class Solution {

public int repeatedStringMatch(String A, String B) {

if(A == null || A.length() == 0){

return -1;

}

if(B == null || B.length() == 0){

return 0;

}

String str = "";

int result = 0;

while(str.length() < B.length()){

str += A;

result++;

}

if(str.indexOf(B) >= 0){

return result;

}

str += A;

result++;

if(str.indexOf(B) >= 0){

return result;

}

return -1;

}

}

class Solution {

int max = 0;

public int longestUnivaluePath(TreeNode root) {

if(root == null){

return 0;

}

int result = helper(root);

return Math.max(result, max);

}

private int helper(TreeNode root){

if(root == null){

return 0;

}

int left = helper(root.left);

int right = helper(root.right);

// int l = 0;

// int r = 0;

left = isEqual(root, root.left) ? left + 1 : 0;

right = isEqual(root, root.right) ? right + 1 :0;

// if(root.left != null && root.val == root.left.val){

// l = left + 1;

// }

// if(root.right != null && root.val == root.right.val){

// r = right + 1;

// }

max = Math.max(max, left + right);

return Math.max(left, right);

}

private boolean isEqual(TreeNode root, TreeNode child){

return child != null && root.val == child.val;

}

}