**Problem Name:**

**Topics:**

**Companies:**

**Level:** Hard

**Language:** C++

**Problem Statement:**

**Input Format:**

**Output Format:**

**Constraints:**

**Examples:**

**Brute force Solution:**

**Explanation:** Level order Traversal |

Using **zero based** indexing here - ith node  
-> left child - 2\* i +1  
-> right child - 2\* i + 2

**Code:**

**Time Complexity**: O(N)

**Space Complexity:** O(N)

**Optimized Solution:**

**Explanation:**

The idea is to use heap indexing:

1

2 3

4 5 6 7

8 9 x 11 x 13 x 15

Regardless whether these nodes exist:

* Always make the id of left child as parent\_id \* 2;
* Always make the id of right child as parent\_id \* 2 + 1;

So we can just:

1. Record the id of left most node when first time at each level of the tree during an pre-order run.(you can tell by check the size of the container to hold the first nodes);
2. At each node, compare the distance from it the left most node with the current max width;

**Code:**

**Time Complexity**: O(N)

**Space Complexity:** O(N)