An [autogram](https://en.wikipedia.org/wiki/Autogram) is a sentence that describes itself in the sense of providing an inventory of its own characters. An essential feature is the use of full cardinal number names such as "one", "two", etc.

A special kind of *autogram* is a *reflexicon* (short for "reflexive lexicon"), which is a self-descriptive word list that describes its own letter frequencies.

Implement the Reflexicon function that returns true if the given sentence is a *reflexicon* and false otherwise.

**Example**

For

sentence = "sixteen e's, six f's, one g, three h's, nine i's,

nine n's, five o's, five r's, sixteen s's, five t's,

three u's, four v's, one w, four x's"

the output should be  
Reflexicon(sentence) = true.

There are indeed 16 'e's, 6 'f's, 3 'h's, 9 'i's, 9'n's, 5 'o's, 5 'r's, 16 's's, 3 'u's, 4 'v's, 1'w' and 4 'x's in the given sentence.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] string sentence**

Each sentence consists of several phrases of format *number* *letter*['s], separated by ,. It is guaranteed that each *number* is in range [1, 19].

*Constraints:*  
5 ≤ sentence.length ≤ 200.

* **[output] boolean**

true if the sentence is a *reflexicon* and falseotherwise.

<https://codefights.com/challenge/KmevSoGg4zJKfJaHd?utm_source=featuredChallenge&utm_medium=email&utm_campaign=email_notification>

static bool Reflexicon(string sentence)

{

string[] spl = sentence.Split(',');

string[] partes = new string[spl.Length];

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

{

partes[i] = spl[i].Trim();

}

Dictionary<char, int> frec = new Dictionary<char, int>();

Dictionary<char, string > frecSentencia = new Dictionary<char, string >();

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

foreach (string s in partes)

{

// Console.WriteLine(s);

//Console.Write(s.Split(' ')[1] + " ");

numeros.Add(s.Split(' ')[0]);

char key = s.Split(' ')[1][0];

// Console.WriteLine(key);

frec[key] = 0;

frecSentencia[key] = s.Split(' ')[0];

}

foreach (char ch in sentence)

{

//if(char.IsLetter(ch)){

if(frec.ContainsKey(ch)) {

frec[ch]++;

}

}

foreach (KeyValuePair<char, int> kvp in frec)

{

Console.WriteLine(kvp.Key + " " + kvp.Value);

}

string[] numsLetras = { "one", "two","three","four","five","six","seven", "eight","nine","ten","eleven","twelve" ,"thirteen","fourteen","fifteen","sixteen", "seventeen", "eighteen","nineteen"};

//foreach (string elem in numeros)

//{

// Console.WriteLine(elem);

//}

foreach (KeyValuePair<char, string> kvp in frecSentencia)

{

Console.WriteLine(kvp.Key + " " + kvp.Value);

}

foreach (KeyValuePair<char, int> kvp in frec)

{

if (kvp.Value != numsLetras.ToList().IndexOf( frecSentencia[kvp.Key]) + 1) {

return false;

}

}

return true;

}