Pseudo code that checks a given binary tree is a min heap

```
isMinHeap ( inout treePtr : TreeNodePtr , inout index : int, in size : int)
// Checks if the binary tree is complete and each node has higher key
// than its parent (min heap)
  if( treePtr is NULL) { Empty tree, return true }
// Check if tree is complete by comparing size and index
  if( index >= size ) { Not complete, return false }
// check if node has higher value than its left or right child
  if ((treePtr -> leftChild -> item < treePtr -> item) AND treePtr -> leftChild
      is not NULL) OR (treePtr -> rightChild -> item < treePtr -> item AND
      treePtr -> rightChild is not NULL)) {
        It is not min-heap, return false
      }
// Check left and right subtree
  leftMinHeap = isMinHeap (treePtr -> leftChild, leftChildIndex, size)
  rightMinHeap = isMinHeap ( treePtr -> rightChild, rightChildIndex, size)
// if both left and subtree are minheap, return true
  return (leftMinHeap AND rightMinHeap)
```

treePtr = root
index = 0
size = total no of nodes

leftChildIndex = 2 * index + 1
rightChildIndex = 2 * index + 2