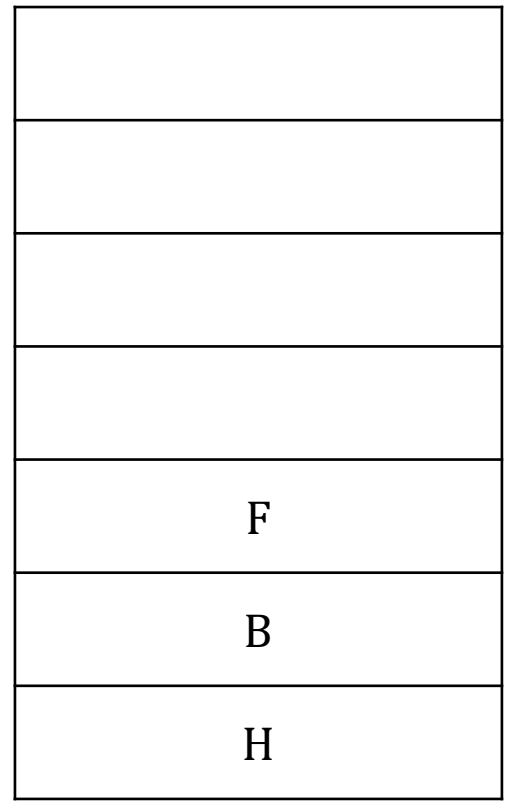
Stack



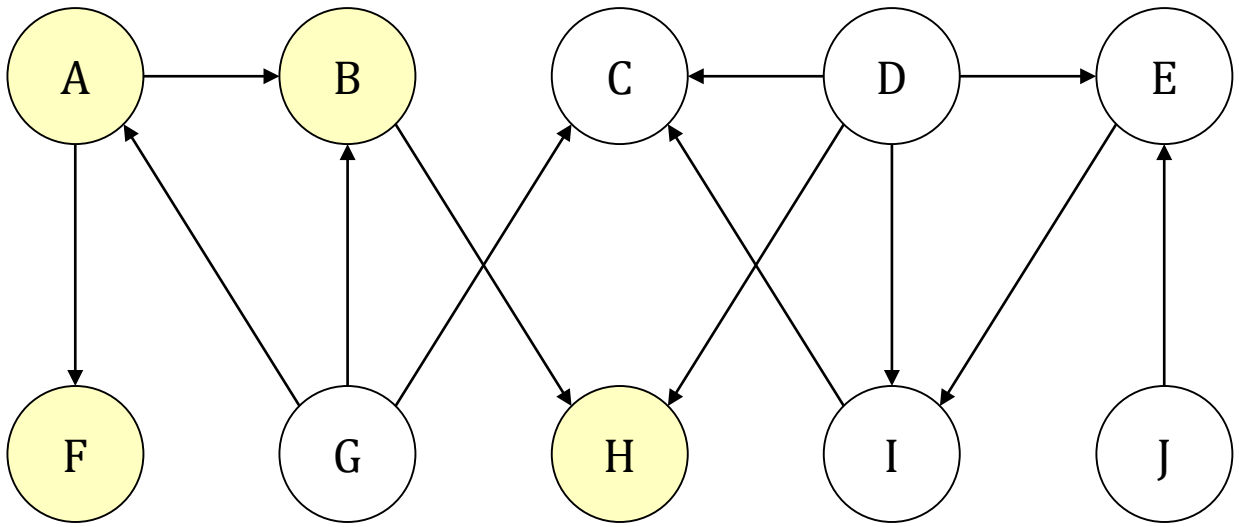
위상 정렬
Topological sort



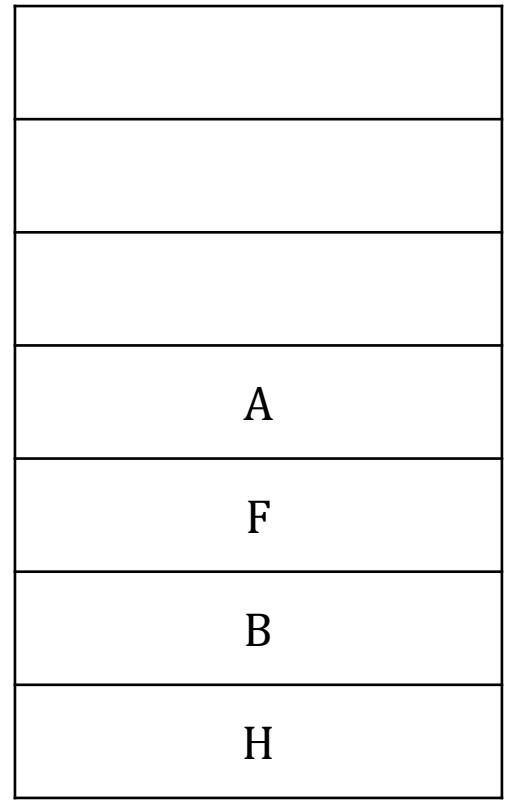
Stack



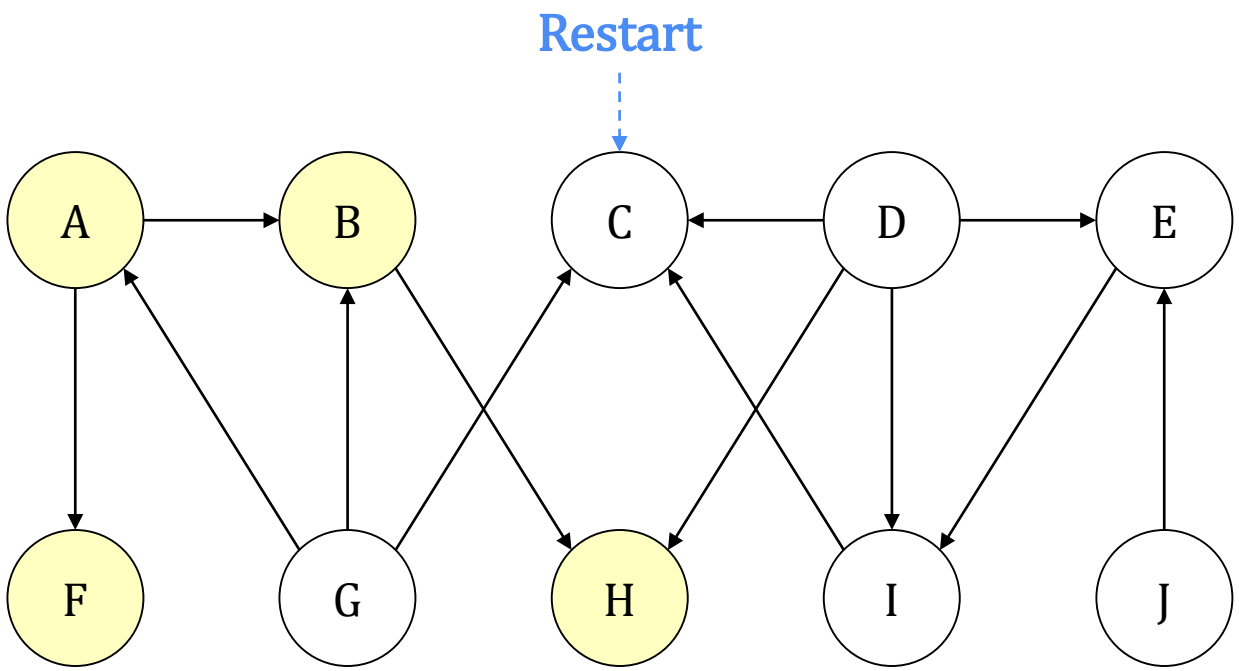
위상 정렬
Topological sort



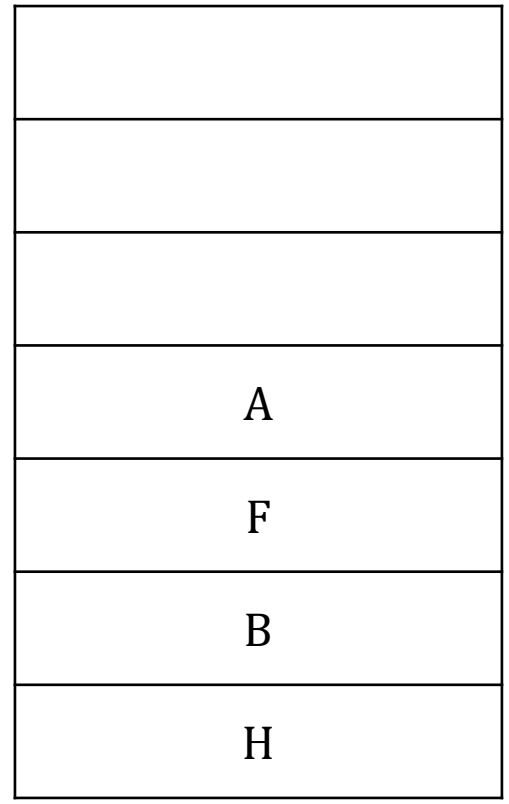
Stack



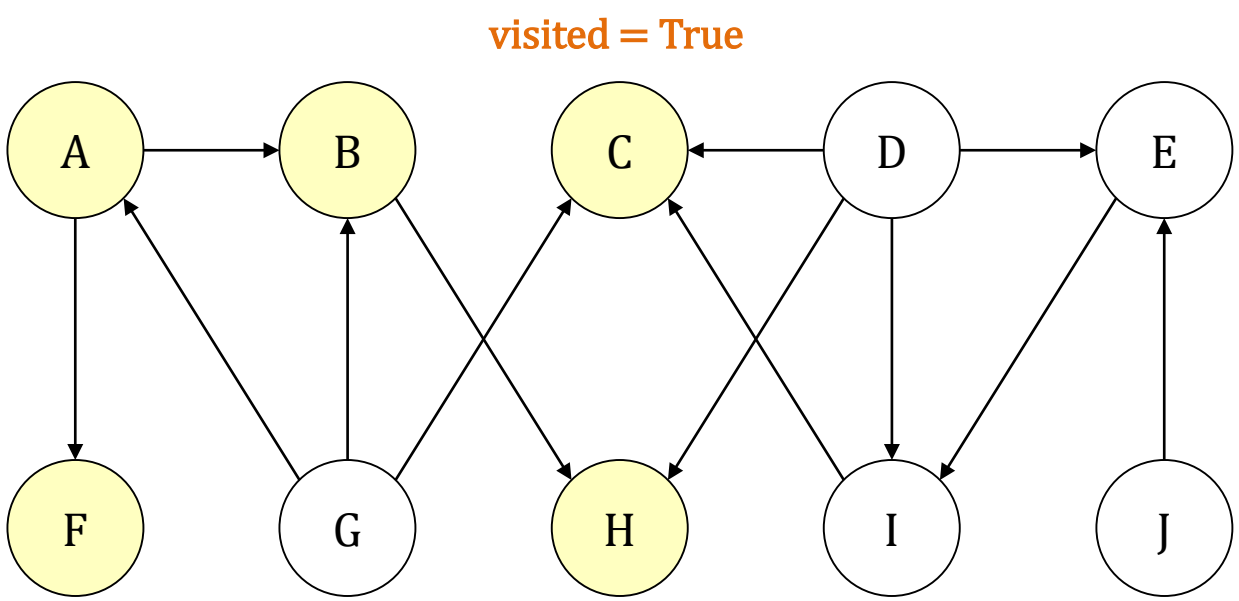
위상 정렬
Topological sort



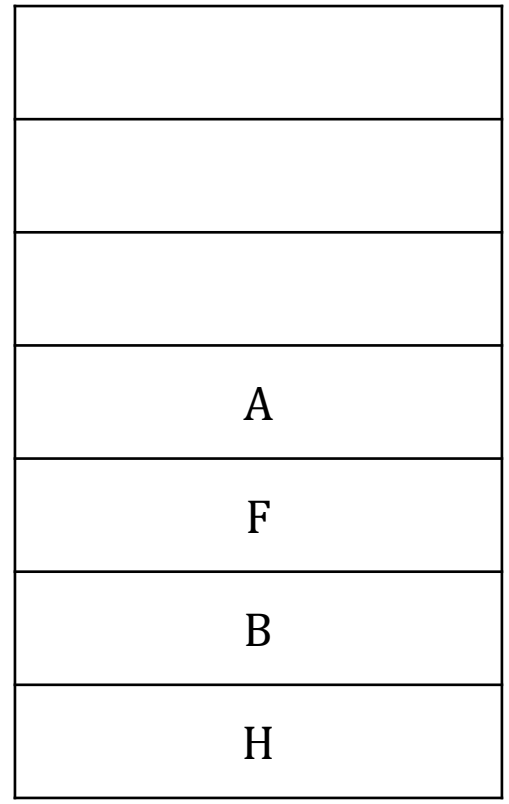
Stack



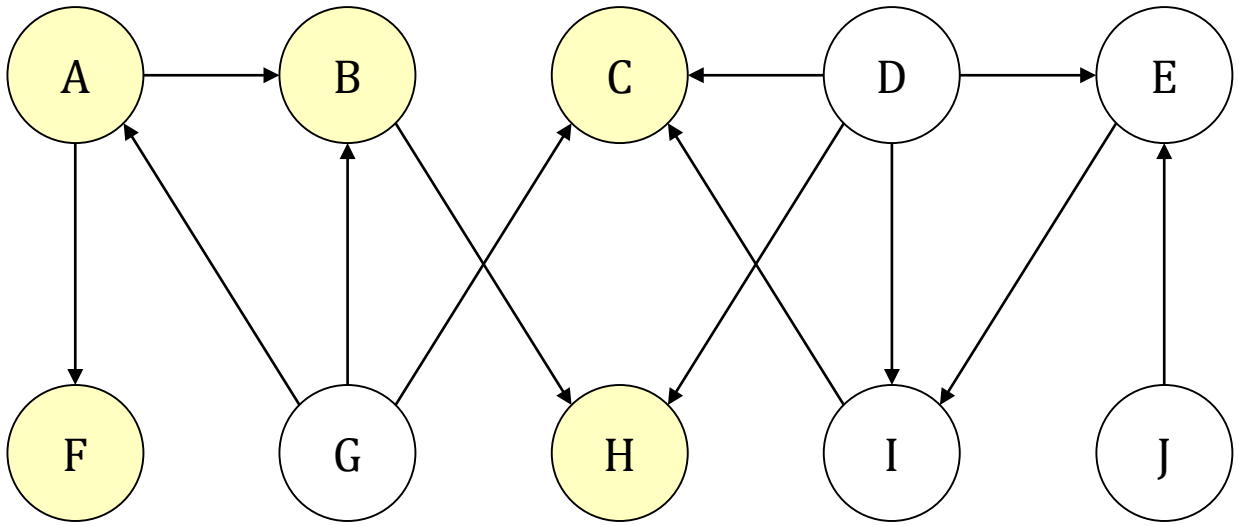
위상 정렬
Topological sort



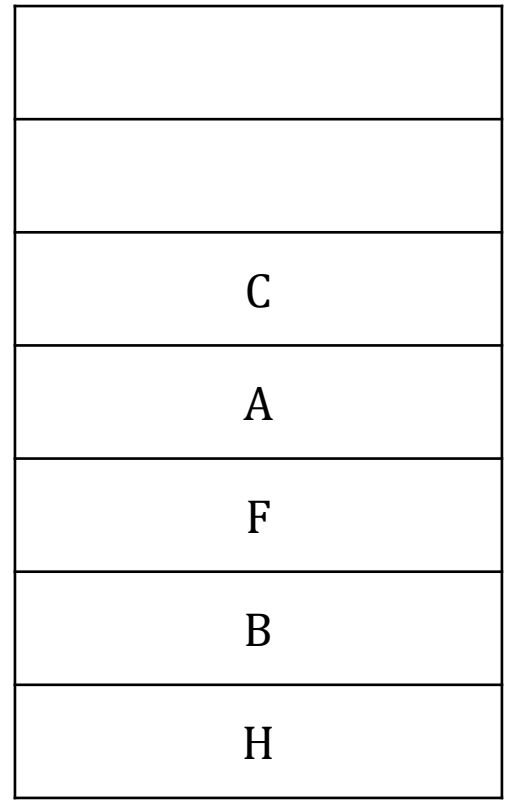
Stack



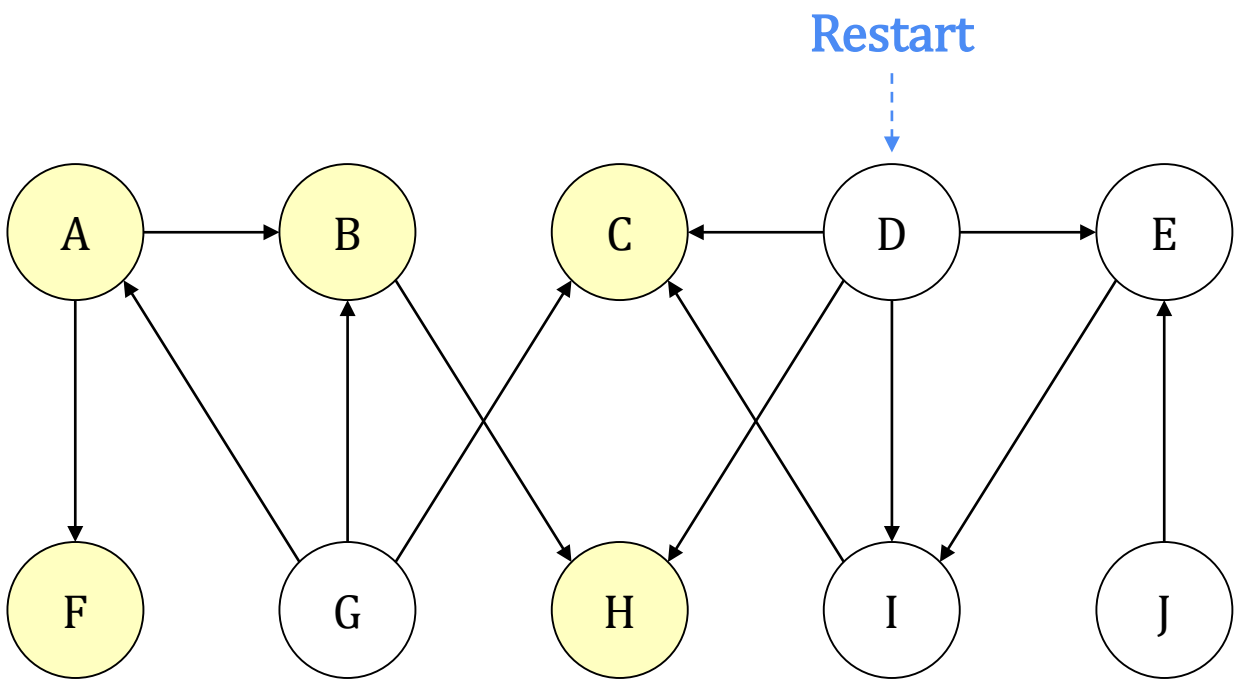
위상 정렬
Topological sort



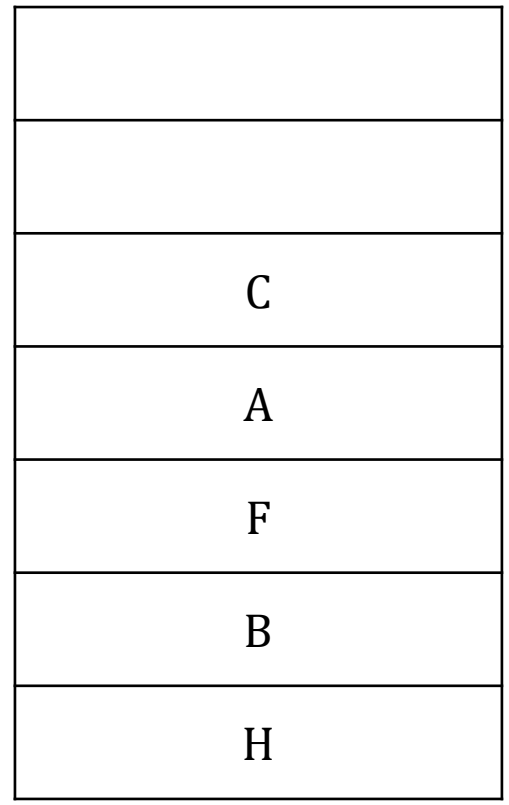
Stack



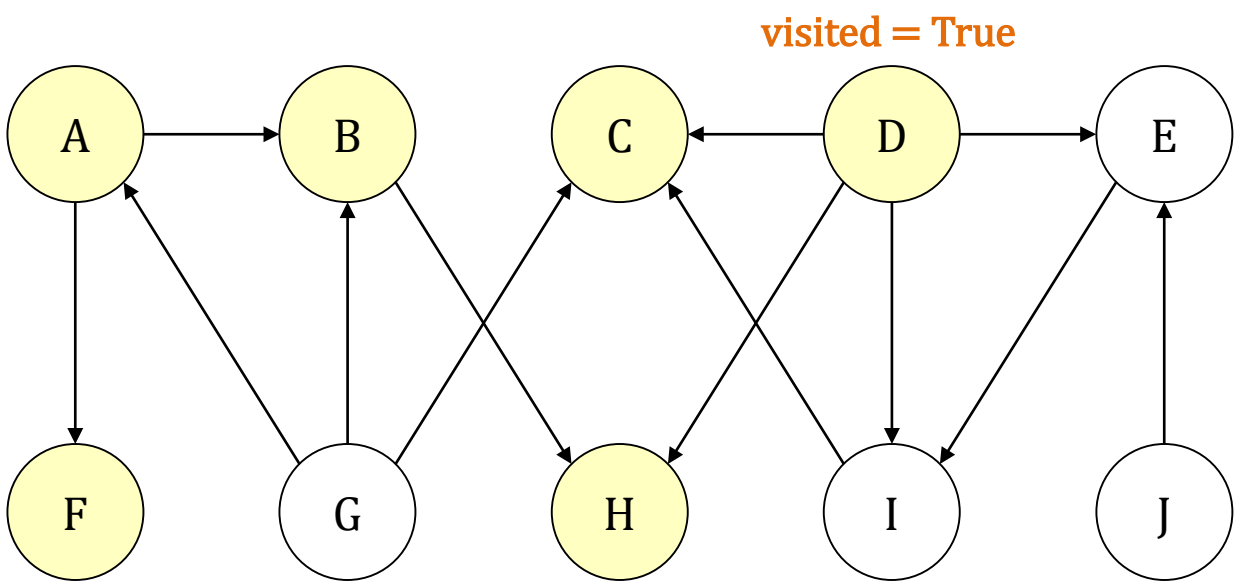
위상 정렬
Topological sort



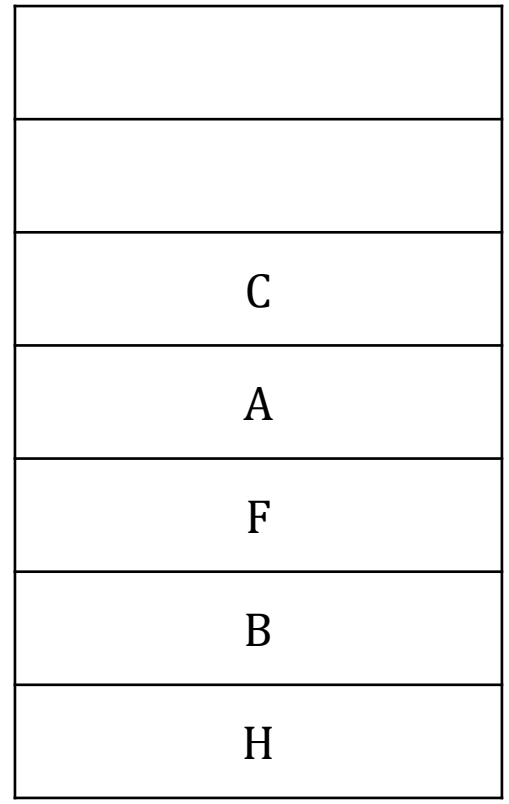
Stack



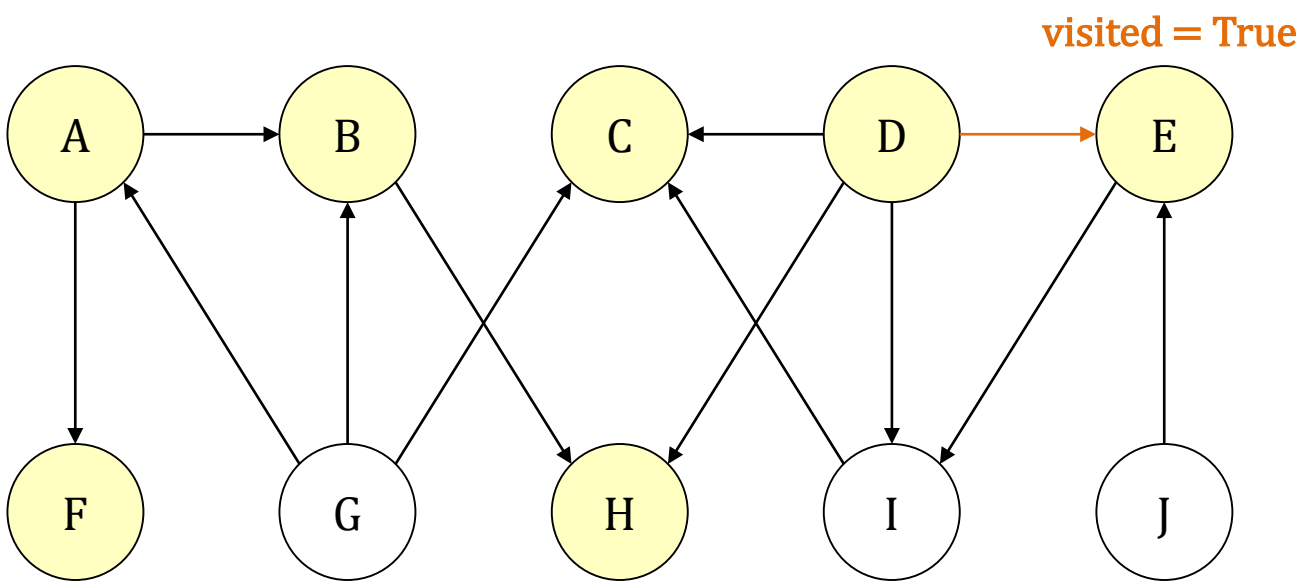
위상 정렬
Topological sort



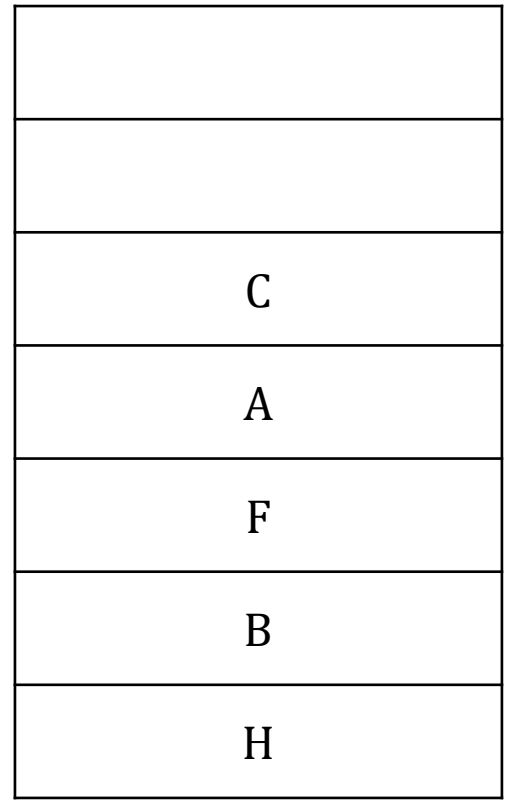
Stack



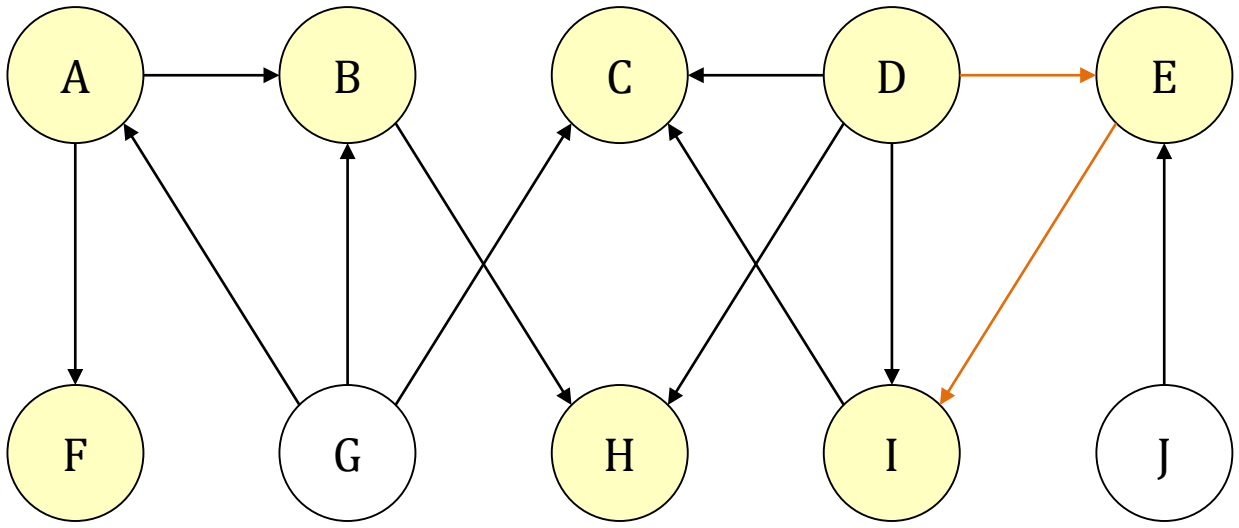
위상 정렬
Topological sort



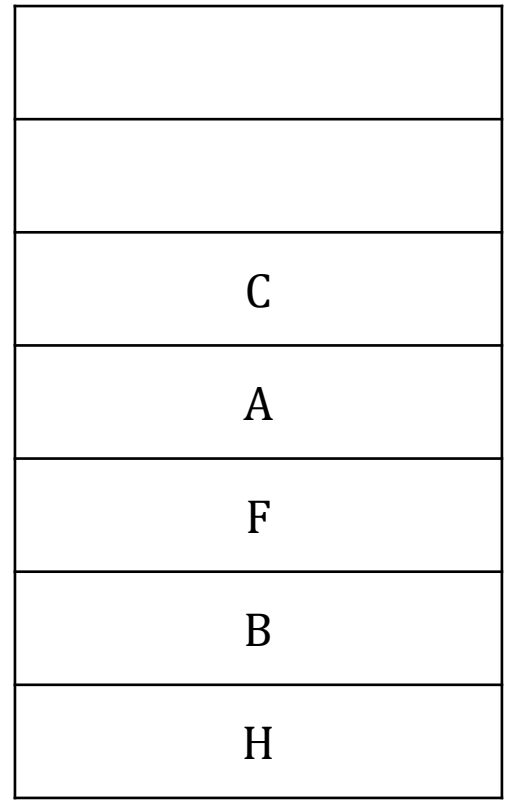
Stack



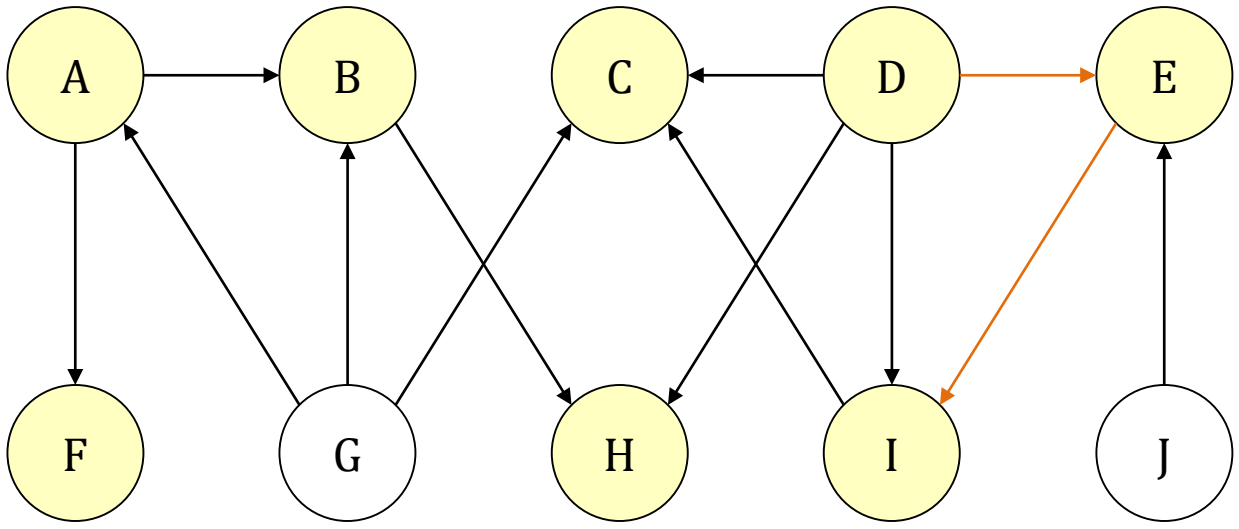
위상 정렬
Topological sort



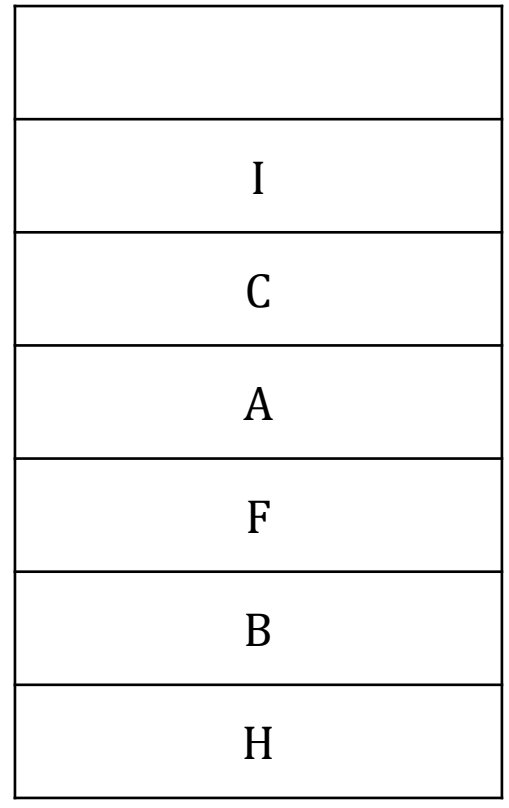
Stack



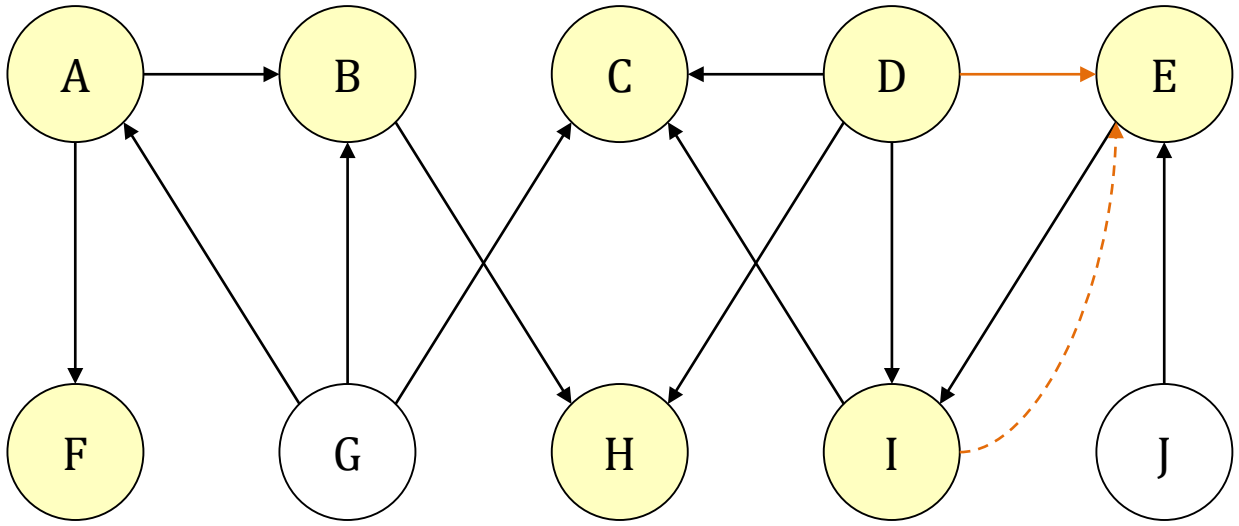
위상 정렬
Topological sort



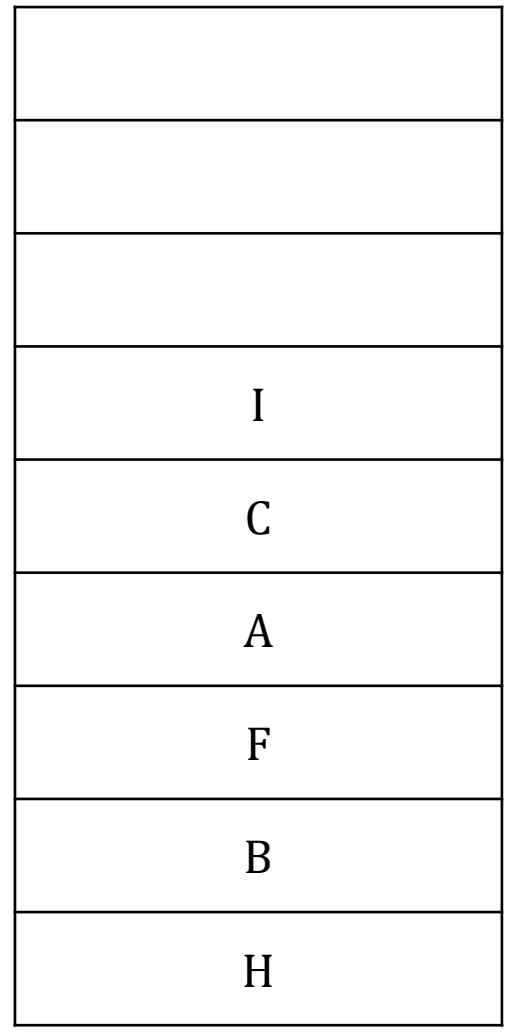
Stack



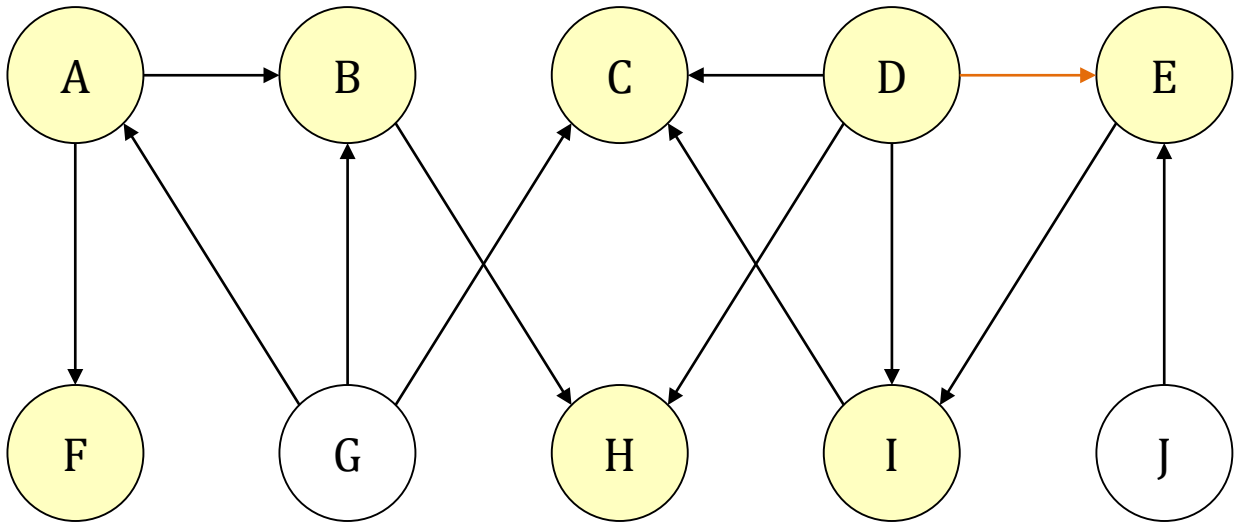
위상 정렬
Topological sort



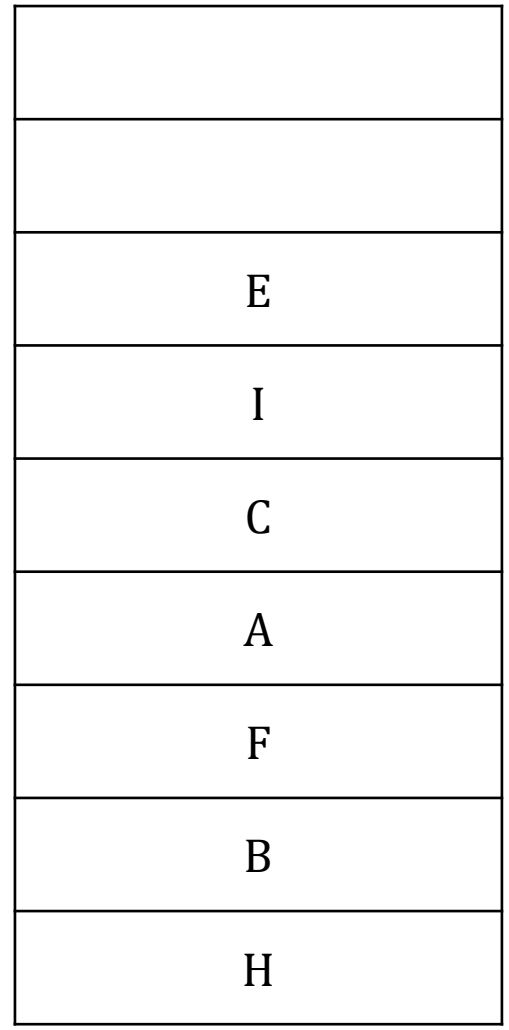
Stack



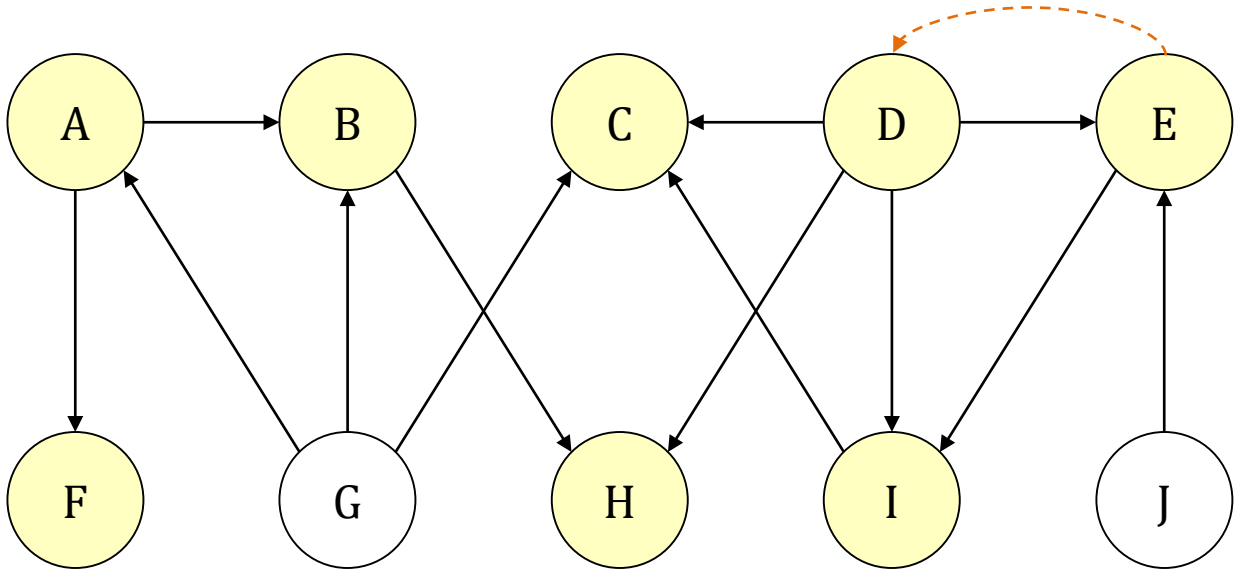
위상 정렬
Topological sort



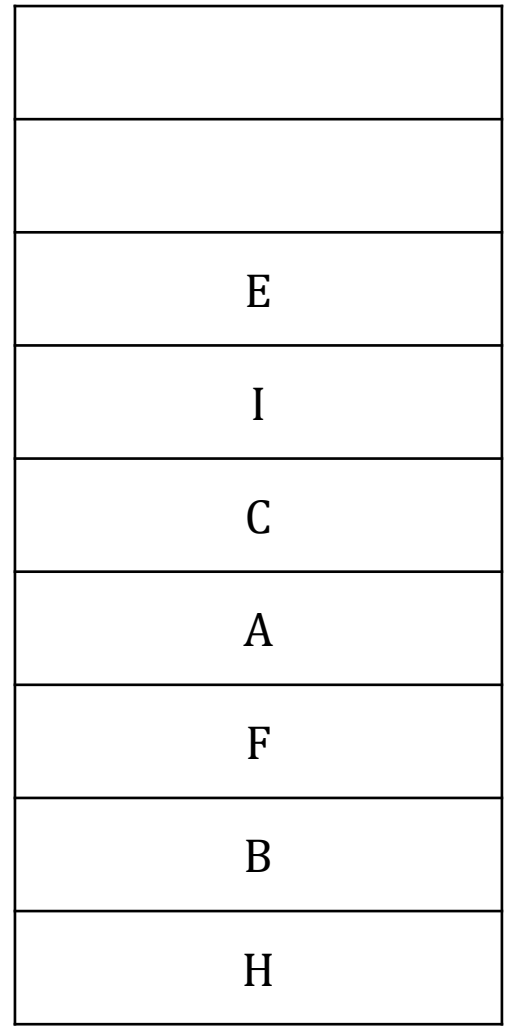
Stack



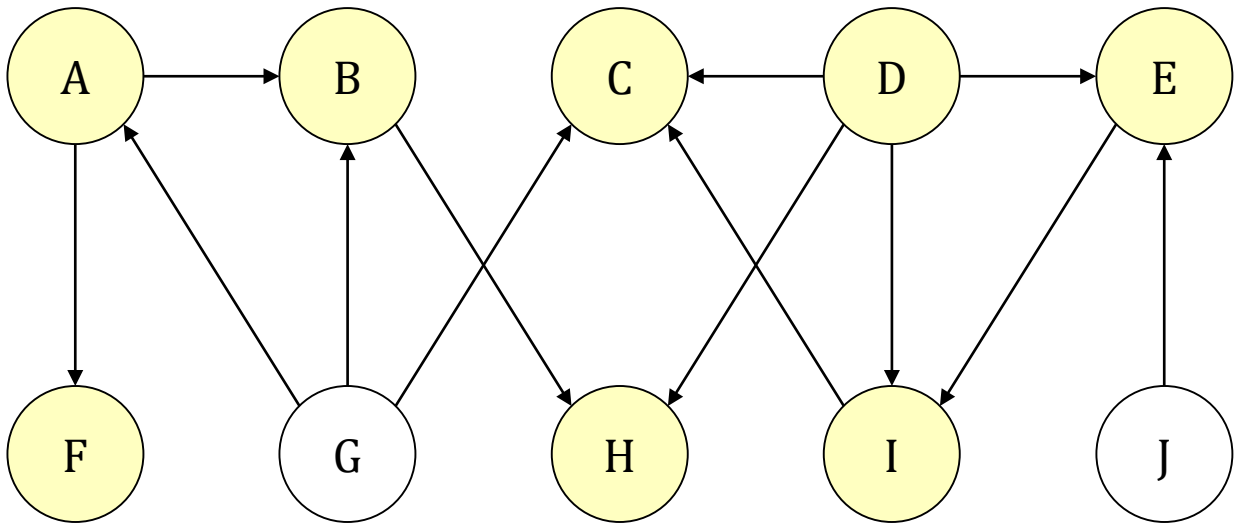
위상 정렬
Topological sort



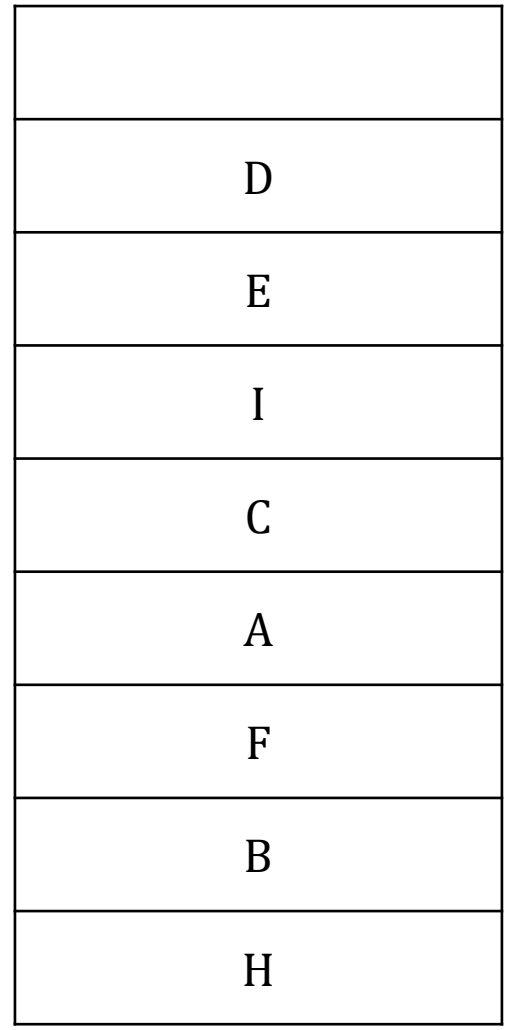
Stack



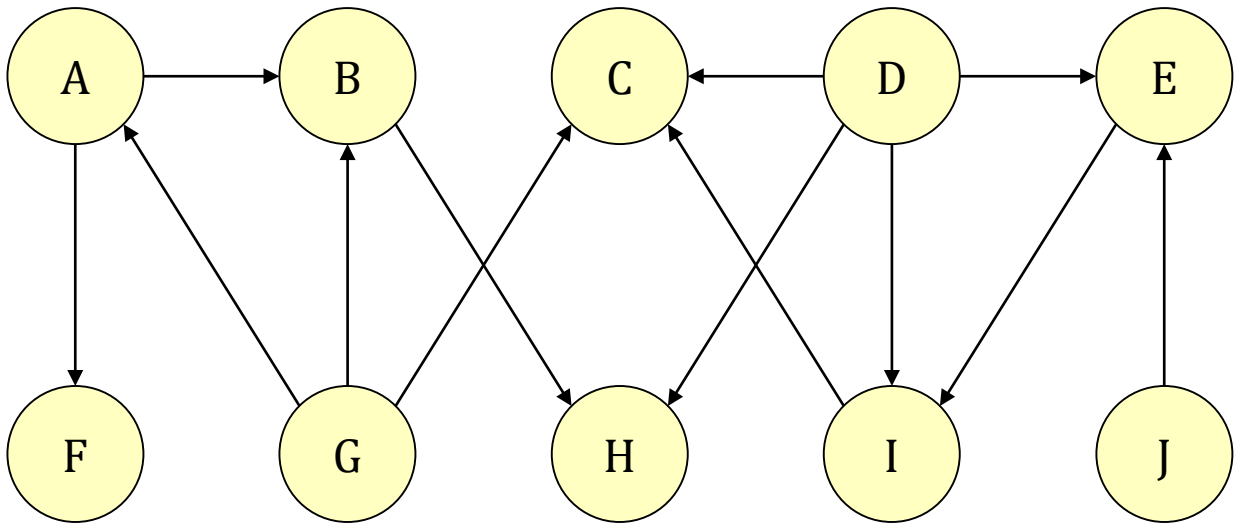
위상 정렬
Topological sort



Stack



위상 정렬
Topological sort



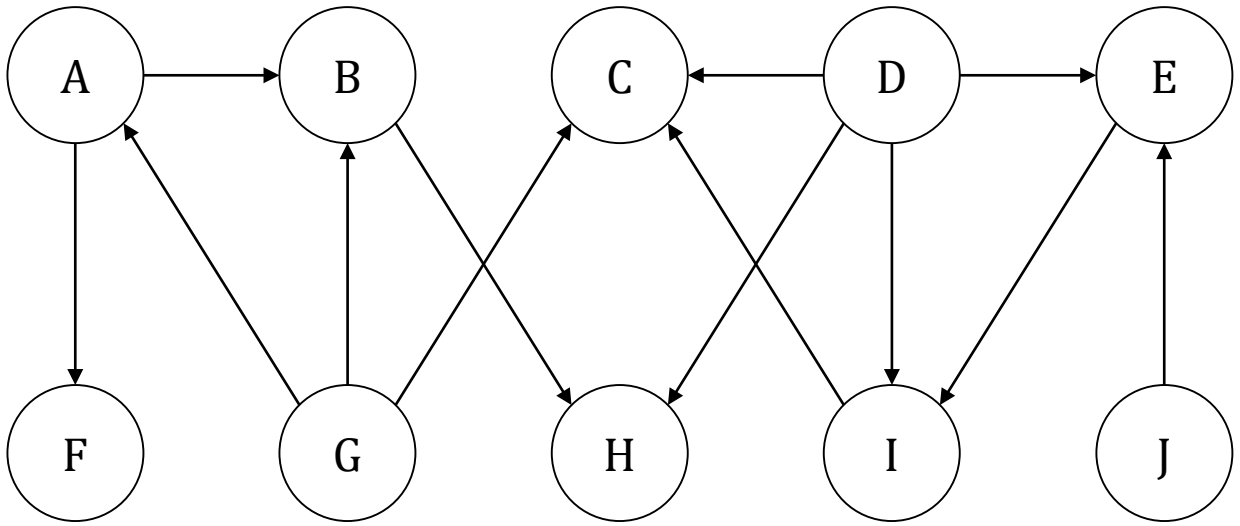
visited = True

visited = True

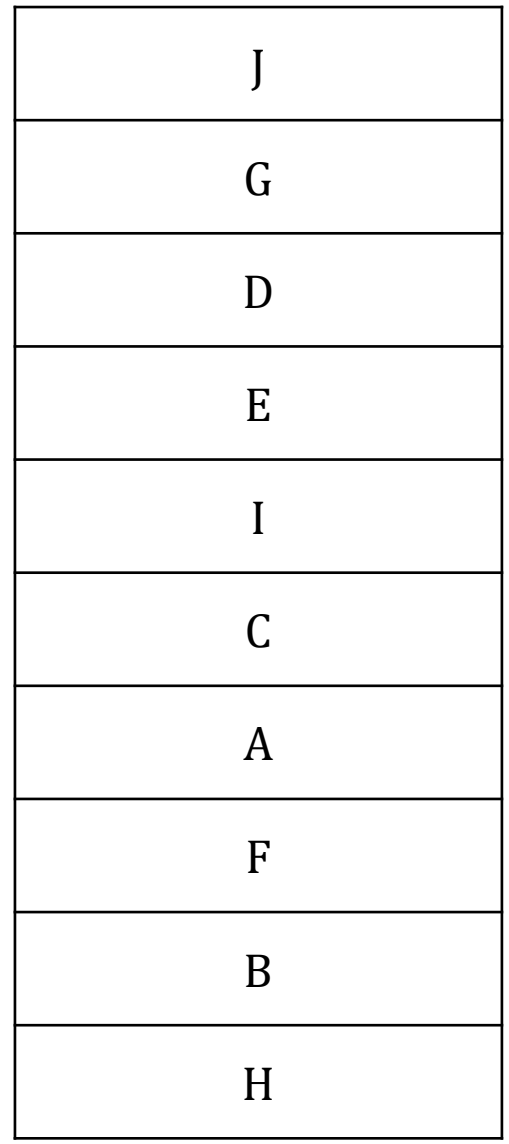
Stack

J
G
D
E
I
C
A
F
B
H

위상 정렬
Topological sort



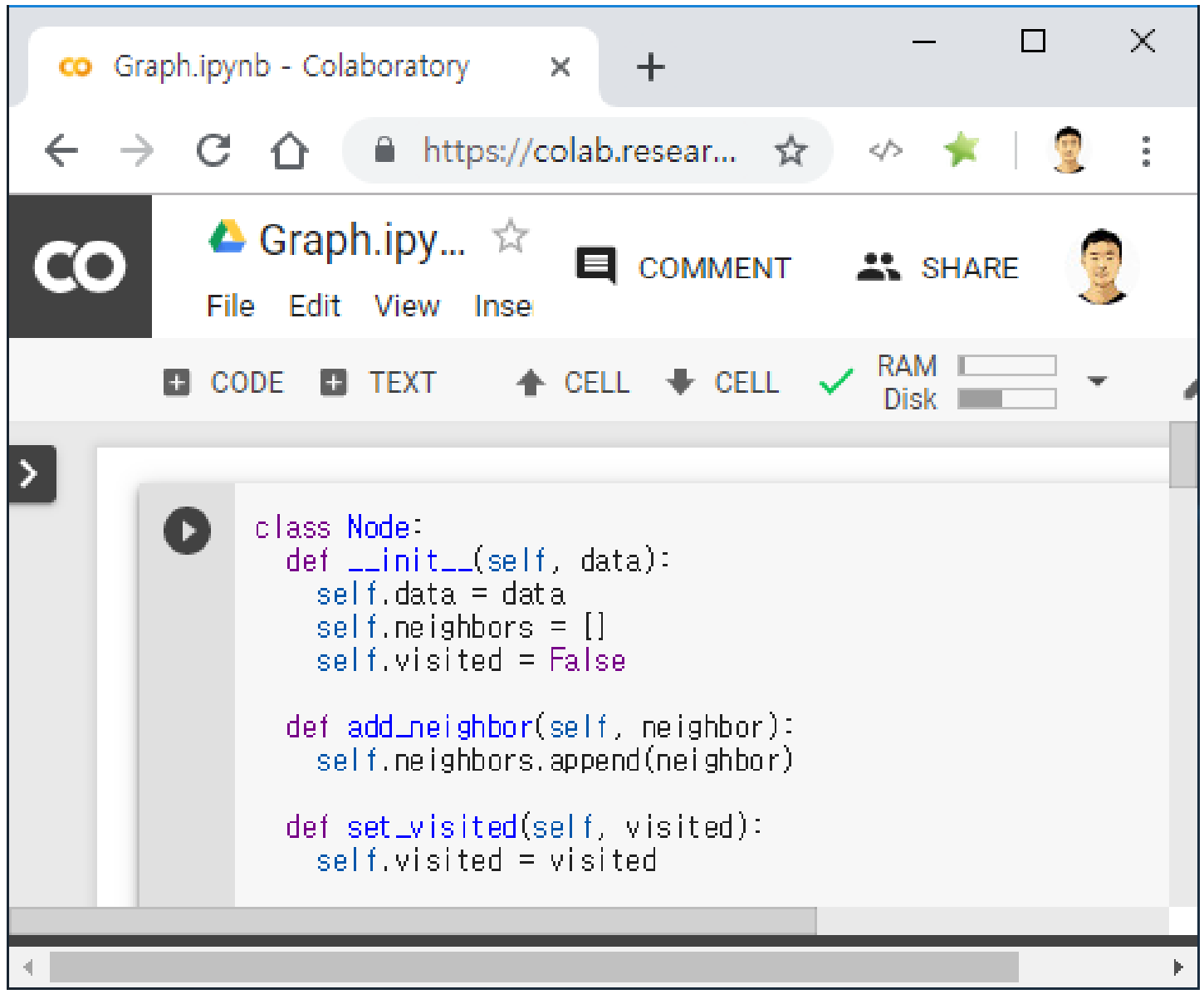
Stack



위상 정렬 J G D E I C A F B H

실습 1. Implementation of topological sort

- e-강의동 > 컴퓨터데이터구조 > 과제 > [실습 1] Implementation of topological sort
  - Graph.ipynb / Graph.py 다운로드 및 오픈



실습 1. Implementation of topological sort

- Node 클래스

```
class Node:
  def __init__(self, data):
    self.data = data
    self.neighbors = []
    self.visited = False

  def add_neighbor(self, neighbor):
    self.neighbors.append(neighbor)

  def set_visited(self, visited):
    self.visited = visited

  def get_data(self):
    return self.data

  def get_neighbors(self):
    return self.neighbors

  def get_visited(self):
    return self.visited
```

실습 1. Implementation of topological sort

- Graph 클래스

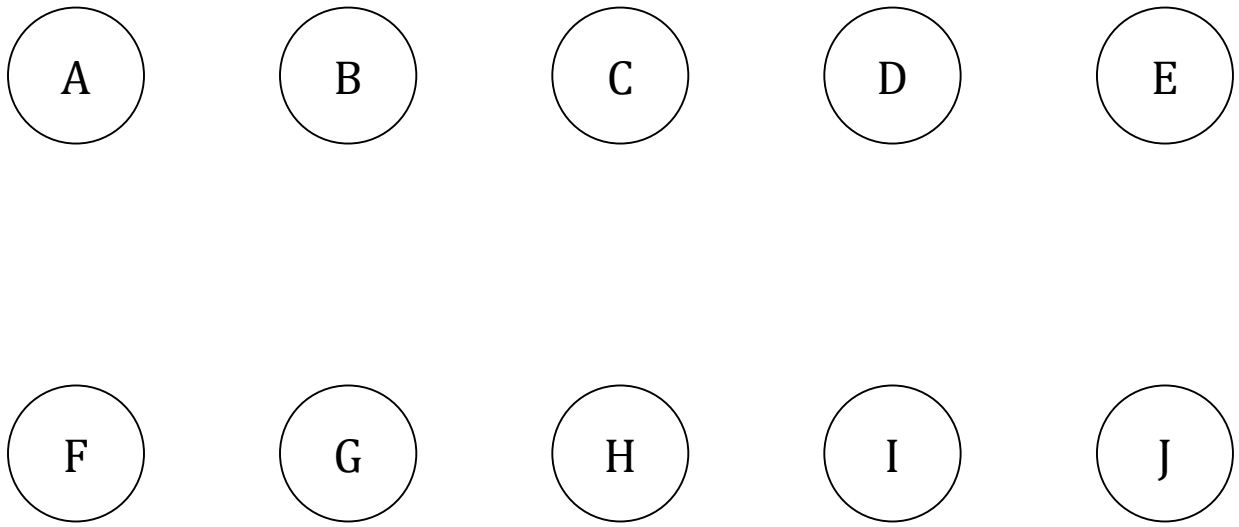
```
class Graph:
  def __init__(self):
    self.nodes = []

  def add_node(self, node):
    self.nodes.append(node)

  def topological_sort(self):
```

실습 1. Implementation of topological sort

- Body

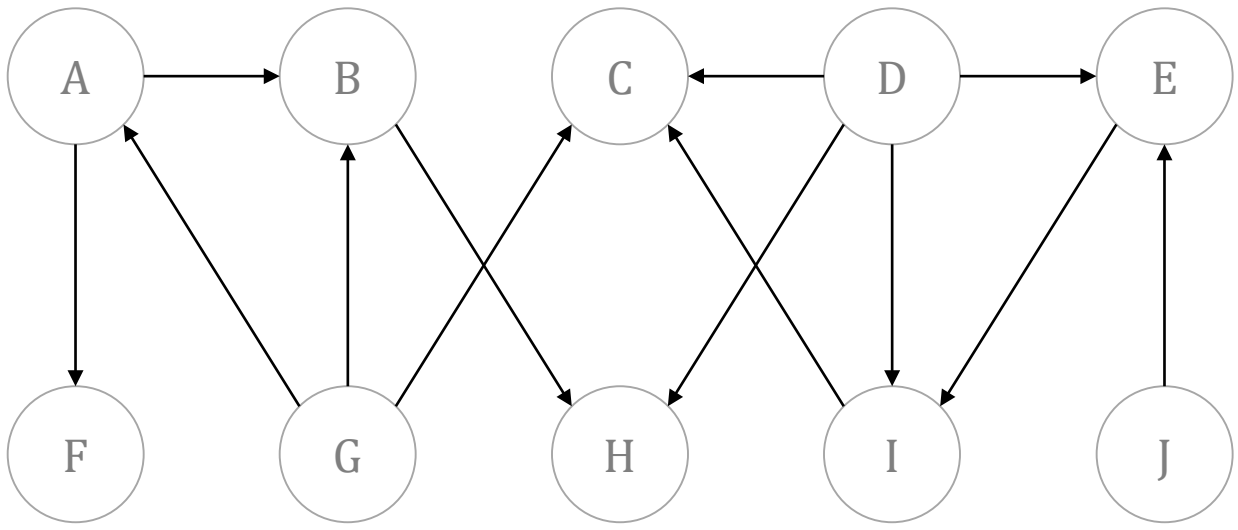



```
graph = Graph()

node_A = Node('A')
graph.add_node(node_A)
node_B = Node('B')
graph.add_node(node_B)
node_C = Node('C')
graph.add_node(node_C)
node_D = Node('D')
graph.add_node(node_D)
node_E = Node('E')
graph.add_node(node_E)
node_F = Node('F')
graph.add_node(node_F)
node_G = Node('G')
graph.add_node(node_G)
node_H = Node('H')
graph.add_node(node_H)
node_I = Node('I')
graph.add_node(node_I)
node_J = Node('J')
graph.add_node(node_J)
```

실습 1. Implementation of topological sort

- Body




```
node_A.add_neighbor(node_B)
node_A.add_neighbor(node_F)

node_B.add_neighbor(node_H)

node_D.add_neighbor(node_C)
node_D.add_neighbor(node_E)
node_D.add_neighbor(node_I)

node_E.add_neighbor(node_I)

node_G.add_neighbor(node_A)
node_G.add_neighbor(node_B)
node_G.add_neighbor(node_C)

node_I.add_neighbor(node_C)

node_J.add_neighbor(node_E)
```

실습 1. Implementation of topological sort

- Question, answer sheet and answer

Question

```
node_I.add_neighbor(node_C)

node_J.add_neighbor(node_E)

graph.topological_sort()
```

Answer sheet

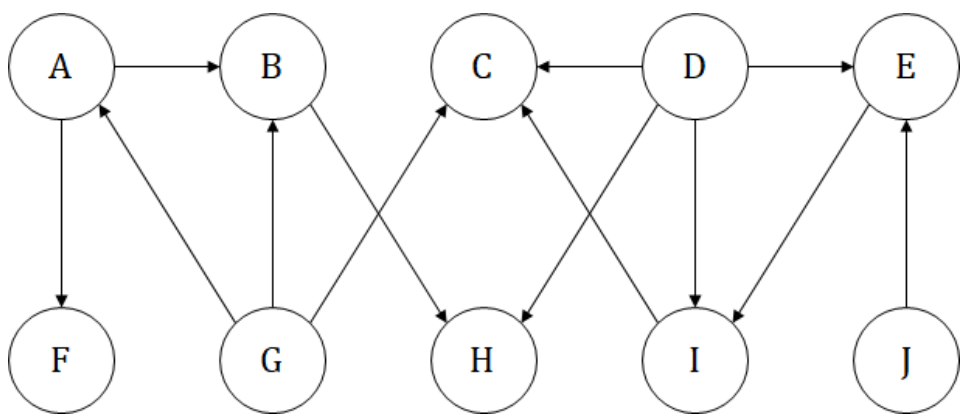
```
class Graph:
  def __init__(self):
    self.nodes = []

  def add_node(self, node):
    self.nodes.append(node)

  def topological_sort(self):
```

Answer

```
J
G
D
E
I
C
A
F
B
H
```



위상 정렬 J G D E I C A F B H

실습 1. Implementation of topological sort

- Question, answer sheet and answer

Question

```
node_I.add_neighbor(node_C)

node_J.add_neighbor(node_E)

graph.t
```

Answer sheet

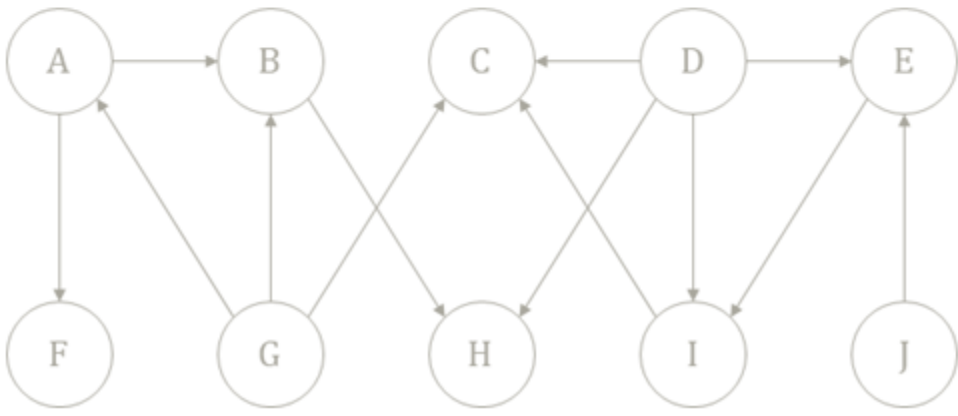
```
class Graph:
  def __init__(self):
    self.nodes = []

  def topological_sort(self):
```

Answer

J
G
D
E
I
C
A
F
B
H

제출 마감: 2019.05.28 오후 11:59



위상 정렬

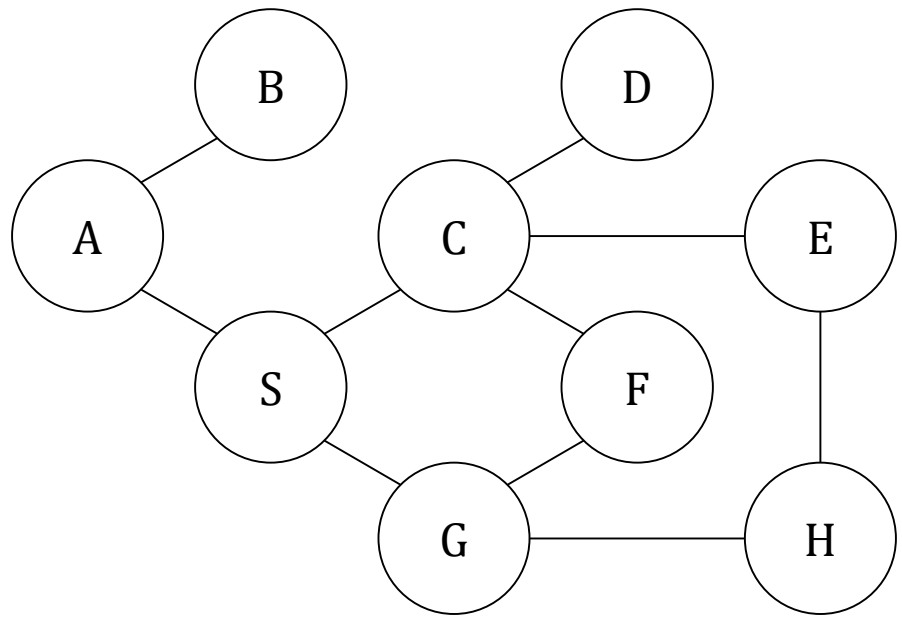
J G D E I C A F B H

최소 신장 트리

MST: Minimum Spamming Tree

신장 트리
Spanning

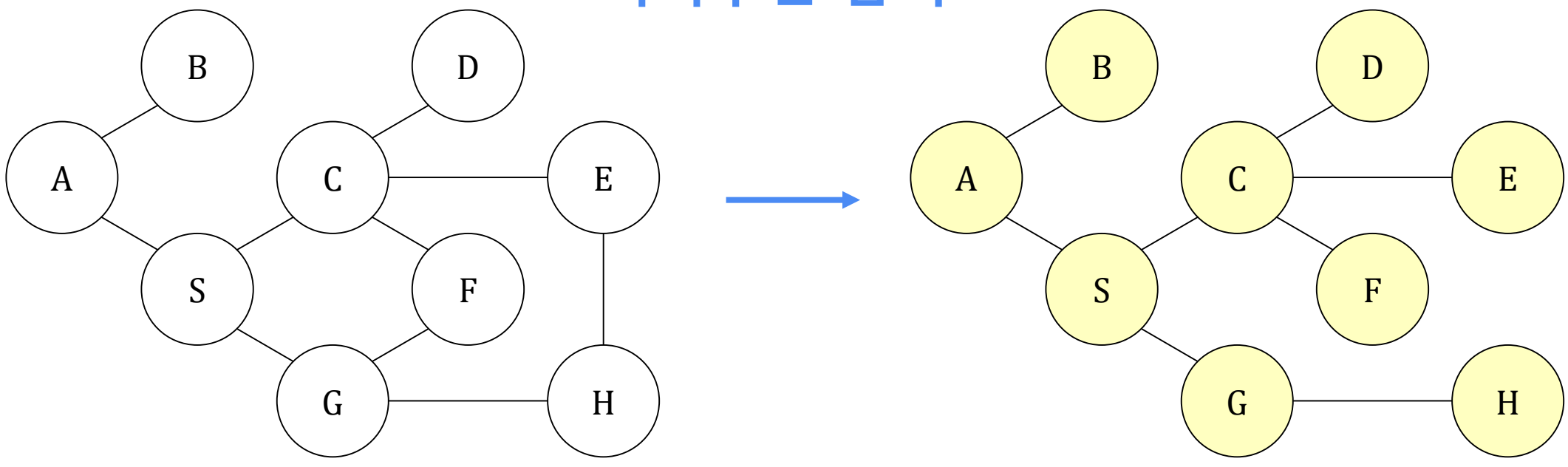
- **신장 트리Spanning tree:** 그래프의 모든 정점들을 싸이클 없이 연결하는 부분 그래프



신장 트리
Spanning

- **신장 트리**Spanning tree: 그래프의 모든 정점들을 싸이클 없이 연결하는 부분 그래프

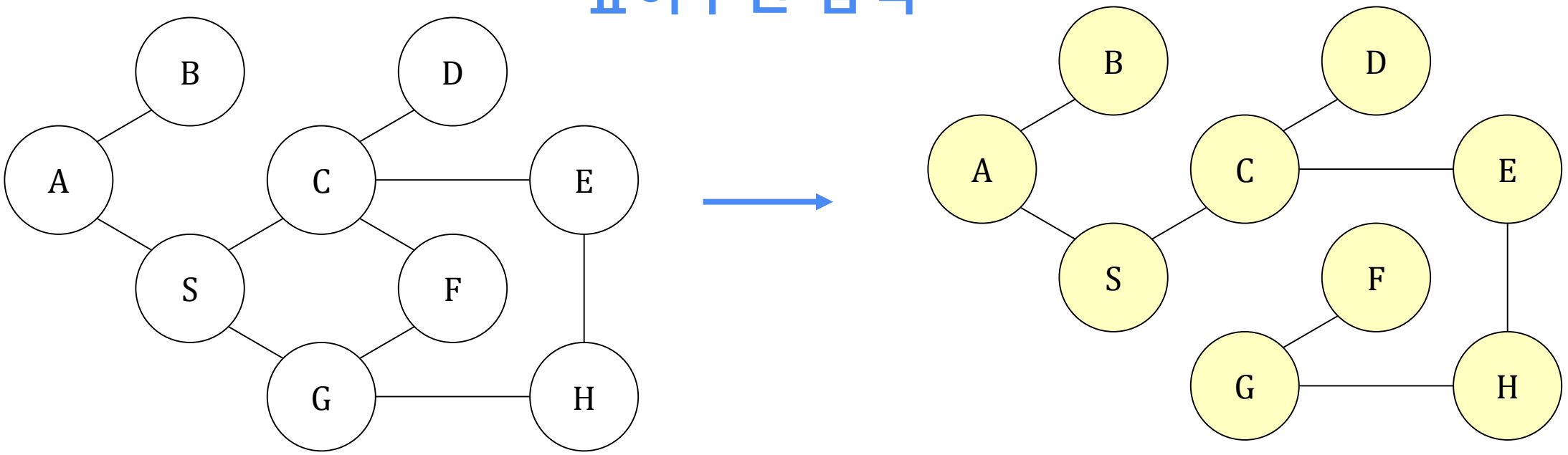
너비우선 탐색



신장 트리
Spanning

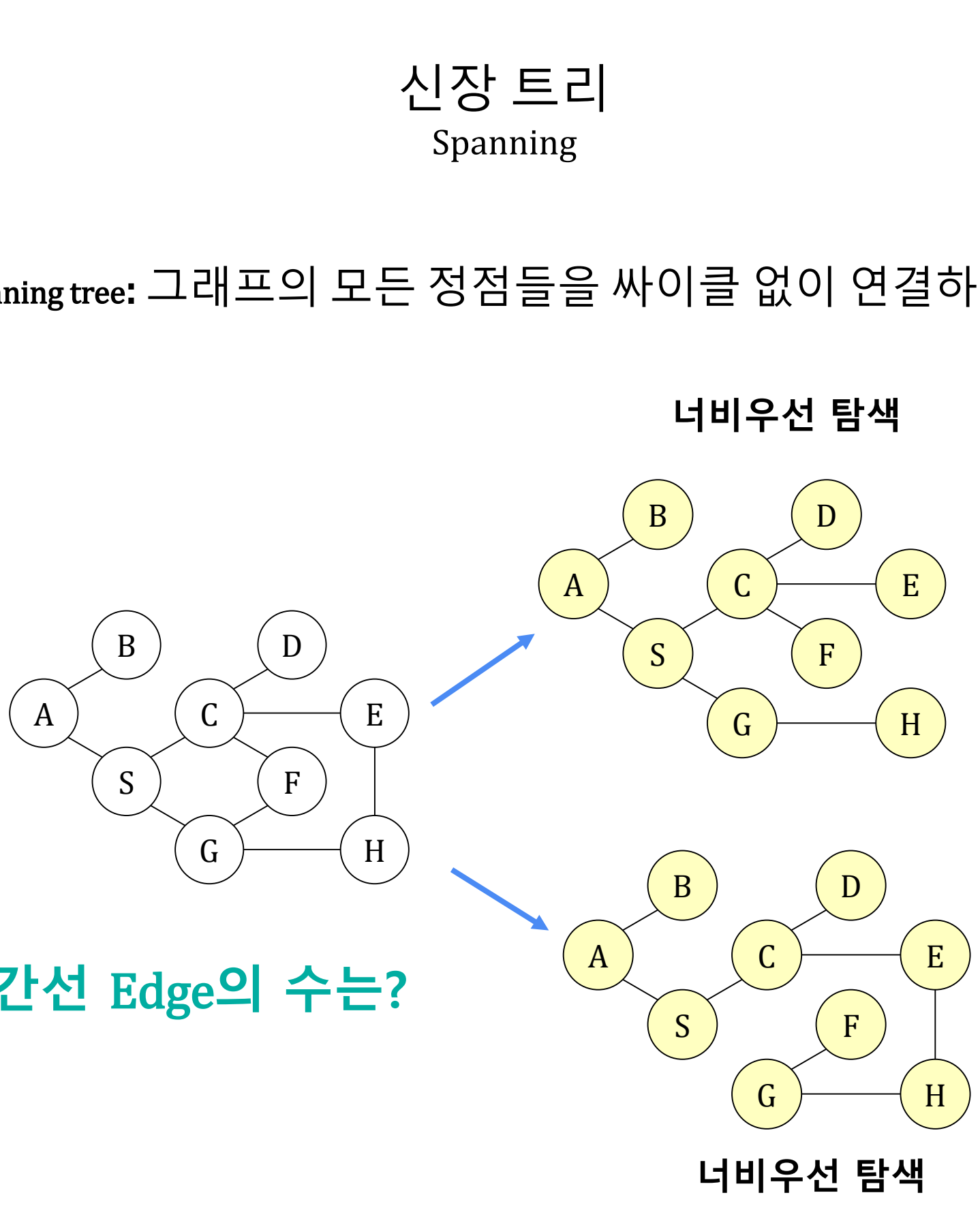
- **신장 트리**Spanning tree: 그래프의 모든 정점들을 싸이클 없이 연결하는 부분 그래프

깊이우선 탐색



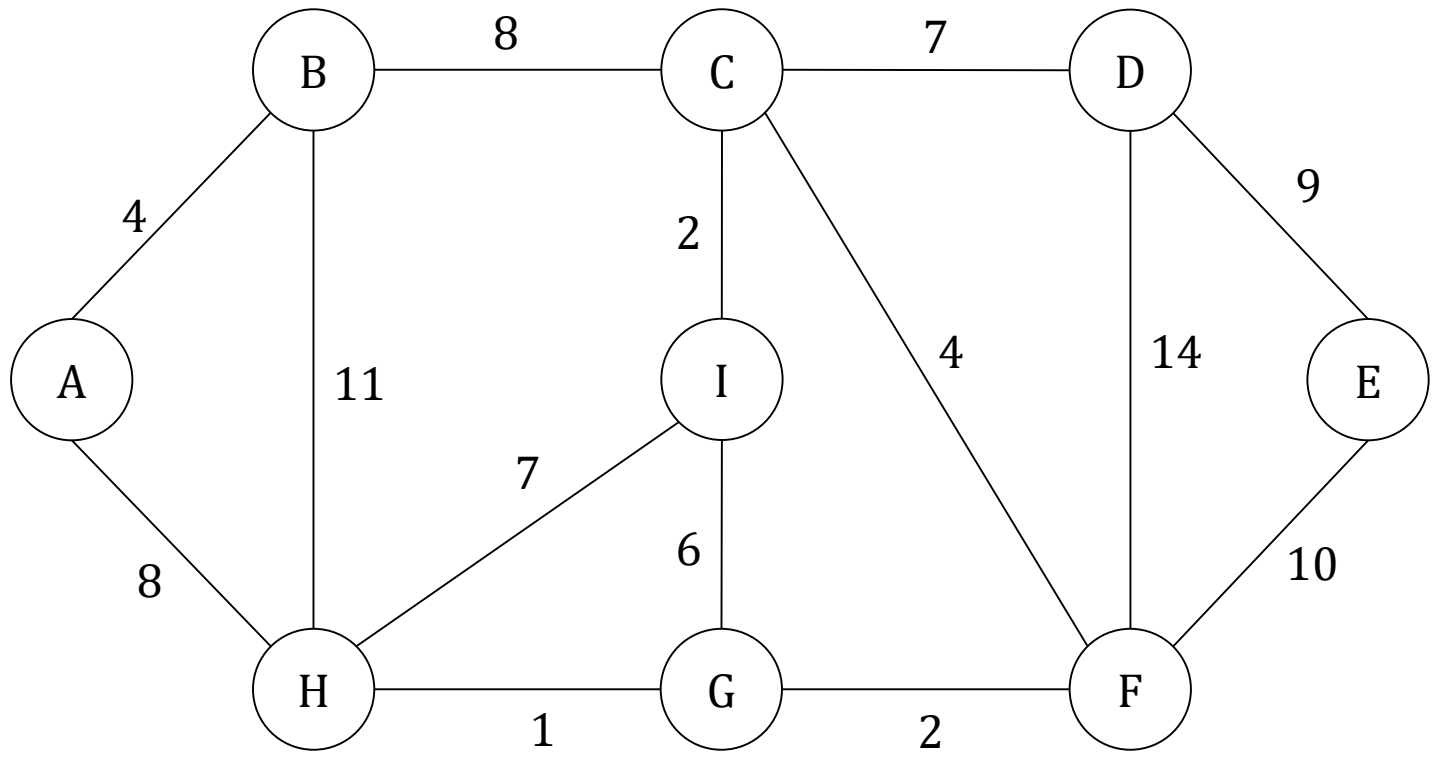
신장 트리
Spanning

- **신장 트리Spanning tree:** 그래프의 모든 정점들을 싸이클 없이 연결하는 부분 그래프



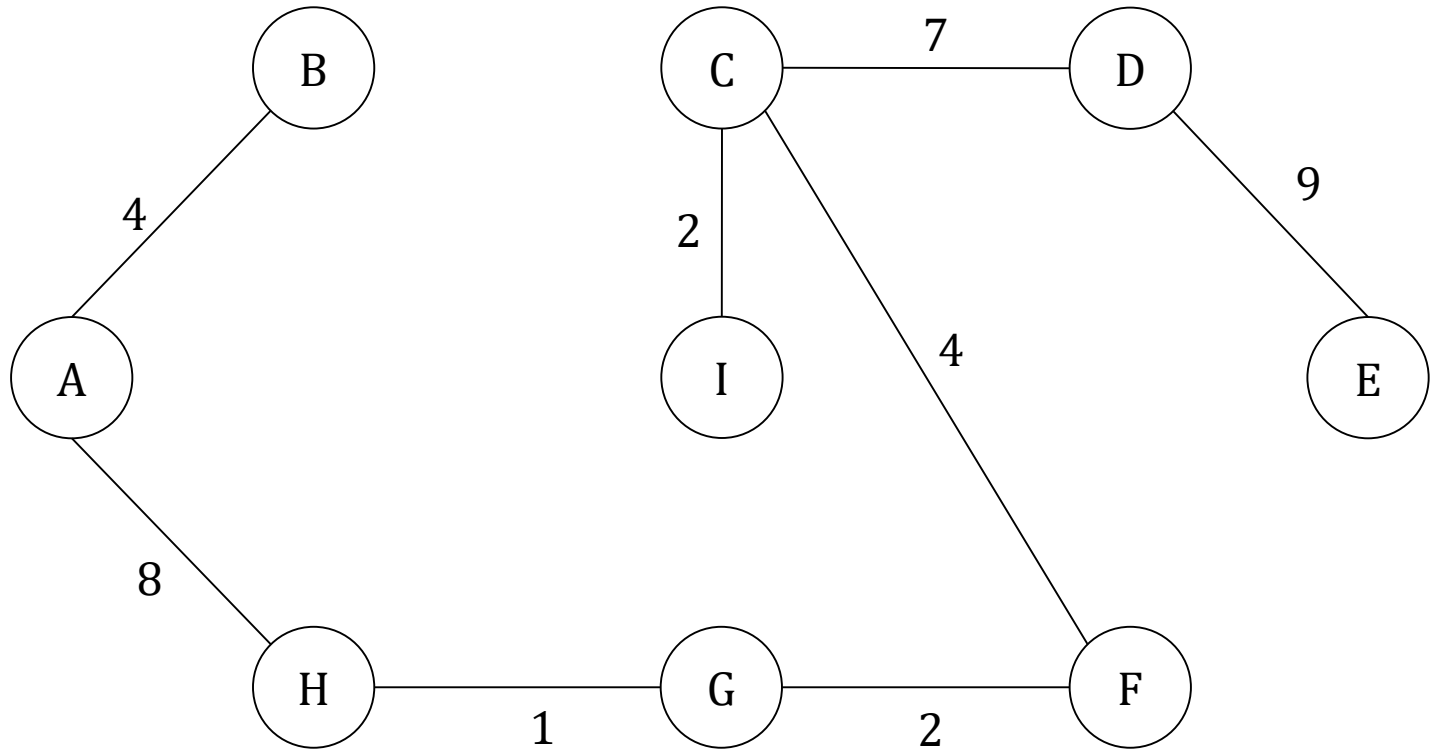
최소 신장 트리 추출 알고리즘

- 최소 신장 트리 MST: Minimum spanning tree
  - 대상: **가중치 그래프**



최소 신장 트리 추출 알고리즘

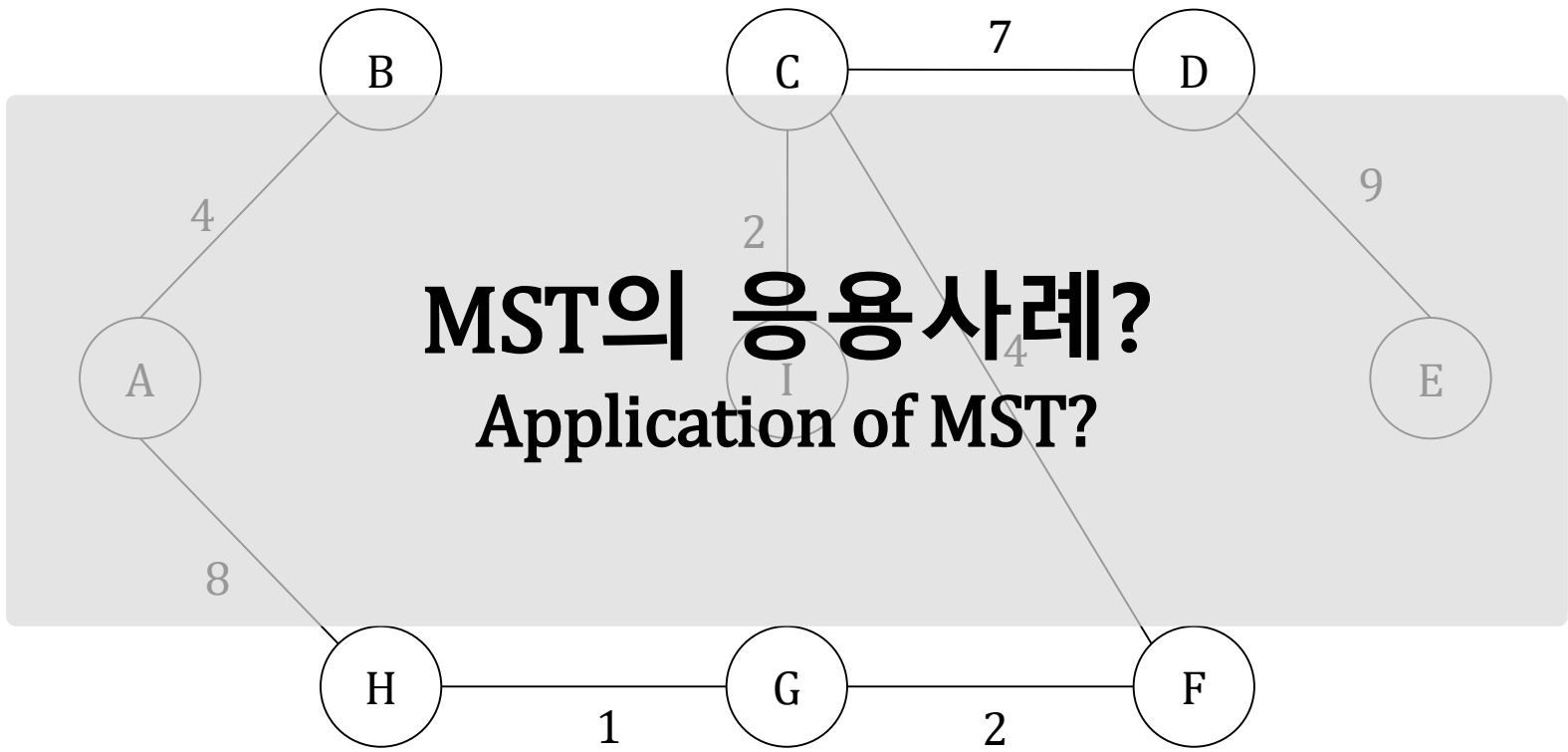
- 최소 신장 트리 MST: Minimum spanning tree
  - 대상: **가중치 그래프**



최소 신장 트리 신장 트리 중, **가중치의 합이 최소인 신장 트리**

최소 신장 트리 추출 알고리즘

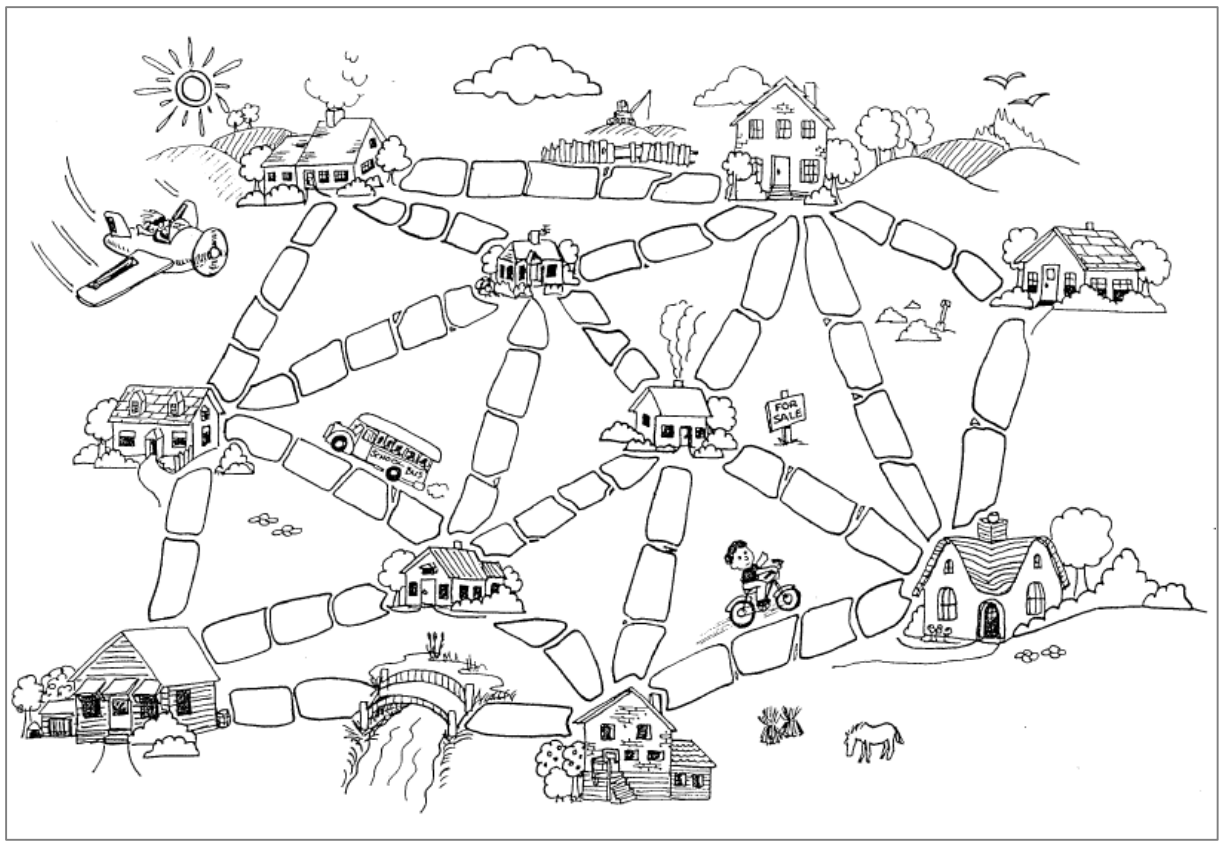
- 최소 신장 트리 MST: Minimum spanning tree
  - 대상: **가중치 그래프**

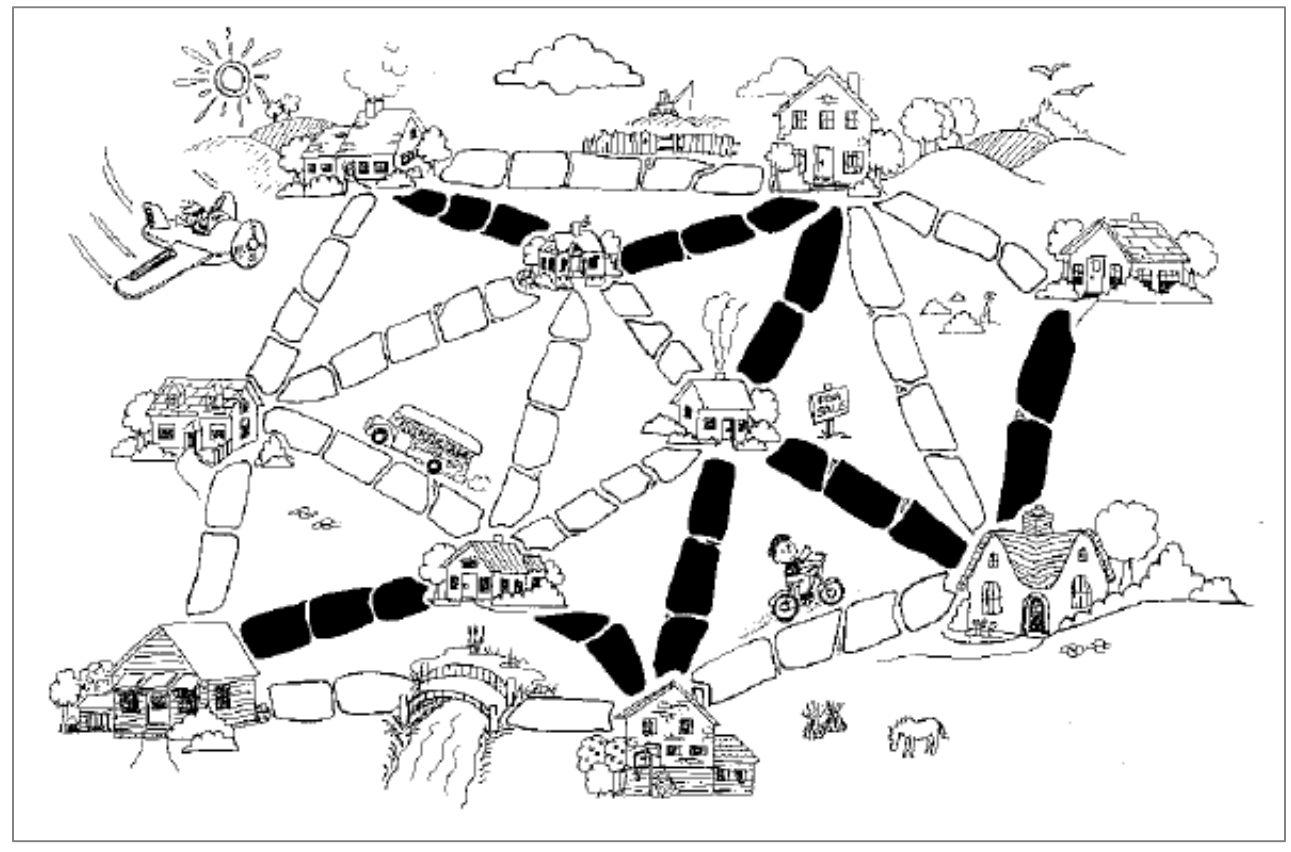


최소 신장 트리 신장 트리 중, **가중치의 합이 최소**인 신장 트리

최소 신장 트리 추출 알고리즘

- 최소 신장 트리 MST: Minimum spanning tree
  - 대상: **가중치 그래프**



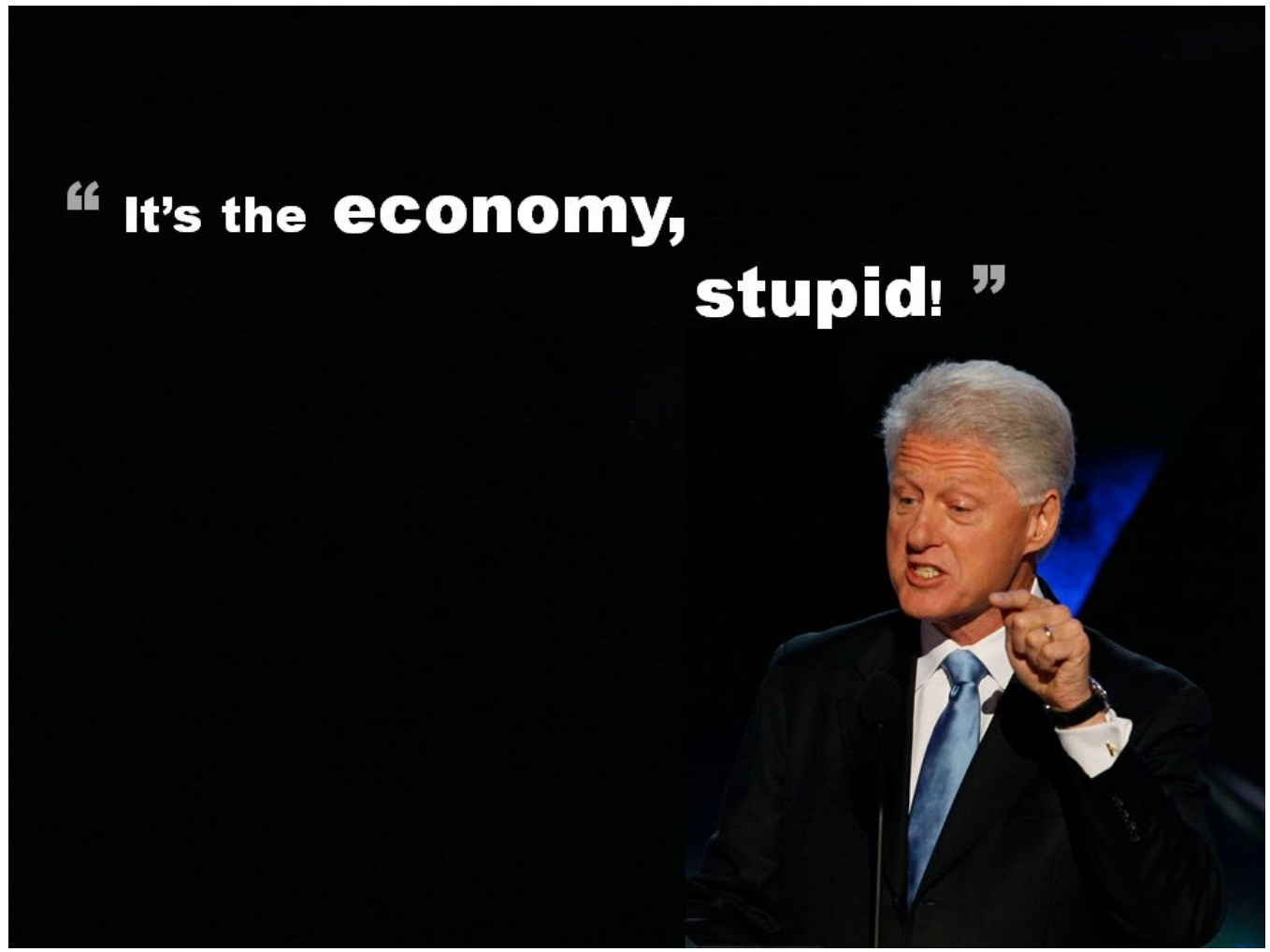


비포장 → 포장

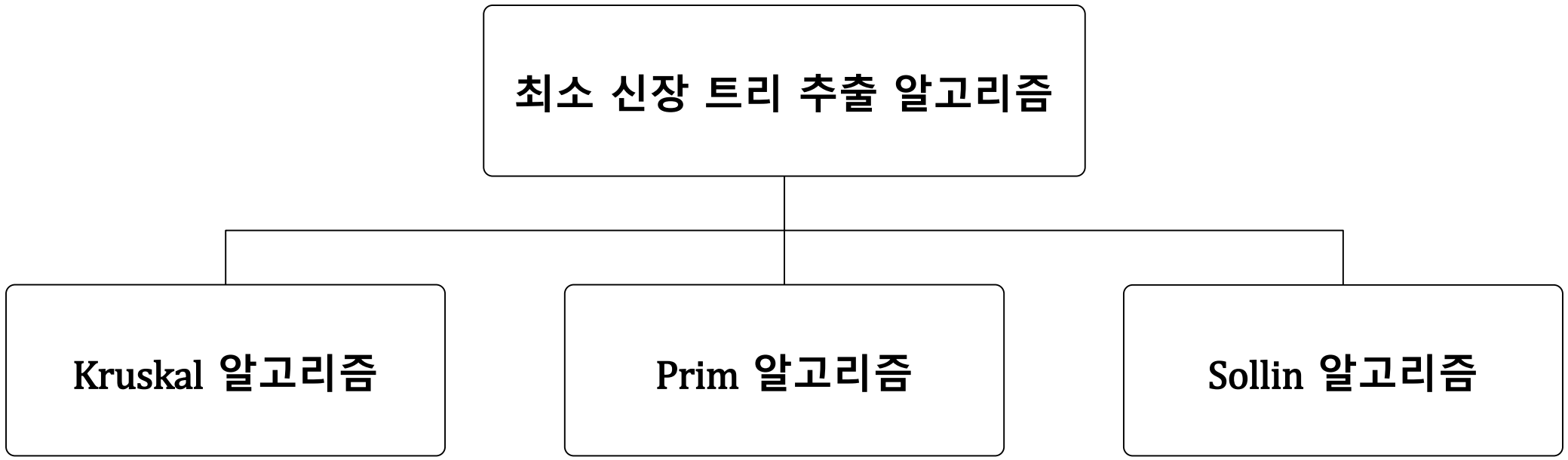
Muddy city and paving

최소 신장 트리 추출 알고리즘

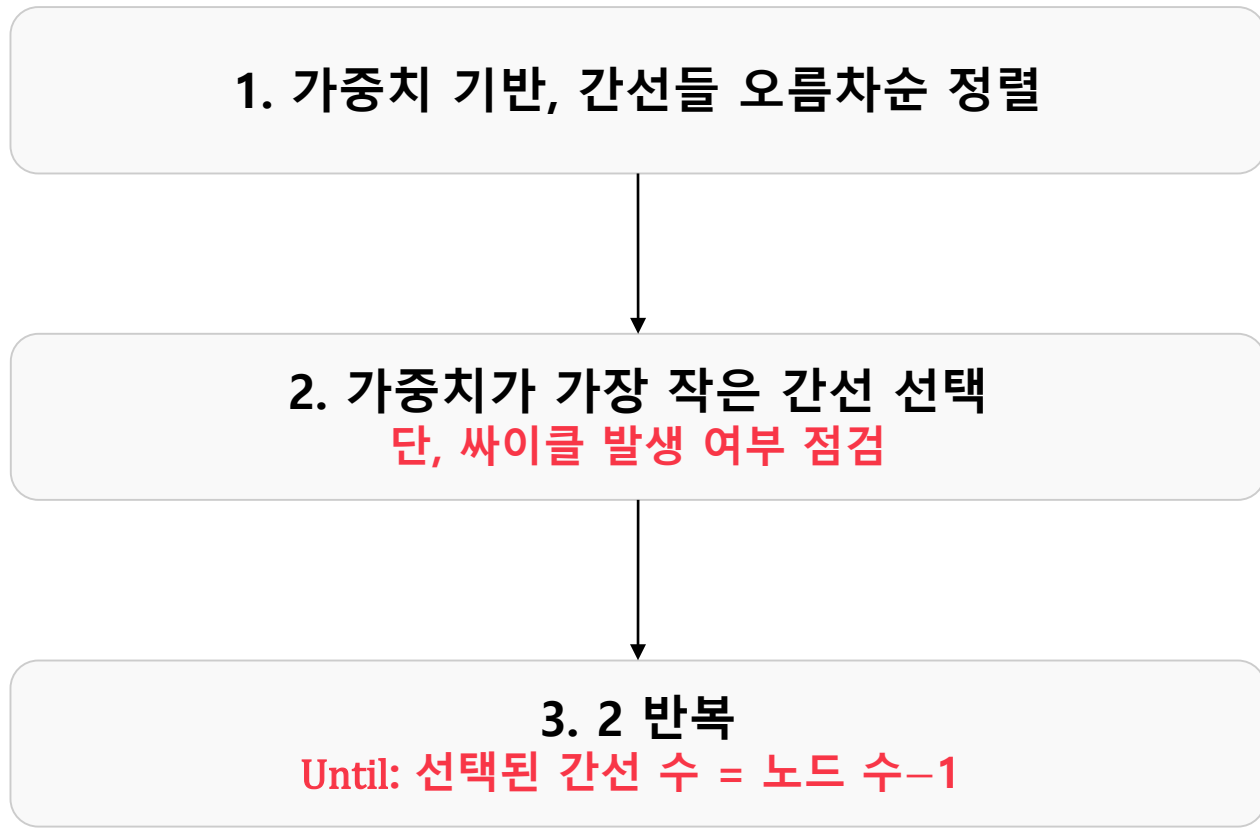
- 최소 신장 트리 MST: Minimum spanning tree
  - 대상: **가중치 그래프**



최소 신장 트리 추출 알고리즘

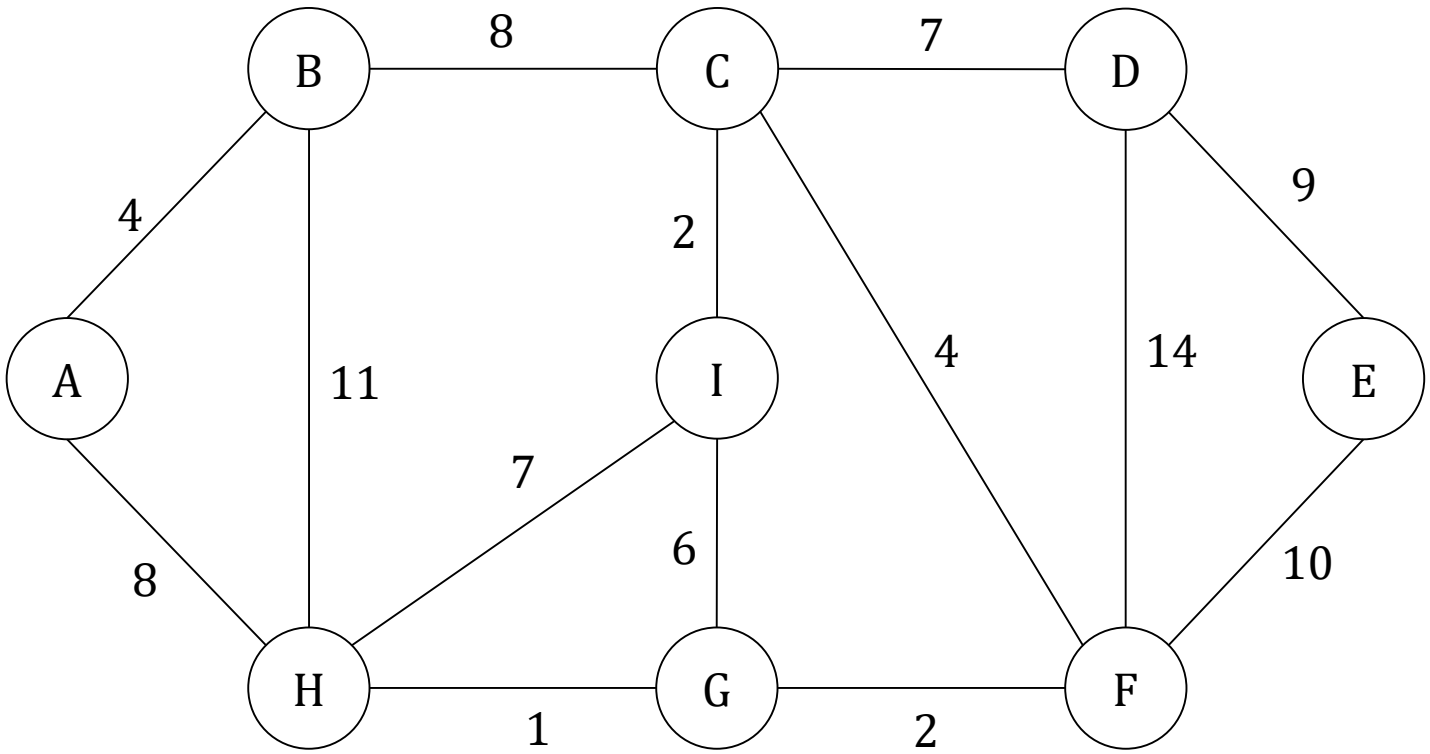


Kruskal 알고리즘



Kruskal 알고리즘

1. 가중치 기반, 간선들 오름차순 정렬

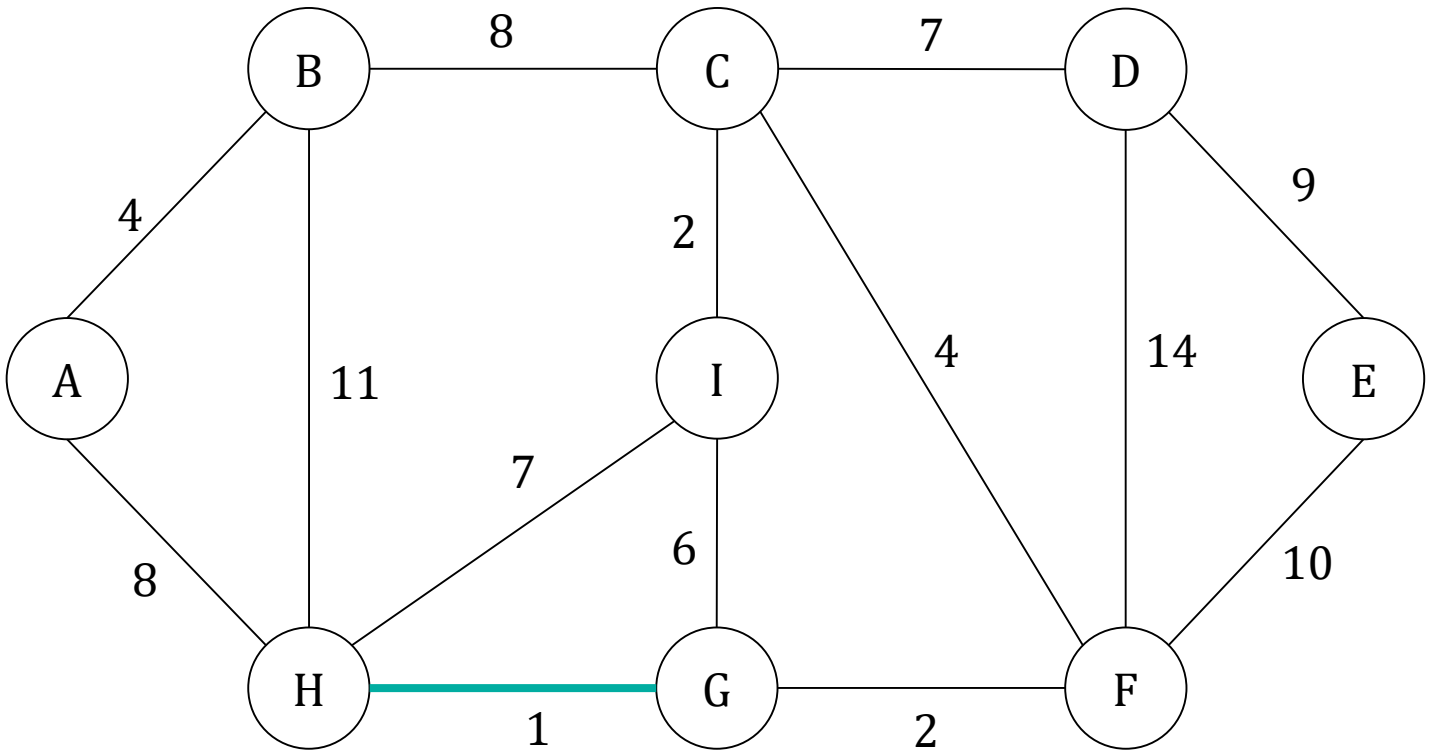


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	
(C, I)	2	
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

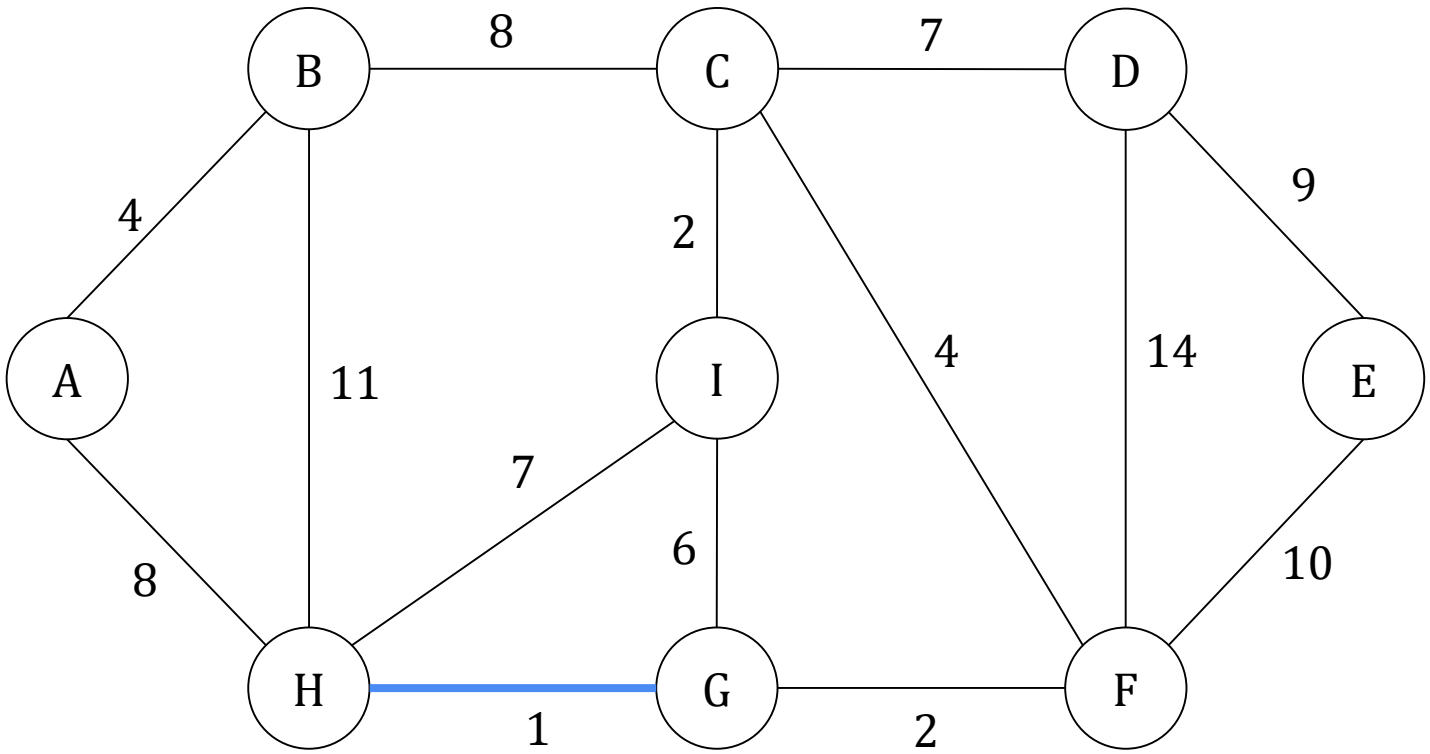


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	
(C, I)	2	
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

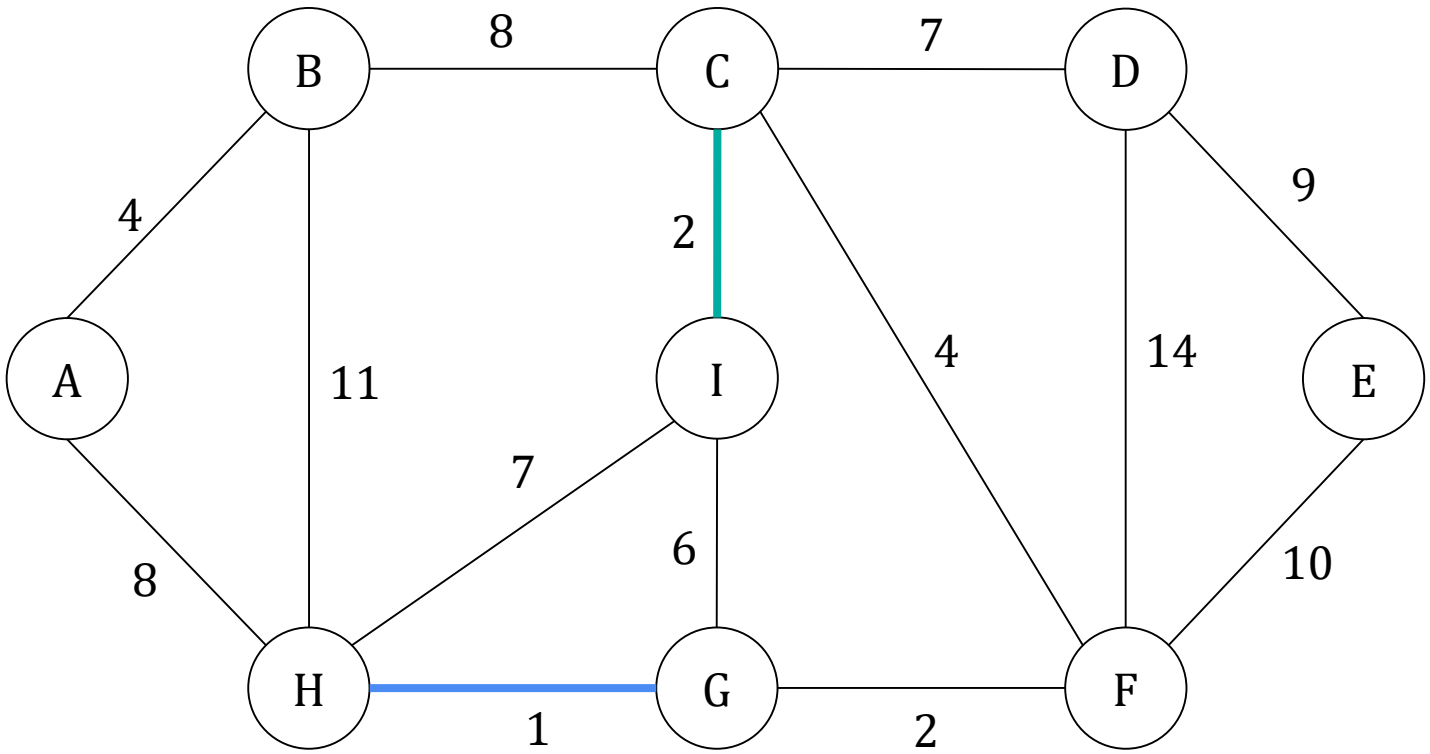


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

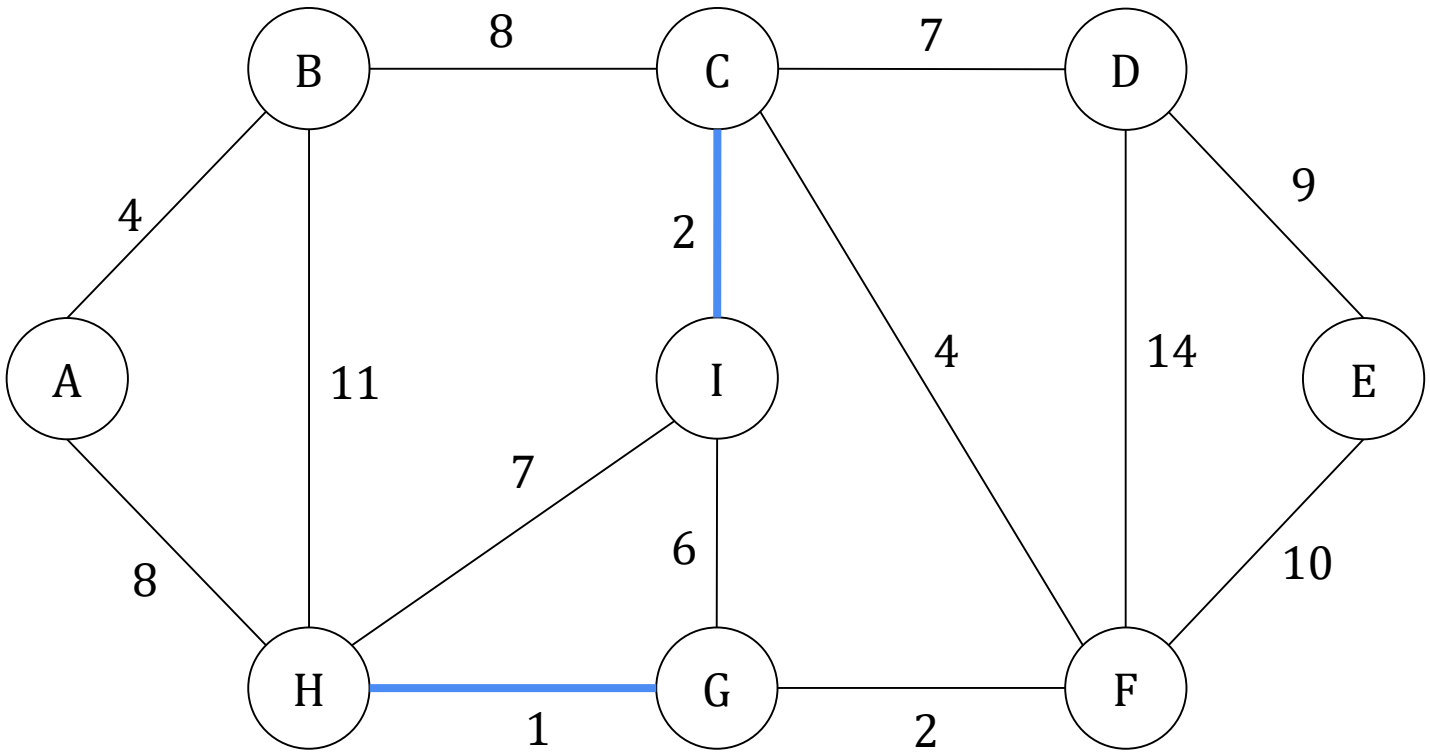


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

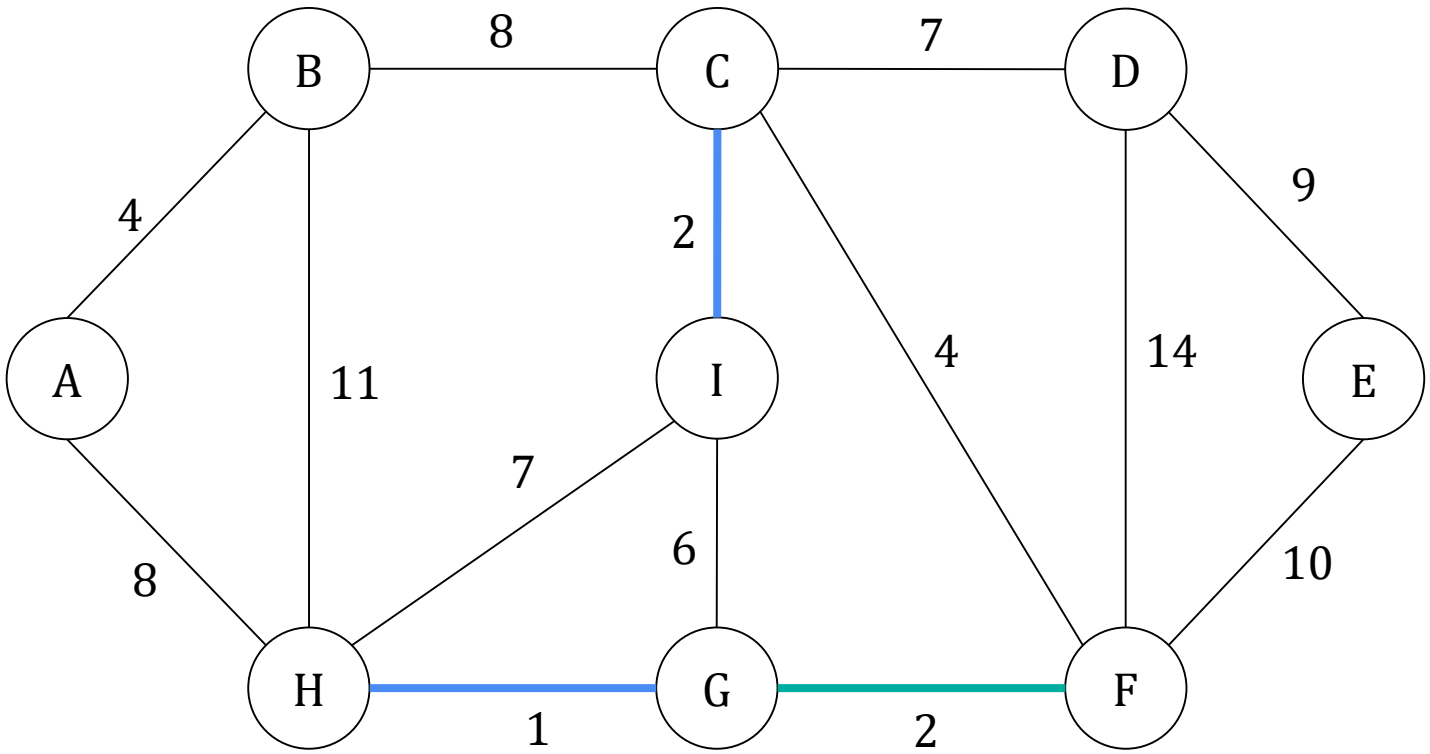


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

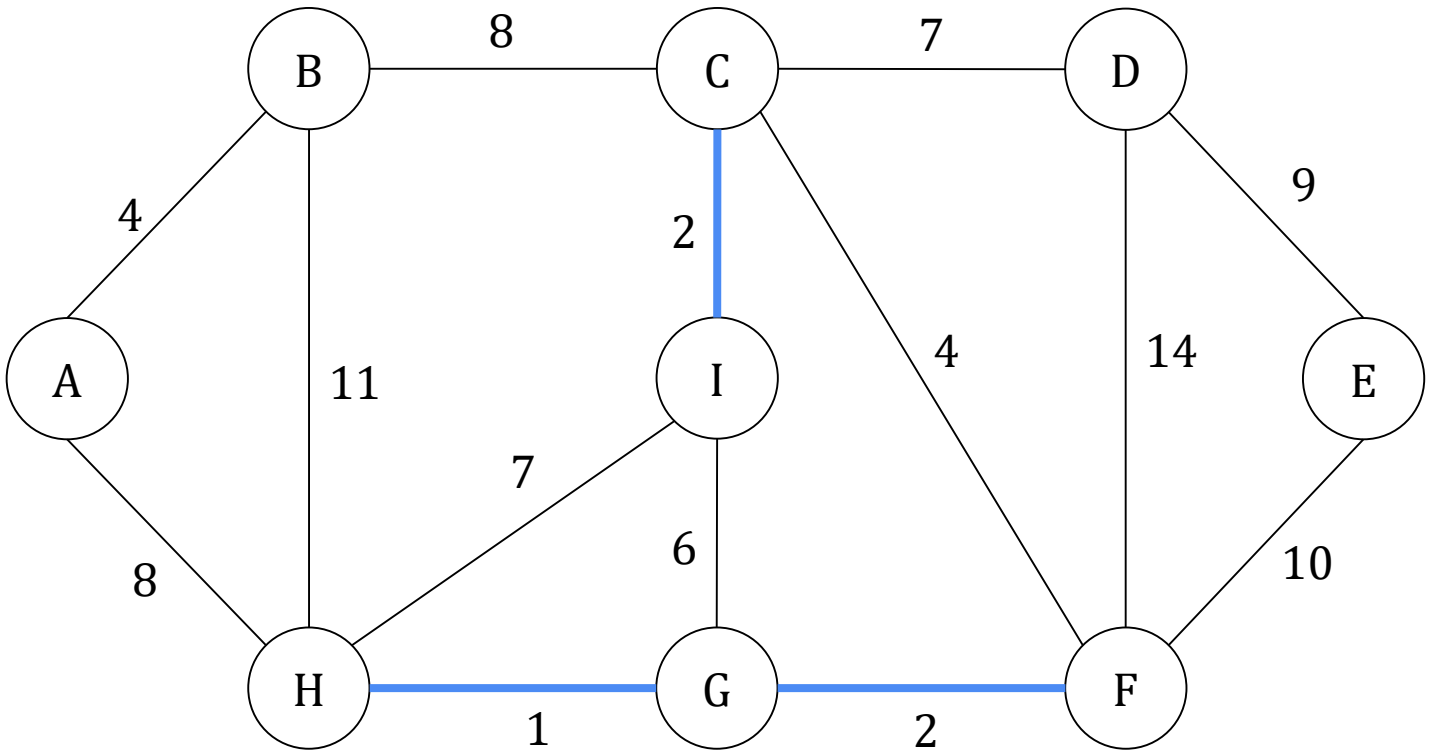


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

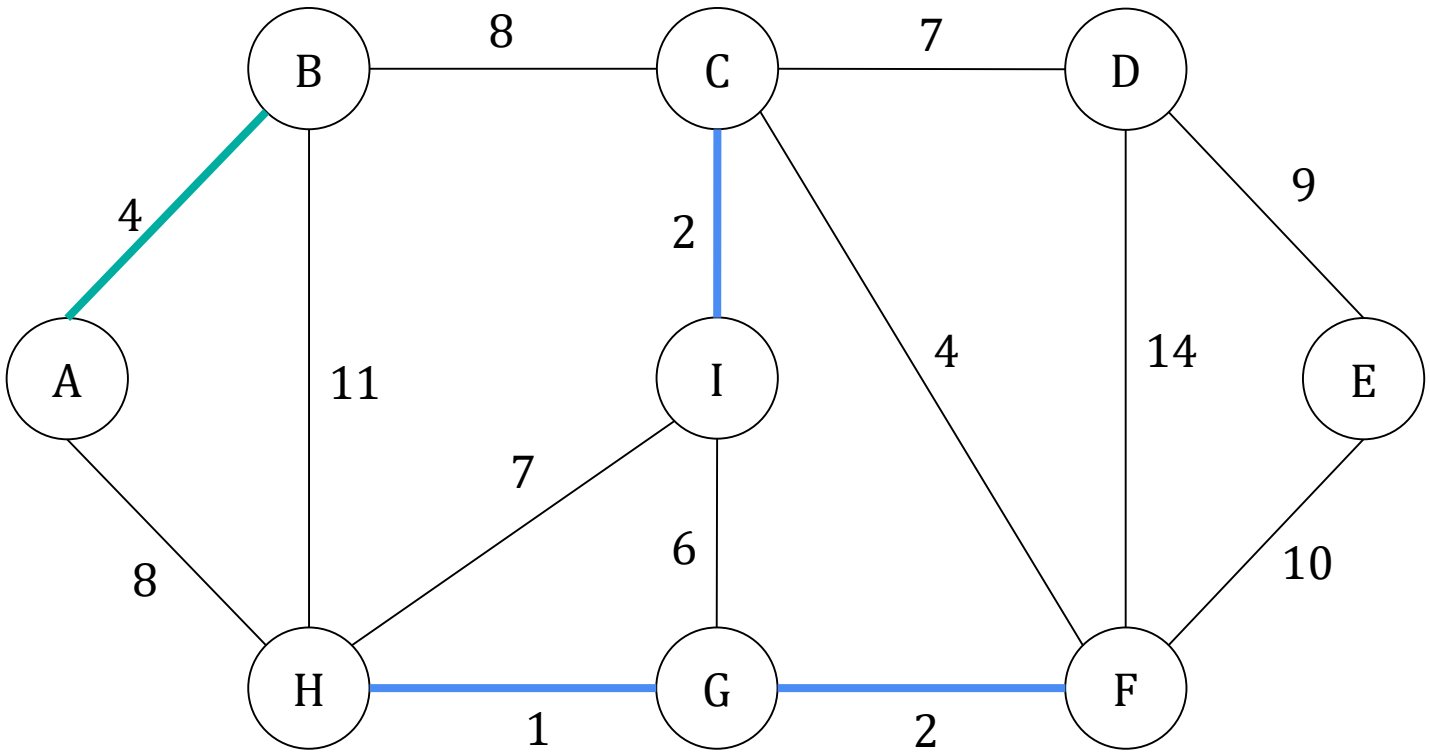


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

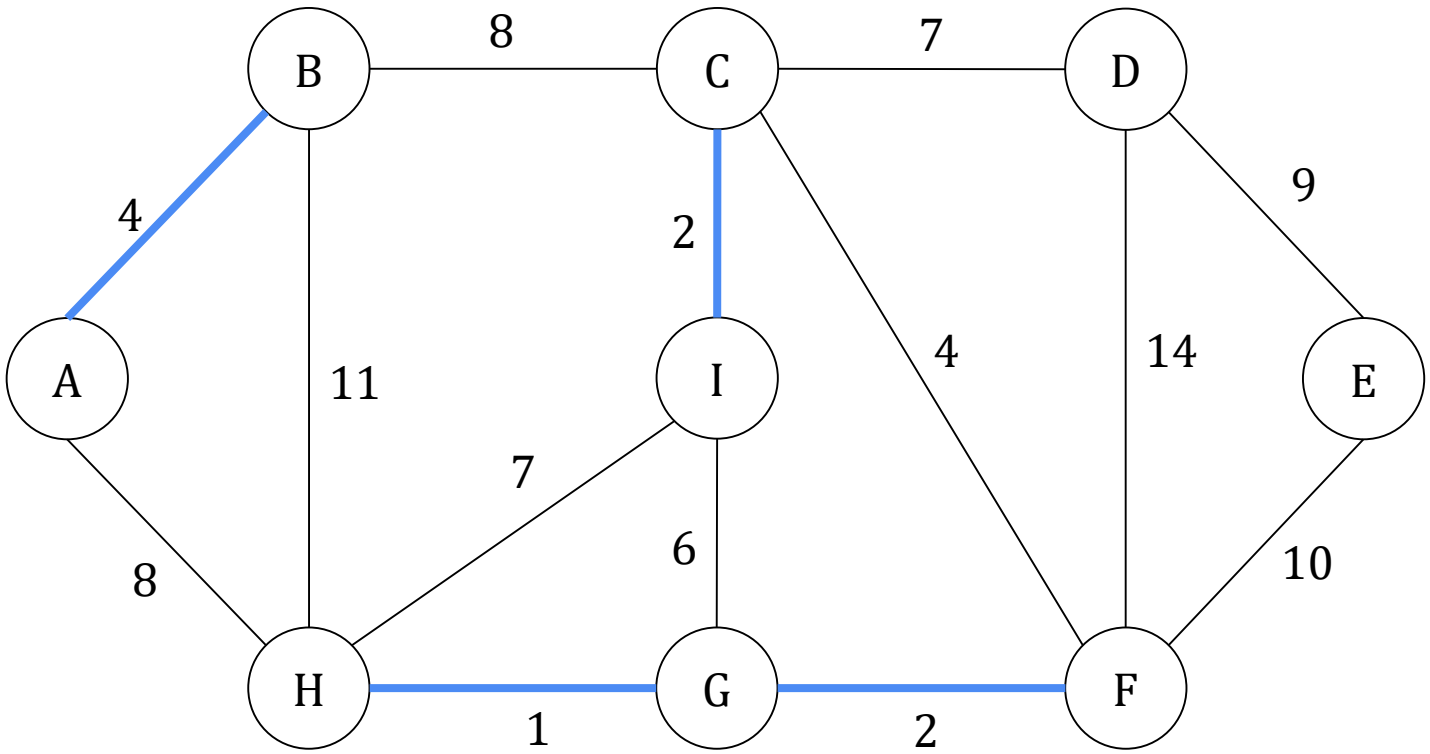


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

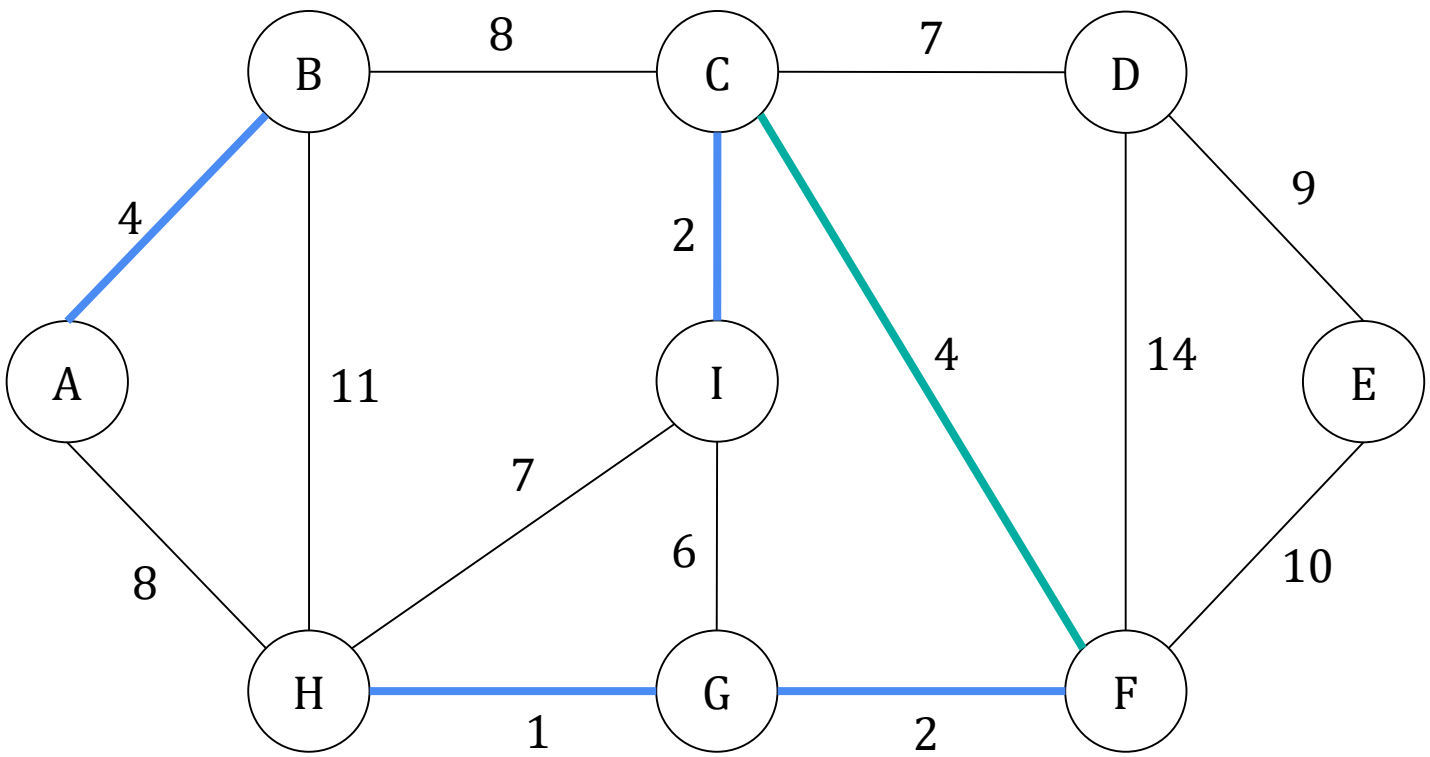


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

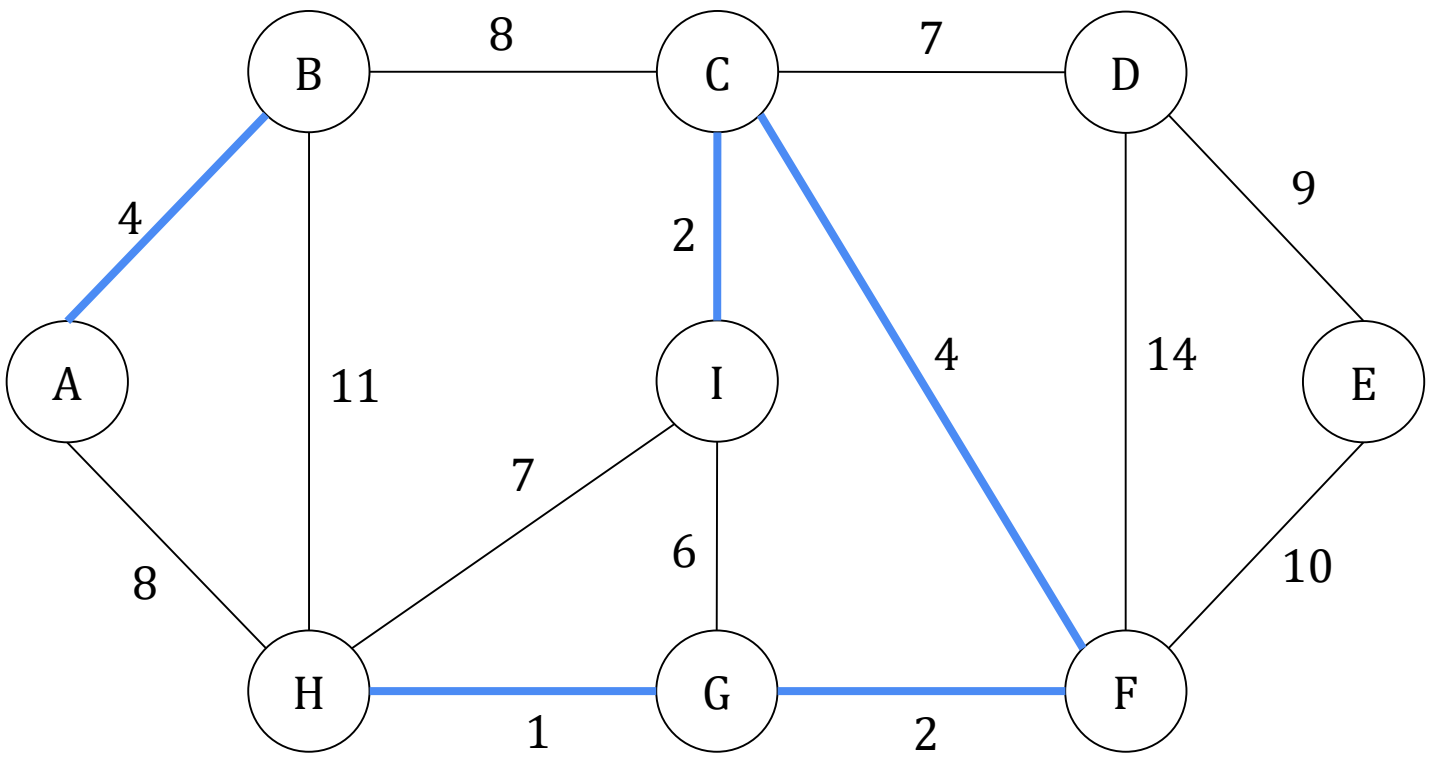


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

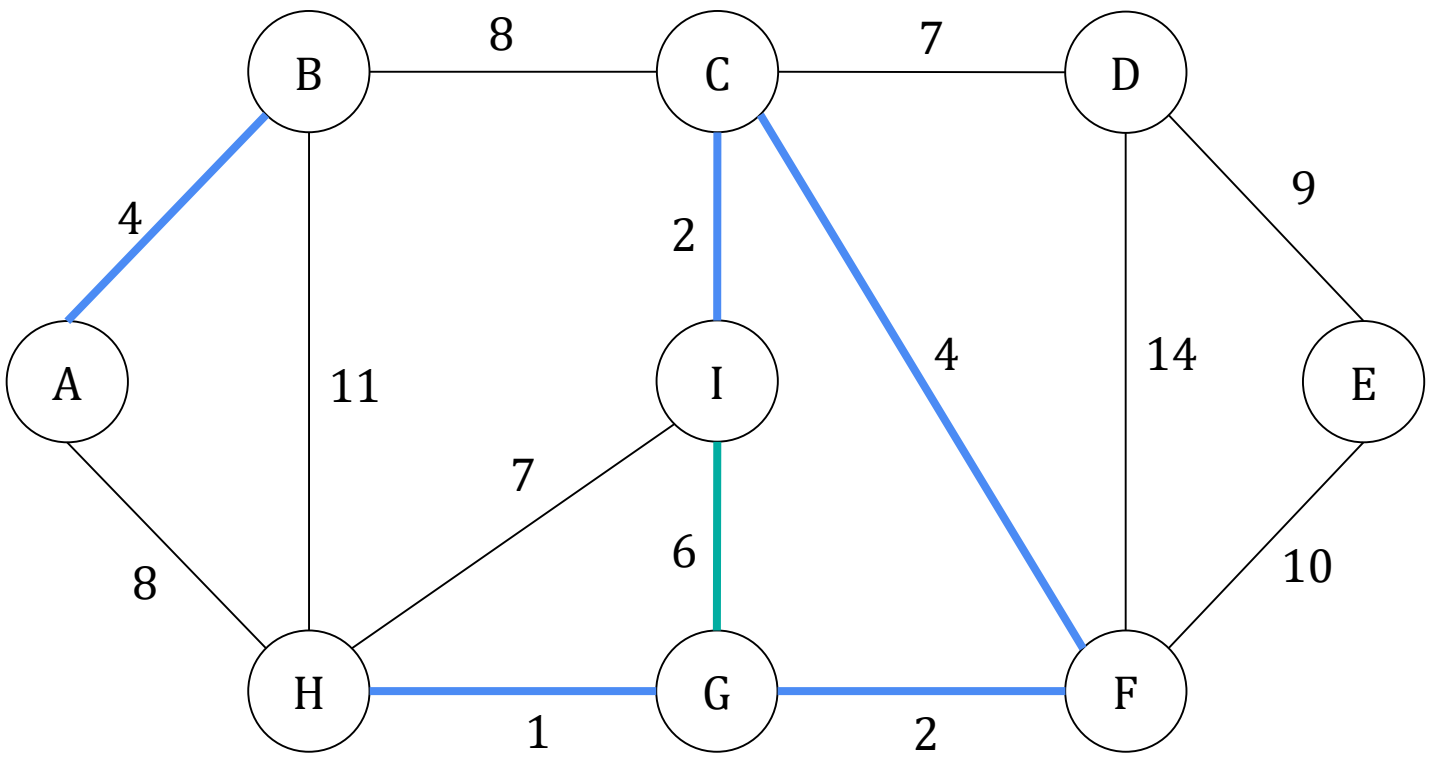


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

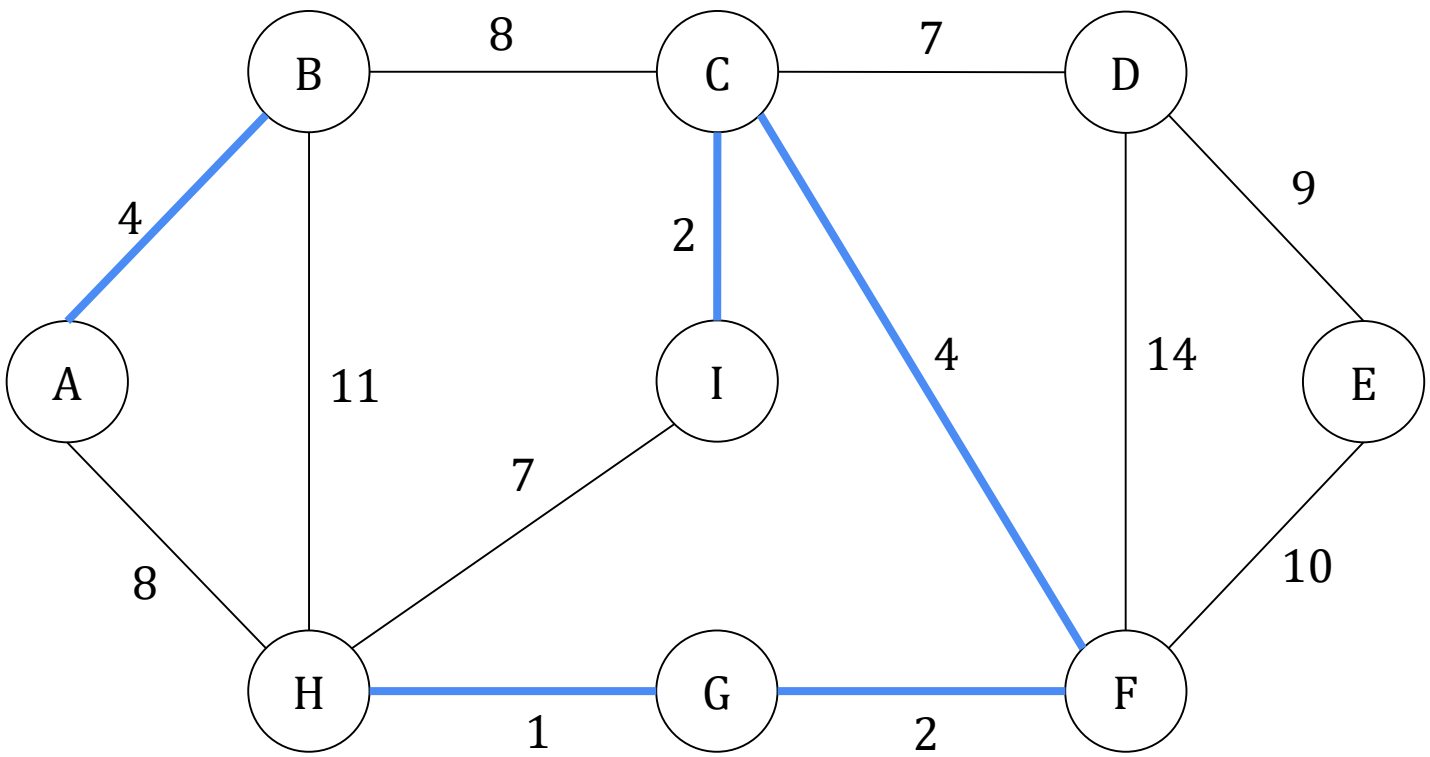


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

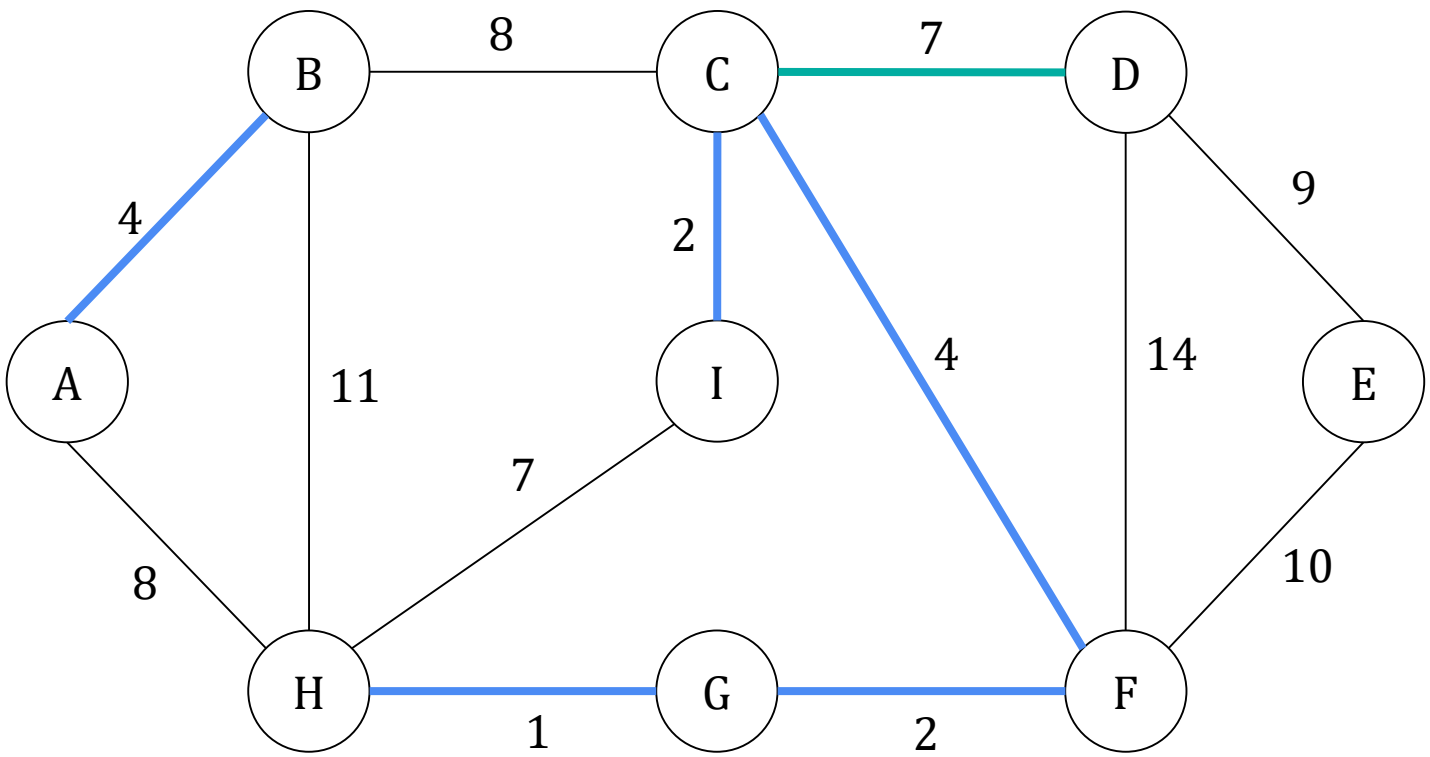


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

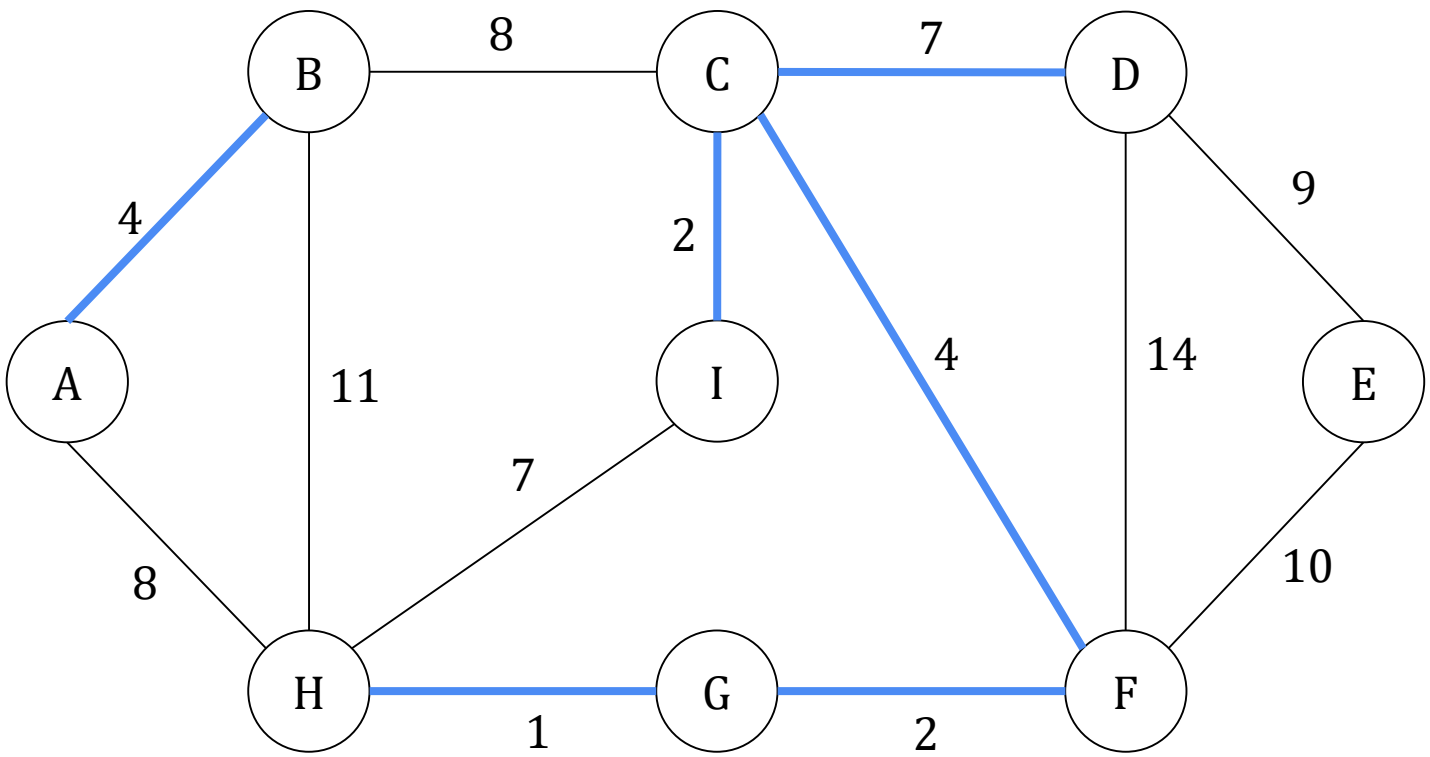


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

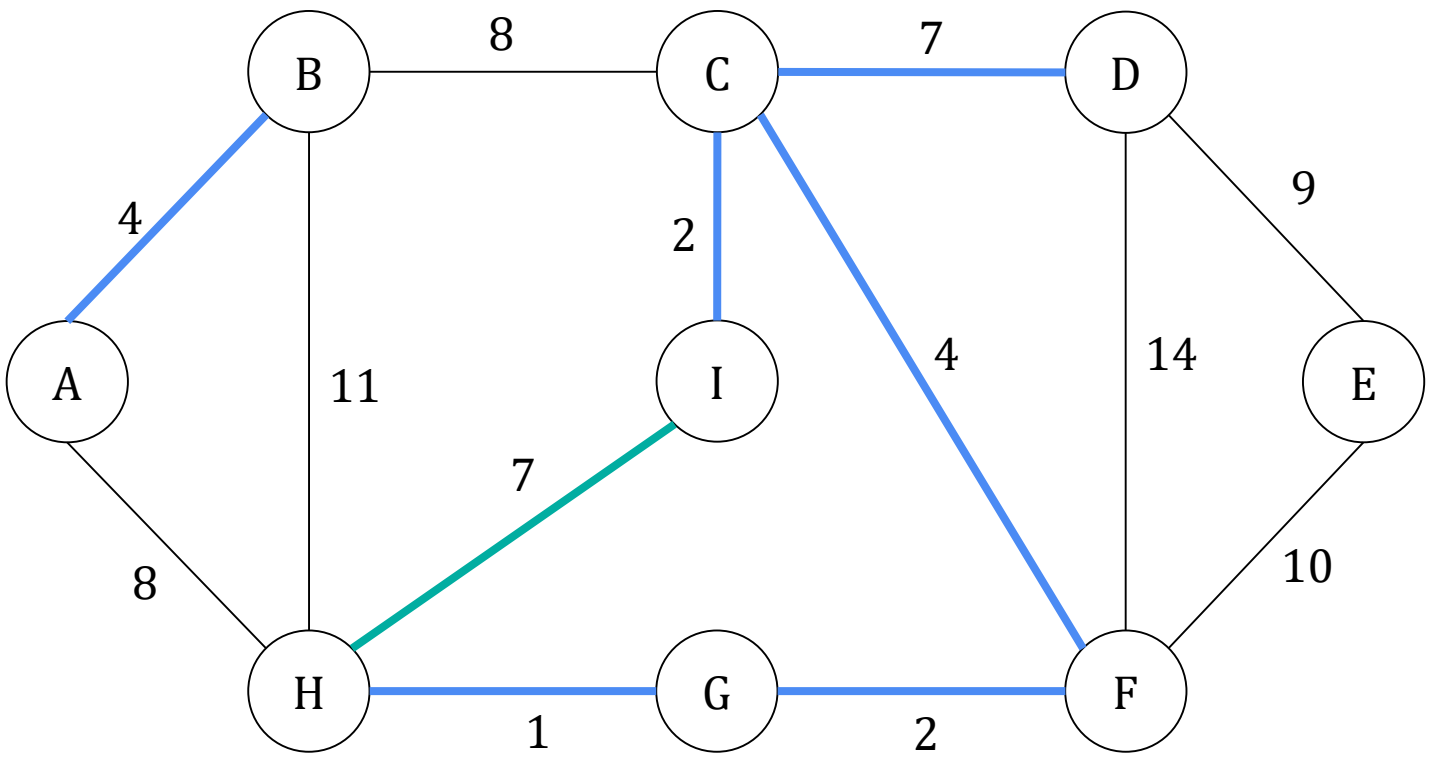


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

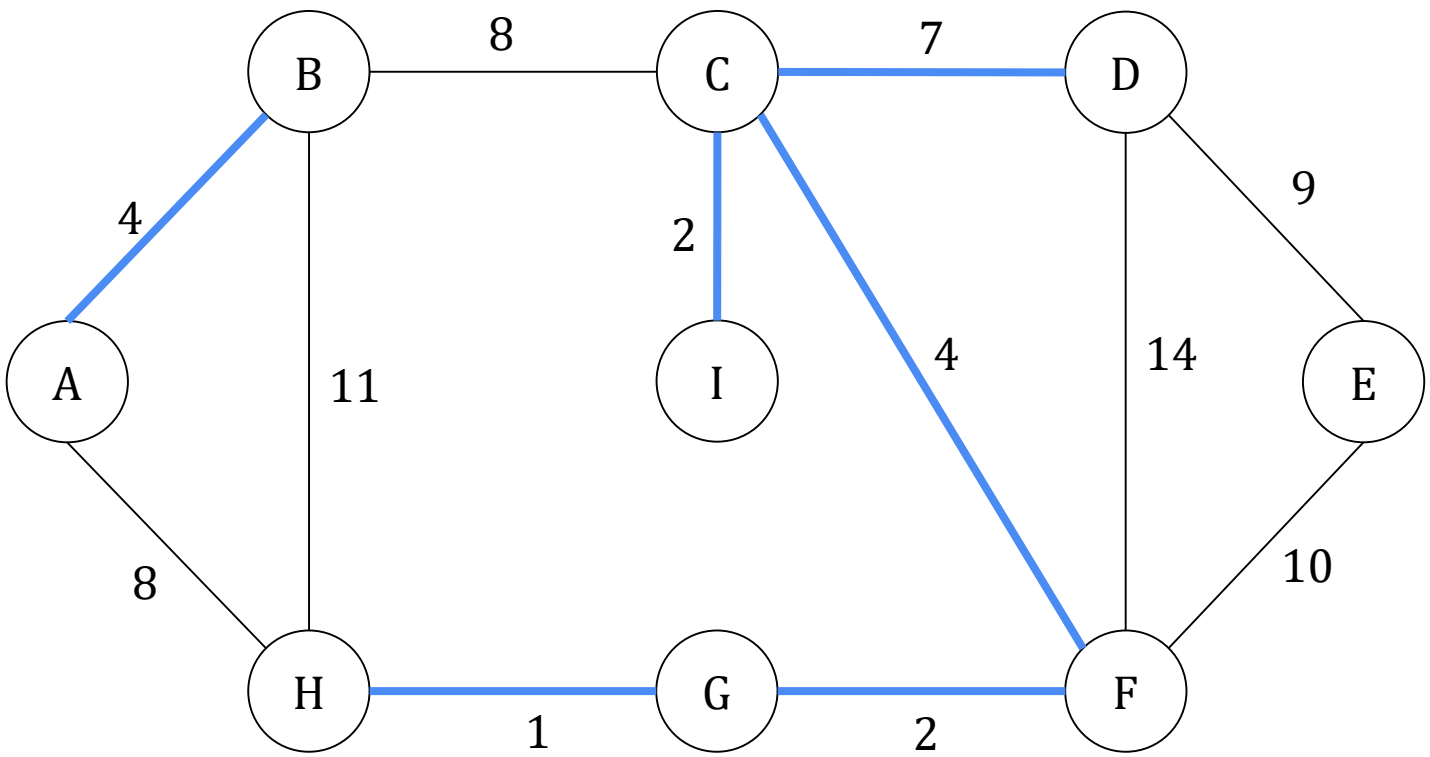


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

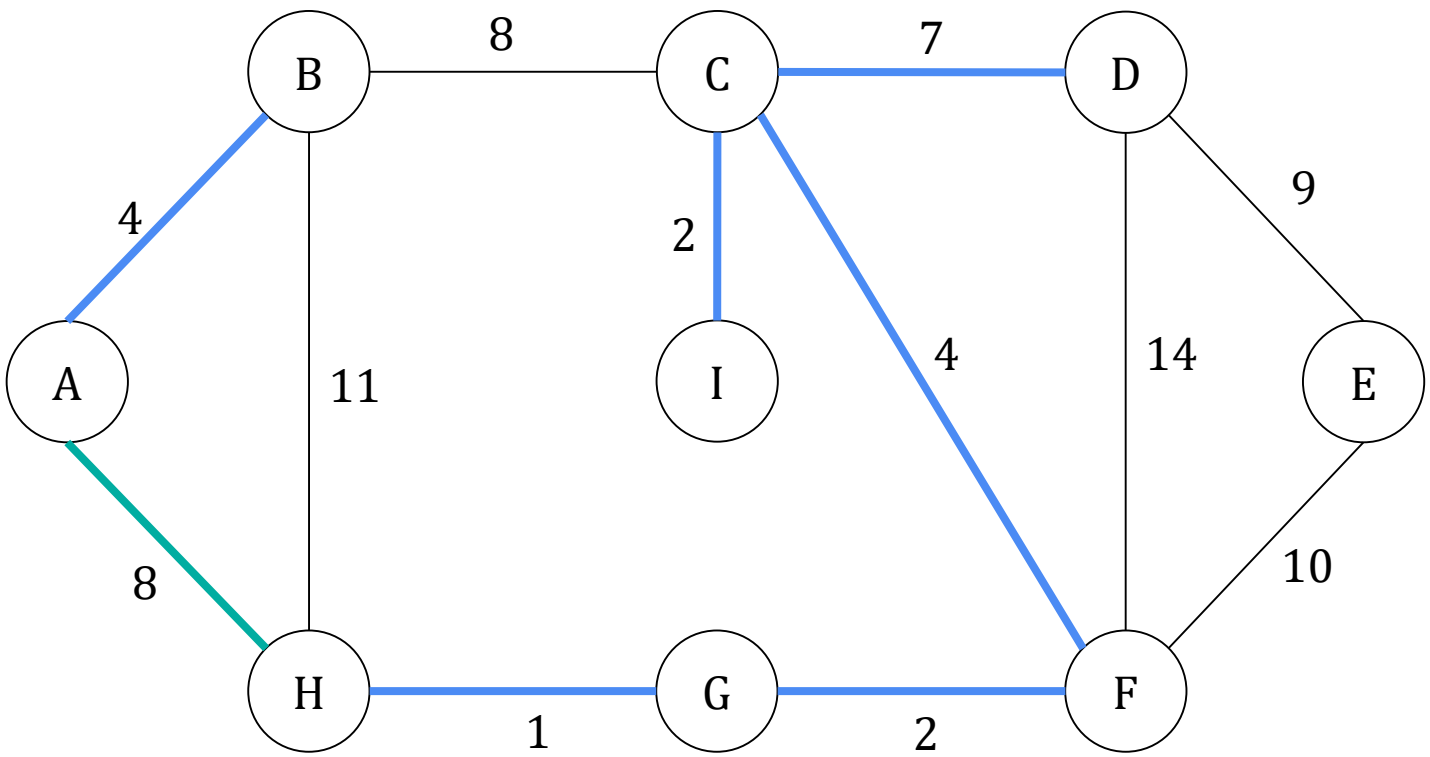


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

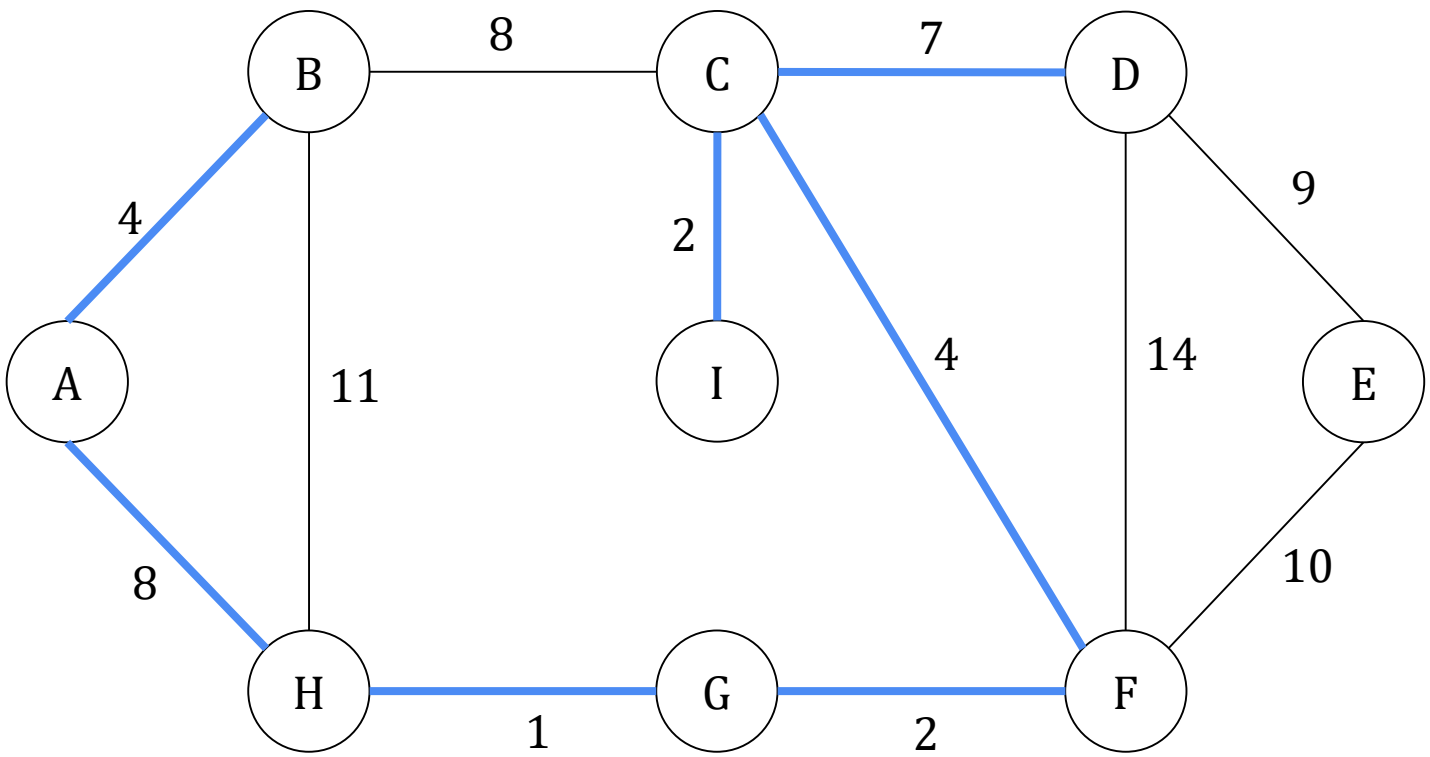


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

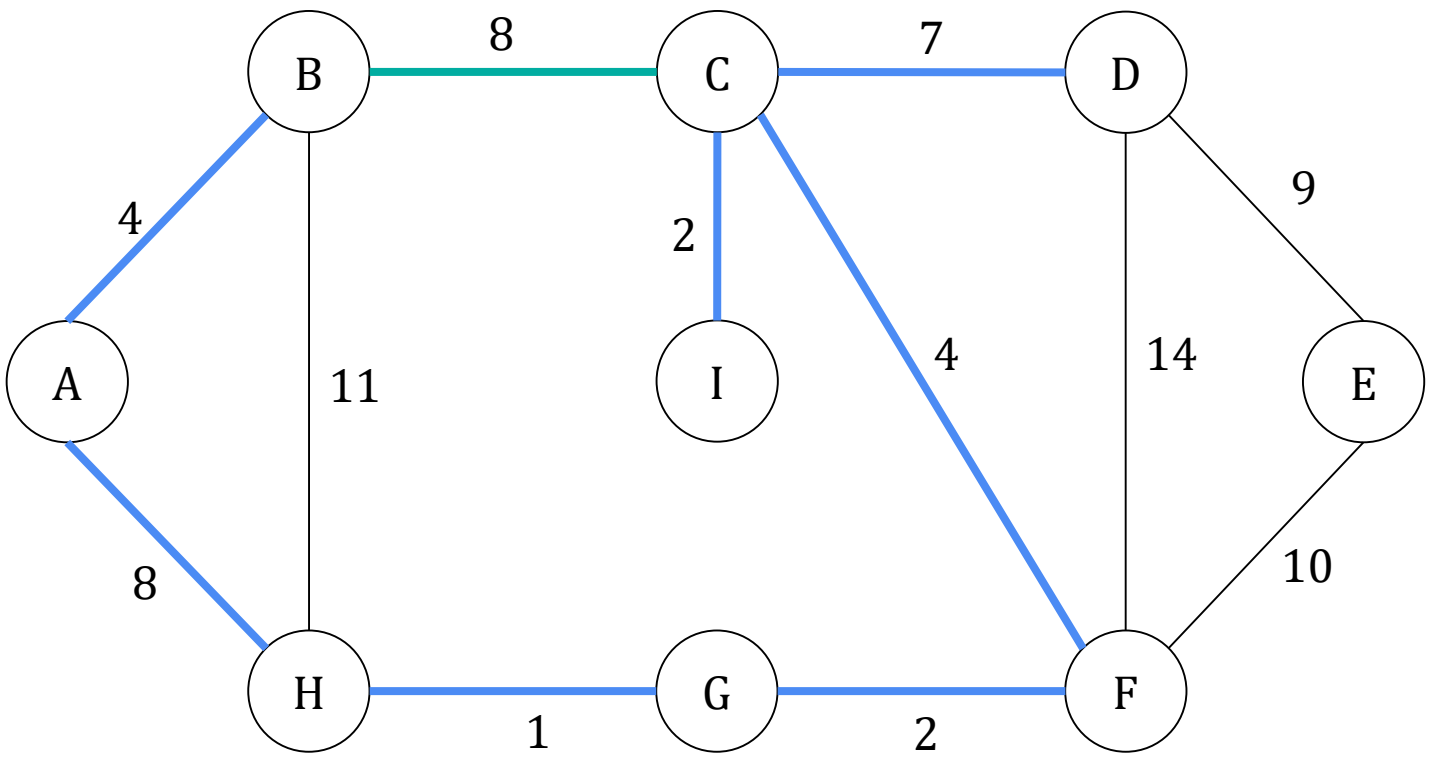


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

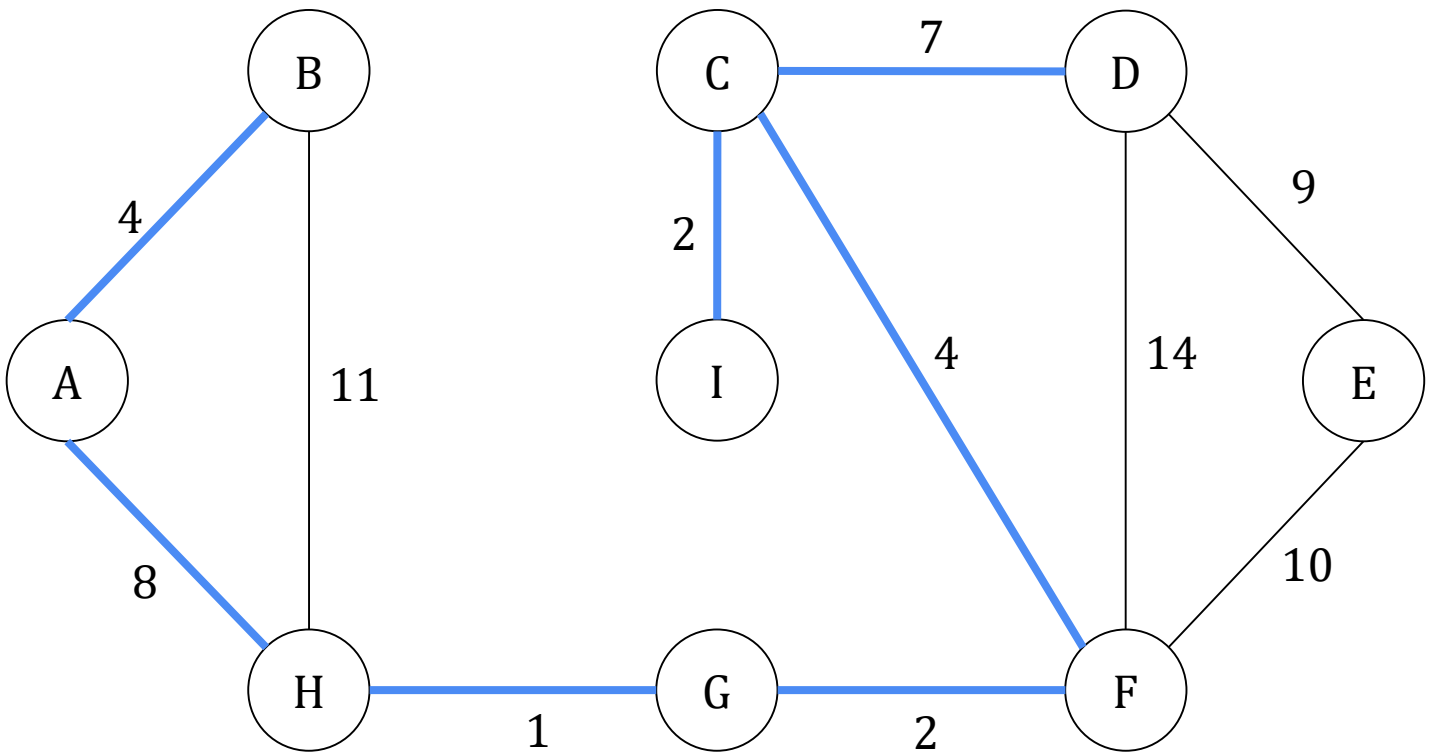


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

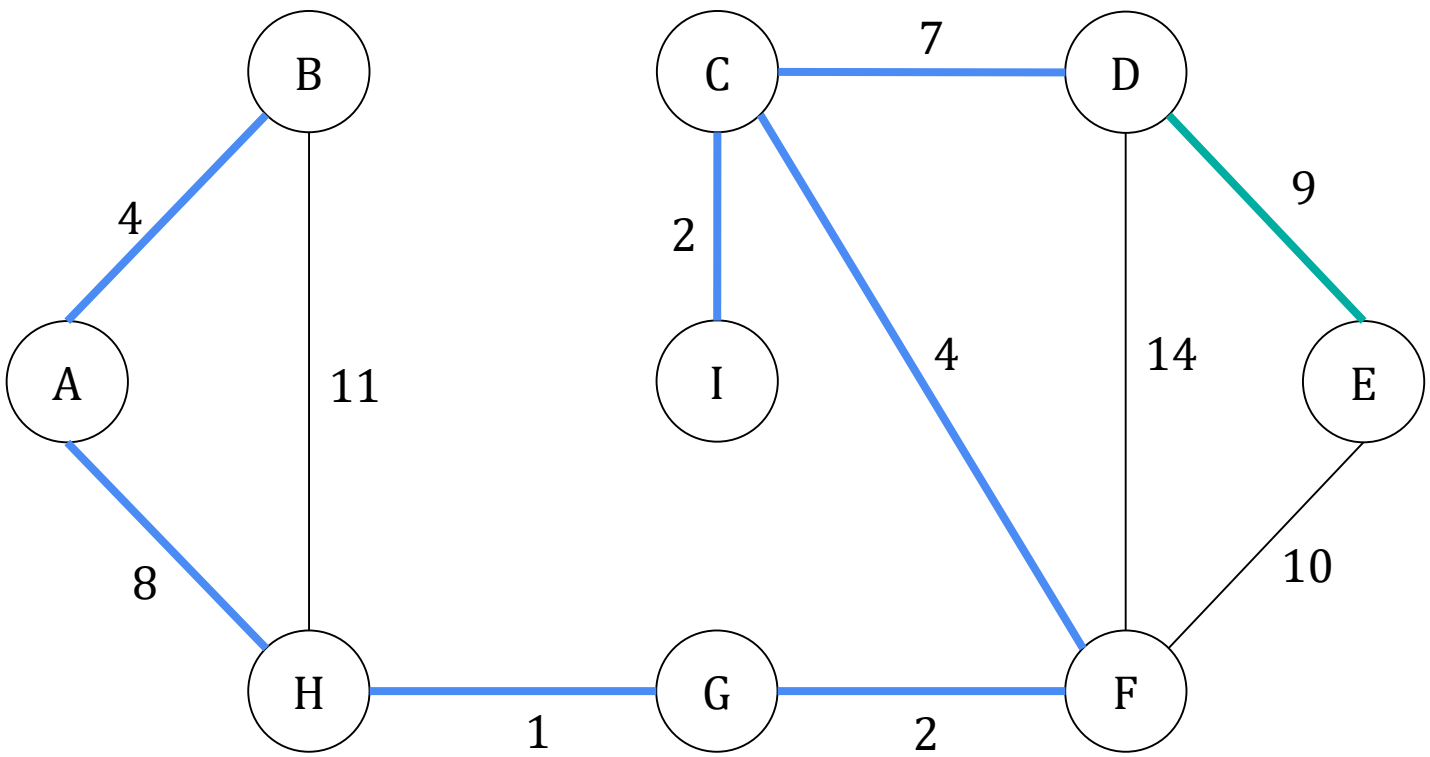


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	×
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

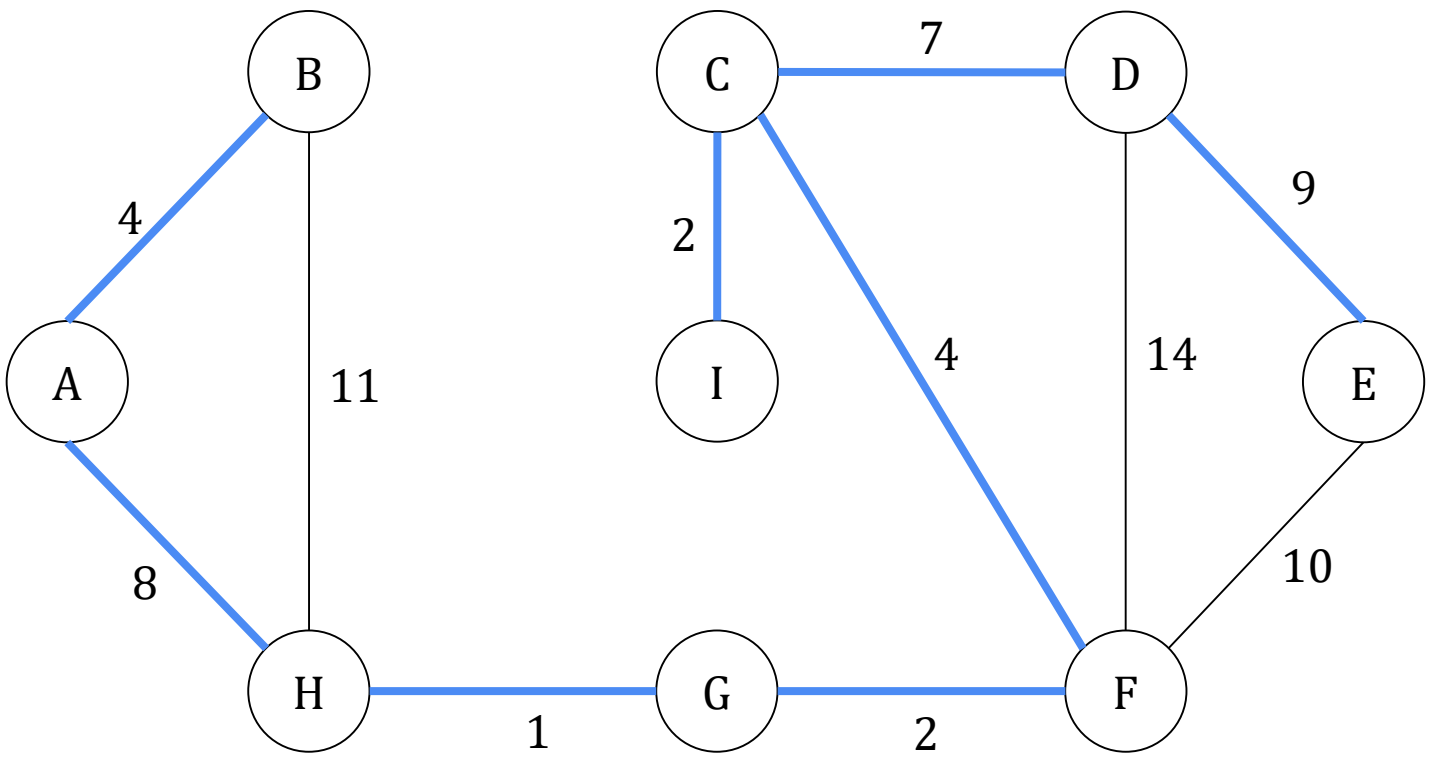


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	×
(D, E)	9	
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검

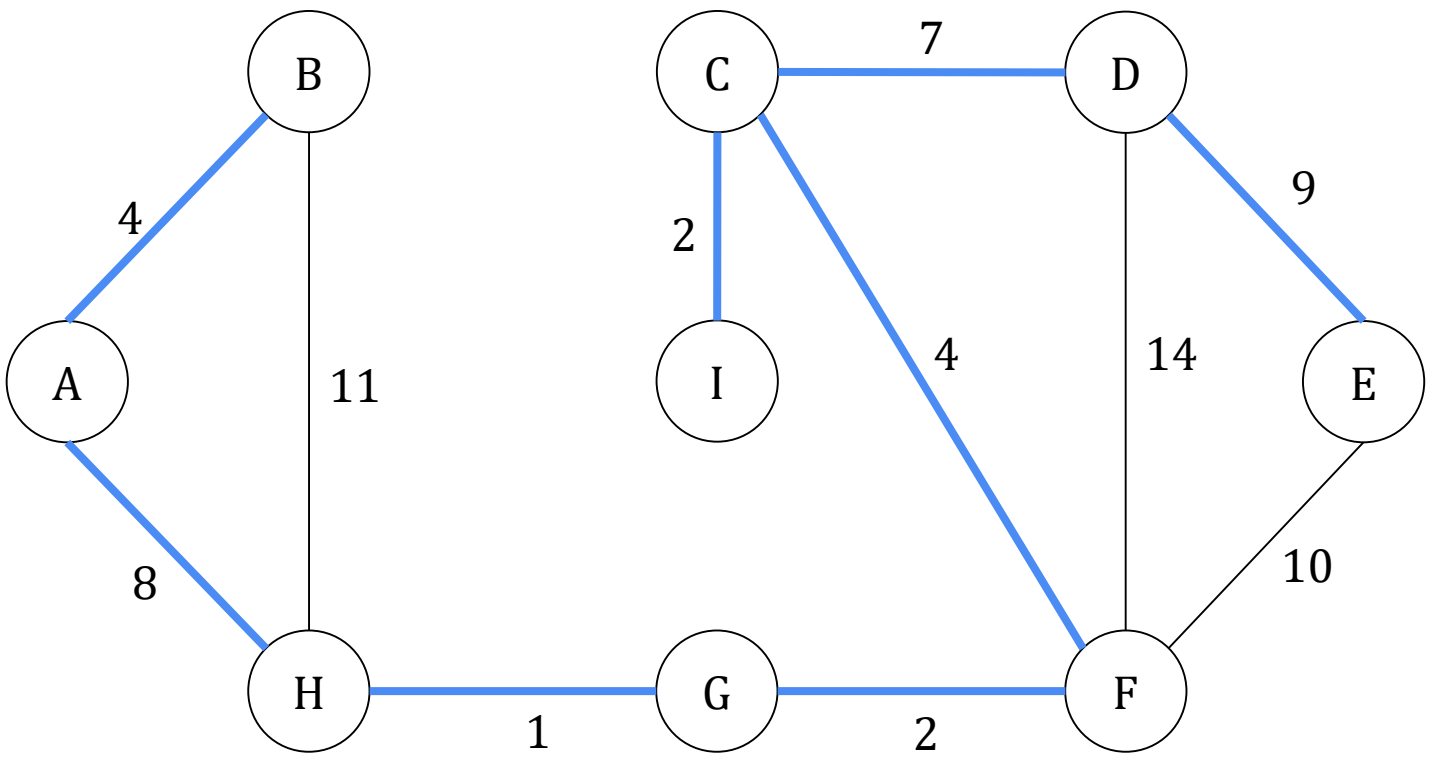


노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	×
(D, E)	9	O (8)
(E, F)	10	
(B, H)	11	
(D, F)	14	

Kruskal 알고리즘

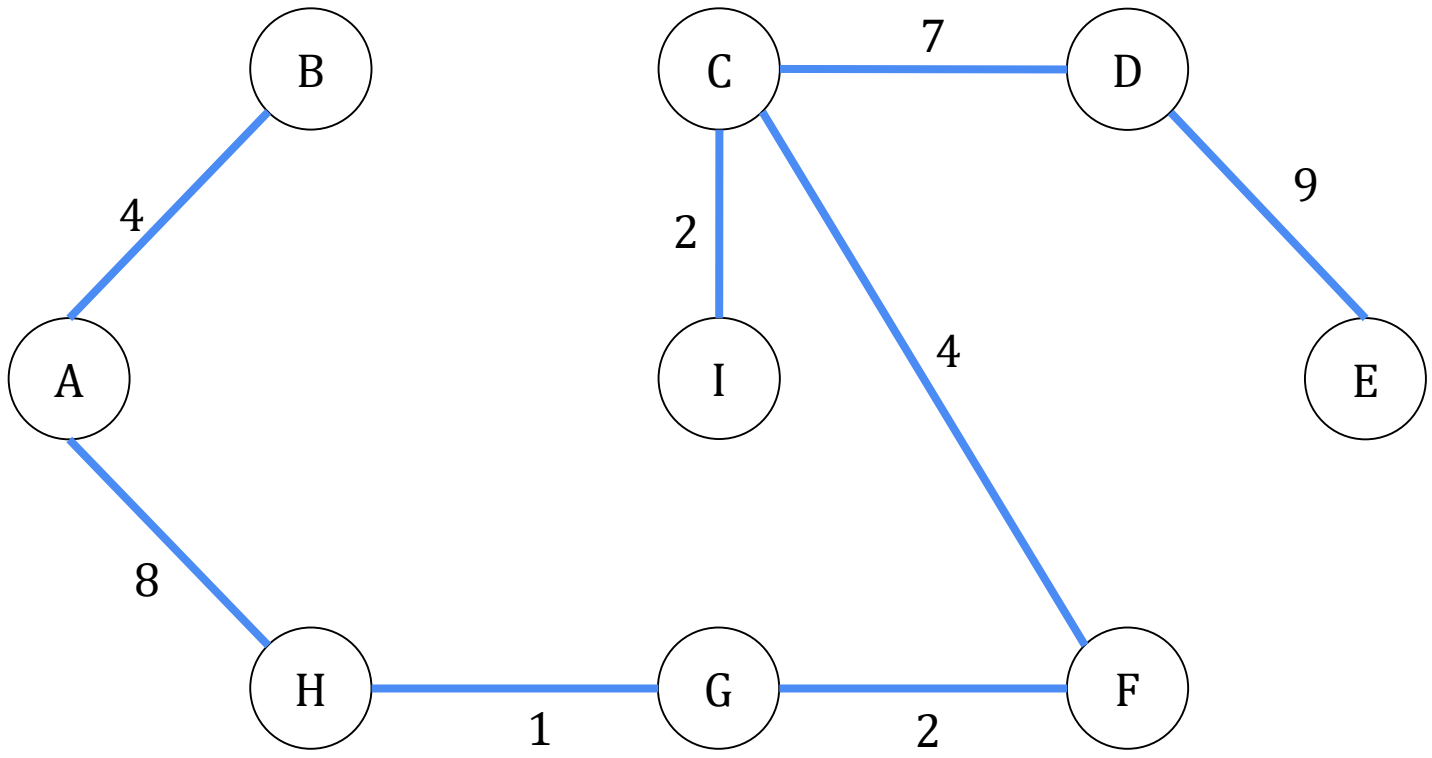
2. 가중치가 가장 작은 간선 선택
단, 싸이클 발생 여부 점검



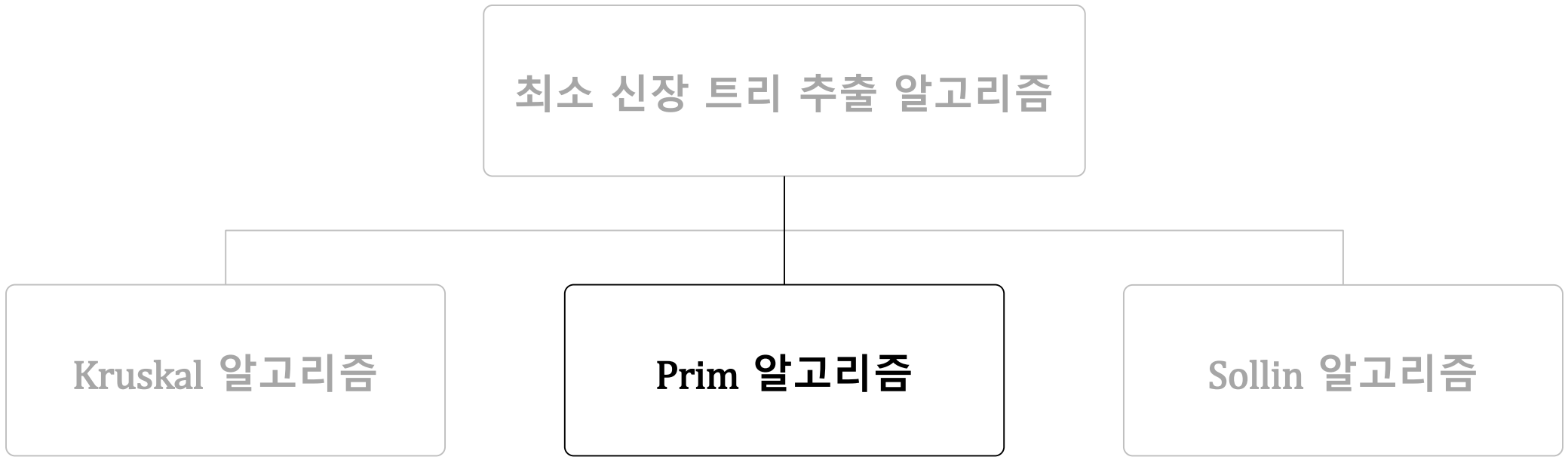
노드 수: 9

간선	가중치	선택여부 (간선 수)
(G, H)	1	O (1)
(C, I)	2	O (2)
(F, G)	2	O (3)
(A, B)	4	O (4)
(C, F)	4	O (5)
(G, I)	6	×
(C, D)	7	O (6)
(H, I)	7	×
(A, H)	8	O (7)
(B, C)	8	×
(D, E)	9	O (8)
(E, F)	10	
(B, H)	11	
(D, F)	14	

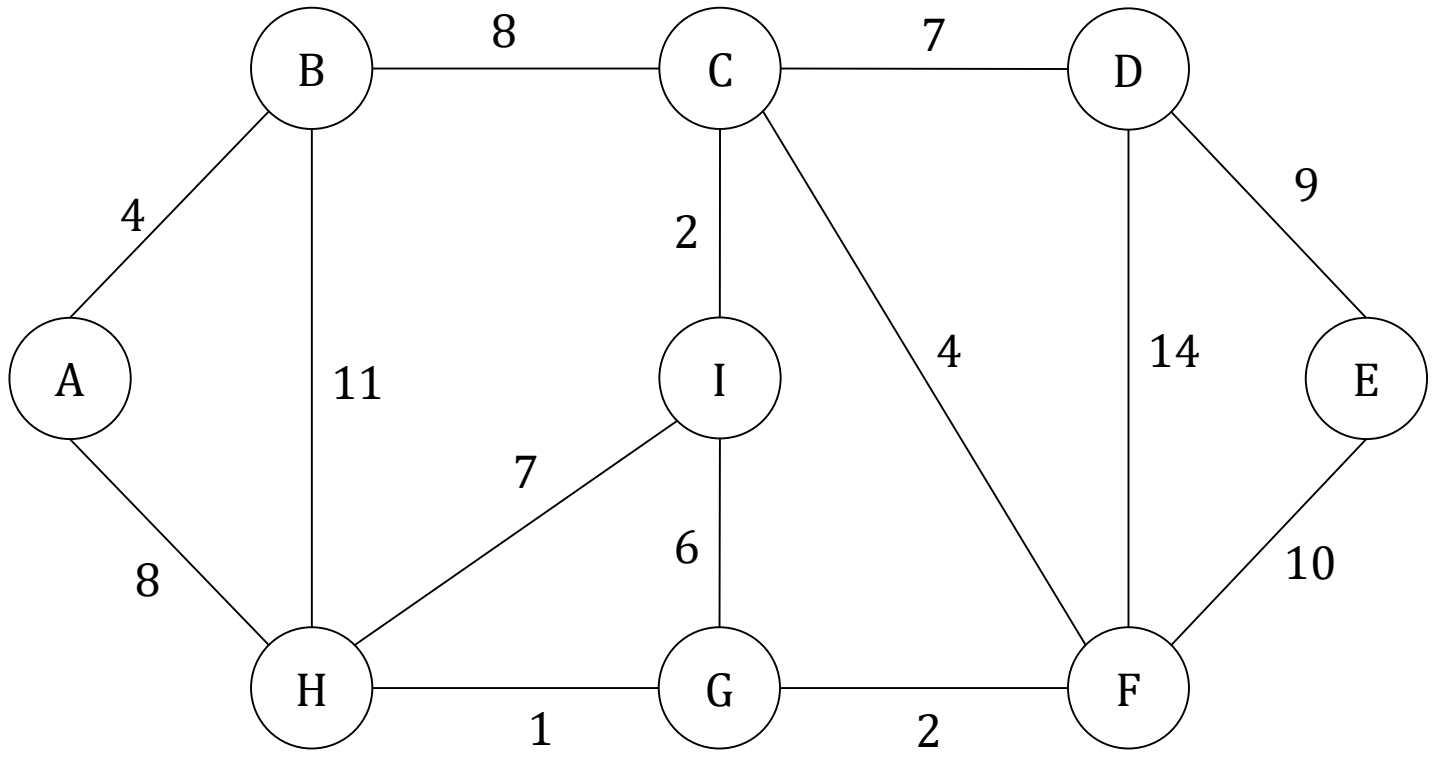
Kruskal 알고리즘



최소 신장 트리 추출 알고리즘

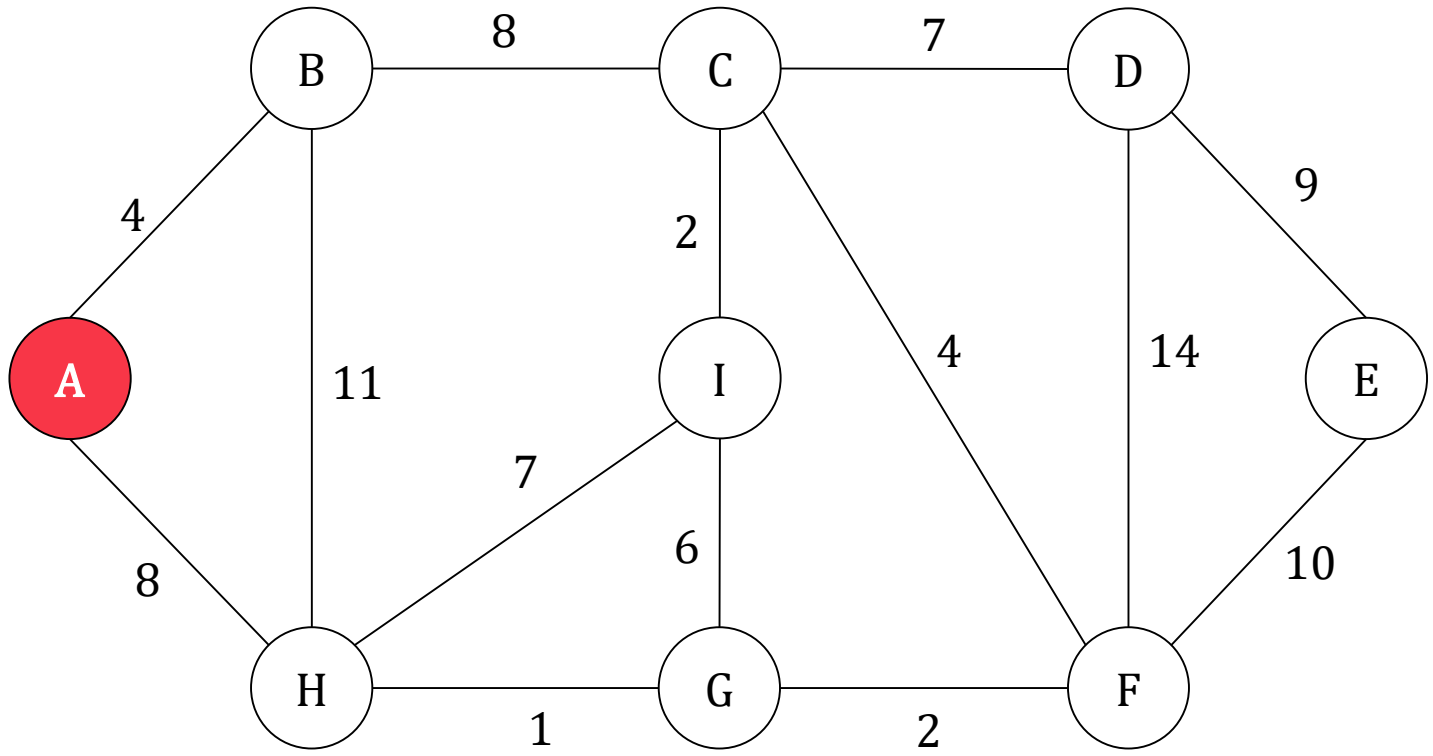


Prim 알고리즘



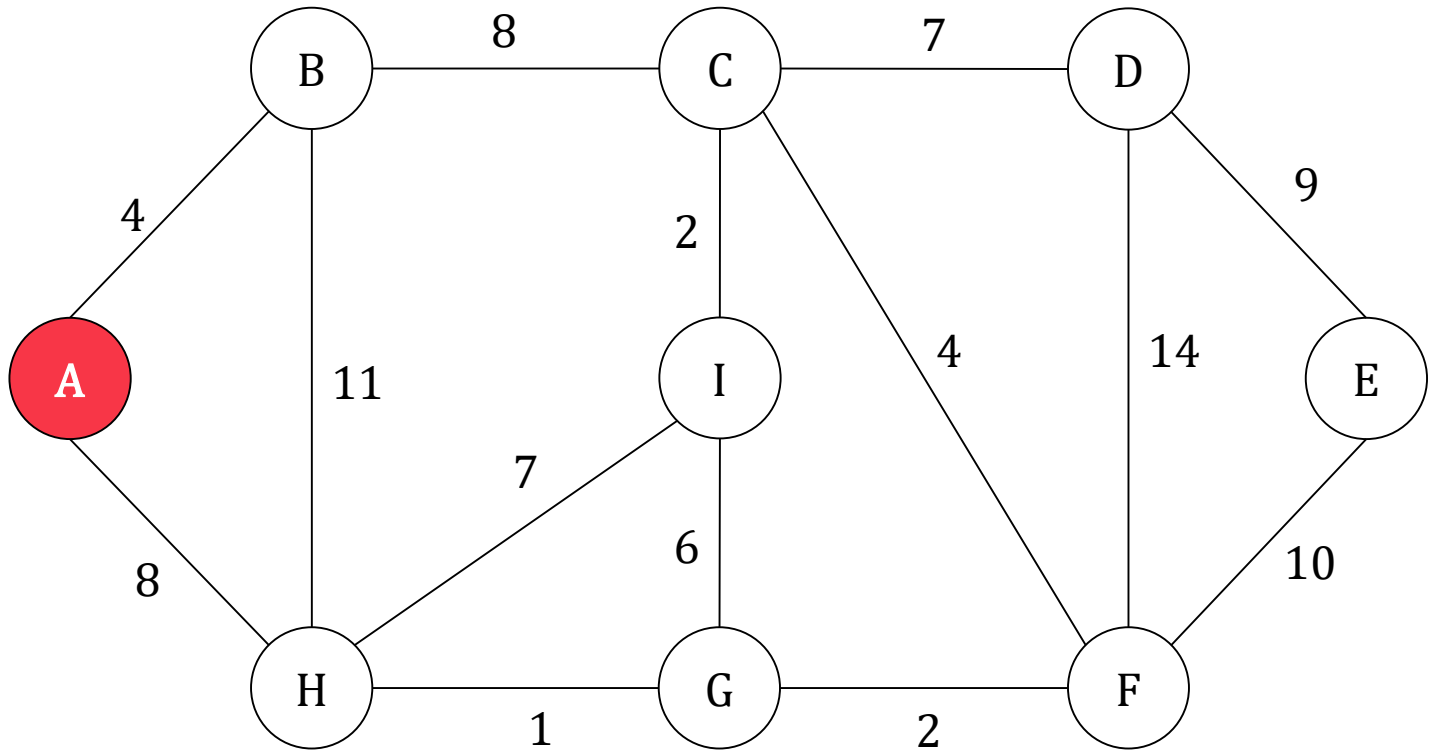
Node	A	B	C	D	E	F	G	H	I
Key									

Prim 알고리즘



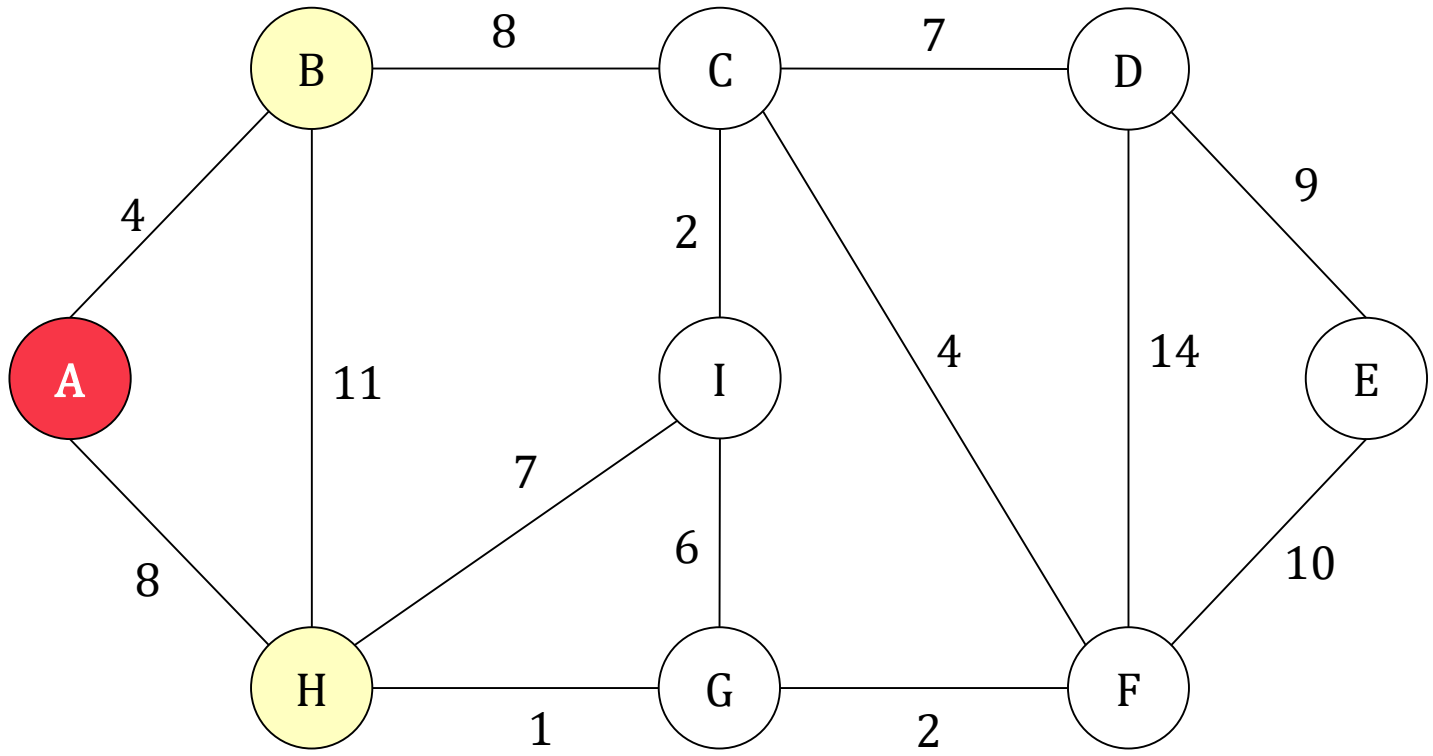
Node	A	B	C	D	E	F	G	H	I
Key	0	∞	∞	∞	∞	∞	∞	∞	∞

Prim 알고리즘



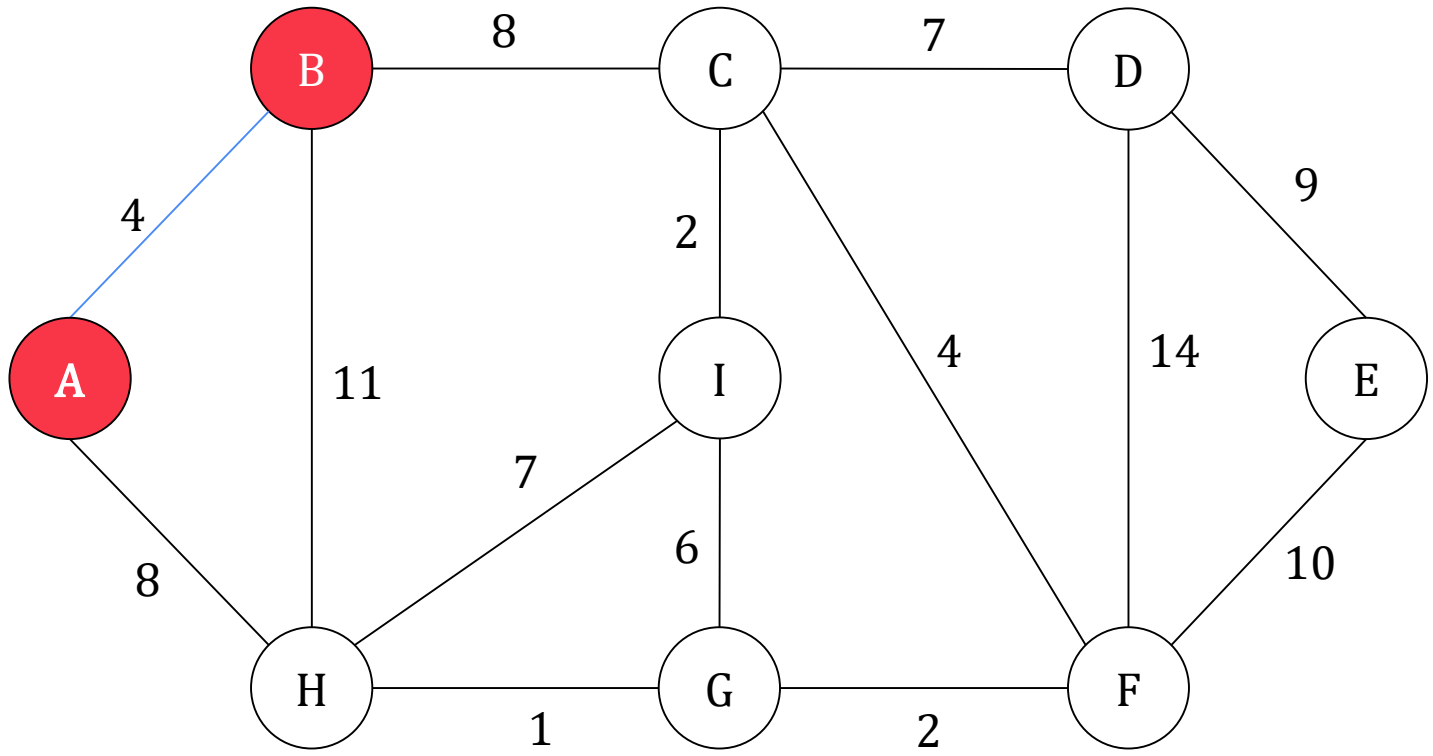
Node	A	B	C	D	E	F	G	H	I
Key	0	∞	∞	∞	∞	∞	∞	∞	∞

Prim 알고리즘



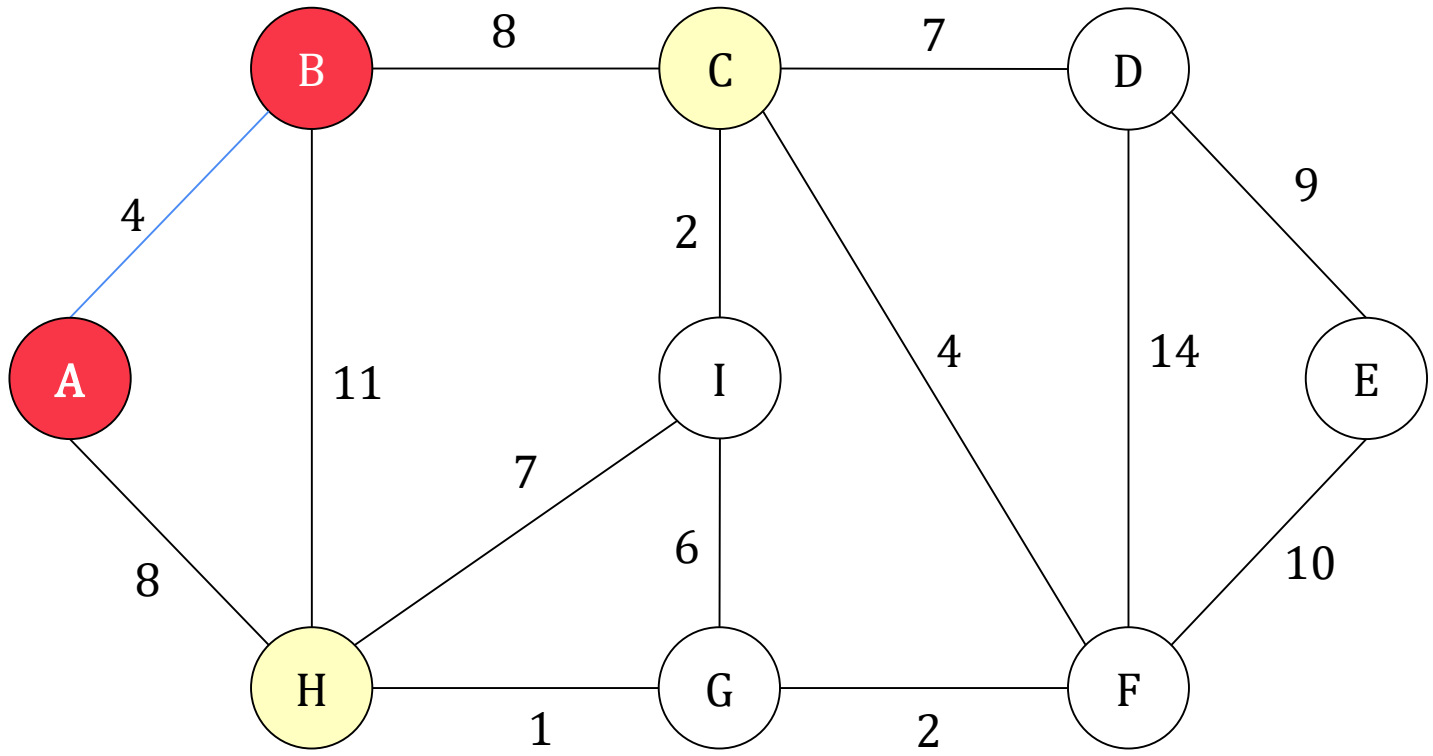
Node	A	B(A)	C	D	E	F	G	H(A)	I
Key	0	**4**	∞	∞	∞	∞	∞	**8**	∞

Prim 알고리즘



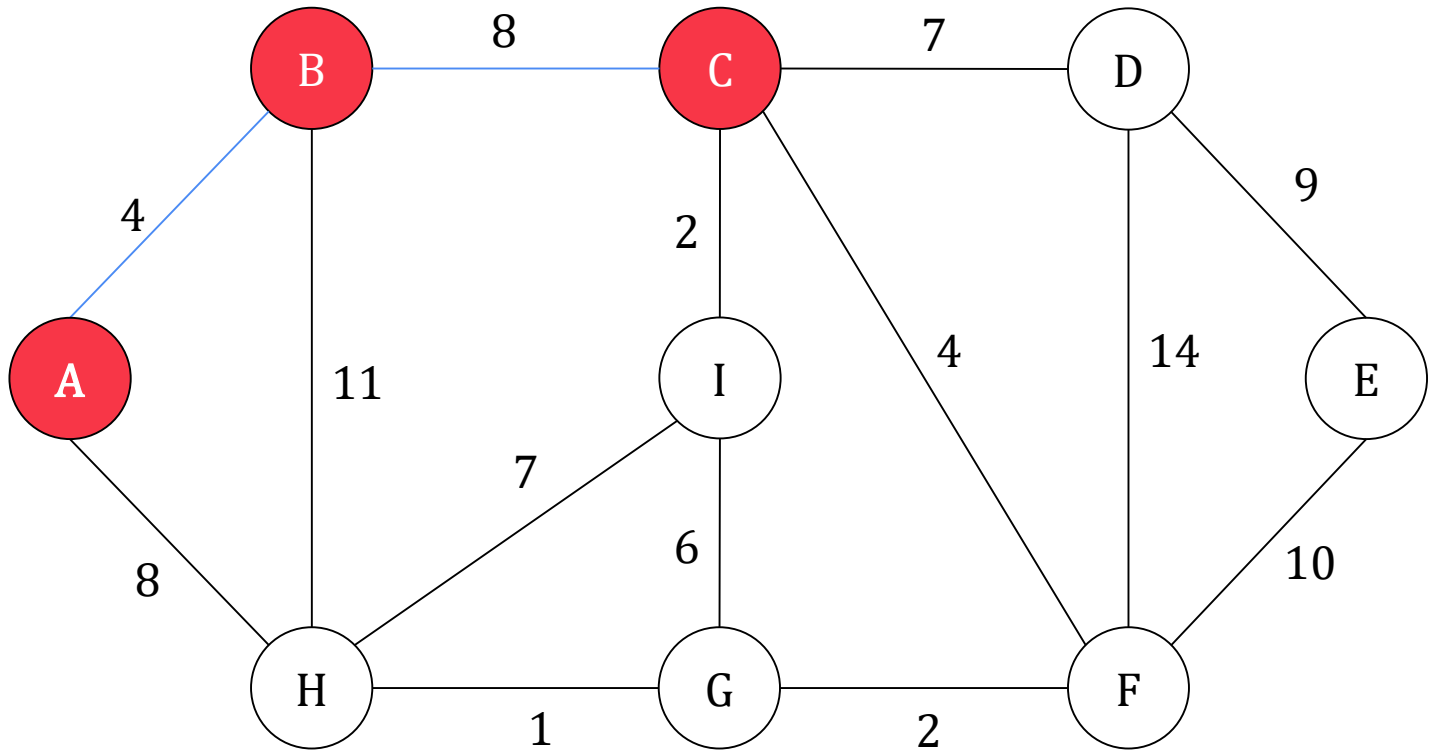
Node	A	B(A)	C	D	E	F	G	H(A)	I
Key	0	4	∞	∞	∞	∞	∞	8	∞

Prim 알고리즘



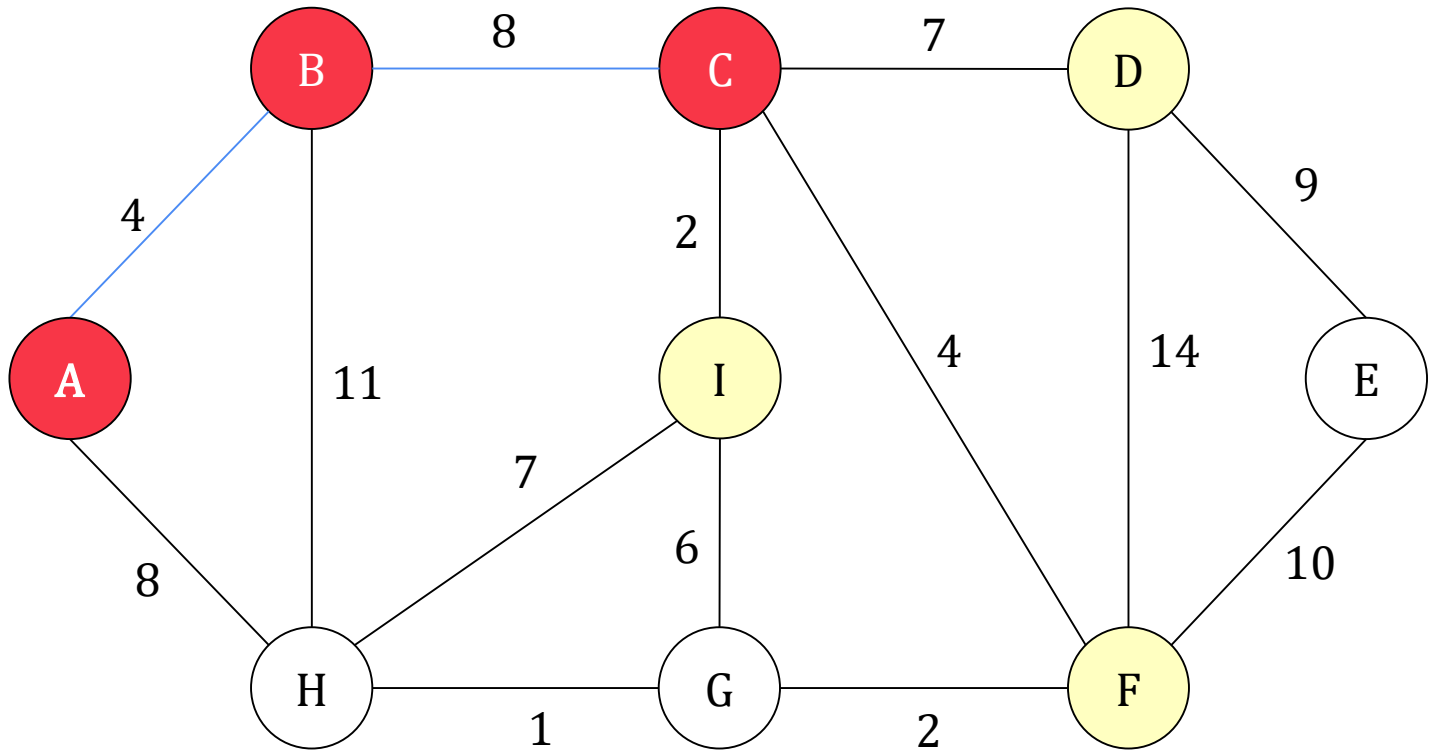
Node	A	B(A)	C(B)	D	E	F	G	H(A)	I
Key	0	4	**8**	∞	∞	∞	∞	8	∞

Prim 알고리즘



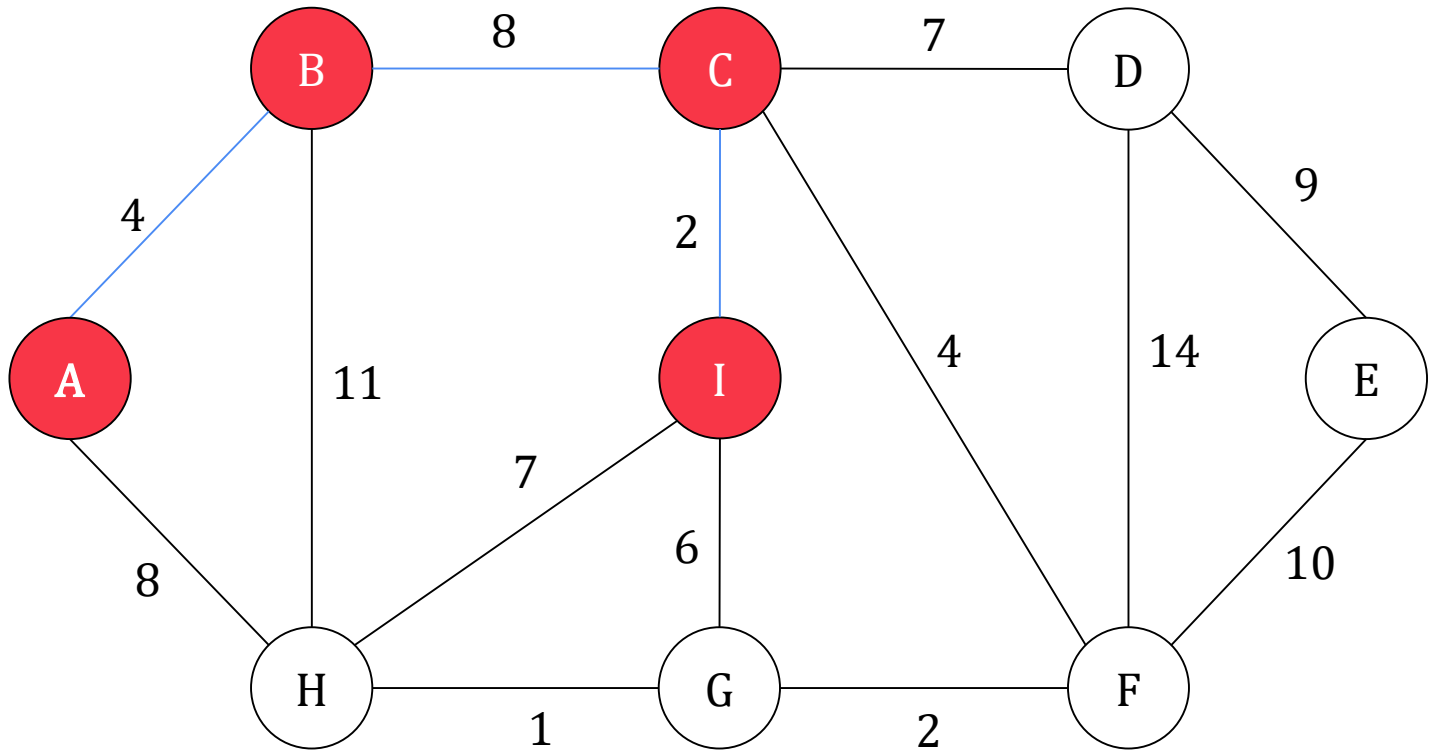
Node	A	B(A)	C(B)	D	E	F	G	H(A)	I
Key	0	4	8	∞	∞	∞	∞	8	∞

Prim 알고리즘



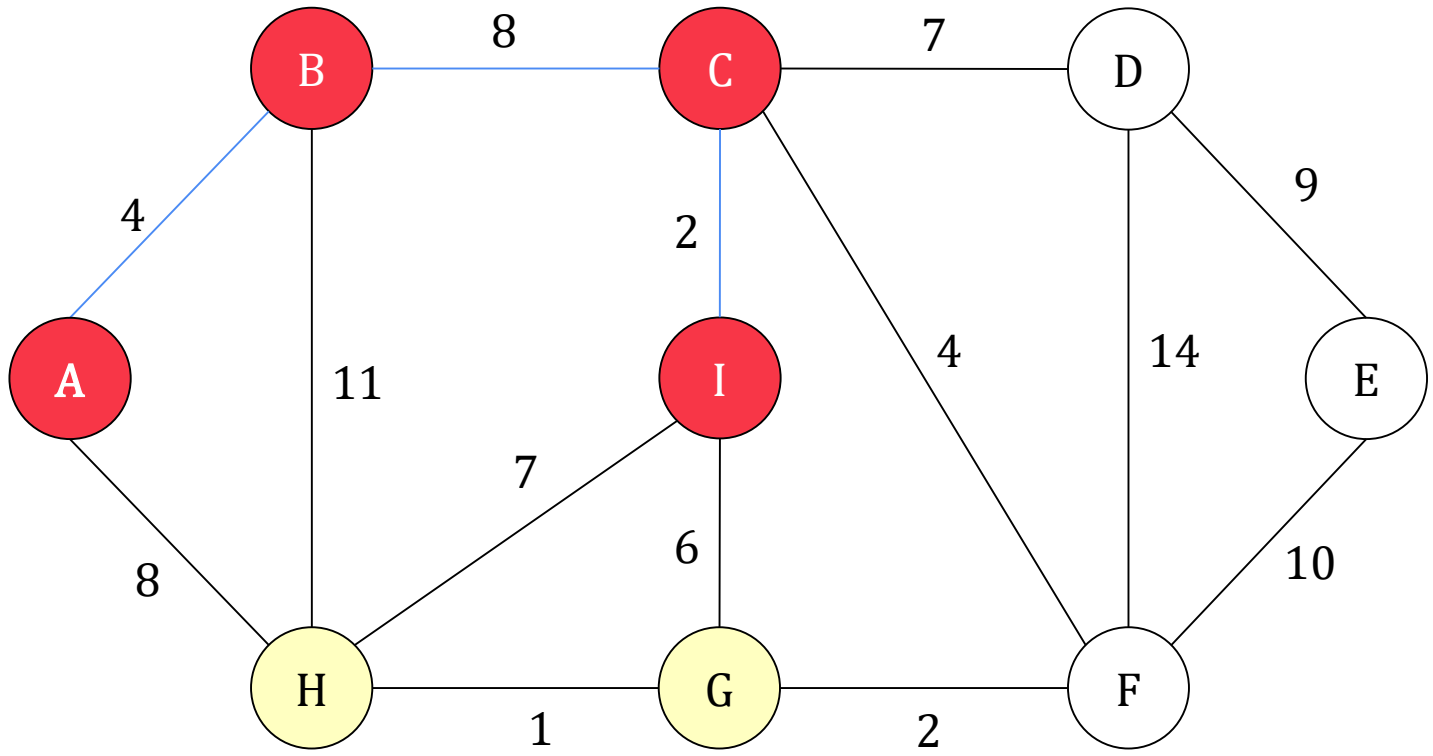
Node	A	B(A)	C(B)	D(C)	E	F(C)	G	H(A)	I(C)
Key	0	4	8	**7**	∞	**4**	∞	8	**2**

Prim 알고리즘



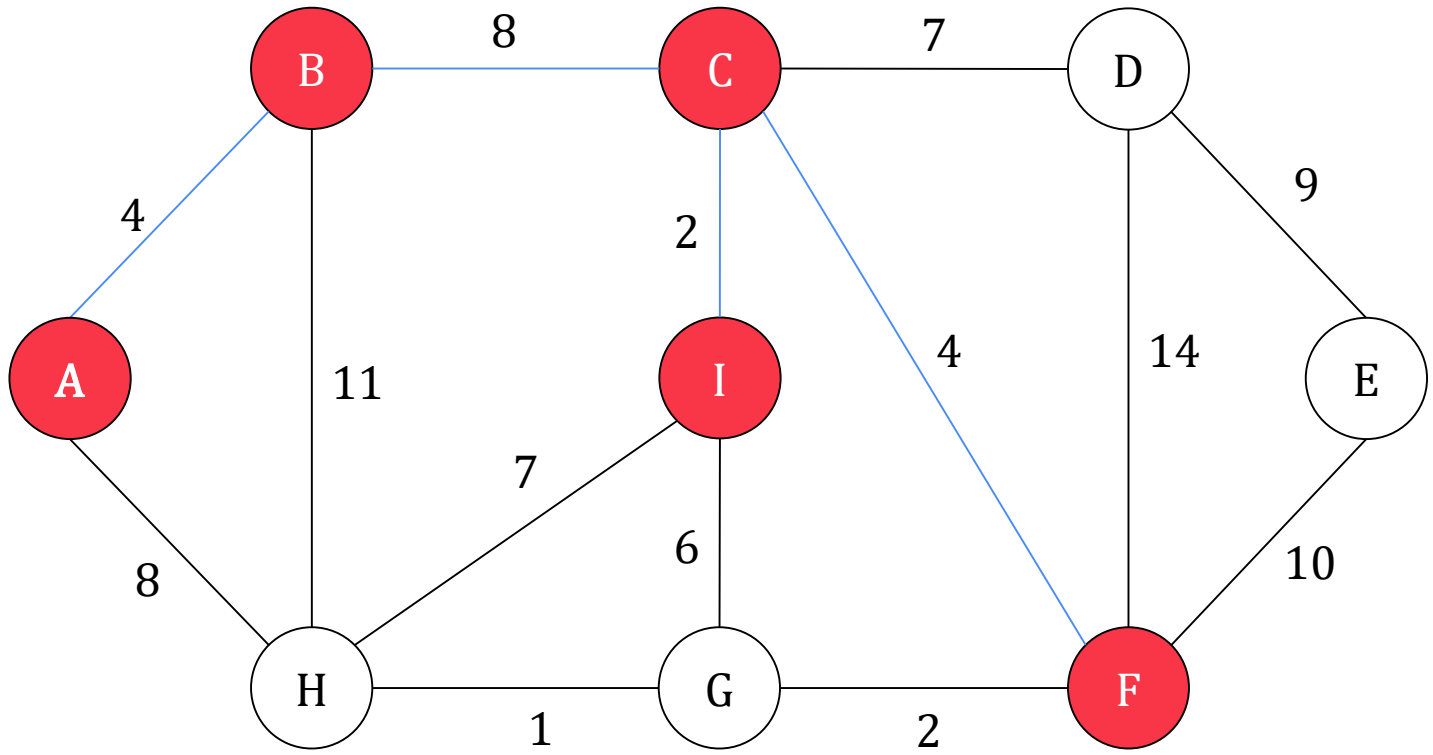
Node	A	B(A)	C(B)	D(C)	E	F(C)	G	H(A)	I(C)
Key	0	4	8	7	∞	4	∞	8	2

Prim 알고리즘



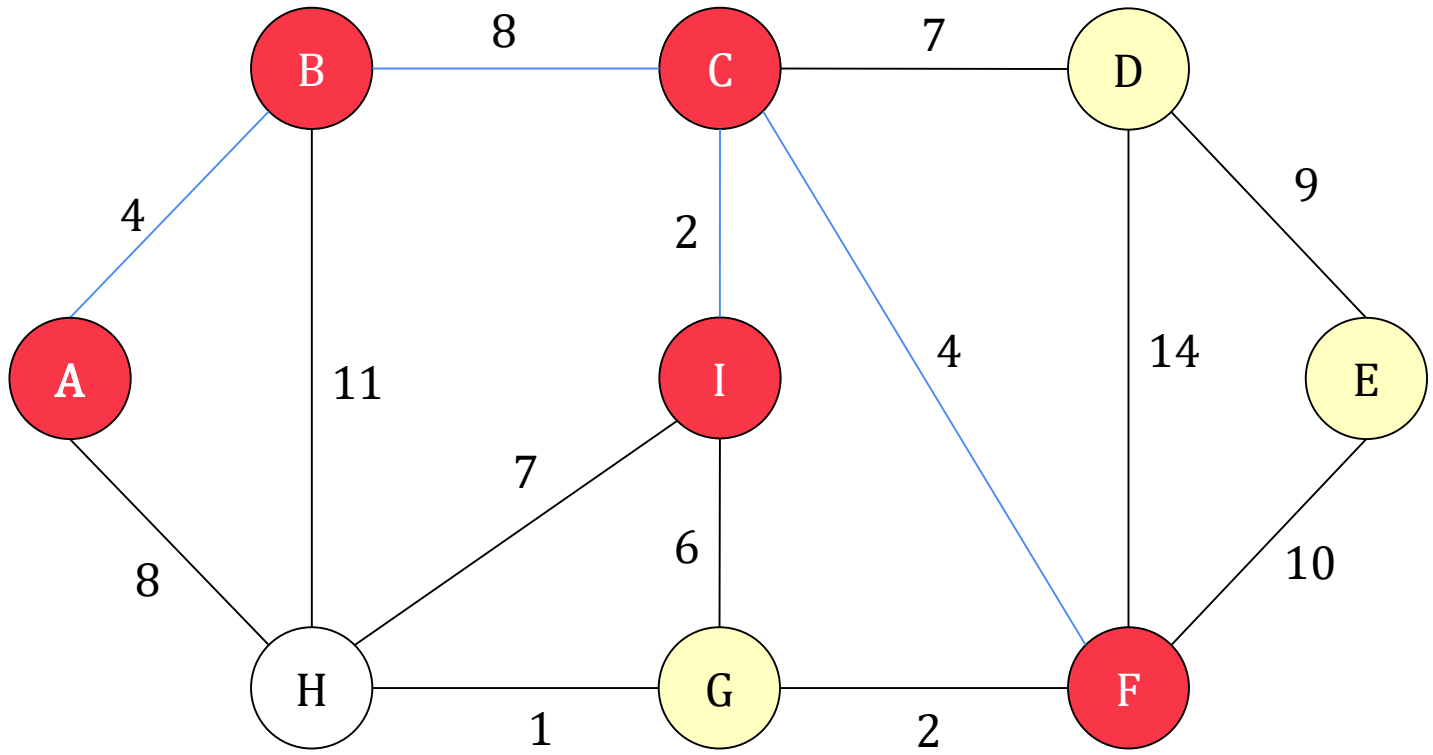
Node	A	B(A)	C(B)	D(C)	E	F(C)	G(I)	H(I)	I(C)
Key	0	4	8	7	∞	4	**6**	**7**	2

Prim 알고리즘



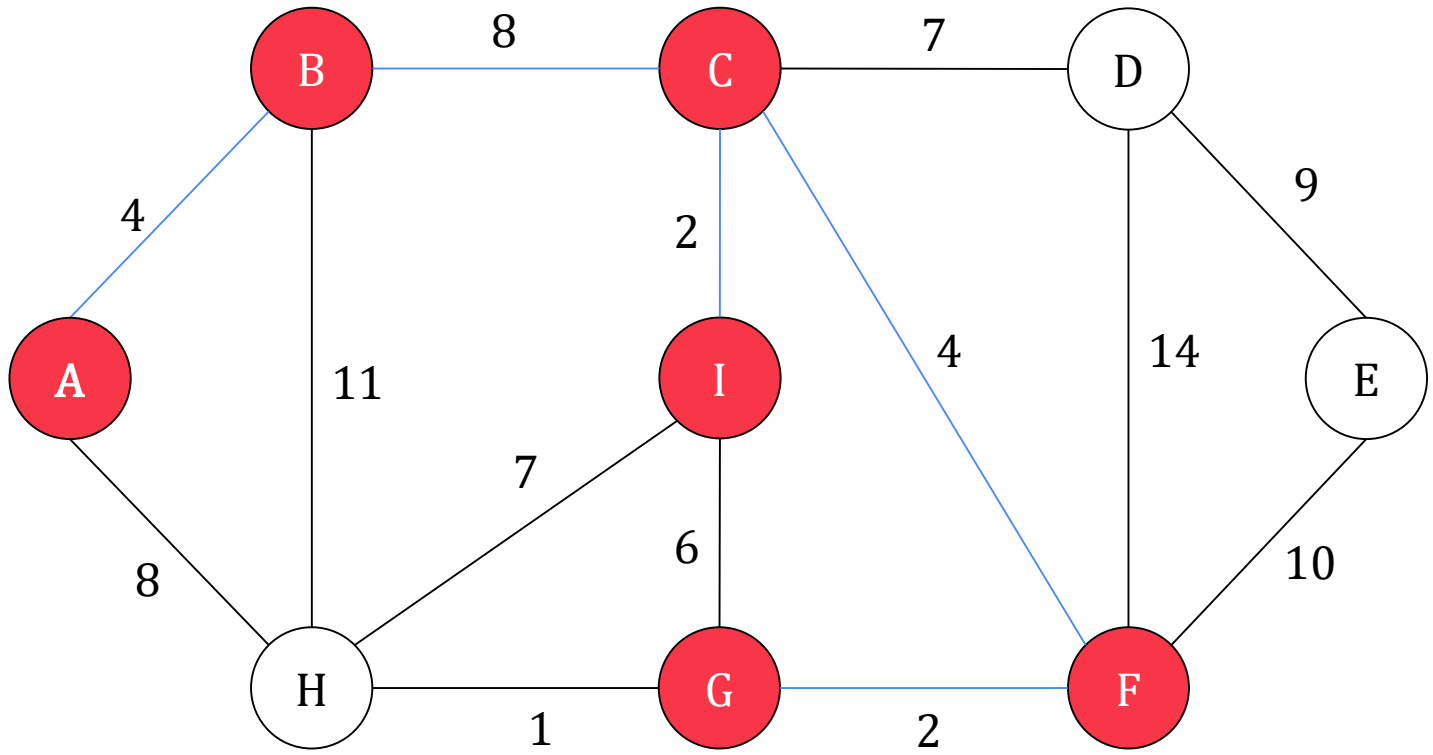
Node	A	B(A)	C(B)	D(C)	E	F(C)	G(I)	H(I)	I(C)
Key	0	4	8	7	∞	4	6	7	2

Prim 알고리즘



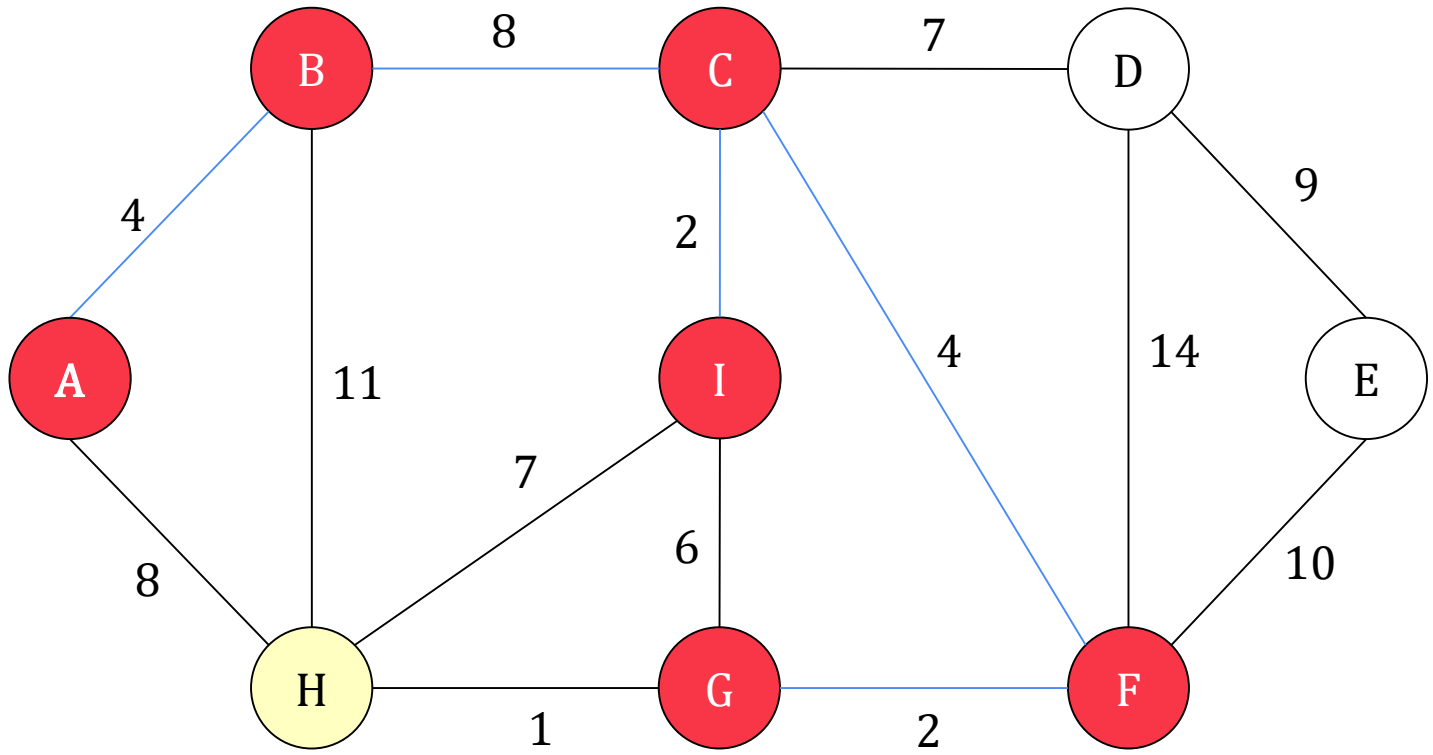
Node	A	B(A)	C(B)	D(C)	E(F)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	**10**	4	**2**	7	2

Prim 알고리즘



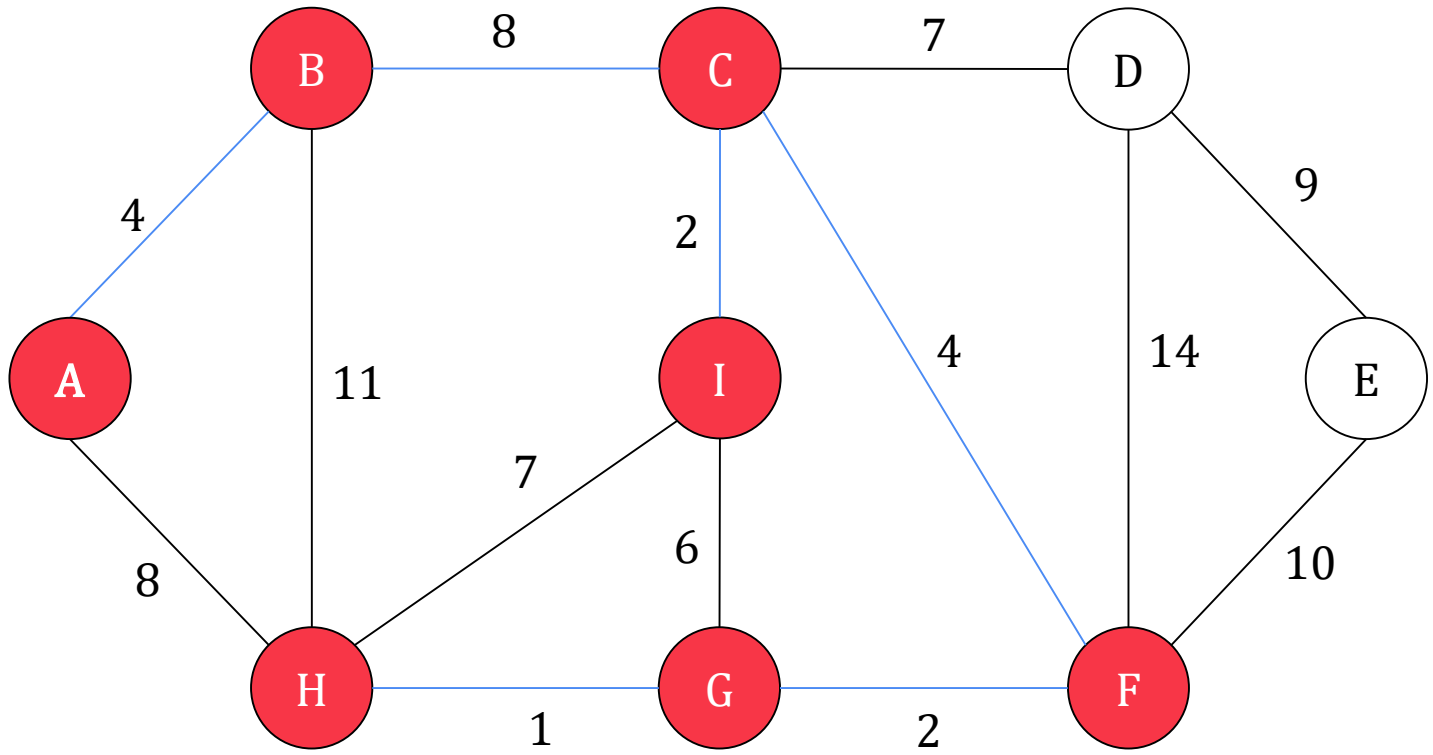
Node	A	B(A)	C(B)	D(C)	E(F)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	10	4	2	7	2

Prim 알고리즘



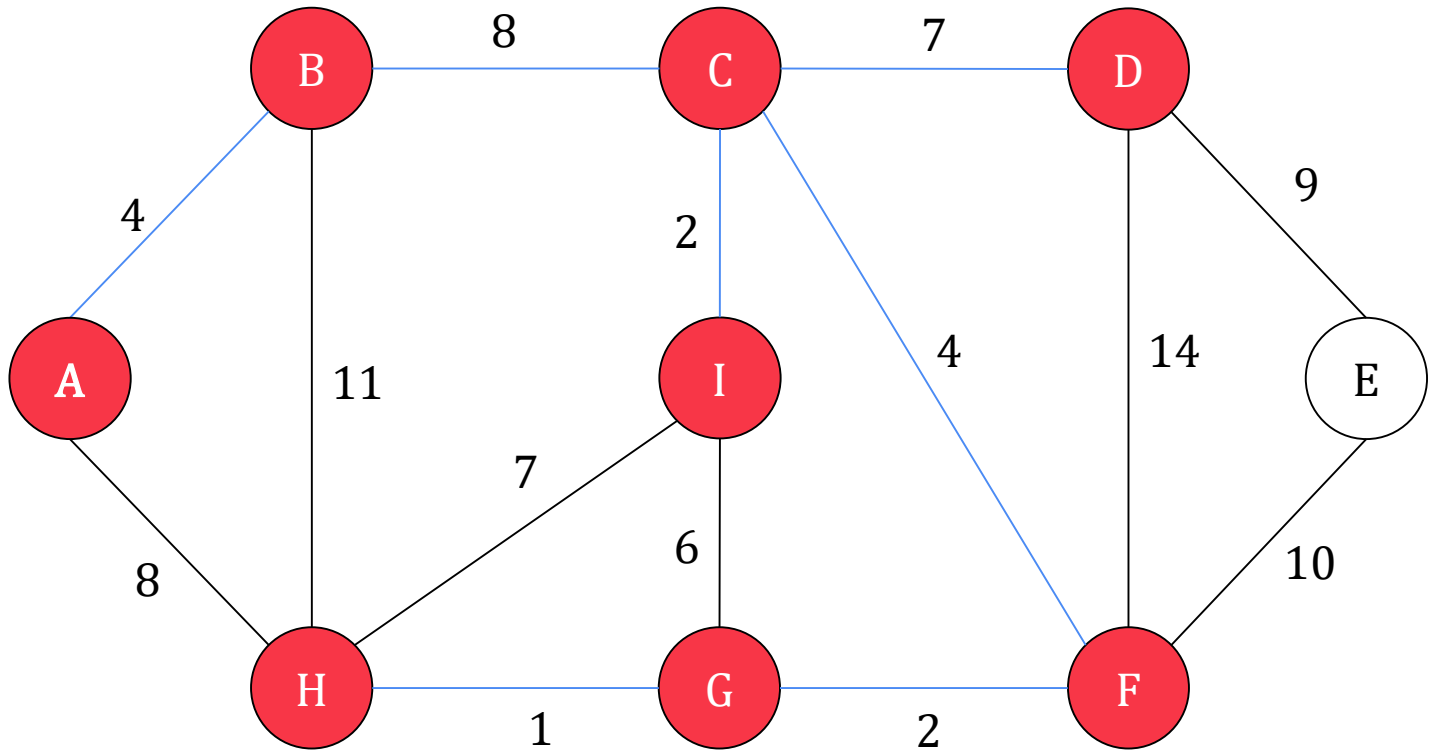
Node	A	B(A)	C(B)	D(C)	E(F)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	10	4	2	1	2

Prim 알고리즘



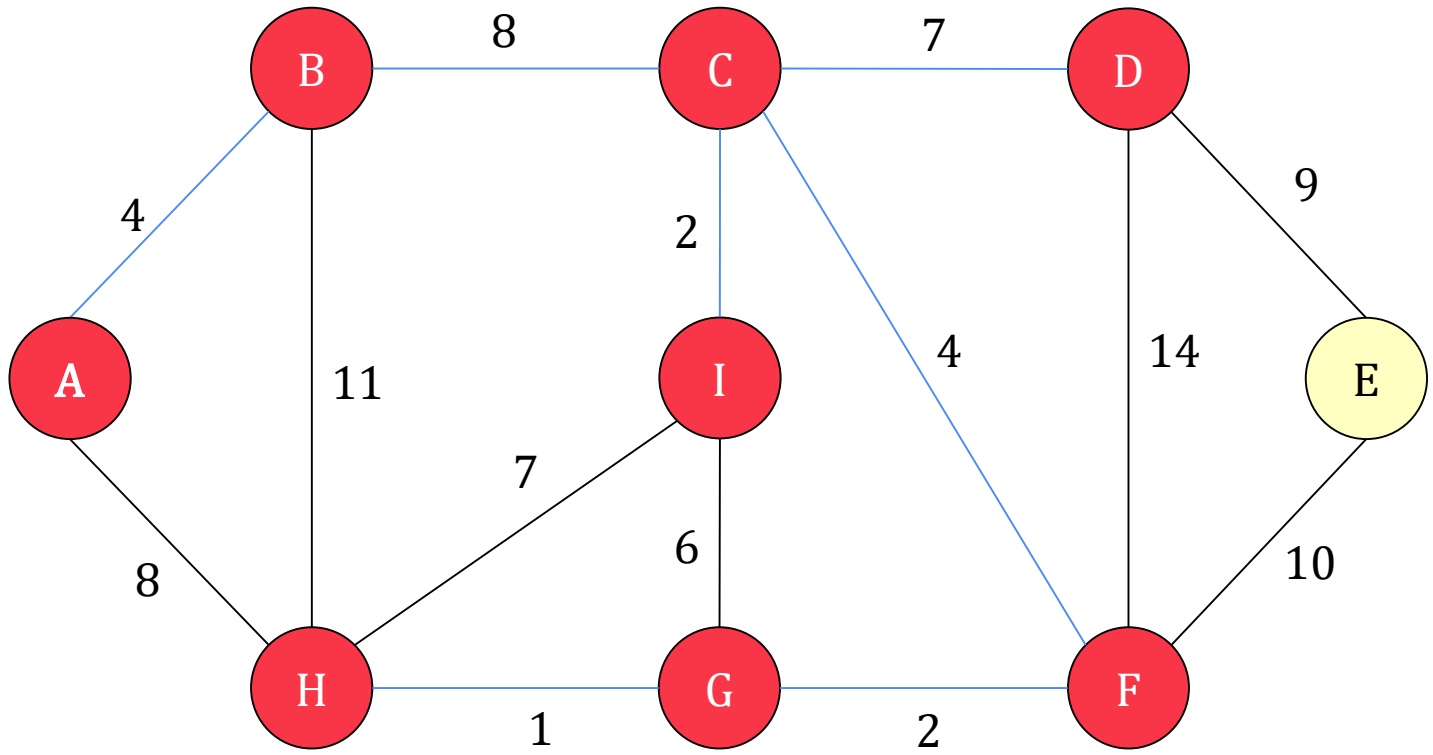
Node	A	B(A)	C(B)	D(C)	E(F)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	10	4	2	1	2

Prim 알고리즘



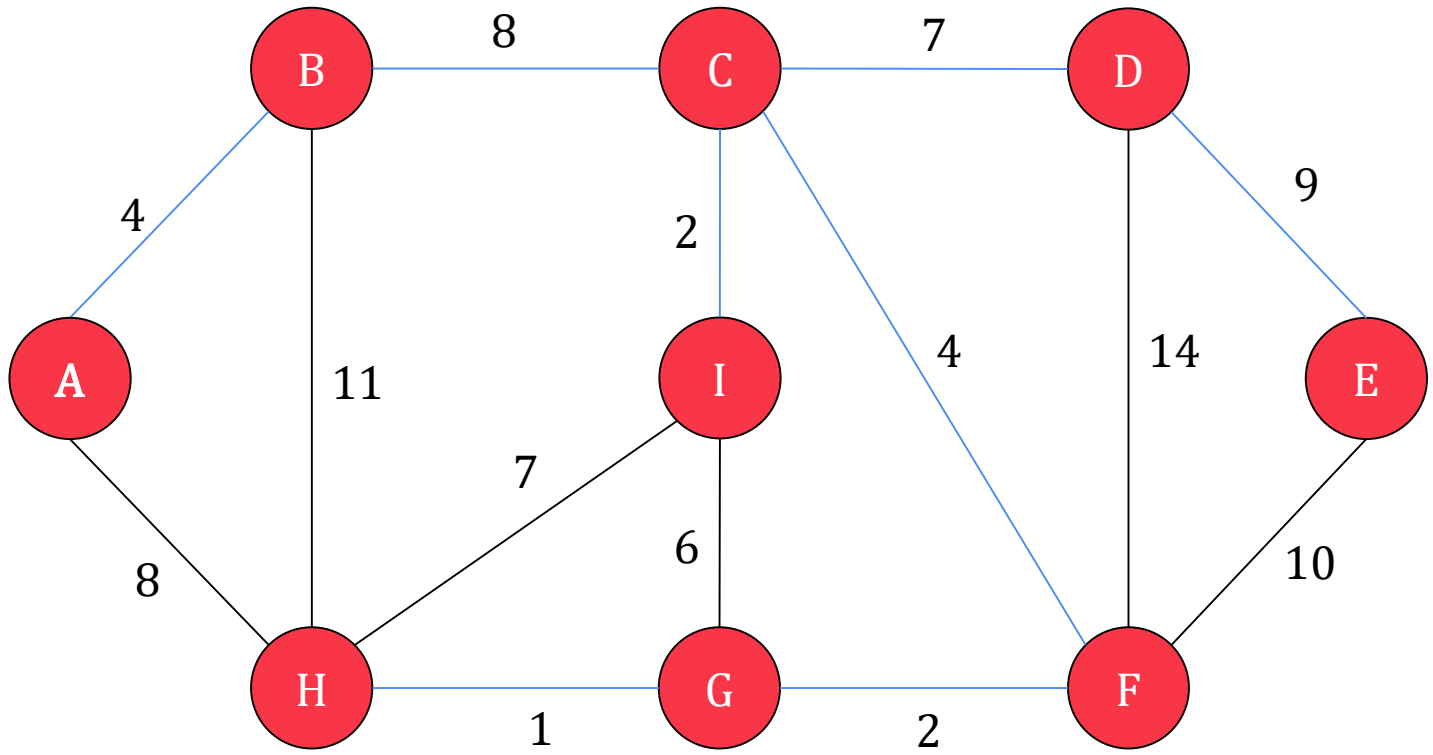
Node	A	B(A)	C(B)	D(C)	E(F)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	10	4	2	1	2

Prim 알고리즘



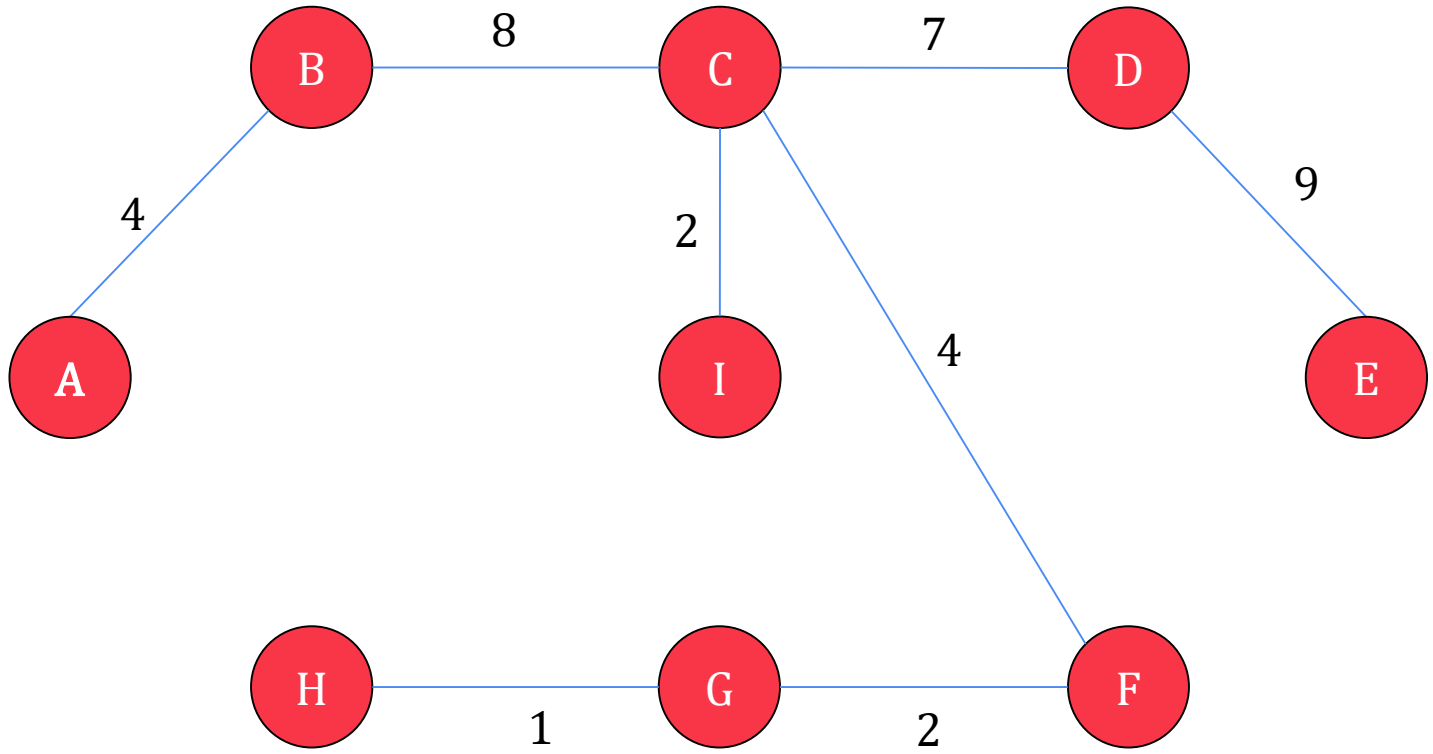
Node	A	B(A)	C(B)	D(C)	E(D)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	9	4	2	1	2

Prim 알고리즘



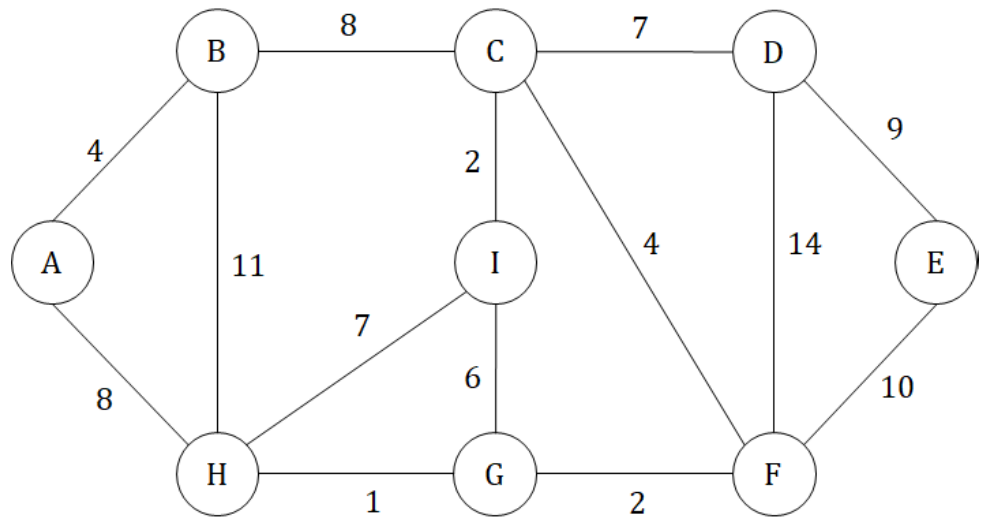
Node	A	B(A)	C(B)	D(C)	E(D)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	9	4	2	1	2

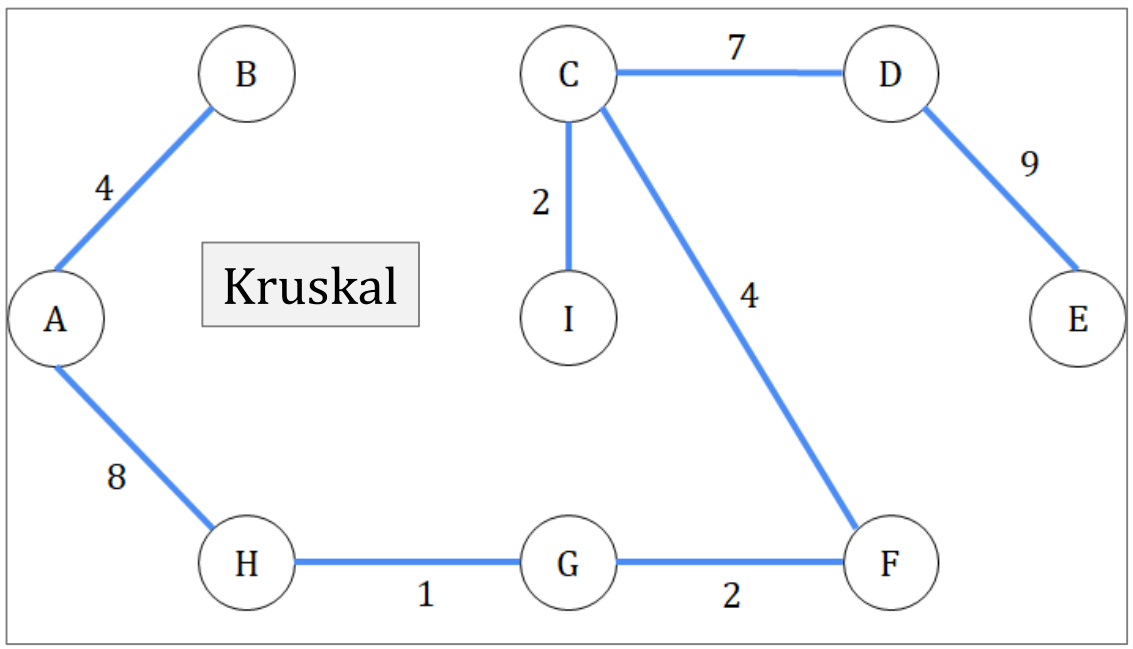
Prim 알고리즘

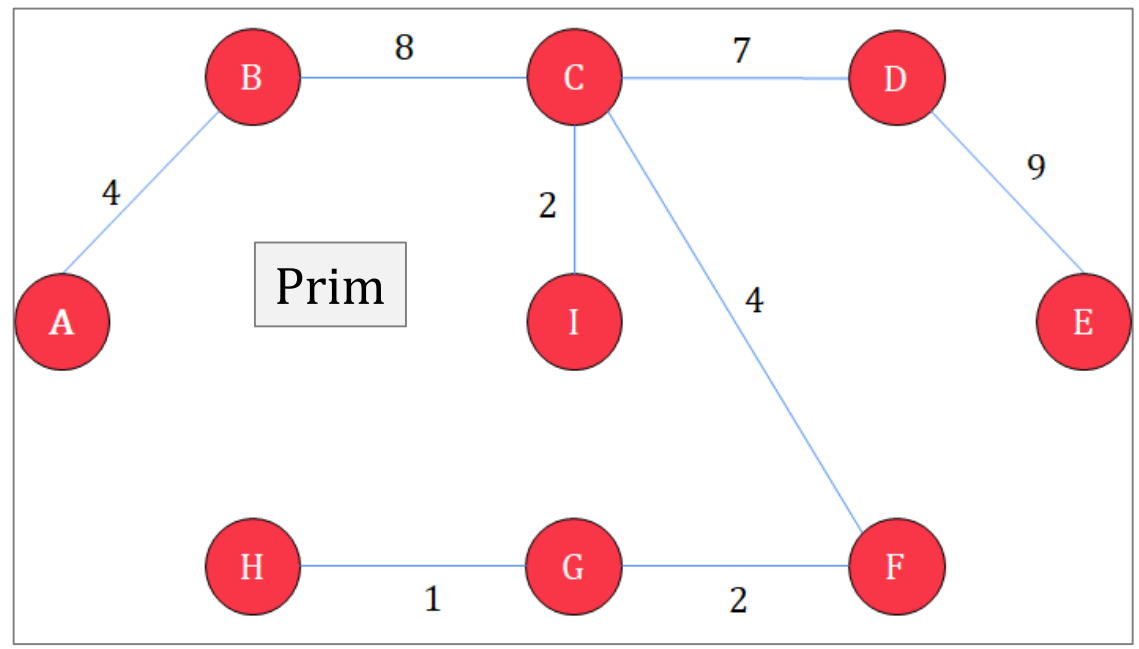


Node	A	B(A)	C(B)	D(C)	E(D)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	9	4	2	1	2

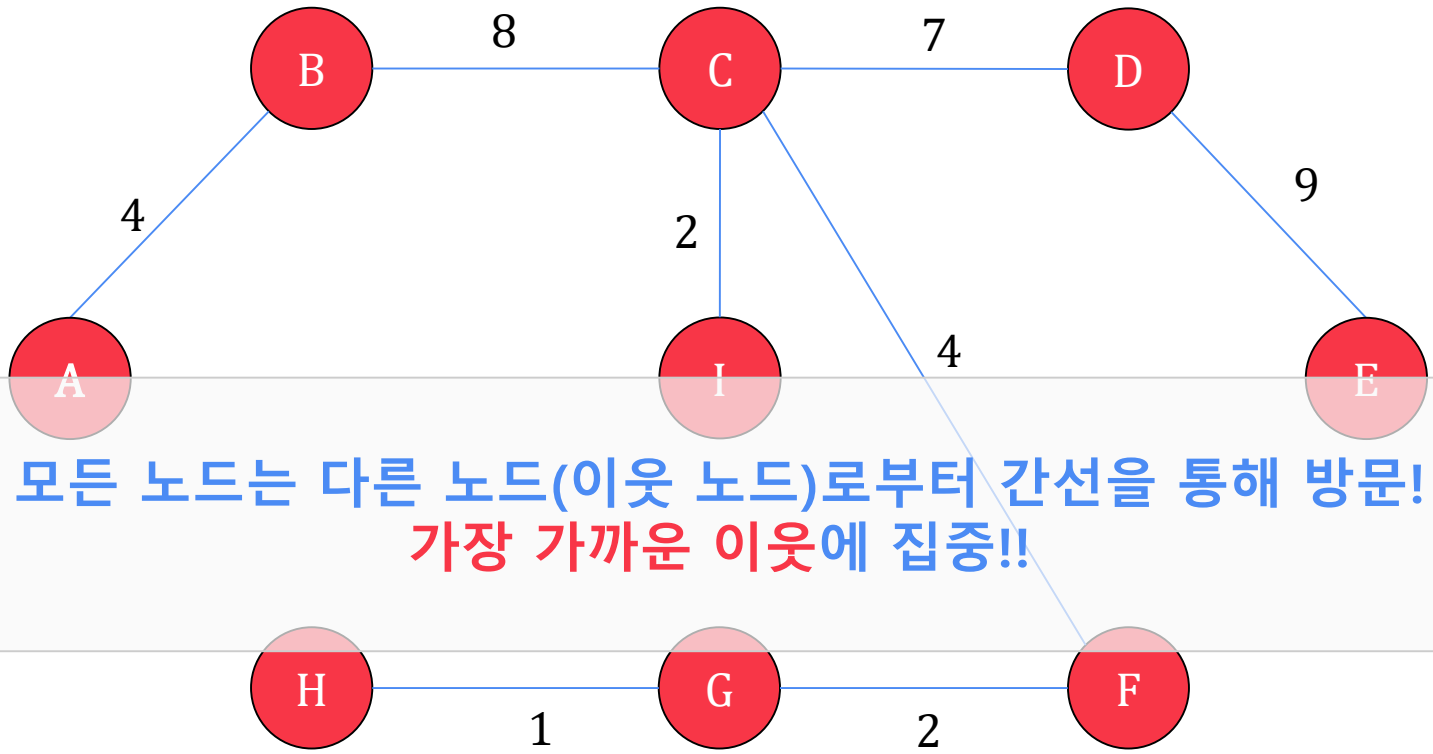
Kruskal vs. Prim 알고리즘







Prim 알고리즘



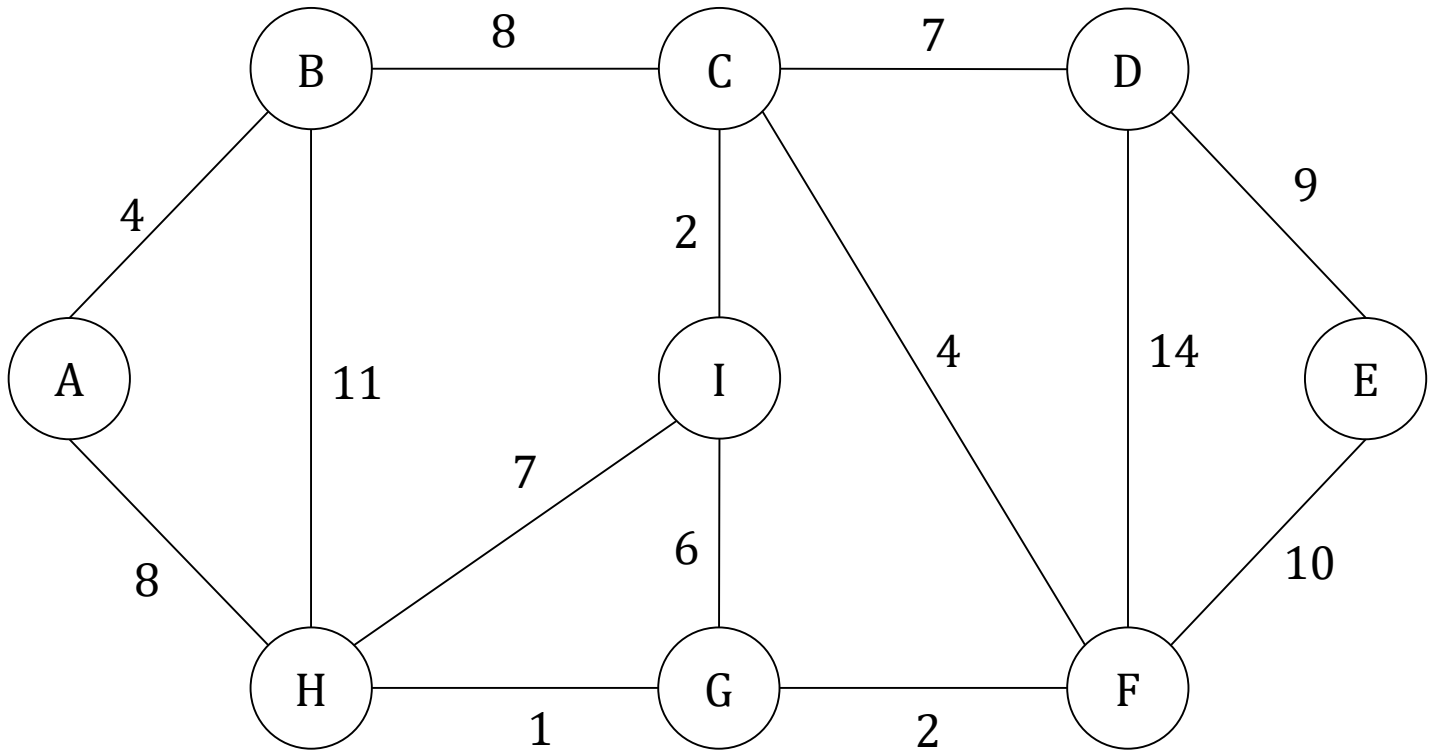
Node	A	B(A)	C(B)	D(C)	E(D)	F(C)	G(F)	H(I)	I(C)
Key	0	4	8	7	9	4	2	1	2

최소 신장 트리 추출 알고리즘

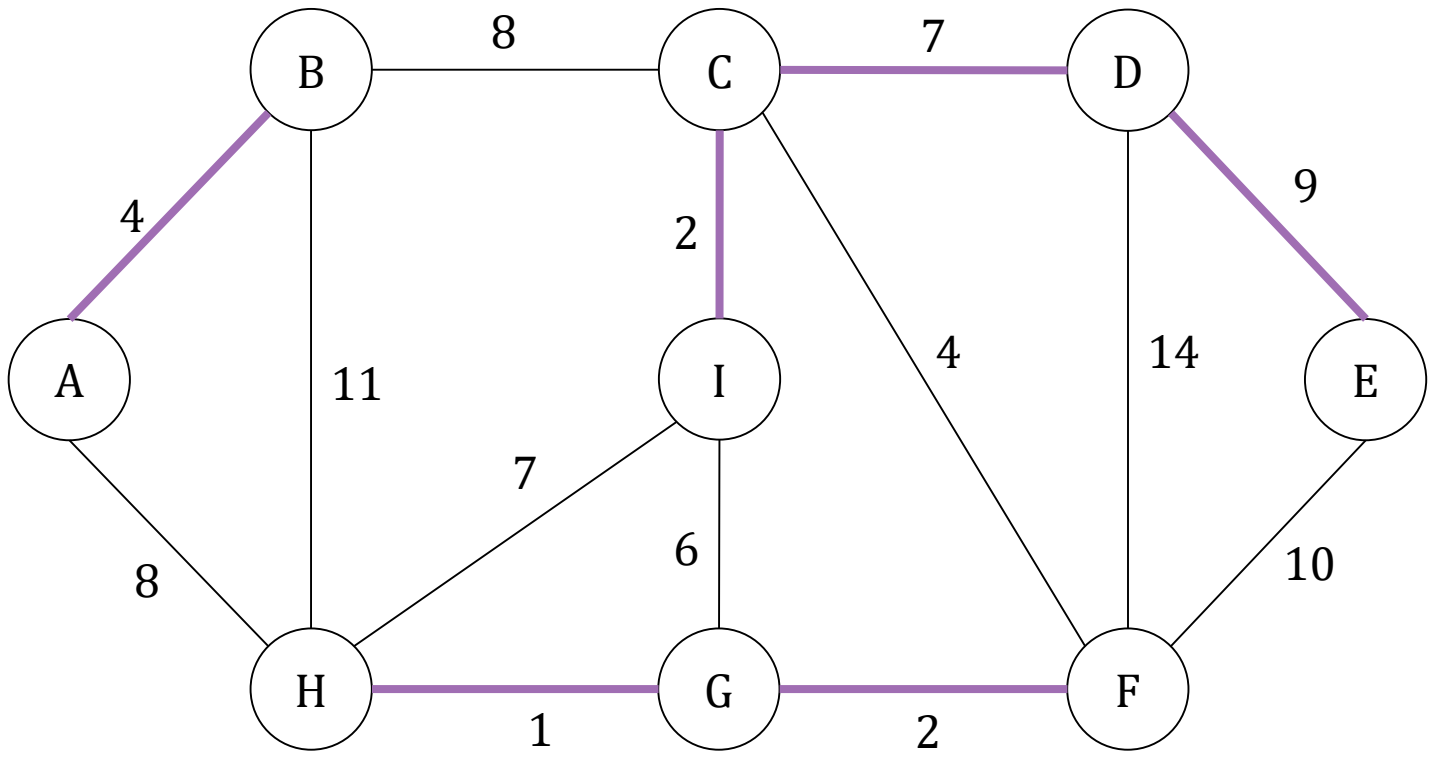
Sollin (Brouvka) 알고리즘

모든 노드는 다른 노드(이웃 노드)로부터 간선을 통해 방문!
가장 가까운 이웃에 집중 → 가족이 되자!

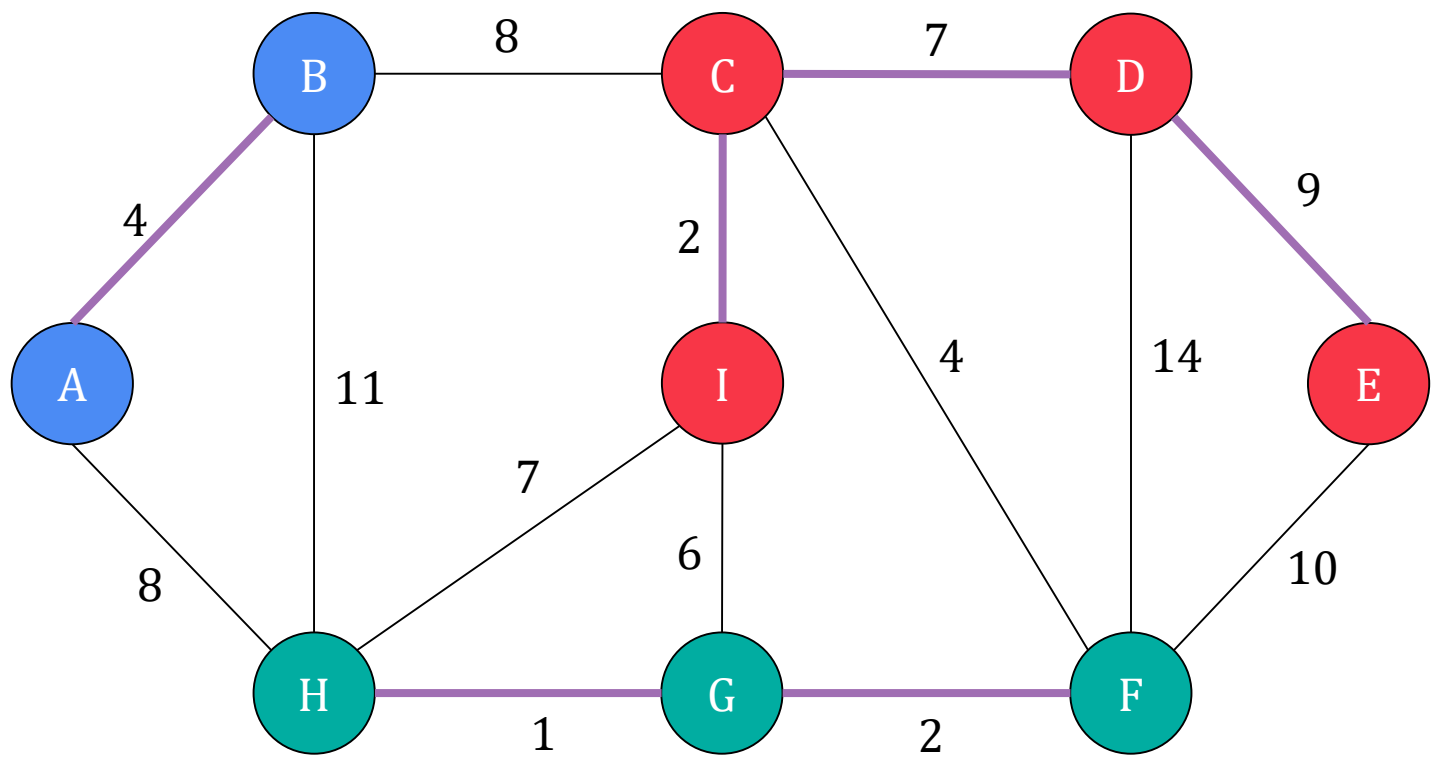
Sollin (Brouvka) 알고리즘



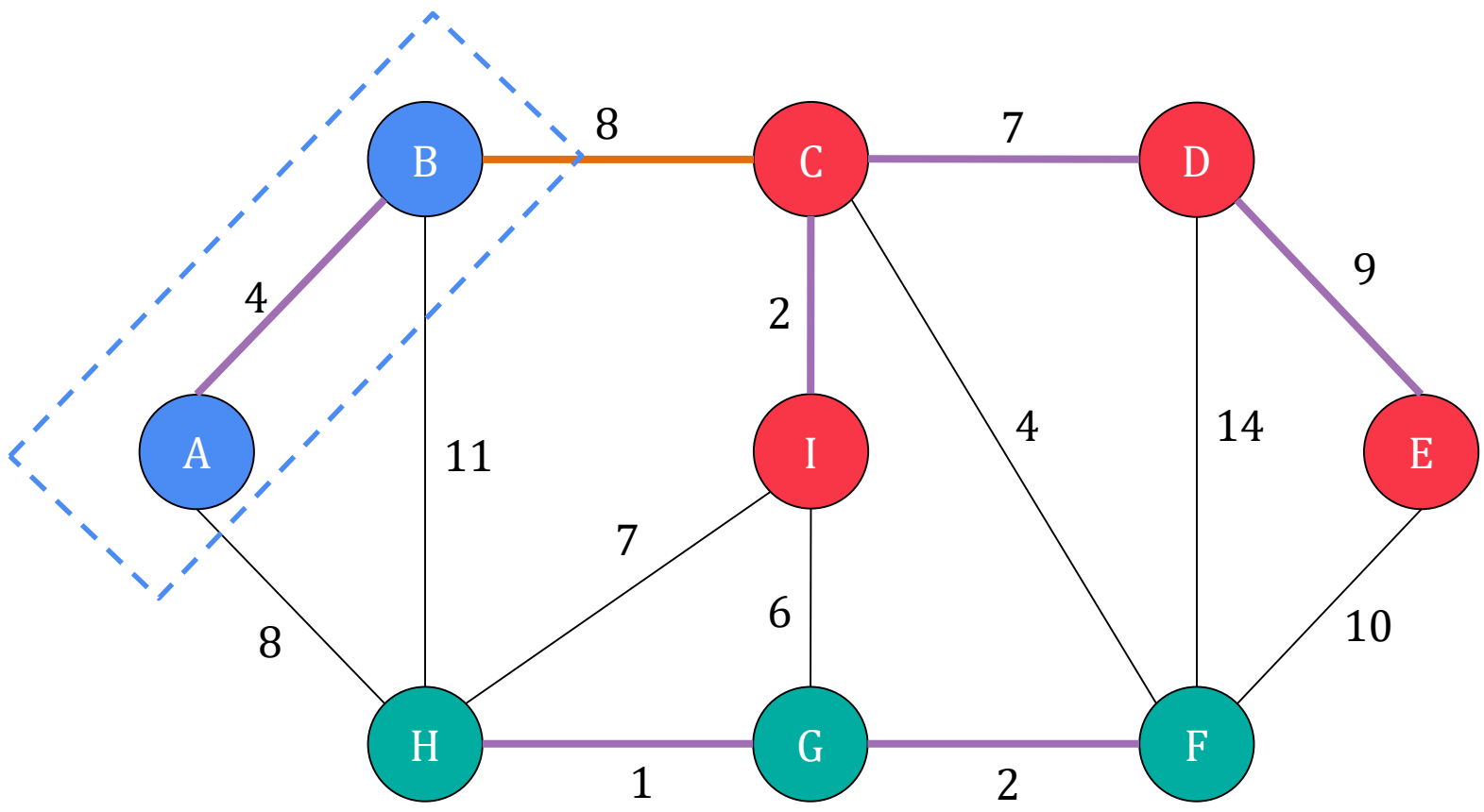
Sollin (Brouvka) 알고리즘



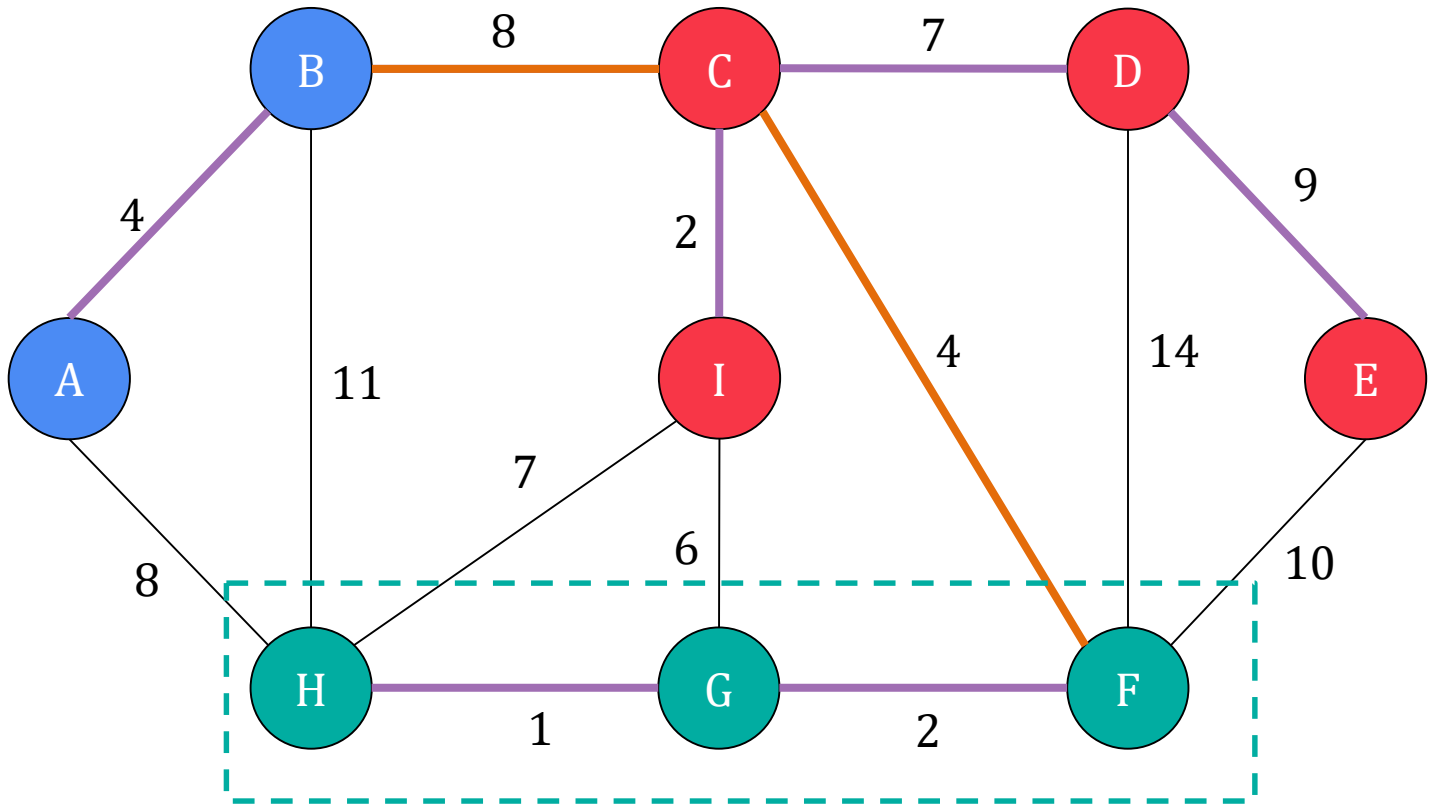
Sollin (Brouvka) 알고리즘



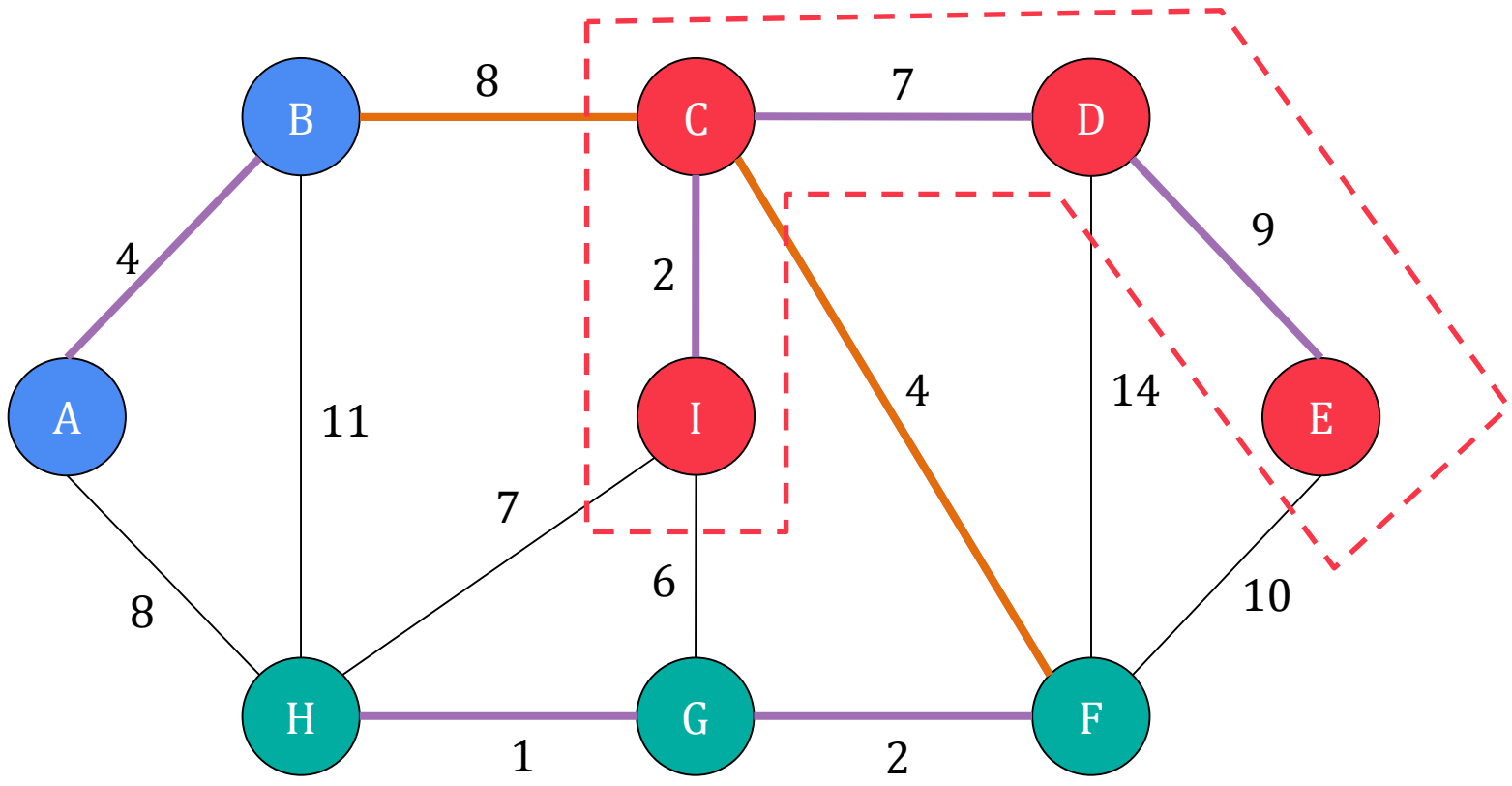
Sollin (Brouvka) 알고리즘



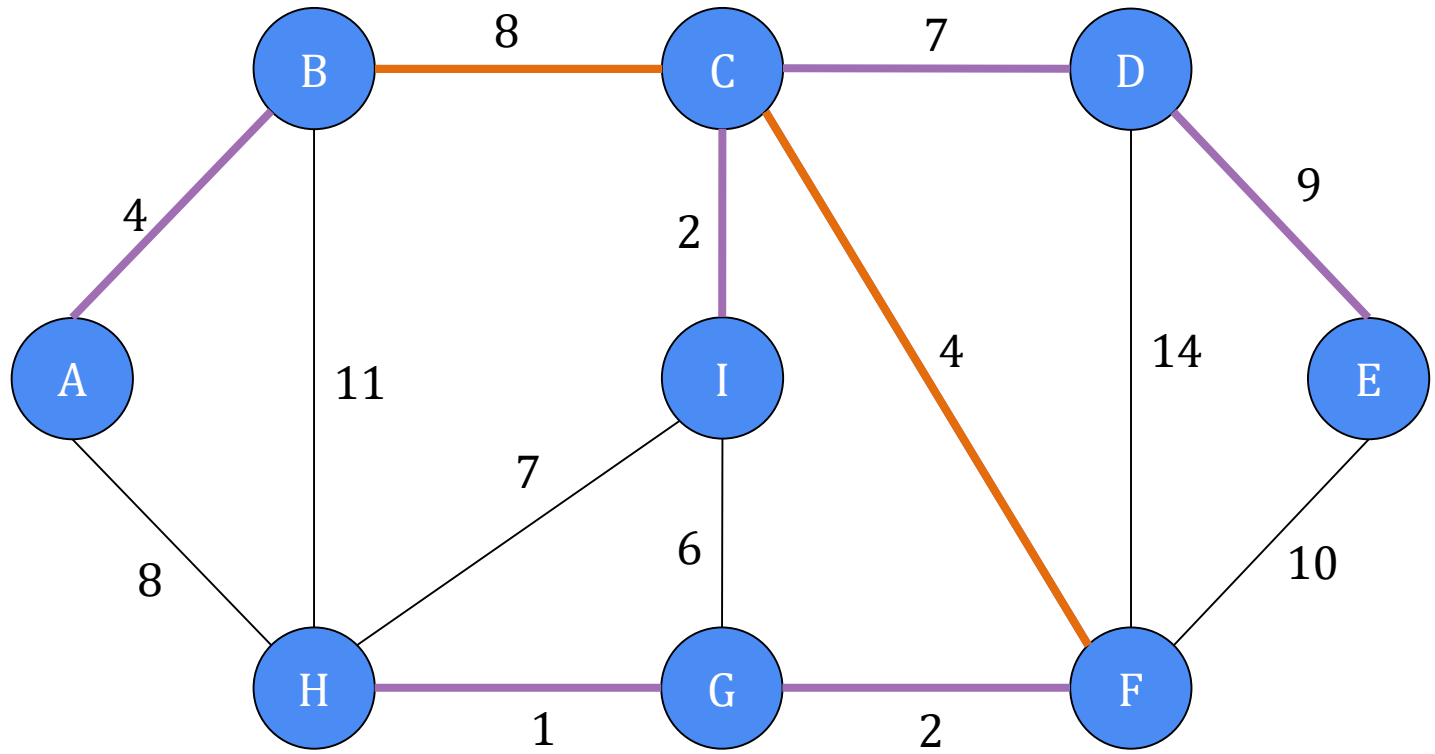
Sollin (Brouvka) 알고리즘



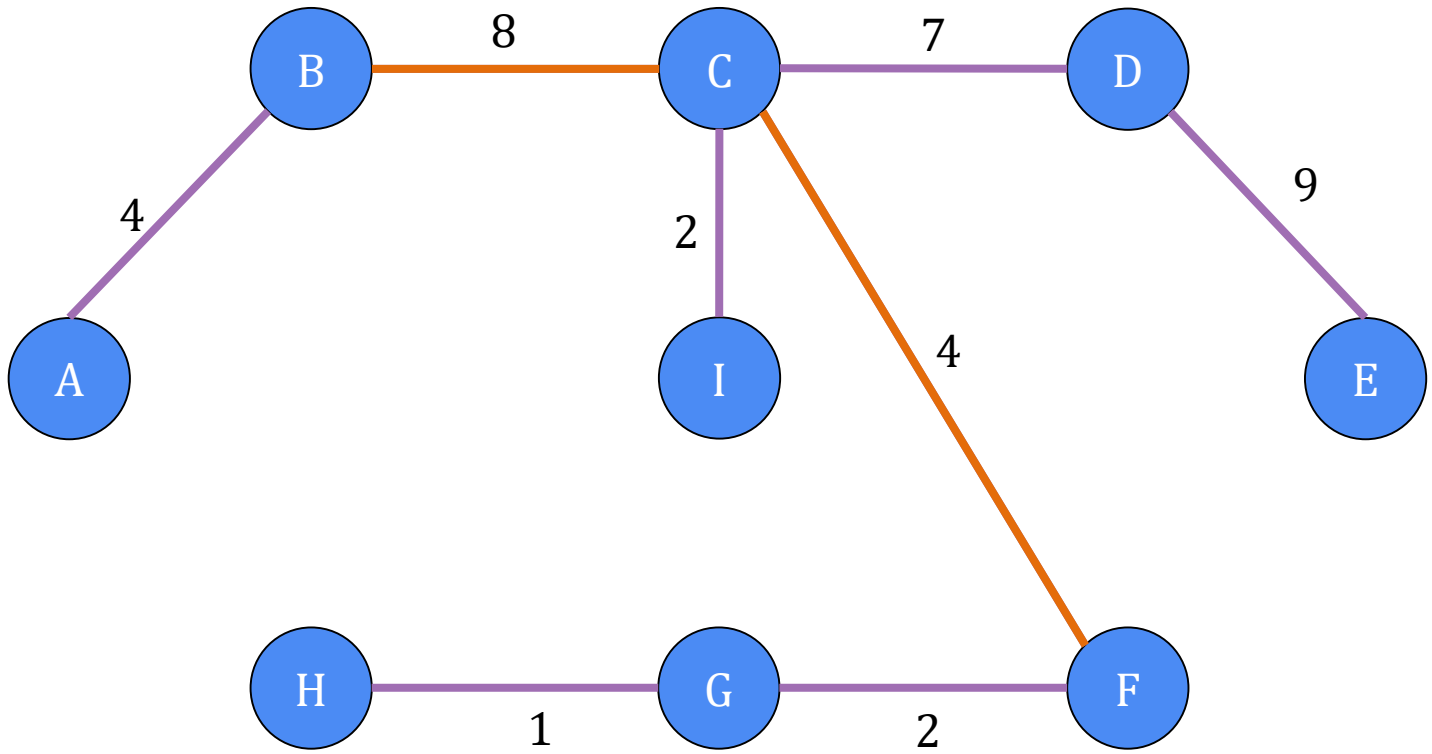
Sollin (Brouvka) 알고리즘



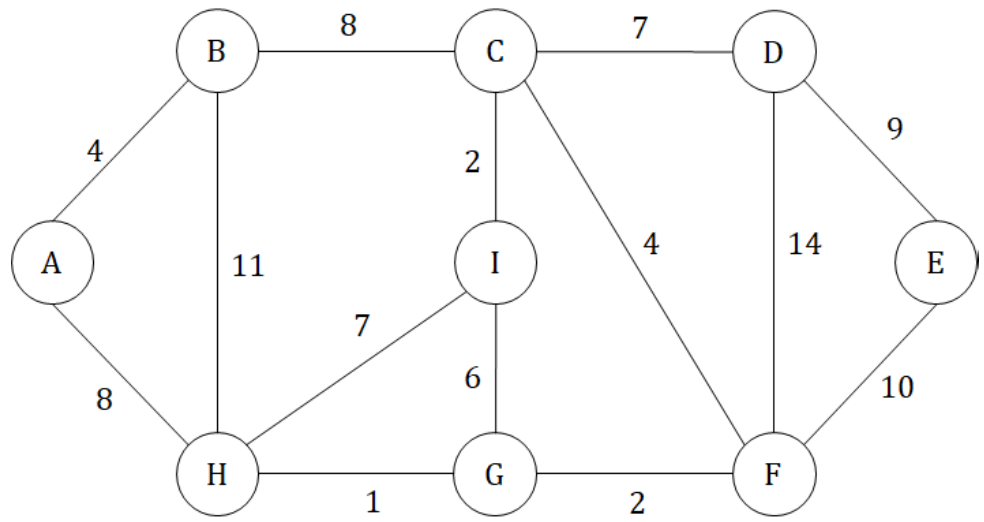
Sollin (Brouvka) 알고리즘

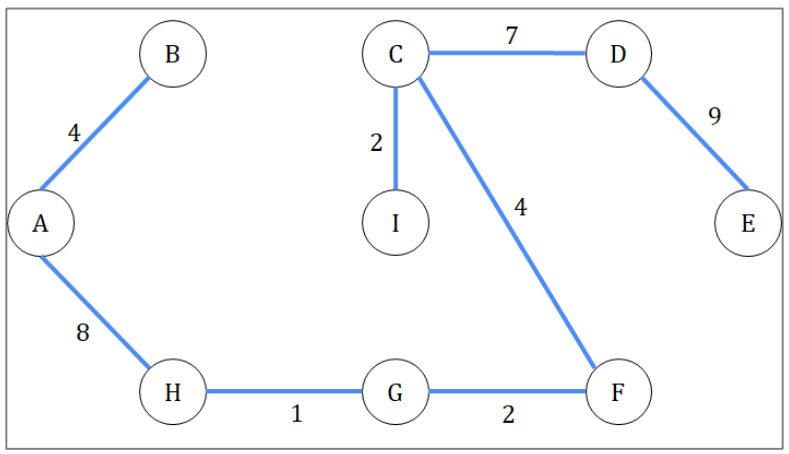


Sollin (Brouvka) 알고리즘

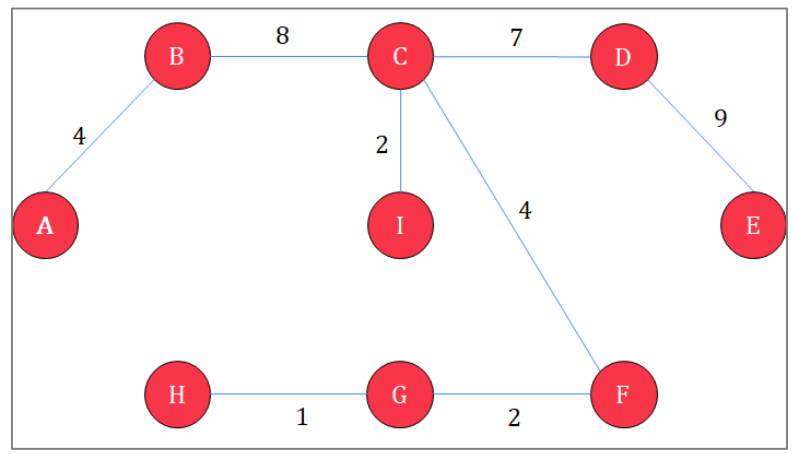


Kruskal vs. Prim vs. Sollin 알고리즘

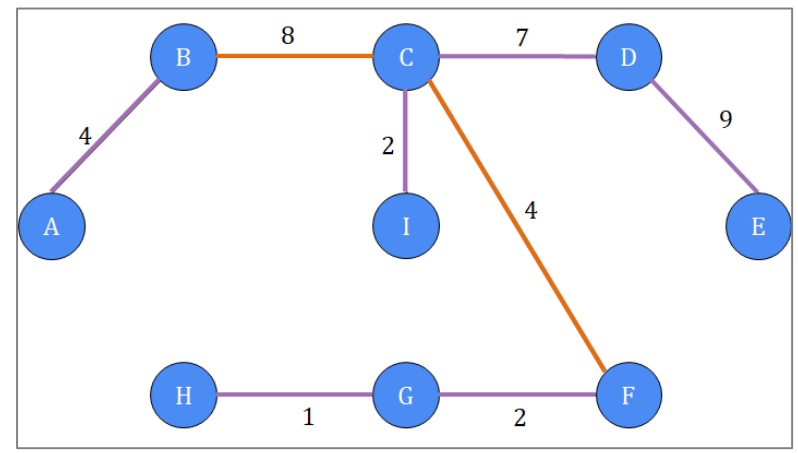




Kruskal



Prim



Sollin

Q & A