Ulric Aird

String Permutation Algorithm

Step 1: Input 2 string values.

Step 2: Check if the 2 input values are the same length. If you return false.

Step 3: Sort the 2 input values.

Step 4: Check if the 2 sorted input values are the same.

Step 5: If they return the same, output "YES"; if they donot match, output "NO".

The time complexity is O(n! \* n)

**Pseudo Code Demonstration**

Module Boolean has\_same\_characters(String str1, String str2) {

// Check if the string lengths are the same

if (str1.length does not equals str2.length) {

return false

}

// We split the string, sort it and join them together to find a match

string sortedStr1 = str1.split('').sort().join('')

string sortedStr2 = str2.split('').sort().join('')

// return true if string matches the other string. If it doesn’t match it will return false.

return sortedStr1 == sortedStr2

}

String str1

String str2

// Get the user input for the string

Input str1

Input str2

// runs the function block to execute the code

Boolean isSame = has\_same\_characters(str1, str2)

// Display whether the values are or are not the same.

If (isSame) {

Output “Both strings are the same”

} else {

Output "Both strings are not the same”

}