

# BTS(Binary Tree for Searching)

**[문제]** 탐색을 위한 이진트리(Binary Tree for Searching)에 string 데이터를 추가(insert), 삭제 그리고 몇 가지 작업을 한다. 이 작업 도중에 관리하고 있는 BTS의 depth  $k$ 에 있는 node 또는 leaf node를 모두 찾아 왼쪽부터 오른쪽 순으로, 사전식 순서대로 모두 출력해서 작업을 확인한다.

명령어 형식	동작 내용	추가 설명
<code>&gt;&gt; + string “+ box”</code>	문자열 <code>string</code> 을 현재 BTS에 들어갈 위치를 찾아 <code>leaf</code> 위치에 이것을 추가한다.	이미 <code>key</code> 값이 <code>Tree</code> 에 있는 경우에는 변함이 없다.
<code>&gt;&gt; - string “- soju”</code>	제거할 <code>data</code> 가 <code>leaf</code> 에 있다면 바로 지운다. 만일 그것이 내부 노드라면 그것을 지운 다음 오른쪽 부트리의 최소값으로 대치한다. 만일 오른쪽 부트리가 없으면 왼쪽 부트리의 최대값으로 교체하고 <code>leaf</code> 에 도달할 때까지 계속한다.	해당되는 <code>Key</code> 가 없는 경우에는 무시한다. 따라서 BTS는 그대로 유지된다.
<code>&gt;&gt; depth k</code>	<code>depth k</code> 에 있는 노드를 찾아 사전식 순서로 모두 출력한다. 없을 경우에는 "NO"를 출력.	<code>root</code> 의 <code>depth</code> 는 1이며 <code>depth k</code> 노드의 자식 노드의 <code>depth</code> 는 $k + 1$ .
<code>&gt;&gt; leaf</code>	BTS의 <code>leaf</code> 를 왼쪽부터 사전식으로 모두 출력. <code>root</code> 만 있을 경우에는 <code>root</code> 를 출력	항상 한 개 이상 존재함.

**[입출력]** 입력의 첫 줄에는 명령어의 개수  $N$  ( $5 \leq N \leq 100$ )이 주어진다. 그리고 이어지는  $N$  개의 각 줄에 한 개의 명령어가 주어진다. 여러분은 명령어 “`depth k`”나 ”`leaf`”에 대하여 해당되는 원소를 사전식 순서대로 한 줄에 모두 출력해야 한다.

## 【예제】

입력 <b>stdin</b>	출력 <b>stdout</b>
13 // 명령어 개수 + phone + banana - cola // 없으면 무시함 + chip <b>leaf</b> + pizza + soccer - phone <b>depth 3</b> + machine <b>depth 2</b> - pizza <b>leaf</b>	<b>chip // leaf</b> 의 결과 <b>chip // depth 3</b> <b>banana soccer //depth 2</b> <b>machine // leaf</b>

**【제한조건】** 프로그램의 이름은 BTS.{py,c,cpp,java}이다. 제출 횟수는 최대 15번이며 허용 시간은 데이터 당 제한 시간은 1초, 허용가능 코드의 최대 크기는 3000 bytes 이다. 문제 풀이 마감시간은 12월7일(화요일) 11:00(PM) 이다. 제시간에 제출하지 못한 학생은 1일 유예시간이 주어진다. 유예시간에는 10%의 감점이 적용된다.