Q1:False

Q2. checks if the height has an equal number of nodes on either side of the tree that is the right child and the left child

Q3. (A,B)

Q4: 32 nodes

Q5. 2

Q6.
```
import java.util.LinkedList;
        import java.util.Queue;


    class Node
{
   int data;
   Node left, right;

   Node(int item)
   {
      data = item;
      left = right;
   }
}

class BinaryTree
{
   Node root;



   int treeHeight(Node node)
   {
        if (node == null)
```

```
        return 0;

    Queue<Node> q = new LinkedList();



    q.add(node);
    int height = 0;

    while (1 == 1)
    {

        int nodeCount = q.size();
        if (nodeCount == 0)
            return height;
        height++;



        while (nodeCount > 0)
        {
            Node newnode = q.peek();
            q.remove();
            if (newnode.left != null)
                q.add(newnode.left);
            if (newnode.right != null)
                q.add(newnode.right);
            nodeCount--;
        }
    }
}
```

```
    public static void main(String args[])
    {
        BinaryTree tree = new BinaryTree();



        tree.root = new Node(1);
        tree.root.left = new Node(2);
        tree.root.right = new Node(3);
        tree.root.left.left = new Node(4);
        tree.root.left.right = new Node(5);
        System.out.println("Height of tree is " + tree.treeHeight(tree.root));
    }
}
```

Q7.
```
static void maxValueBFS(Node<Integer> root) {

                if (root == null) {
                        System.out.println("Tree is empty");
                } else {
                        int max = Integer.MIN_VALUE;
                        Queue<Node<Integer>> bfs = new LinkedList<>();

                        bfs.add(root);

                        while (!bfs.isEmpty()) {
                                Node<Integer> front = bfs.poll();
                                if (front.data > max)
```

```
                    max = front.data;

                if (front.left != null) {
                    bfs.add(front.left);
                }

                if (front.right != null) {
                    bfs.add(front.right);
                }

            }
            System.out.println("Max Value of tree = " + max);

        }

}
```