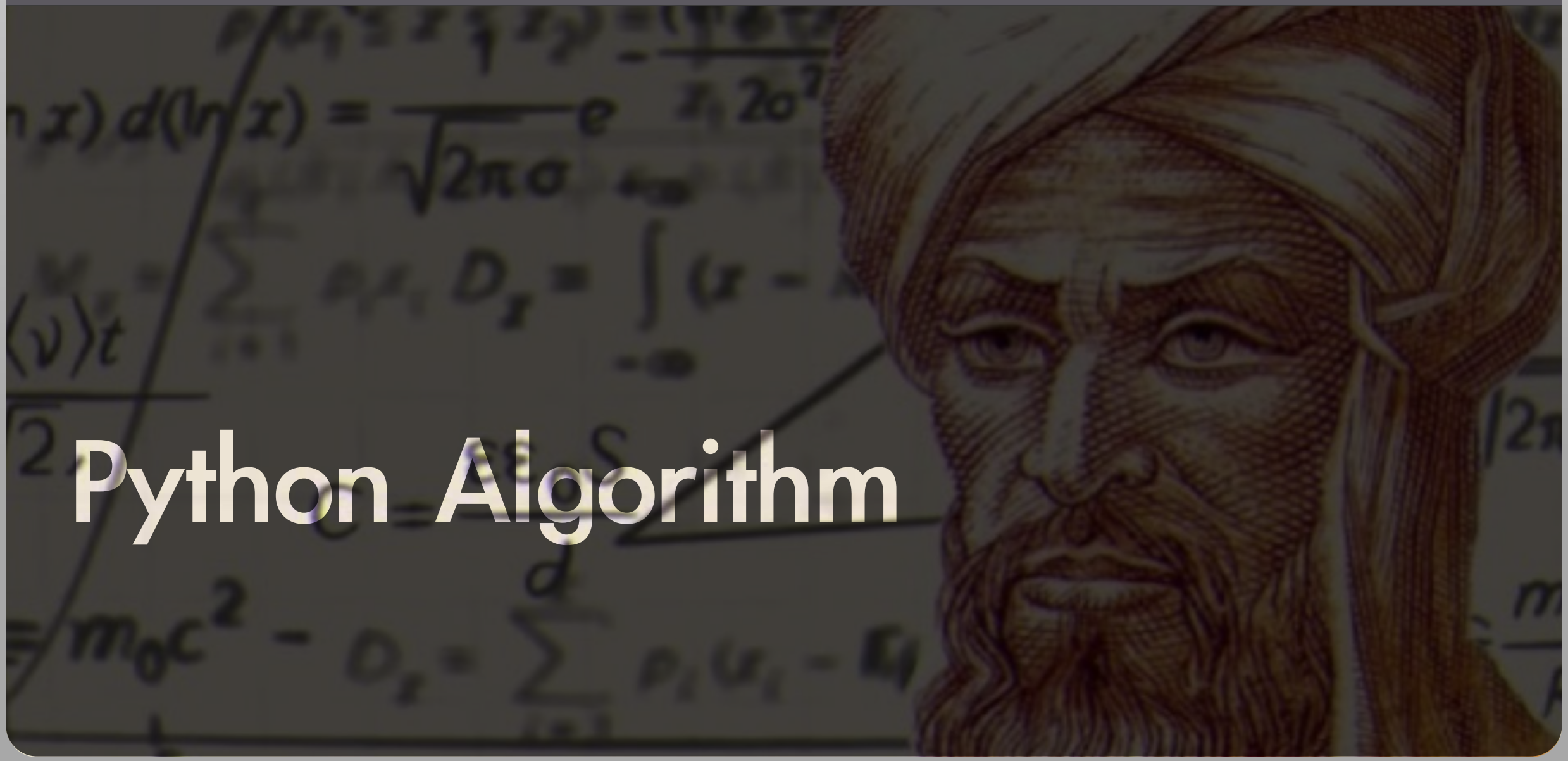
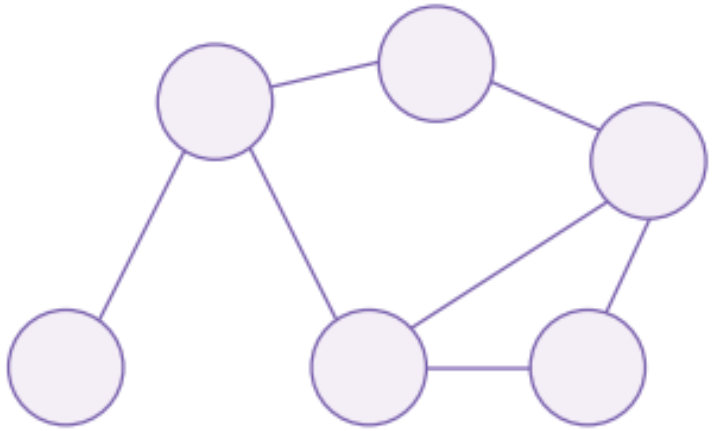

Python Algorithm

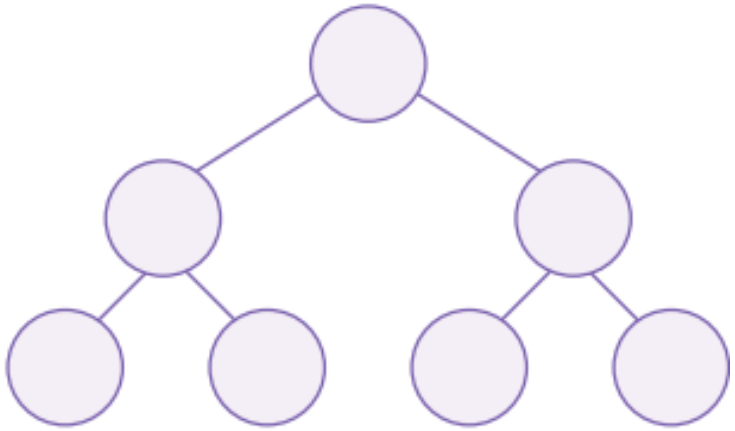
비선형 자료구조

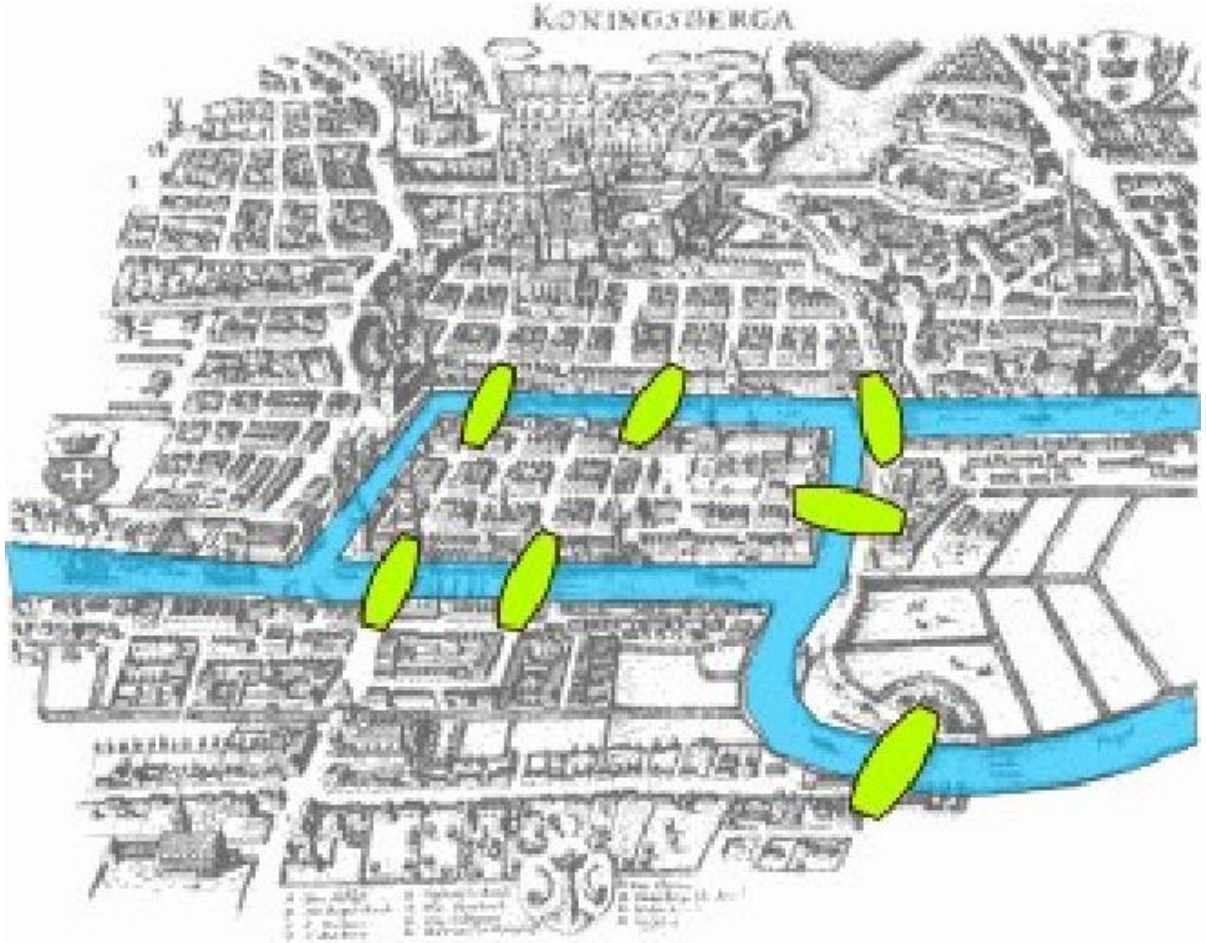
데이터 요소가 순차적으로 배열되지 않는 자료구조

그래프



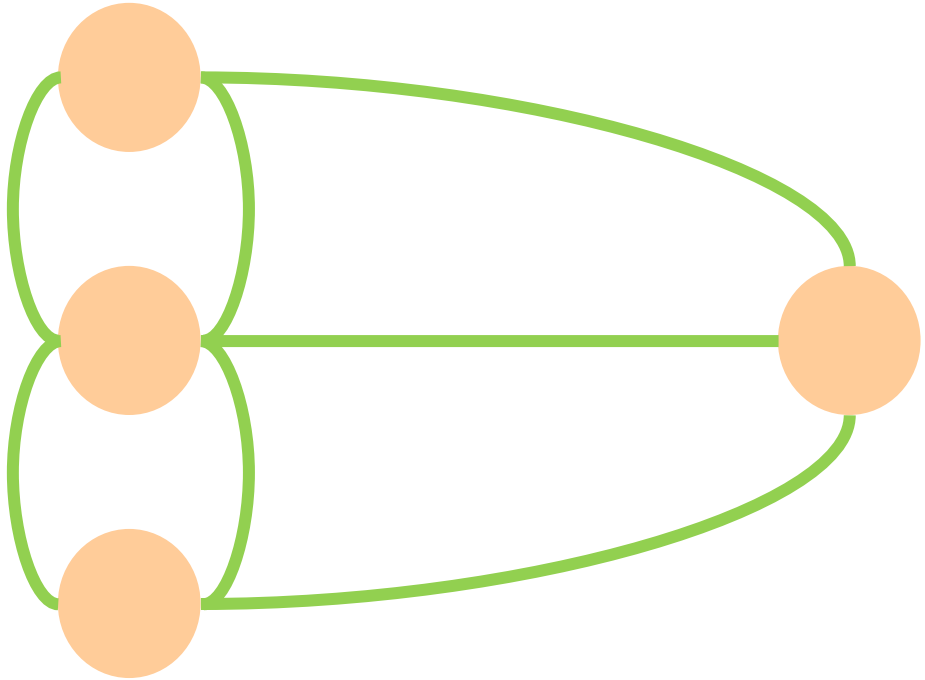
트리




KONINGSBERGA

비선형 자료구조 - 그래프

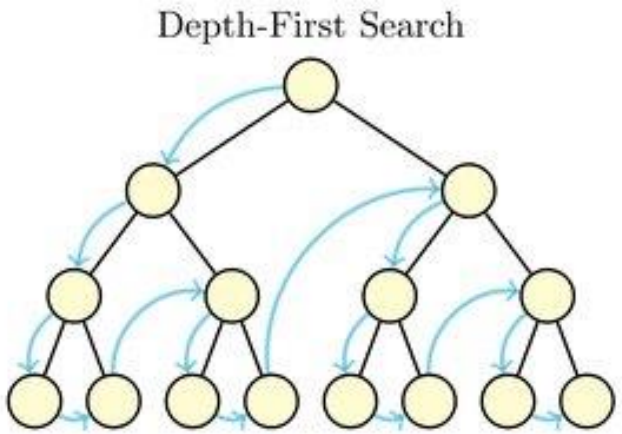
노드와 각 노드를 연결하는 간선으로 구성된 자료 구조



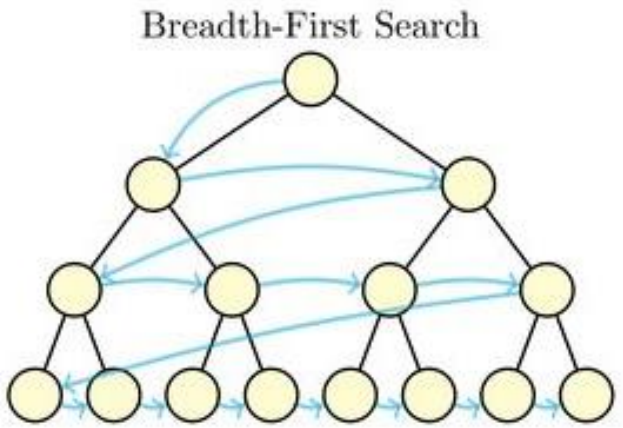
비선형 자료구조 - 그래프의 특징

항목	설명
연결 방법	어떤 방향이든 가능 (양방향, 단방향, 무방향)
구성	어떻게 연결되어도 상관없음 (순환 구조 가능)
시작 노드의 기준	어느 쪽에서 시작해도 무방
다른 노드와의 관계	수평 관계
순회 방법	DFS, BFS

비선형 자료구조 - 그래프 순회(그래프 탐색)



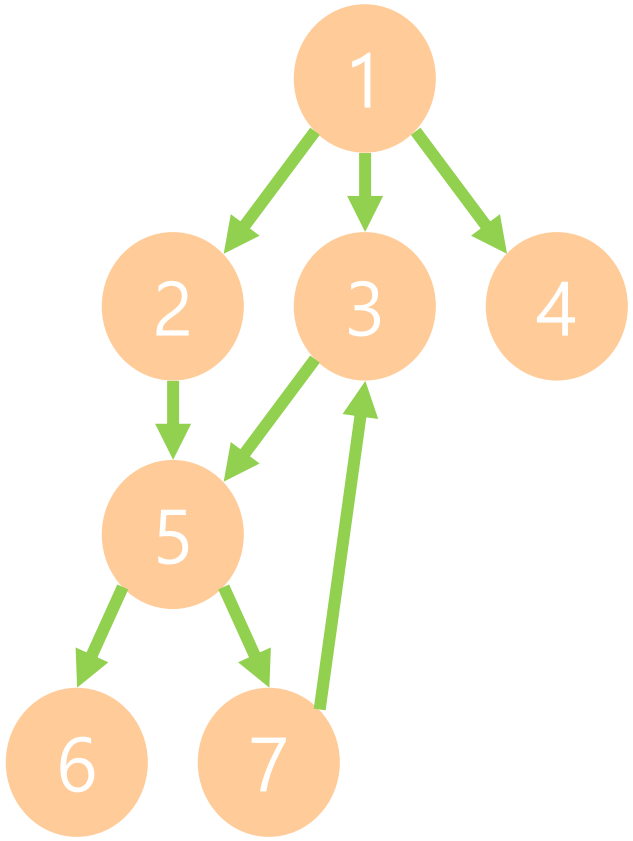
깊이 우선 탐색 (DFS)
- 스택 또는 재귀를 통해 구현
- 정답이 몇 가지인지 물어보는 문제에 적합



너비 우선 탐색 (BFS)
- 큐를 이용해 구현
- 최단 경로를 찾는 문제에 적합

비선형 자료구조 - 그래프의 표현 (인접 행렬)

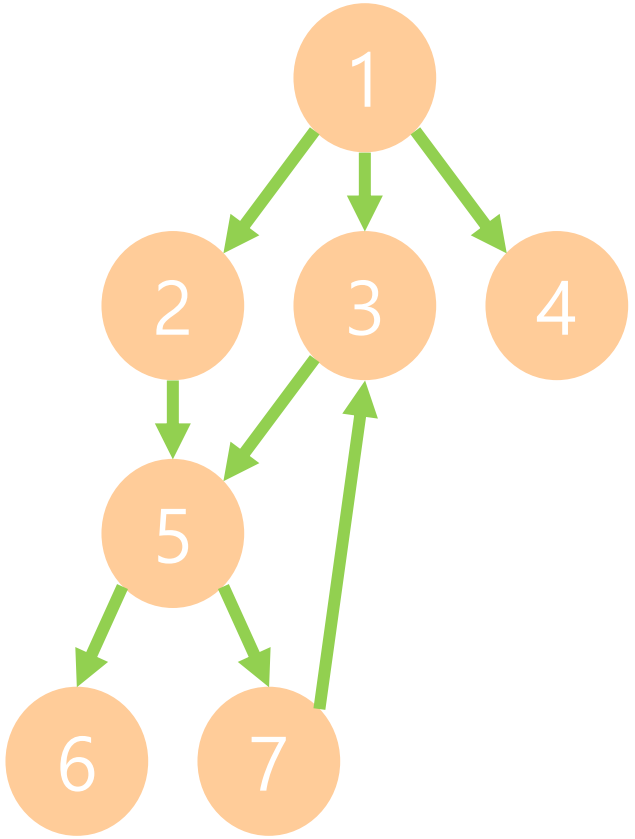
인접 행렬은 2차원 노드의 개수만큼 2차원 배열을 만듭니다.



	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	0	0	0	0	1	0	0
3	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	0	1	1
6	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0

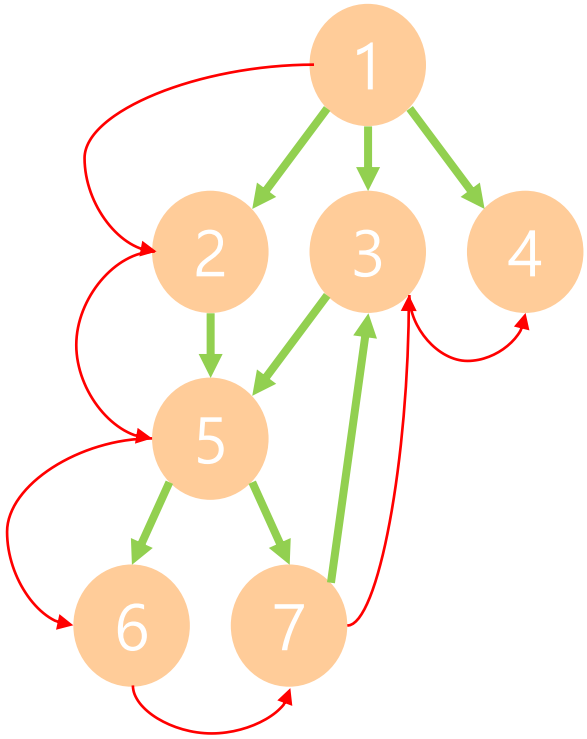
비선형 자료구조 - 그래프 표현 (인접 리스트)

인접 리스트는 출발 노드를 키로, 도착 노드를 값으로 표현합니다.




```
graph = {
    1 : [2,3,4],
    2 : [5],
    3 : [5],
    4 : [],
    5 : [6, 7],
    6 : [],
    7 : [3]
}
```

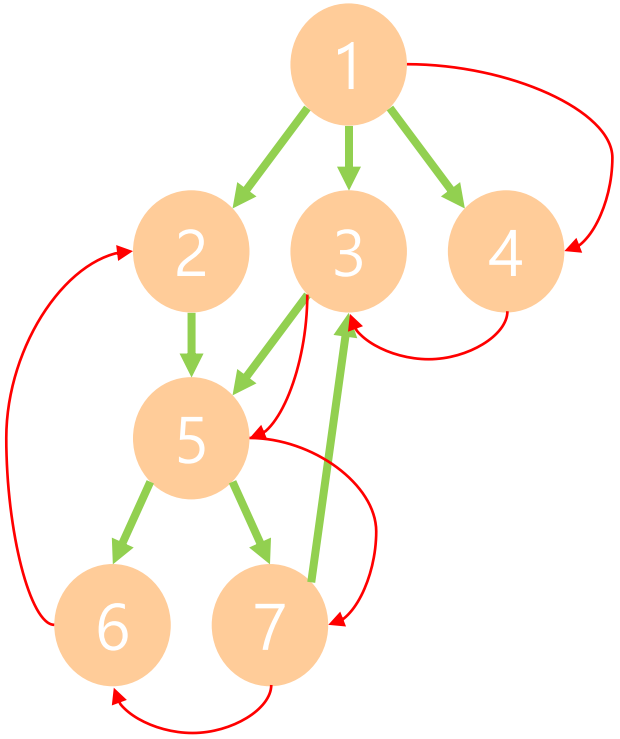
비선형 자료구조 - 재귀구조로 구현한 깊이 우선 탐색 (DFS)




```
def recursive_dfs(v, discovered=[]):
    discovered.append(v)
    for w in graph[v]:
        if not w in discovered:
            discovered = recursive_dfs(w, discovered)
    return discovered

print(recursive_dfs(1))
```

비선형 자료구조 - 스택을 이용해 구현한 깊이 우선 탐색 (DFS)




```
def stack_dfs(start_v):
    discovered = []
    stack = [start_v]
    while stack:
        v = stack.pop()
        if v not in discovered:
            discovered.append(v)
            for w in graph[v]:
                stack.append(w)
    return discovered

print(stack_dfs(1))
```

문제. 부분수열의 합

https://www.acmicpc.net/problem/1182

N개의 정수로 이루어진 수열이 있을 때,

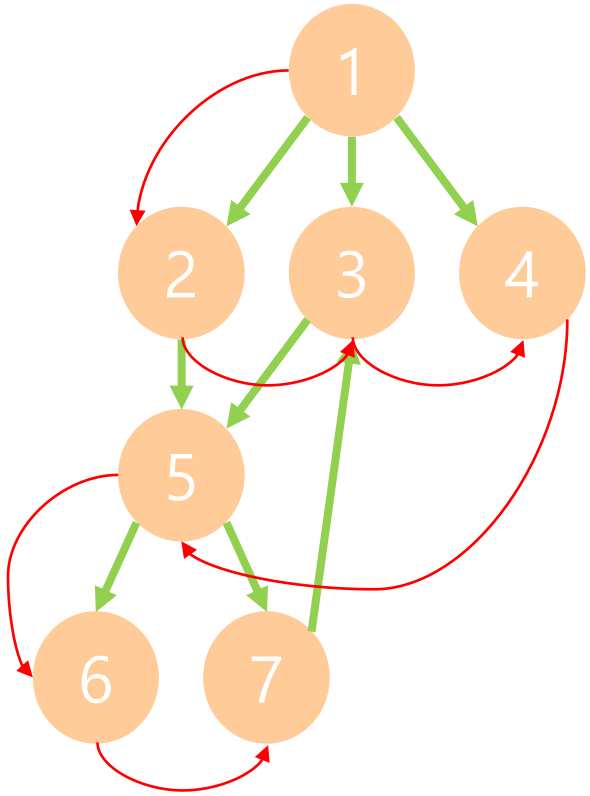
크기가 양수인 부분수열 중에서
그 수열의 원소를 다 더한 값이 S가 되는 경우의 수를 구하는 프로그램을 작성하시오.

입력	출력
5 0 -7 -3 -2 5 8	1

풀이. 부분수열의 합

https://www.acmicpc.net/problem/1182

비선형 자료구조 - 너비 우선 탐색 (BFS)




```
from collections import deque

def queue_bfs(start_v):
    discovered = [start_v]
    queue = deque([start_v])
    while queue:
        v = queue.popleft()
        for w in graph[v]:
            if w not in discovered:
                discovered.append(w)
                queue.append(w)
    return discovered

print(queue_bfs(1))
```

문제. 미로 탐색

https://www.acmicpc.net/problem/2178

N×M크기의 배열로 표현되는 미로가 있다.
미로에서 1은 이동할 수 있는 칸을 나타내고, 0은 이동할 수 없는 칸을 나타낸다.

(1, 1)에서 출발하여 (N, M)의 위치로 이동할 때 지나야 하는 최소의 칸 수를 구하는 프로그램을 작성하시오. 칸을 셀 때에는 시작 위치와 도착 위치도 포함한다.

입력	출력
4 6 101111 101010 101011 111011	15

풀이. 미로 탐색

https://www.acmicpc.net/problem/2178

비선형 자료구조 – 순열

https://leetcode.com/problems/permutations

서로 다른 정수 배열을 입력받아 가능한 모든 순열을 반환하는 프로그램을 작성하시오.

입력	출력

nums = [1, 2, 3]	[
	  [1, 2, 3],
	  [1, 3, 2],
	  [2, 1, 3],
	  [2, 3, 1],
	  [3, 1, 2],
	  [3, 2, 1]
	]

비선형 자료구조 – 조합

https://leetcode.com/problems/combinations

입력값으로 두 개의 정수 n과 k가 주어지면,
1~n 범위에서 k개의 조합을 모두 반환하는 프로그램을 작성하시오

입력	출력
n = 4 k = 2	[[1, 2], [1, 3], [1, 4], [2, 3], [2, 4], [3, 4]]

비선형 자료구조 – 부분 집합

https://leetcode.com/problems/subsets

주어진 배열로 구성할 수 있는 가능한 모든 부분 집합을 반환하는 프로그램을 작성하시오.

입력	출력
nums = [1, 2, 3]	[[], [1], [1,2], [1,2,3], [1,3], [2], [2,3], [3]]

비선형 자료구조 – 조합을 통해 숫자 합 구하기

https://leetcode.com/problems/combination-sum

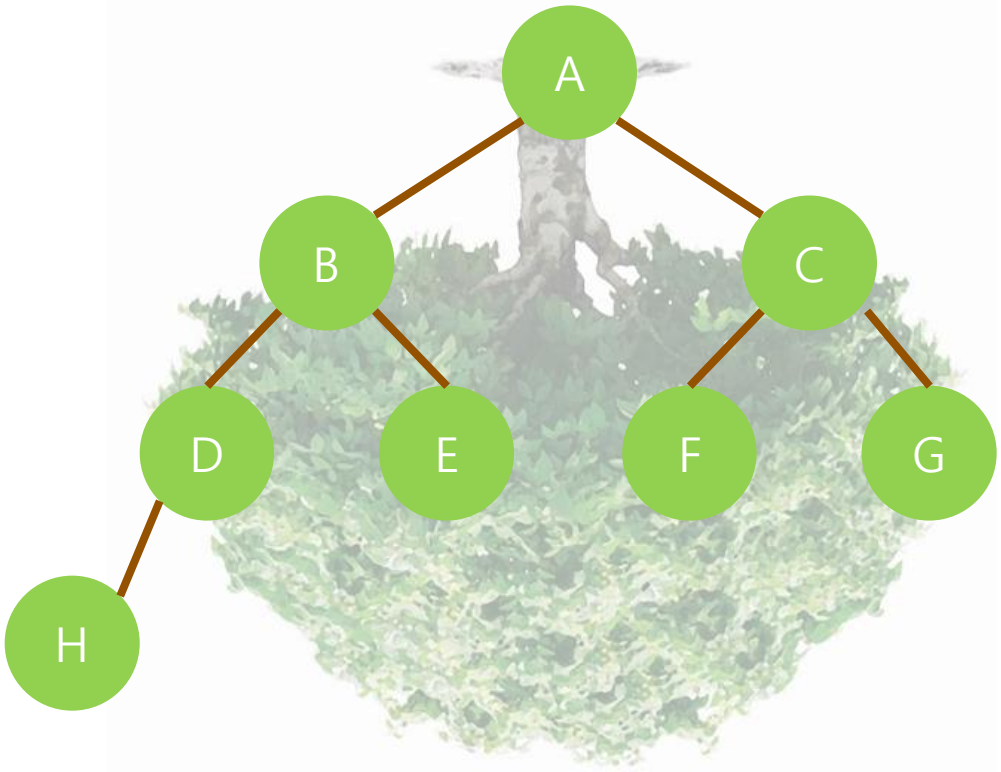
숫자 집합 candidates를 조합하여,
합이 target이 되는 원소를 나열하는 프로그램을 작성하시오.

각 원소는 중복으로 나열 가능하다.

입력	출력
candidates = [2, 3, 6, 7] target = 7	[[7], [2, 2, 3]]

비선형 자료구조 – 트리

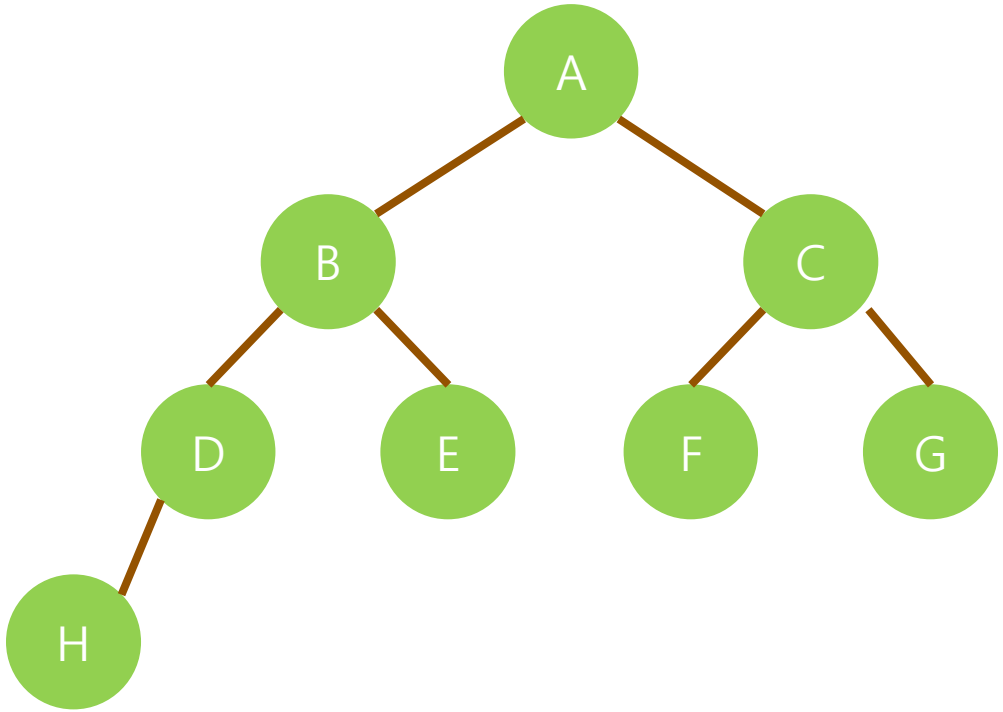
그래프의 일종으로 정점과 간선을 이용하여 데이터의 배치 형태를 추상화한 자료구조
서로 다른 두 노드를 연결하는 길이 하나뿐인 그래프!



- 노드 : 트리를 구성하고 있는 요소 (A~H)
- 루트노드 : 트리의 가장 윗부분 (A)
- 간선 : 노드와 노드를 연결하고 있는 선
- 부모노드 : 노드와 간선으로 연결된 윗부분 노드
- 자식노드 : 노드와 간선으로 연결된 아랫부분 노드
- 리프노드 : 자식노드가 없는 노드
- 서브트리 : 하나의 노드와 그 노드들의 자손들로 이뤄진 트리
- 레벨 : 트리의 각 계층
- 높이 : 트리의 높이

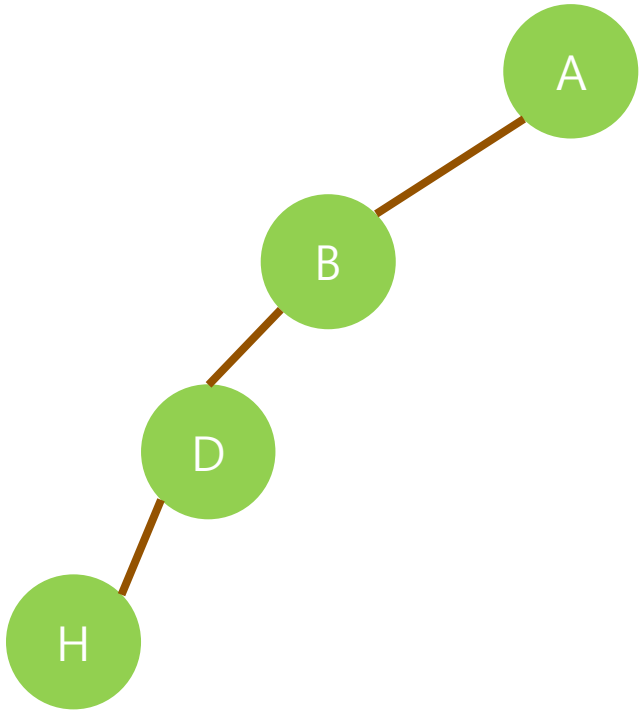
비선형 자료구조 – 이진 트리

이진트리는 각 노드가 최대 2개의 자식을 갖는 트리를 의미합니다.



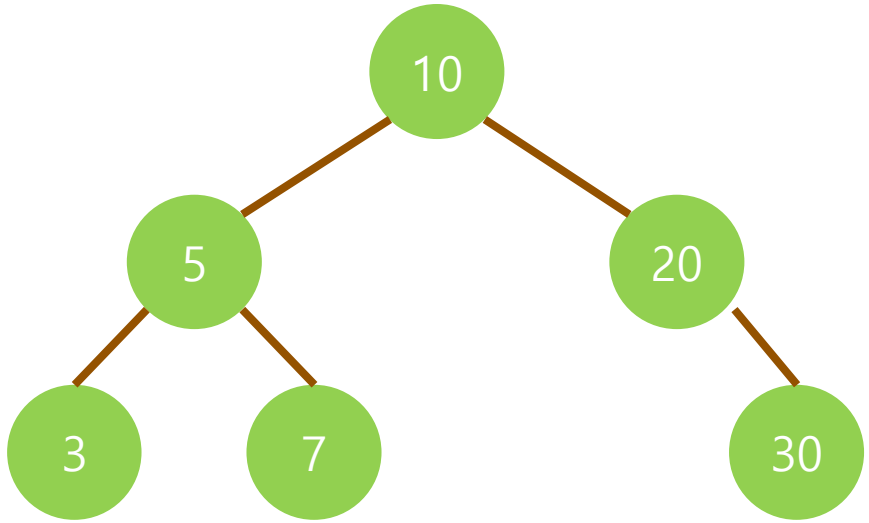
비선형 자료구조 – 이진 트리

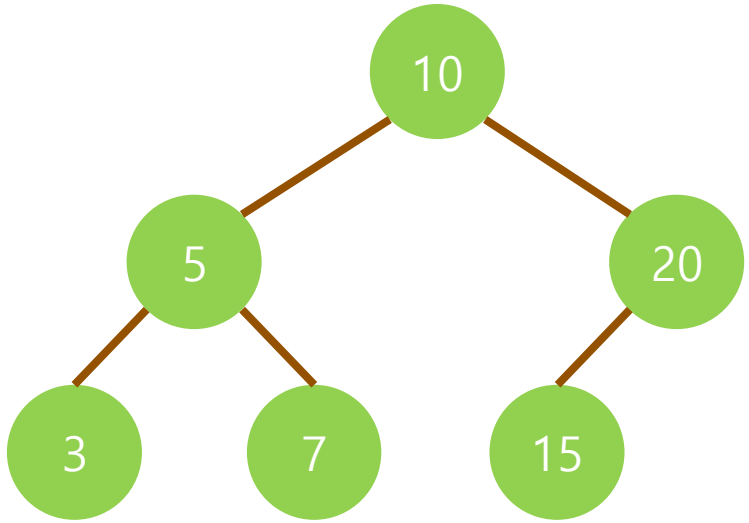
이진트리는 각 노드가 최대 2개의 자식을 갖는 트리를 의미합니다.



비선형 자료구조 – 완전 이진 트리

이진트리는 각 노드가 최대 2개의 자식을 갖는 트리를 의미합니다.





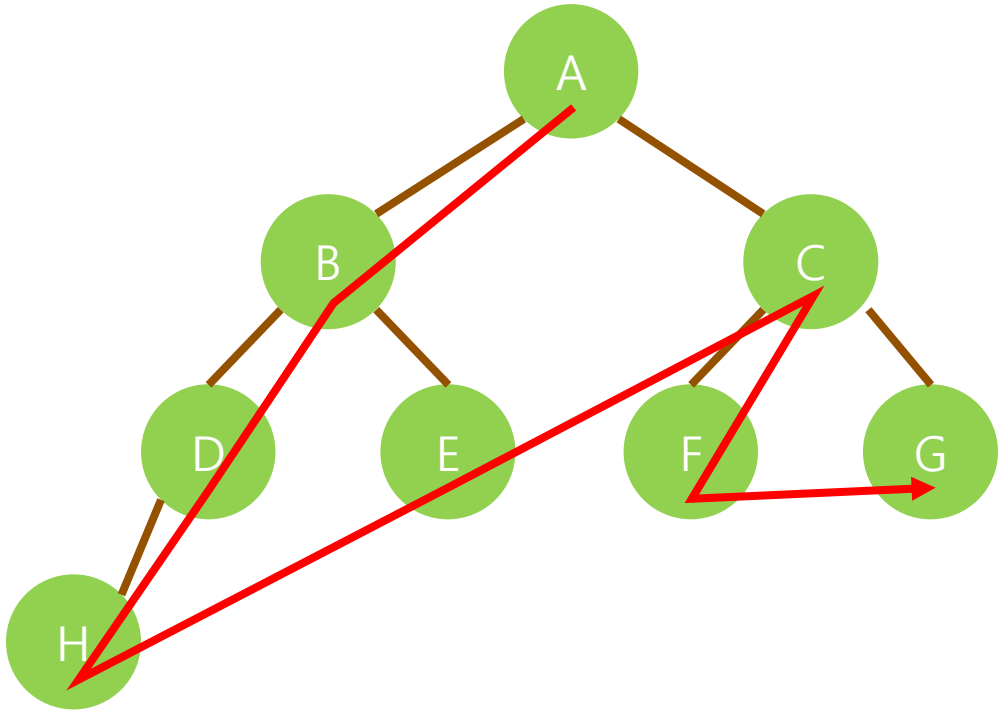
비선형 자료구조 – 이진 트리의 순회

- 전위 순회(preorder traversal) - 중간 왼쪽 오른쪽
  1. 노드를 방문한다
  2. 왼쪽 서브 트리를 전위 순회한다.
  3. 오른쪽 서브 트리를 전위 순회한다.

- 중위 순회(inorder traversal) - 왼쪽 중간 오른쪽
  1. 왼쪽 서브 트리를 중위 순회한다.
  2. 노드를 방문한다.
  3. 오른쪽 서브 트리를 중위 순회한다.

- 후위 순회(postorder traversal) - 왼쪽 오른쪽 중간
  1. 왼쪽 서브 트리를 후위 순회한다.
  2. 오른쪽 서브 트리를 후위 순회한다.
  3. 노드를 방문한다.

비선형 자료구조 – 이진 트리의 순회 (전위 순회)

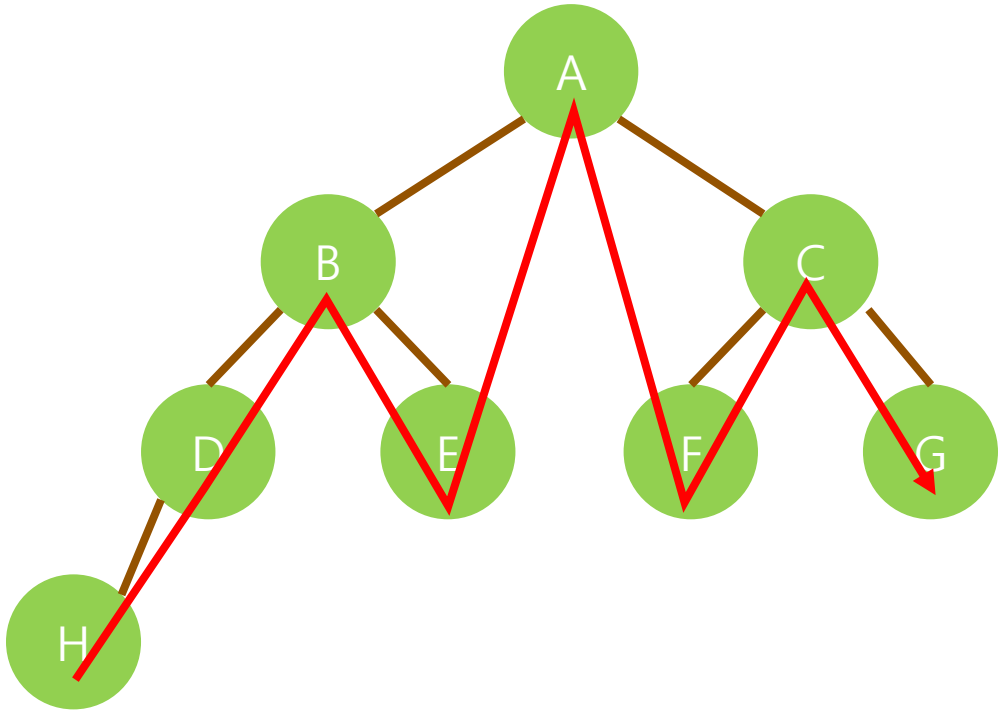


비선형 자료구조 – 이진 트리의 순회 (전위 순회)

```
class Node:
    def __init__(self, item, left, right):
        self.item = item
        self.left = left
        self.right = right

# 전위순회 : 루트 -> 왼쪽 -> 오른쪽
def preorder(node):
    print(node.item, end="")
    if node.left != '.':
        preorder(tree[node.left])
    if node.right != '.':
        preorder(tree[node.right])
```

비선형 자료구조 – 이진 트리의 순회 (중위 순회)

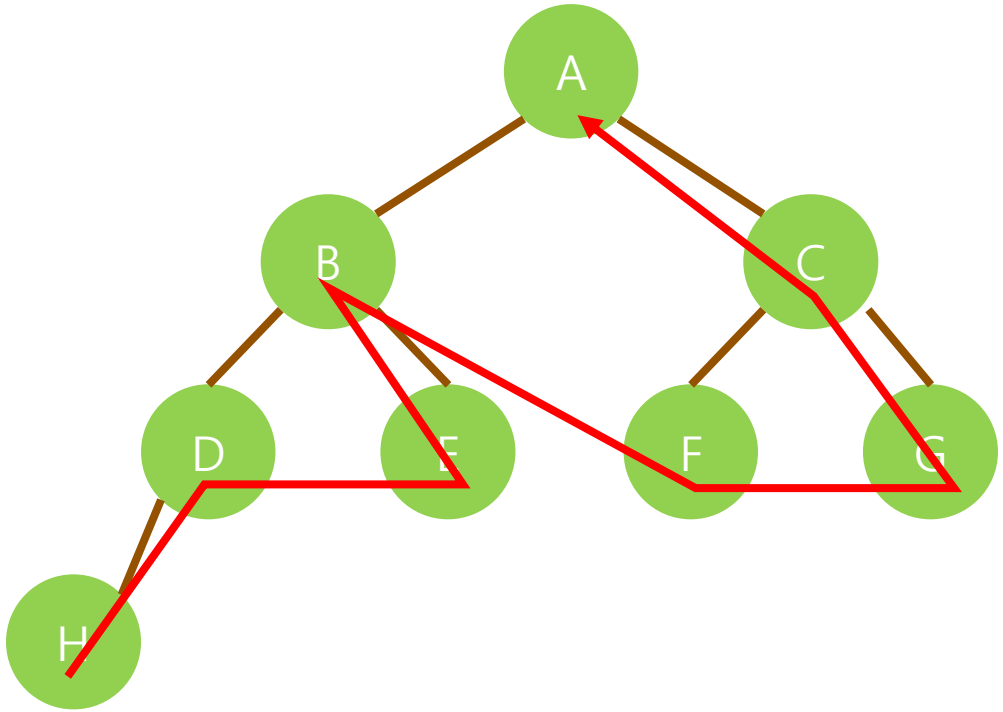


비선형 자료구조 – 이진 트리의 순회 (중위 순회)

```
class Node:
    def __init__(self, item, left, right):
        self.item = item
        self.left = left
        self.right = right

# 중위순회 : 왼쪽 -> 루트 -> 오른쪽
def inorder(node):
    if node.left != '.':
        inorder(tree[node.left])
    print(node.item, end="")
    if node.right != '.':
        inorder(tree[node.right])
```

비선형 자료구조 – 이진 트리의 순회 (후위 순회)



비선형 자료구조 – 이진 트리의 순회 (후위 순회)

```
class Node:
    def __init__(self, item, left, right):
        self.item = item
        self.left = left
        self.right = right

# 후위순회 : 왼쪽 -> 오른쪽 -> 루트
def postorder(node):
    if node.left != '.':
        postorder(tree[node.left])
    if node.right != '.':
        postorder(tree[node.right])
    print(node.item, end="")
```

문제. 트리 순회

https://www.acmicpc.net/problem/1991

이진트리를 입력받아 전위순회, 중위순회, 후위순회한 결과를 출력하는 프로그램을 작성하시오.

입력	출력
7 A B C B D . C E F E . . F . G D . . G . .	ABDCEFG DBAECFG DBEGFCA

풀이. 트리 순회

https://www.acmicpc.net/problem/1991

비선형 자료구조 – 이진 검색 트리

이진 검색 트리란 노드에 있는 값을 Key라고 할 때, 아래의 조건을 만족하는 트리이다.

- 모든 노드의 Key 값은 유일해야 한다. (Key는 중복되지 않는다)
- 왼쪽 서브 트리의 Key들은 루트의 Key보다 작다.
- 오른쪽 서브 트리의 Key들은 루트의 Key보다 크다.
- 노드의 자식 노드는 최대 2개가 올 수 있다.
- 왼쪽과 오른쪽 서브 트리도 이진 검색 트리이다.

* 이진 검색 트리는 이진트리의 특징을 가질 뿐, 서로 다른 트리이다.

문제. 이진 검색 트리

https://www.acmicpc.net/problem/5639

이진 검색 트리를 전위 순회한 결과가 주어졌을 때,
이 트리를 후위 순회한 결과를 구하는 프로그램을 작성하시오.

입력	출력
50 30 24 5 28 45 98 52 60	5 28 24 45 30 60 52 98 50

풀이. 이진 검색 트리

https://www.acmicpc.net/problem/5639

비선형 자료구조 – 힙

힙은 부모가 항상 자식보다 작거나 같다라는 힙의 특성을 만족하는 트리 기반의 자료 구조로,
heapq 모듈이 힙으로 구현되어 있다. 파이썬에서는 최소 힙만 구현되어 있습니다.

힙의 자료구조

- 최대 힙 자료구조 : 최대값을 구하기 위해, 부모노드의 값을 자식노드의 값보다 항상 크게 트리를 만든다.
- 최소 힙 자료구조 : 최소값을 구하기 위해, 부모노드의 값을 자식노드의 값보다 항상 작게 트리를 만든다.

비선형 자료구조 – 최대 힙

데이터의 삽입, 삭제가 많은 경우에 최대값을 빠른 속도로 찾을 수 있는 자료 구조

- 최대 힙은 부모노드가 자식노드보다 값이 큰 완전 이진트리를 의미한다.
- 부모노드의 모든 값이 자식노드보다 커야 한다.
- 최대 힙의 루트노드는 항상 최대값을 가진다.

비선형 자료구조 – 최대 힙의 데이터 추가

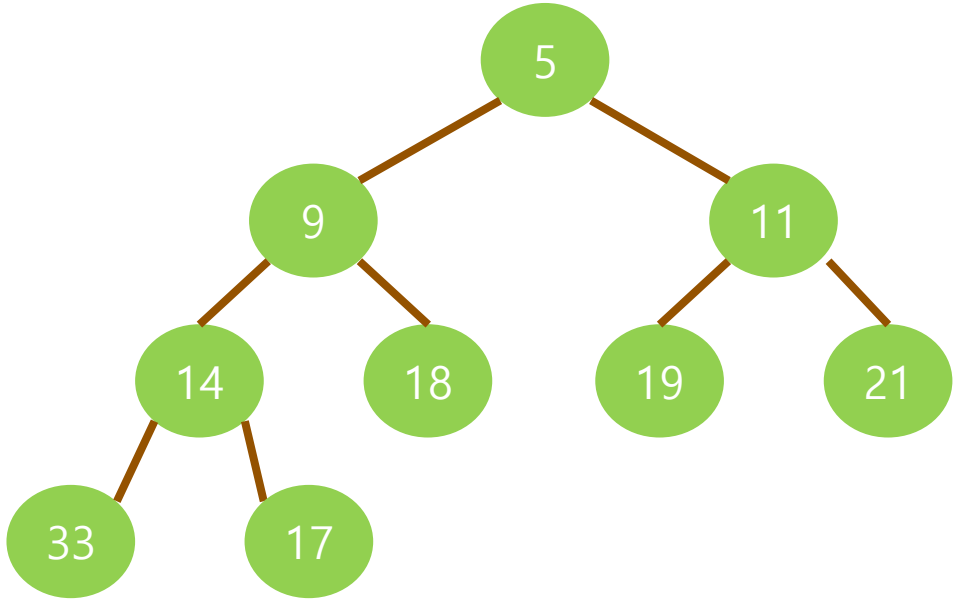
- 처음 원소는 루트 노트에 삽입
- 그 후로는 완전 이진트리를 유지하는 형태로 데이터를 추가
- 만약 루트노드보다 큰 값이 들어오는 경우,
  먼저 완전 이진트리를 유지하는 형태로 데이터를 추가한 뒤,
  부모노드와 삽입한 노드 위치를 비교하여 위치를 바꿈

비선형 자료구조 – 최대 힙의 데이터 삭제

- 최대힙에서의 데이터의 삭제는 루트노드를 삭제하는 것을 의미
- 루트노드를 삭제하고, 가장 밑에서 오른쪽에 있는 노드를 루트노드에 위치
- 새롭게 위치한 루트노드에서부터 출발해서, 자식노드가 더 크다면 가장 큰 자식노드와 위치를 바꿔감

비선형 자료구조 – 최소 힙

데이터의 삽입, 삭제가 많은 경우에 최소값을 빠른 속도로 찾을 수 있는 자료 구조



- 최소 힙은 부모노드가 자식노드보다 값이 작은 완전 이진트리를 의미한다.
- 부모노드의 모든 값이 자식노드보다 작아야 한다.
- 최소 힙의 루트노드는 항상 최소값을 가진다.

비선형 자료구조 – 최소 힙의 데이터 추가

- 처음 원소는 루트 노트에 삽입
- 그 후로는 완전 이진트리를 유지하는 형태로 데이터를 추가
- 만약 루트노드보다 작은 값이 들어오는 경우,
  먼저 완전 이진트리를 유지하는 형태로 데이터를 추가한 뒤,
  부모노드와 삽입한 노드 위치를 비교하여 위치를 바꿈

비선형 자료구조 – 최소 힙의 데이터 삭제

- 최소힙에서의 데이터의 삭제는 루트노드를 삭제하는 것을 의미
- 루트노드를 삭제하고, 가장 밑에서 오른쪽에 있는 노드를 루트노드에 위치
- 새롭게 위치한 루트노드에서부터 출발해서, 자식노드가 더 작다면 가장 작은 자식노드와 위치를 바꿔감

비선형 자료구조 – 우선순위 큐

우선순위 큐는 삽입의 순서에 상관없이 우선순위가 높은 데이터가 가장 먼저 출력되는 구조

문제. 최소 힙

https://www.acmicpc.net/problem/1927

넓리 잘 알려진 자료구조 중 최소 힙이 있다.
최소 힙을 이용하여 다음과 같은 연산을 지원하는 프로그램을 작성하시오.

첫째줄에 연산의 개수 n이 주어진다.
그 이후에 연산 정보 x가 자연수로 주어지면 배열에 자연수 x를 넣는다.

x가 0이면 배열에서 가장 작은 값을 출력하고, 그 값을 배열에서 제거한다.
배열에 값이 없으면 0을 출력한다.

프로그램은 처음에 비어있는 배열에서 시작하게 된다.

입력	출력
9 0 123456 1 2 0 0 0 0 32	0 1 2 123456 0

풀이. 최소 힙

https://www.acmicpc.net/problem/ 1927

문제. 최대 힙

https://www.acmicpc.net/problem/11279

널리 잘 알려진 자료구조 중 최대 힙이 있다.
최대 힙을 이용하여 다음과 같은 연산을 지원하는 프로그램을 작성하시오.

첫째줄에 연산의 개수 n이 주어진다.
그 이후에 연산 정보 x가 자연수로 주어지면 배열에 자연수 x를 넣는다.

x가 0이면 배열에서 가장 큰 값을 출력하고, 그 값을 배열에서 제거한다.
배열에 값이 없으면 0을 출력한다.

프로그램은 처음에 비어있는 배열에서 시작하게 된다.

입력	출력
13 0 1 2 0 0 3 2 1 0 0 0 0 0	0 2 1 3 2 1 0 0

풀이. 최대 힙

https://www.acmicpc.net/problem/11279