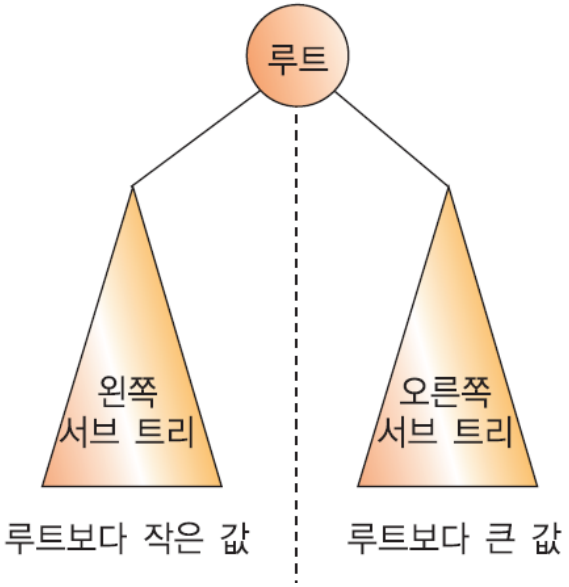
CSE2010 자료구조론

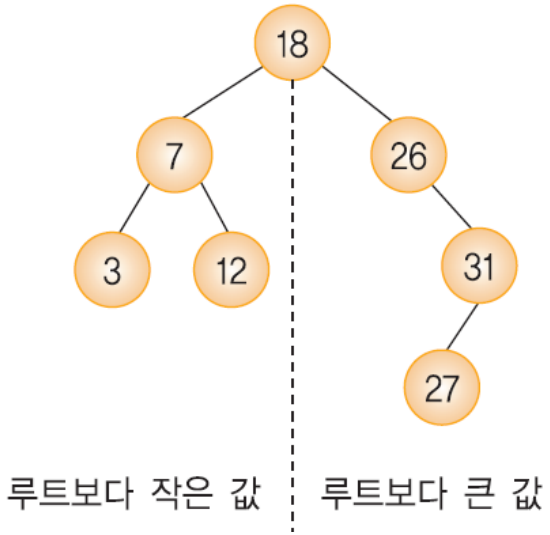
Week 7: Binary Search Tree 1

ICT융합학부 한진영

이진 탐색 트리(Binary Search Tree)

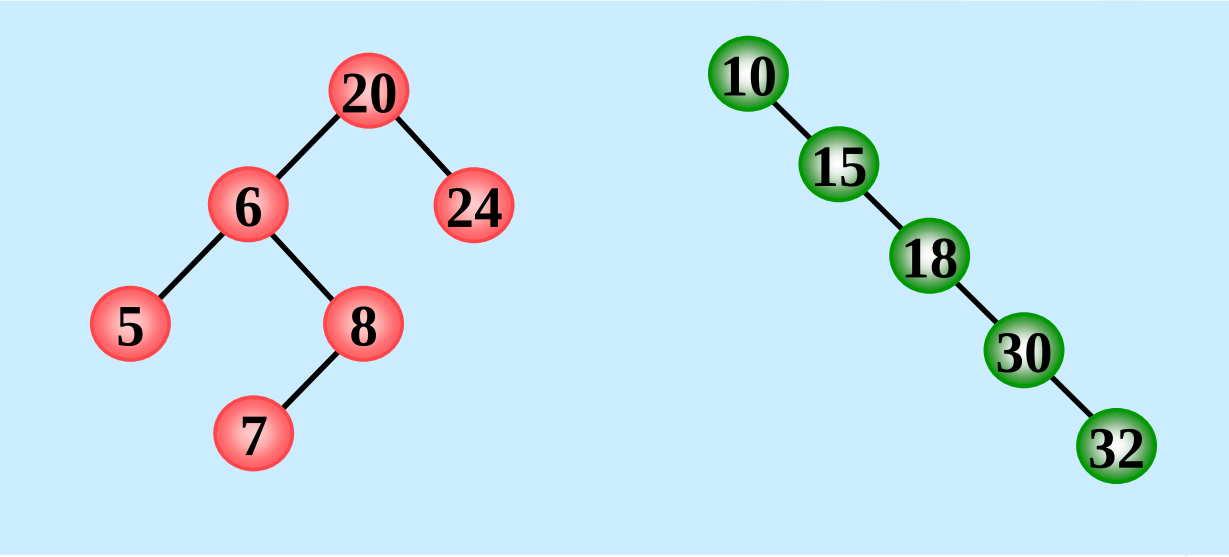
- 탐색작업을 효율적으로 하기 위한 자료구조
- key(왼쪽서브트리)≤key(루트노드)≤key(오른쪽서브트리)
  - 모든 노드의 Key는 유일함
- 이진탐색를 중위순회하면 오름차순으로 정렬된 값을 얻을 수 있음





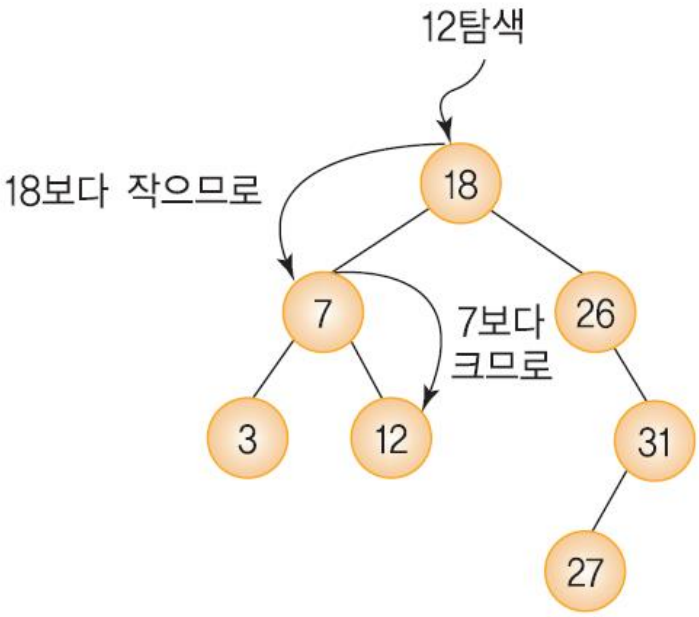
이진 탐색 트리 정의

- 이진 탐색 트리: 공집합이거나 다음을 만족하는 이진 트리
  - 공집합이 아닌 왼쪽 부분 트리의 모든 키값은 루트의 키값 보다 작음
  - 공집합이 아닌 오른쪽 부분 트리의 모든 키값은 루트의 키값보다 큼
  - 왼쪽 부분 트리와 오른쪽 부분 트리도 이진 탐색 트리



이진 탐색 트리 탐색 연산(1)

- 아이디어
  - 비교한 결과가 같으면 탐색이 성공적으로 끝남
  - 주어진 키 값이 루트 노드의 키값보다 작으면 탐색은 이 루트 노드의 왼쪽 자식을 기준으로 다시 시작
  - 주어진 키 값이 루트 노드의 키값보다 크면 탐색은 이 루트 노드의 오른쪽 자식을 기준으로 다시 시작



이진 탐색 트리 탐색 연산(2)

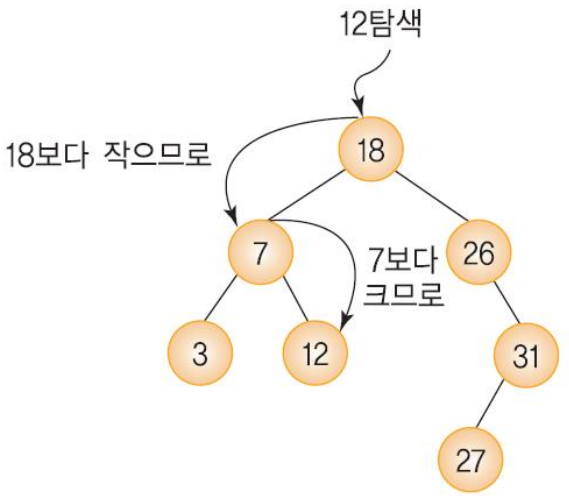
- 알고리즘: 재귀적인 탐색 방법 (찾고자 하는 값 : key)
  - 루트가 null 이면 탐색 실패
  - 루트의 원소값 = key 이면 탐색 성공 & 종료
  - 루트의 원소값 > key 이면 왼쪽 부분 트리를 재귀적으로 탐색
  - 루트의 원소값 < key 이면 오른쪽 부분 트리를 재귀적으로 탐색

- 시간 복잡도 : $O(h)$, 단 h: 트리의 높이

이진 탐색 트리 탐색 연산 알고리즘

```
search(x, k)

if x==NULL
        then return NULL;
if k==x->key
        then return x;
else if k < x->key
     then return search(x->left, k);
     else return search(x->right, k);
```



이진 탐색 트리 탐색 연산 구현(1)

```
//순환적인 탐색 함수

TreeNode  *search(TreeNode  *node,  int  key)
{
    if ( node == NULL )  return NULL;
    if ( key == node->key ) return node;
    else if ( key < node->key )
        return  search(node->left, key);
    else
        return  search(node->right, key);
}
```

이진 탐색 트리 탐색 연산 구현(2)

```
// 반복적인 탐색 함수

TreeNode *search(TreeNode *node, int key)
{
        while(node != NULL){
                if( key == node->key ) return node;
                else if( key < node->key )
                        node = node->left;
                else
                        node = node->right;
        }
        return NULL;   // 탐색에 실패했을 경우 NULL 반환
}
```

Week 7: Binary Search Tree 1