Your task is to determine the number of moves it will take to get from the first character of the given string maze to the last one.

Each character of maze is a lowercase letter of the English alphabet. Each letter is assigned an integer value: 'a' = 1, 'b' = 2, 'c' = 3, and so on, with 'z' = 26. Starting from the first character of the maze, you can move n characters to the right, where n is the value assigned to the letter at your current position. For example, if you are standing on the character 'c', you should move 3 positions to the right.

If you land up on a character that lies farther than the last character of the maze, start over from the beginning and keep counting. For example, if you pass over the last character by four, you will end up at the fourth character from the beginning.

Given the maze, return the number of moves required to get to its last character from its first one as described above. If the algorithm turns out to be an infinite loop that never lands on the last character, return -1 instead.

**Example**

* For maze = "able", the output should be  
  stringMaze(maze) = 2.

Starting on 'a', move one character right to the 'b'. Since 'b' equals 2, move two more characters right to the 'e'. 'e' is the last character in the maze, so you are finished. This took two moves, so the output should be 2.

* For maze = "aced", the output should be  
  stringMaze(maze) = -1.

Starting on the 'a', move one character right to the 'c'. Since 'c'equals 3, move three more characters right, wrapping around to the 'a'again. 'a' and 'c' form an infinite loop, so the answer is -1.

**Input/Output**

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

A string of lowercase English letters.

*Constraints:*  
2 ≤ maze.length ≤ 25.

* **[output] integer**

The number of moves required to land on the last character in the maze, or -1 if it's impossible to get there.

<https://codefights.com/challenge/Sa6rfk78Qmvw293FY?utm_source=featuredChallenge&utm_medium=email&utm_campaign=email_notification>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static int stringMaze(string maze)

{

HashSet<int> indices = new HashSet<int>();

int i = 0, movimientos = 0;

while (true)

{

//Console.WriteLine(i);

if (indices.Contains(i))

{

return -1;

}

movimientos++;

indices.Add(i);

int avanza = (maze[i] - 96) % maze.Length;

//Console.WriteLine(avanza);

i += avanza;

if (i > maze.Length-1)

{

i = i - maze.Length;

}

if (i == maze.Length-1)

{

return movimientos;

}

}

//Console.WriteLine(movimientos);

return movimientos;

}

static void Main(string[] args)

{

//Console.WriteLine('a' - 96);

//for (char i = 'a'; i <= 'z'; i++)

//{

// Console.Write( (i - 96) + " ");

//}

//string maze = "able";

//string maze = "aced";

// string maze = "uber"; //2

string maze = "fffff";

//string maze = "aced";

Console.WriteLine(stringMaze(maze));

Console.ReadLine();

}

}

}