Pavan K
RVCE24MCA042

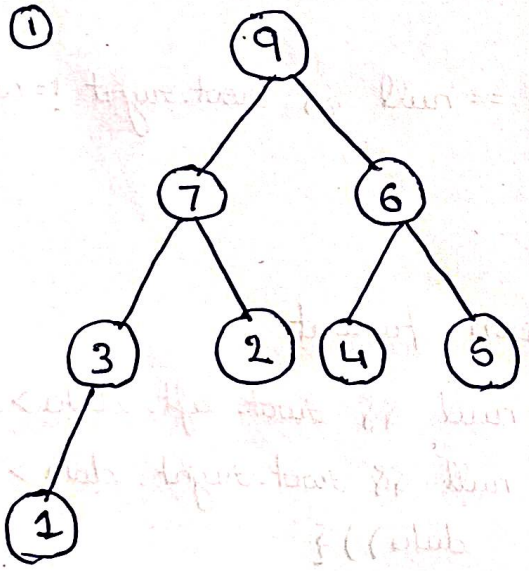
Check if a Binary Tree is Heap :-

A Binary Tree to be heap it has to satisfy 2 conditions :-

1) It's complete → All levels are fully filled expect the last & nodes are added left to right.

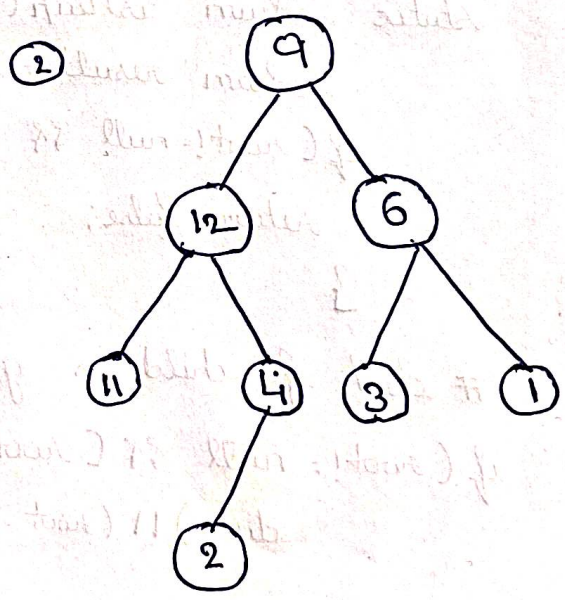
2) Parents are bigger :- (max heap property) Every parent node is greater than or equal to its children

Example :-



true ✓

It is complete & satisfy max-heap property.



false ✗

It is incomplete & doesn't satisfy max-heap property also

Program :-

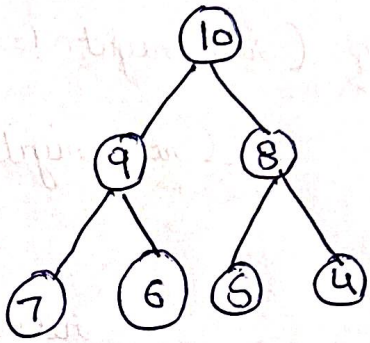
```
class Node {
    int data;
    Node left;
    Node right;
Node (int val) {
    data = val;
    left = null;
    right = null;
  }
}
class Gfg {
  # Function to check if the binary tree is max heap.
  static boolean isHeap (Node root) {
      boolean result = true;
    if (root != null && root.left == null && root.right != null) {
      return false;
    }
  # check if child is greater than parent :
  if (root != null && (root.left != null && root.left.data > root.
          data) || (root.right != null && root.right.data > root.
                                              data)) {
                  return false;
              }
  # check if left subtree has children but right is null.
    if (root.left != null) {
        if ((root.left != null || root.left.right != null) &&
                                          root.left == null) {
          return false;
    }
```

```
// check if right subtree has children & left is null.
If (root.right != null) {
    If ((root.right.left != null || root.right.right != null) &&
            root.right == null){
        return false;
    }
}

// check if right subtree has children but not left subtree
if (root.left != null) {
  if (root.left.left == null && root.left.right == null) {
      if (root.right.left != null){
          If (root.right.left != null){
              return false;
          }
      }
  }
}
If (root != null && root.left != null){
    boolean left = isLeaf(root.left);
    result = result && left;
}
if (root != null && root.right != null){
    boolean right = isLeaf(root.right);
    result = result && right;
}
return result.
}
public static void main (String[] args){
    Node root = new Node(10);
    root.left = new Node(9);
```

```
        root.left.right = new Node(6)
        root.left.left = new Node(7);
        root.right.left = new Node(5);
        root.right.right = new Node(4);

    If(isHeap(root)){
        System.out.println("true");
    }else{
        System.out.println("false");
    }
  }
}
```



output :- True.