파이썬 심화

이진탐색트리

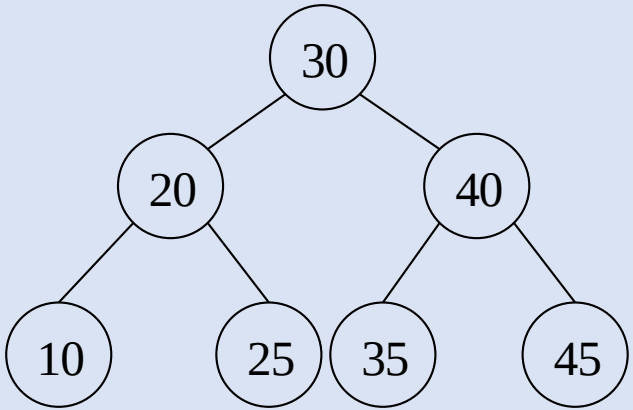
이진탐색트리

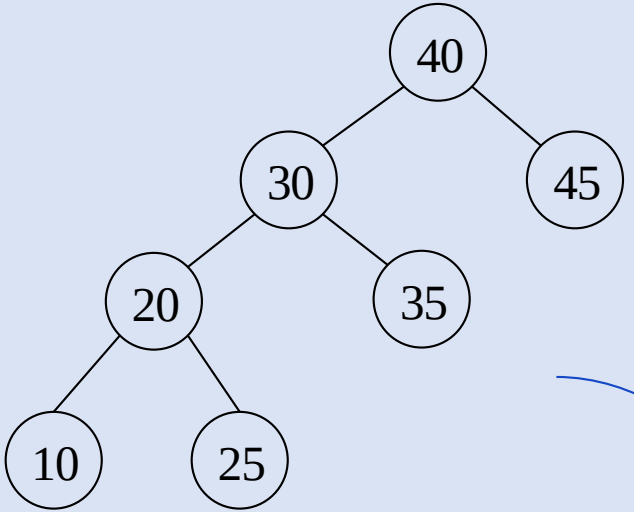
이진탐색트리란?

- 이진탐색트리는 이진트리를 이용하여 원하는 정보를 빨리 찾을 수 있도록 하는 자료구조
  - 탐색은 키(Key)를 이용하여 원하는 값(Value)을 찾음
  - 이진탐색트리에서의 노드에는 키와 값을 저장

이진검색트리의 형태

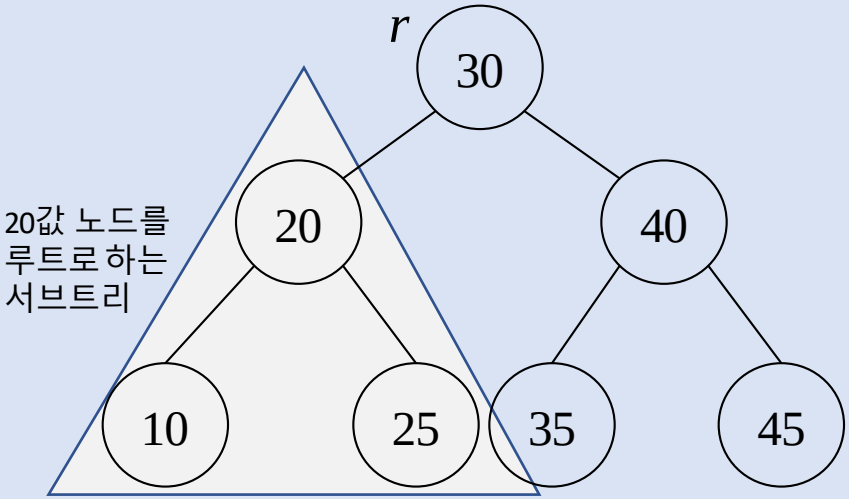
- 이진검색트리는 노드의 왼쪽 자식 트리에는 현재 노드값보다 작은값이, 오른쪽 자식 트리에는 현재 노드값보다 큰값이 저장된 트리
- 같은 데이터라도 다양한 트리 형태가 존재





서브 트리

- 트리에 있는 모든 노드를 루트로 하는 트리를 서브 트리(sub-tree)라고 함



이진탐색트리 연산

검색(Search)

- 주어진 키를 BST에서 검색

```
def search(tree, key):
  if tree == None: return None
  if tree.key == key: return tree.value
  if key < tree.key: return search(tree.left, key)
  return search(tree.right, key)
```

삽입(Insert)

- 주어진 키와 값을 BST에 삽입

```
def insert(tree, key, value):
  if tree == None: return Node(key, value)
  if key < tree.key: return tree.left = insert(tree.left, key, value)
  return tree.right = insert(tree.right, key, value)
```

삭제(Delete)

- 삭제는 삭제하고자 하는 노드의 자식 개수에 따라 다르게 작동
  - 자식의 개수가 0인 경우 : 해당 노드를 삭제
  - 자식의 개수가 1인 경우 : 해당 노드를 삭제하고 그 위치에 자식을 놓음
  - 자식의 개수가 2인 경우 : 오른쪽 서브트리의 가장 왼쪽 노드를 가져오고 해당 노드를 삭제

삭제(Delete)

- 주어진 키에 대한 노드를 BST에서 삭제

```
def delete(tree, key):
  if tree == None: return None
  if tree.key == key: return deleteNode(tree)
  if key < tree.key: return tree.left = delete(tree.left, key)
  return tree.right = delete(tree.right, key)
```

삭제(Delete)

- 주어진 키에 대한 노드를 BST에서 삭제

```
def deleteNode(tree):
  if tree.left == None and tree.right == None: return None
  if tree.right == None: return tree.left
  if tree.left == None: return tree.right
  s = tree.right
  while s.left != None:
    s = s.left
  tree.key, tree.value = s.key, s.value
  tree.right = delete(tree.right, s.key)
  return tree
```