```
indice 0 x:xs = x
indice n x:xs = indice (n-1) xs
```

Evaluamos

```
indice 2 1:[2,3,4] = indice (2-1) [2,3,4]
indice 1 2:[3,4] = indice (1-1) [3,4]
={Por caso base}
indice 0 3:[4] = 3
```

13. Considerando la función *soloPares* : $[Num] \to [Num]$ definida de la siguiente manera

$$
\begin{array}{lcll}
soloPares.[\,] & \doteq & [\,] & \\
soloPares.(x \triangleright xs) & \doteq & (& x \bmod 2 = 0 \to x \triangleright soloPares.xs \\
 & & \square & x \bmod 2 \neq 0 \to soloPares.xs \\
 & &) &
\end{array}
$$

demostrá que

$$soloPares.(xs +\!\!+\, ys) = soloPares.xs +\!\!+\, soloPares.ys$$

```
[] ++ ys = ys                        (1)
xs ++ ys = x: xs++ys                 (2)

soloPares [] = []                    (3)
soloPares (x:xs)
   |x `mod` 2 == 0 = x: soloPares xs  (4)
   |x `mod` 2 /= 0 = soloPares xs     (5)
```

Caso base: Reemplazamos a xs por [].

```
soloPares (xs ++ ys) = soloPares xs ++ soloPares ys

soloPares ([] ++ ys) = soloPares [] ++ soloPares ys
={Por (1) y (3)}
soloPares ys = [] ++ soloPares ys
={Por (1)}
soloPares ys = soloPares ys
={Reflexividad del =}
True
```

```
Caso inductivo: reemplazamos a xs por una lista no vacia (x:xs)
  Caso 1: x `mod` 2 == 0

soloPares (xs ++ ys) = soloPares xs ++ soloPares ys

soloPares (x:xs ++ ys) = soloPares (x:xs) ++ soloPares ys
={Por (2) y (4)}
soloPares (x: xs ++ ys) = x: soloPares xs ++ soloPares ys
={Por (4), x:=x, xs:=(xs++ys)}
x: soloPares (xs ++ ys) = x: soloPares xs ++ soloPares ys
={Por HI}
x: soloPares xs ++ soloPares ys = x: soloPares xs ++ soloPares ys
={Por reflexividad del =}
True


  Caso 2: x `mod` 2 /= 0

    soloPares (xs ++ ys) = soloPares xs ++ soloPares ys

    soloPares (x:xs ++ ys) = soloPares x:xs ++ soloPares ys
    ={Por (2) y (5)}
    soloPares (x: xs ++ ys) = soloPares xs ++ soloPares ys
    ={Por (5), x:=x; xs:=(xs++ys)}
    soloPares (xs ++ ys) = soloPares xs ++ soloPares ys
    ={Por HI}
    soloPares xs ++ soloPares ys = soloPares xs ++ soloPares ys
    ={Reflexividad del =}
    True
```