동아대학교

데이터 구조

10주차: BST(Binary Search Tree) 구현

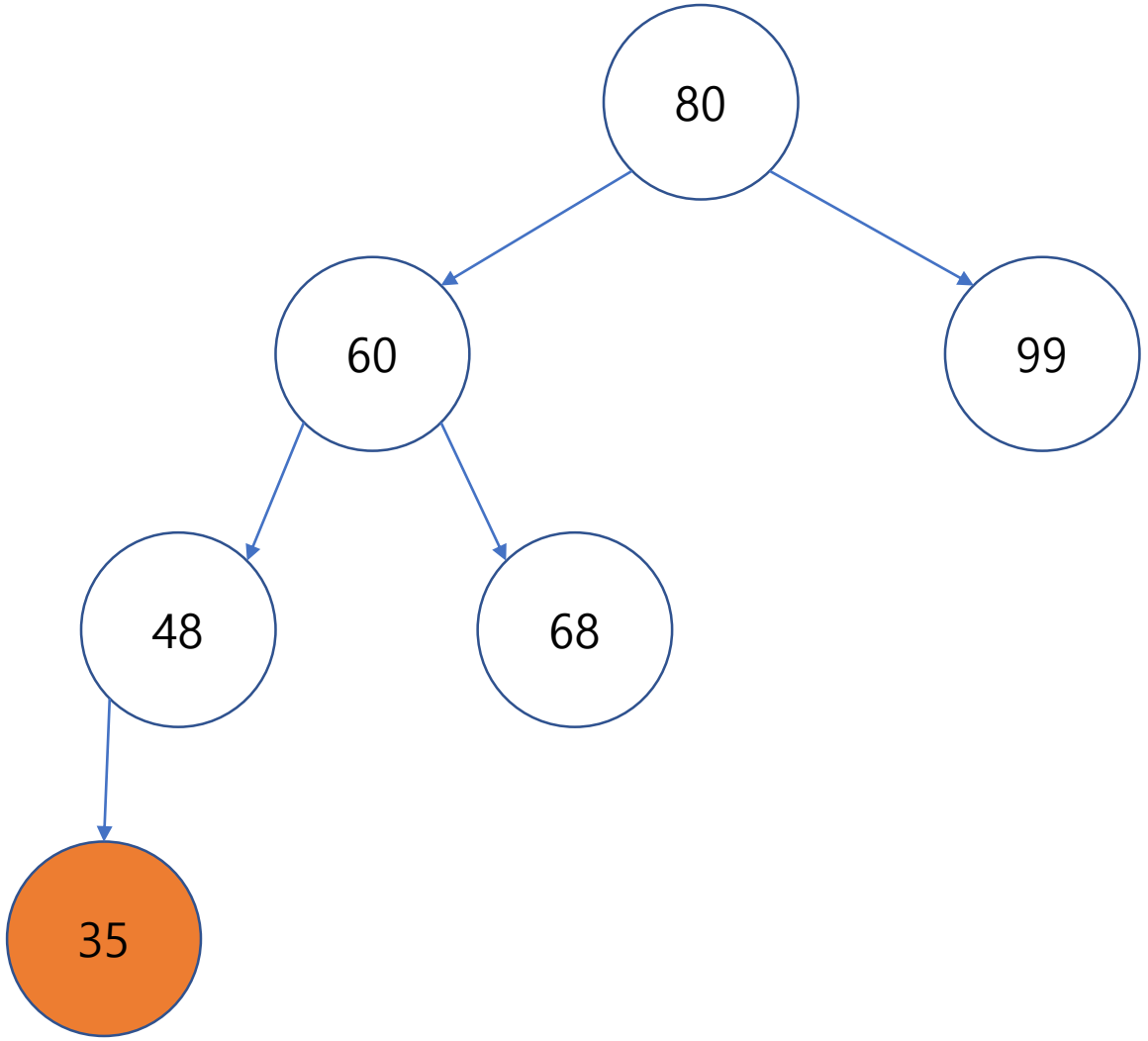
박 경 훈

BST 클래스 구현

● 구성 메소드

1. 삽입 (insert)
2. 탐색 (search)
3. 중위 순회 출력 (inorder_print)
4. (추가) 레벨 별 출력 (levelOrder_print)

BST 클래스 구현 – Insert (삽입)

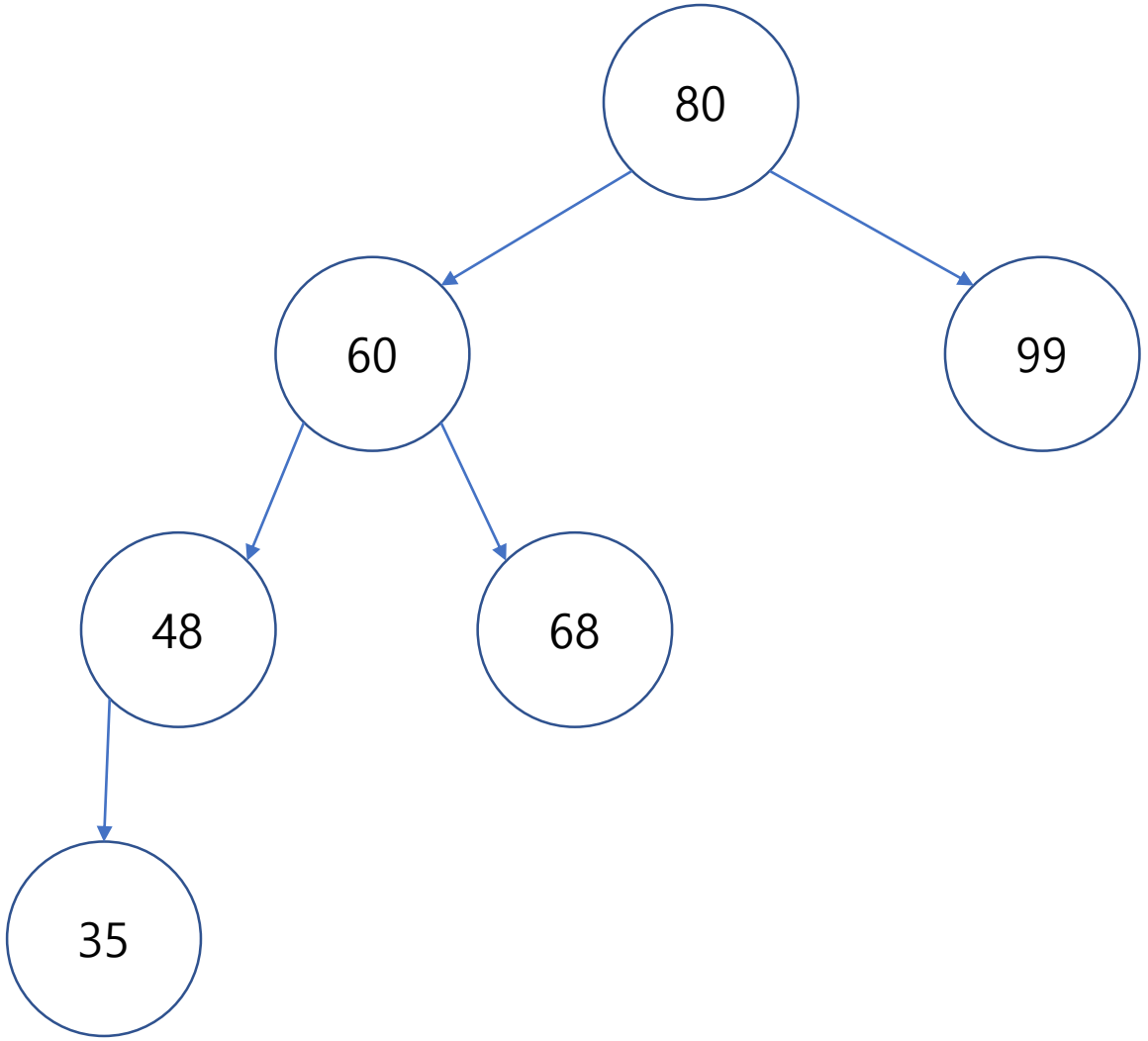


○ 삽입 메소드 동작 예시

- "35" 데이터 입력시

Step 1: "80" 과 "35" 비교,
왼쪽 서브트리 존재유무 판단,
왼쪽 서브트리 이동
Step 2: "60" 과 "35" 비교,
왼쪽 서브트리 존재유무 판단
왼쪽 서브트리 이동
Step 3: "48" 과 "35" 비교,
왼쪽 서브트리 존재유무 판단
왼쪽 서브트리 노드 생성

BST 클래스 구현 – search (탐색)



● 탐색 구현

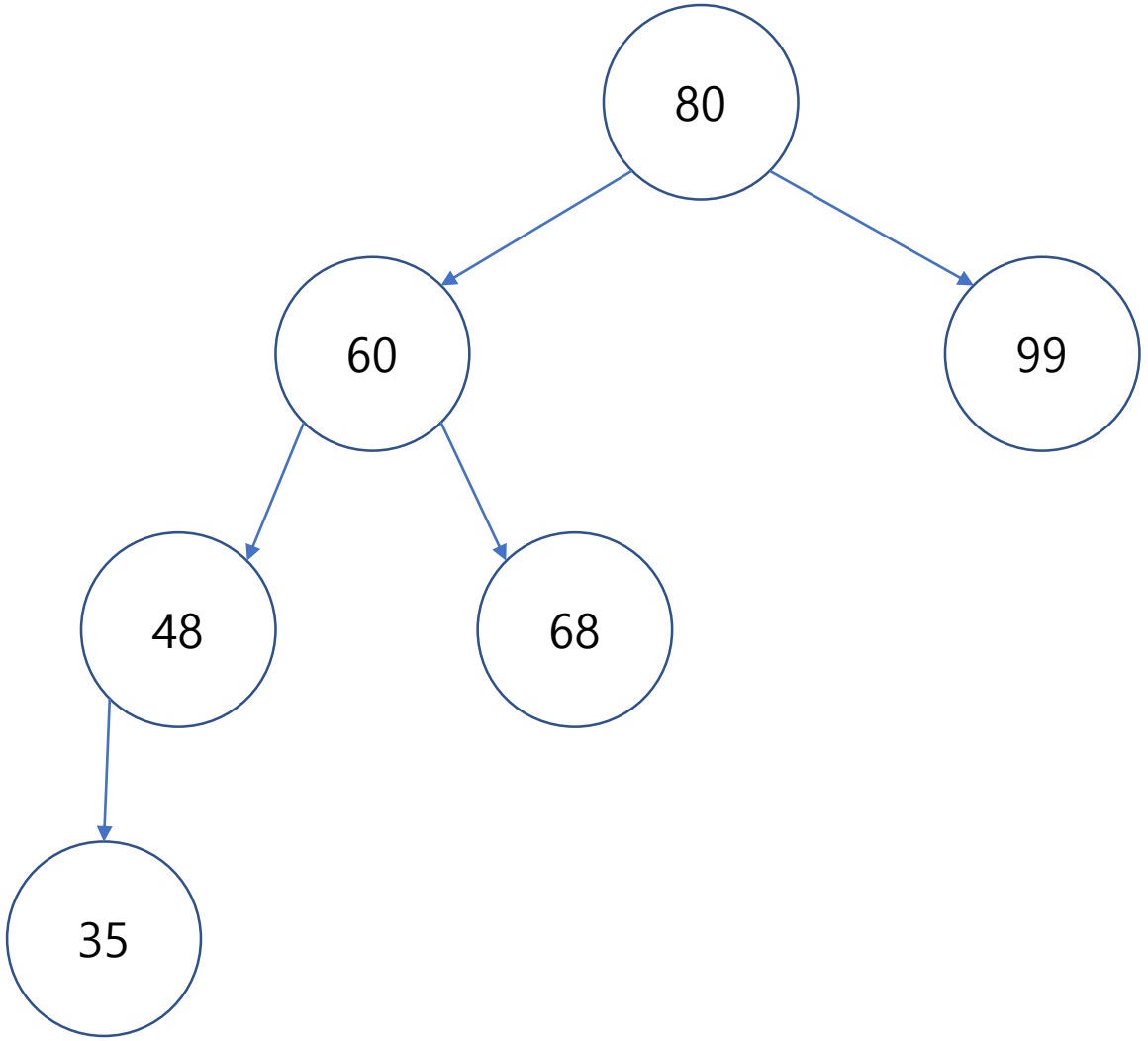
- 삽입 메소드 동작과 유사한 동작
- 삽입 메소드를 참조하여 구현

○ 탐색 메소드 동작 예시

- "68" 데이터 검색시

Step 1: "80" 과 "68" 비교,
왼쪽 서브트리 존재유무 판단,
왼쪽 서브트리 이동
Step 2: "60" 과 "68" 비교,
오른쪽 서브트리 존재유무 판단
오른쪽 서브트리 이동
Step 3: "68" 과 "68" 비교,
return 노드

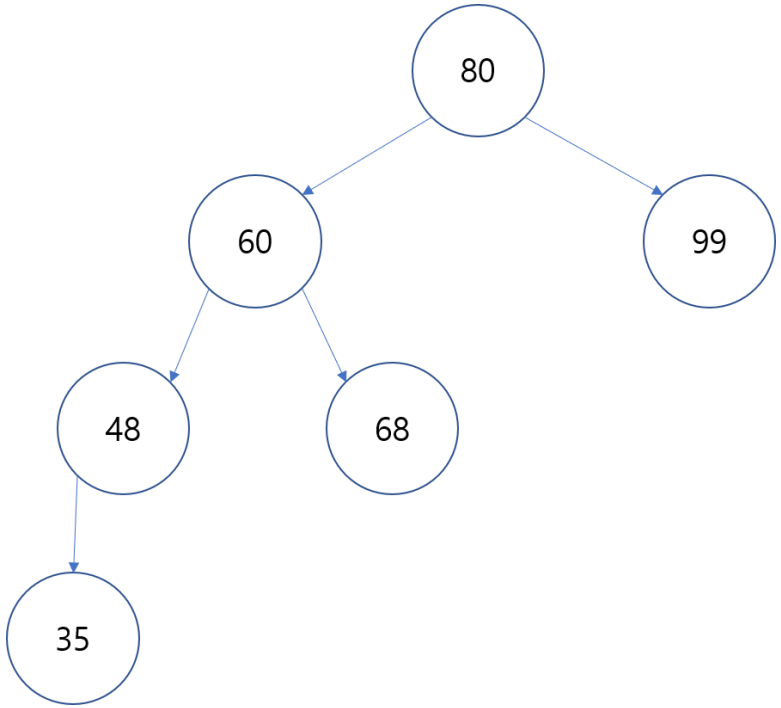
BST 클래스 구현 – 출력



● 출력구현

- 중위순회 방식 출력과 레벨 출력방식
- 중위순회 방식을 이용할 경우 정렬 출력,
  이진 트리 방식 참조
- 레벨 출력 방식은 실제 입력된 형태로 출력
  큐 자료형 또는 데크 자료형 활용

BST 클래스 구현 – Level order 출력



○ Step 1: Queue 자료형 또는 deQue 자료형 생성

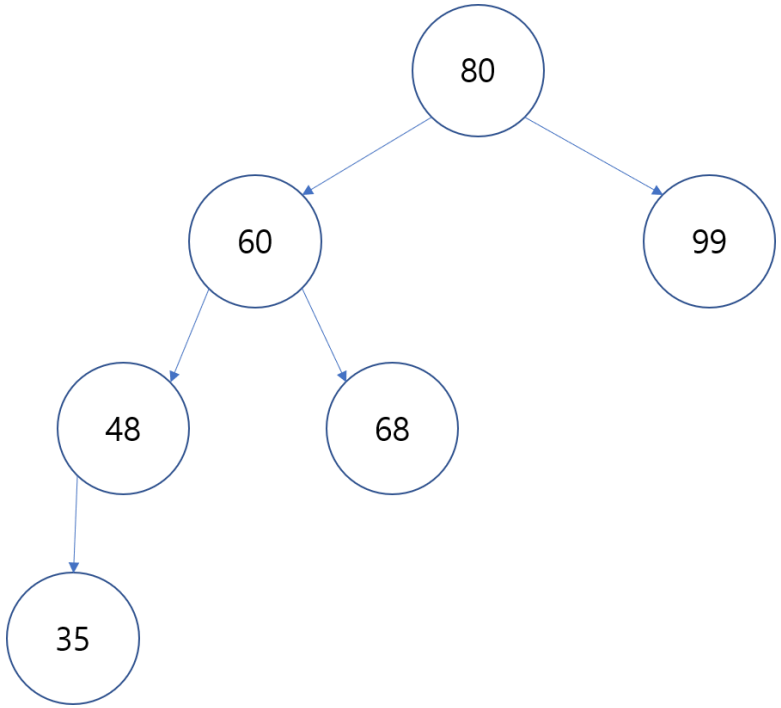
○ Step 2: root 노드 삽입

80					

○ Step 3:
- root 노드 pop() 동작 후 데이터 출력
- root 노드 left, right 노드 append

60	99				

BST 클래스 구현 – Level order 출력



○ Step 4:
- "60" 노드 pop() 동작 후 데이터 출력
- "60" 노드 left, right 노드 append

99	48	68			

○ Step 5:
- "99" 노드 pop() 동작 후 데이터 출력
- "99" 노드 left, right 노드 append
  ("99" 노드 left, right 노드가 "none" 이므로 패스)

48	68				

★ 위 동작을 반복 수행하여 출력