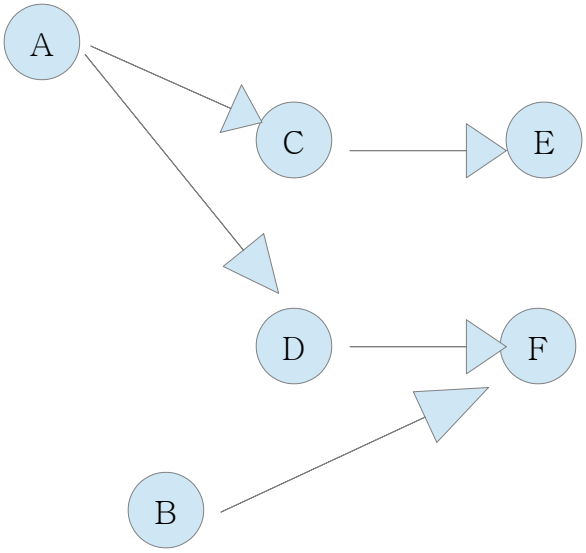
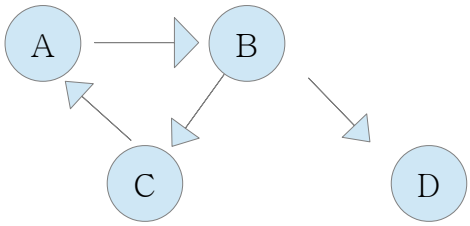
위상정렬 (Topology Sort): 방향그래프에서 간선으로 주어진 정점 간 선후 관계를 위배하지 않도록 나열하는 정렬



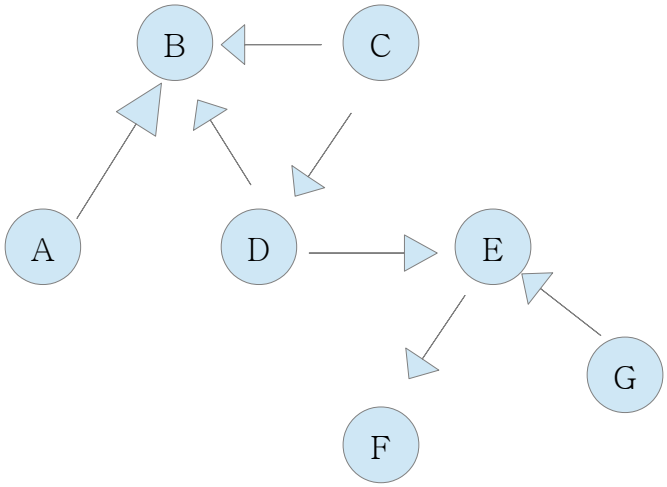
그리기 1: A 부터 F 까지 정렬하되, 선후 관계를 따르게 정렬한다. 예를 들어 A가 C보다 앞에 위치 해야 한다.



그리기 2: 그래프에 사이클이 있는 경우 위상정렬을 할 수 없다. 위 예시의 경우 A는 B보다 먼저 나와야 하지만, 또 한편으로는 B가 A보다 먼저 나와야 한다.

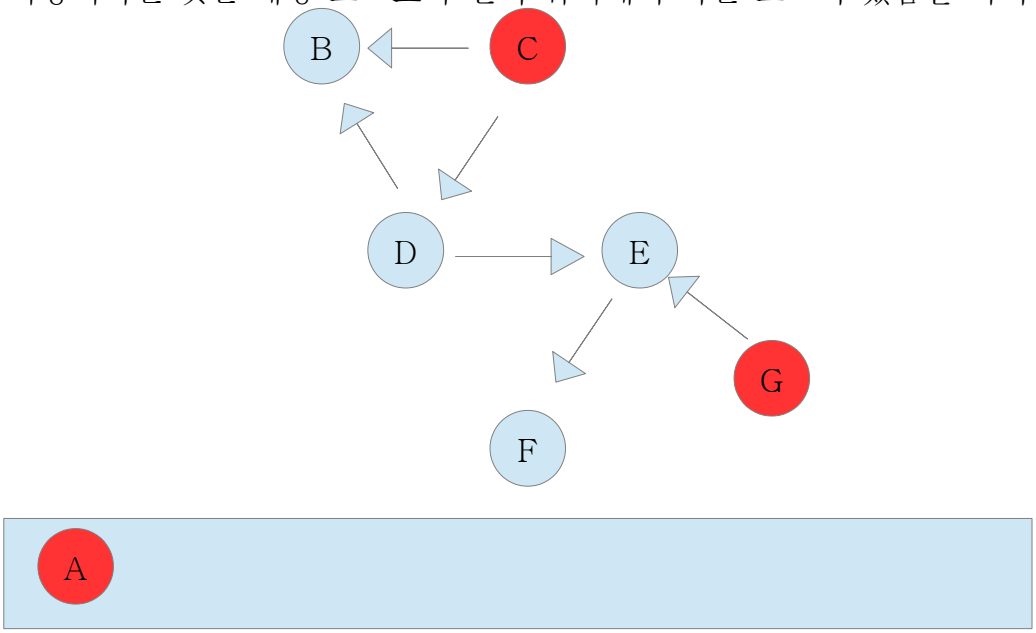
DAG(Directed Acyclic Graph): 사이클이 존재하지 않는 방향 그래프, 위상정렬은 사이클이 존재하지 않는 방향그래프에서 할 수 있다.

위상정렬의 예시



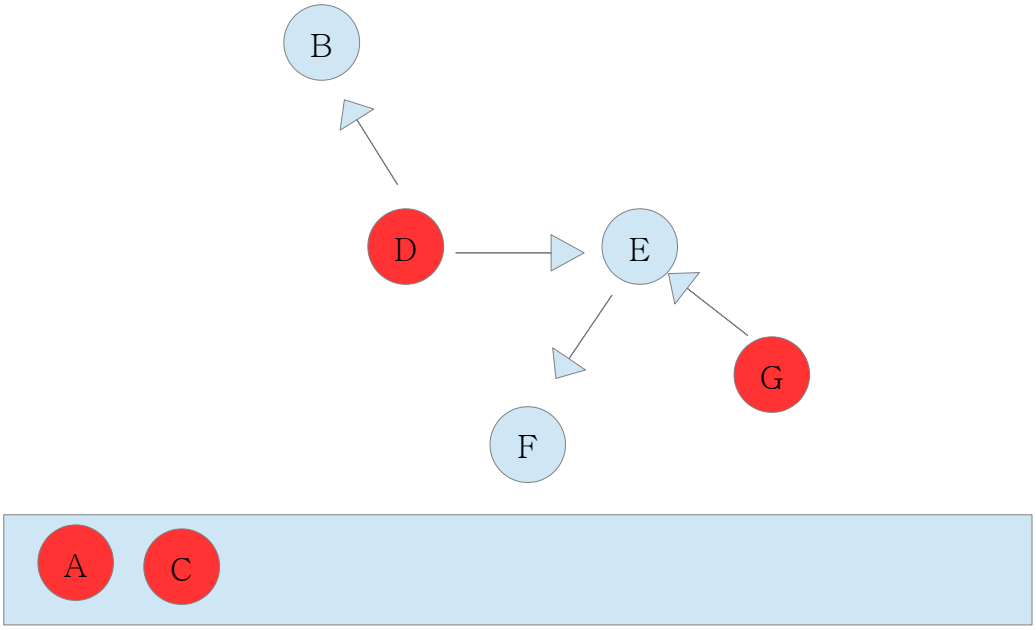
그리기 3: 이 그래프에서 위상정렬을 수행해 본다.

위상정렬 시 처음에 위치할 수 있는 노드 후보는 [indegree가 0이어야] 한다. (어떤 노드의 indegree가 1 이상이라는 것은 해당 노드보다 먼저 위치해야 하는 노드가 있음을 의미한다.)



그리기 4: 특정 노드를 방문 처리 후 해당 노드와 해당 노드에서 나가는 간선을 그래프에서 지운다.

그래프에서 해당 노드를 방문 처리 후 남은 노드는 해당 노드 뒤에 위치함이 분명해 지므로, 그래프에서 해당 노드 및 해당 노드에서 나가는 간선을 지워도 위상정렬에 영향을 끼치지 않는다.



그리기 5: 그래프에서 indegree가 0인 노드가 없어질 때 까지 이 과정을 반복한다.

구현의 편의를 위한 성질 (O(V+E) 시간복잡도로 구현하기 위한 방법)

1. 정점과 간선을 실제로 지울 필요 없이 미리 (각 노드의) indegree 값을 저장해 두었다가 (노드와 간선을 지울 때 마다) 도착하는 정점들의 indegree 값만 1 감소시켜도 과정을 수행할 수 있다.
2. indegree가 0인 정점을 구하기 위해 매번 모든 정점을 다 확인하는 대신, 목록을 따로 저장하고 있다가 직전에 제거한 정점에서 연결된 정점들만 추가한다.
   => indegree를 1 감소시킨 후 해당 노드의 indegree가 0이 되면 특정 queue에 push한다.

위상정렬 알고리즘

1. 맨 처음 모든 간선을 읽으며 indegree 테이블을 채운다.
2. indegree가 0인 정점들을 모두 queue 에 넣는다.
3. queue에서 정점을 꺼내어 위상 정렬 결과에 추가한다.
4. 해당 정점으로부터 연결된 모든 정점의 indegree의 값을 1 감소시킨다. 이 때 indegree가 0가 되었다면 그 정점을 큐에 넣는다.
5. queue가 빌 때 까지 3, 4 번 과정을 반복한다.

*. 정렬의 결과 모든 노드를 전부 거치지 못하면 해당 방향 그래프는 cycle이 존재하는 것이다.
*. 큐에는 각 노드가 최대 한 번씩 들어가고, indegree 감소는 각 간선마다 최대 한 번씩 이루어진다. 위상정렬의 시간복잡도는 O(V+E)가 된다.