**7 kyu**

**All Inclusive?**

1191992% of 354115of 2,243[g964](https://www.codewars.com/users/g964)

C#

* [TRAIN AGAIN](https://www.codewars.com/kata/all-inclusive/train/csharp)
* [NEXT KATA](https://www.codewars.com/trainer/csharp)

Details

[Solutions](https://www.codewars.com/kata/all-inclusive/solutions/csharp)

[Forks (4)](https://www.codewars.com/kata/all-inclusive/forks/csharp)

[Discourse (97)](https://www.codewars.com/kata/all-inclusive/discuss/csharp)

* Add to Collection
* |
* Share this kata:

Input:

* a string strng
* an array of strings arr

Output of function contain\_all\_rots(strng, arr) (or containAllRots or contain-all-rots):

* a boolean true if all rotations of strng are included in arr (C returns 1)
* false otherwise (C returns 0)

#Examples:

contain\_all\_rots(

"bsjq", ["bsjq", "qbsj", "sjqb", "twZNsslC", "jqbs"]) -> true

contain\_all\_rots(

"Ajylvpy", ["Ajylvpy", "ylvpyAj", "jylvpyA", "lvpyAjy", "pyAjylv", "vpyAjyl", "ipywee"]) -> false)

**Note:**

Though not correct in a mathematical sense

* we will consider that there are no rotations of strng == ""
* and for any array arr: contain\_all\_rots("", arr) --> true

Ref: <https://en.wikipedia.org/wiki/String_(computer_science)#Rotations>

**Bash:**

For bash the array is replaced by a string (see "RUN sample tests).

<https://www.codewars.com/kata/all-inclusive/csharp>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

public static Boolean ContainAllRots(string strng, List<string> arr)

{

List<string> rotaciones = new List<string>();

for (int i = 0; i < strng.Length; i++)

{

rotaciones.Add(strng.Substring(i, strng.Length - i) + strng.Substring(0, i));

}

//foreach(string elem in rotaciones)

//{

// Console.WriteLine(elem);

//}

for (int i = 0; i < rotaciones.Count; i++)

{

if (!arr.Contains(rotaciones[i]))

{

return false;

}

}

//arr.Sort();

//for(int i =0; i<rotaciones.Count; i++)

//{

//}

return true;

}

public static Boolean ContainAllRots(string strng, List<string> arr)

{

List<string> rotaciones = new List<string>();

for (int i = 0; i < strng.Length; i++)

{

rotaciones.Add(strng.Substring(i, strng.Length - i) + strng.Substring(0, i));

}

arr.Sort();

for (int i = 0; i < rotaciones.Count; i++)

{

if(arr.BinarySearch(rotaciones[i]) < 0)

{

return false;

}

}

return true;

}

//solucion de Timp

public static Boolean ContainAllRots(string s, List<string> a)

{

if (s == "") return true;

for (int i = 0; i < s.Length; i++)

if (a.IndexOf(s.Substring(i) + s.Substring(0, i)) < 0)

return false;

return true;

}

static void Main(string[] args)

{

Console.WriteLine(ContainAllRots("bsjq", new List<string>(new string[] { "bsjq", "qbsj", "sjqb", "twZNsslC", "jqbs" }))); //-> true

// Console.WriteLine(ContainAllRots("Ajylvpy", new List<string>(new string[]{ "Ajylvpy", "ylvpyAj", "jylvpyA", "lvpyAjy", "pyAjylv", "vpyAjyl", "ipywee"}))); //-> false)

Console.ReadLine();

}

}

}