

Polar circularity recover original list

By Luis Felipe Massena Misiec

(\* Lista de somas consecutivas fornecida \*)

```
consecutiveSums = {1.000, 0.*10^-4, 0.*10^-4, 2.000, -2.000, 10.000, -10.000,  
  4.000, 4.000, -4.000, 4.000, 0.*10^-4, -8.000, 0.*10^-4,  
  6.000, -2.000, 0.*10^-4, 4.000, -4.000, 0.*10^-4,  
  -2.000, -2.000, 8.000, -6.000, 0.*10^-4, 4.000,  
  -4.000, 8.000, -6.000, 10.000, -6.000, -6.000,  
  14.000, -16.000, 4.000, -4.000, 8.000, 0.*10^-4,  
  -8.000, 22.000, -12.000, -10.000, 6.000, 2.000,  
  12.000, -12.000, 10.000, -14.000, 12.000, -10.000,  
  2.000, -2.000, -6.000, 4.000, 10.000, -10.000,  
  -4.000, 12.000, 4.000, -14.000, 0.*10^-4, 2.000};
```

(\* Função para recuperar a lista original \*)

```
recoverOriginal[consecutiveSums_] := Module[{n = Length[consecutiveSums], originalList},  
  originalList = Table[0, {n + 1}]; (* Cria uma lista para armazenar os valores recuperados *)  
  originalList[[1]] = consecutiveSums[[1]]; (* Define o primeiro elemento *)
```

(\* Recupera os elementos originais usando as somas consecutivas \*)

```
Do[  
  originalList[[i + 1]] = consecutiveSums[[i]] - originalList[[i]],  
  {i, 1, n - 1}  
];
```

```
originalList
```

```
];
```

(\* Recupera a lista original a partir das somas consecutivas \*)

```
recoveredOriginalList = recoverOriginal[consecutiveSums];
```

(\* Exibe a lista original recuperada \*)

recoveredOriginalList

Explicação do Código:

Lista de Somas: A lista

consecutiveSums

```
{1.,0.,0.,0.,2.,-4.,14.,-24.,28.,-24.,20.,-16.,16.,-24.,24.,-18.,16.,-16.,20.,-24.,24.,-26.,24.,-16.,10.,-10.,14.,-18.,26.,-32.,42.,-48.,42.,-28.,12.,-8.,4.,4.,-4.,-4.,26.,-38.,28.,-22.,24.,-12.,0.,10.,-24.,36.,-46.,48.,-50.,44.,-40.,50.,-60.,56.,-44.,48.,-62.,62.,0}
```

do Explicação (Código:de Lista Somas:A lista)

```
{1.,0.,0.,2.,-2.,10.,-10.,4.,4.,-4.,4.,0.,-8.,0.,6.,-2.,0.,4.,-4.,0.,-2.,-2.,8.,-6.,0.,4.,-4.,8.,-6.,10.,-6.,-6.,14.,-16.,4.,-4.,8.,0.,-8.,22.,-12.,-10.,6.,2.,12.,-12.,10.,-14.,12.,-10.,2.,-2.,-6.,4.,10.,-10.,-4.,12.,4.,-14.,0.,2.}
```

The next lines show a positive successful result :

(\* Function to sum until two elements, storing intermediate steps with normalization \*)

```
sumUntilTwoElementsWithNormalization[consecutiveSums_] := Module[{sums = consecutiveSums, steps = {}},
```

```
While[Length[sums] > 2,
```

```
AppendTo[steps, sums];
```

```
sums = Table[
```

```
Module[{sum = sums[[i]] + sums[[i + 1]]},
```

```
(* Normalize the sum to be within 1 or 2 digits *)
```

```
If[Abs[sum] >= 10, sum = Mod[sum, 10]];
```

```
sum
```

```
], {i, 1, Length[sums] - 1}];
```

```
];
```

```
{sums, steps}
```

```
];
```

(\* Reverse the process to get original list with correct length \*)

```

reverseSumUntilTwoElementsWithNormalization[finalSums_, steps_, originalLength_] :=
Module[{currentSums = finalSums, reversed = {}},

Do[

currentSums = Flatten[Table[

Module[{sum = steps[[i, j]], diff = currentSums[[j + 1]] - steps[[i, j]]},

(* De-normalize the sum if needed *)

If[Abs[diff] >= 10, diff = diff + 10];

sum

], {j, Length[currentSums] - 1}

]];

PrependTo[reversed, currentSums];

, {i, Length[steps], 1, -1}

];

(* Limit the reversed list to the original length *)

Take[Flatten[{First[steps], reversed}], originalLength]

];

```

(\* Provide the consecutive sums list \*)

```
consecutiveSums = {1, 0, 0, 2, -2, 10, -10,
```

```
4, 4, -4, 4, 0, -8, 0,
```

```
6, -2, 0, 4, -4, 0,
```

```
-2, -2, 8, -6, 0, 4,
```

```
-4, 8, -6, 10, -6, -6,
```

```
14, -16, 4, -4, 8, 0,
```

```
-8, 22, -12, -10, 6, 2,
```

```
12, -12, 10, -14, 12, -10,
```

```
2, -2, -6, 4, 10, -10,
```

```
-4, 12, 4, -14, 0, 2};
```

(\* Execute the function to sum and store intermediate steps \*)

```
result = sumUntilTwoElementsWithNormalization[consecutiveSums];
```

```
(* Extract final sums and steps *)
```

```
finalSums = First[result];
```

```
steps = Last[result];
```

```
(* Display the final sums *)
```

```
finalSums
```

```
(* Execute the function to reverse the process *)
```

```
originalSums = reverseSumUntilTwoElementsWithNormalization[finalSums, steps,  
Length[consecutiveSums]];
```

```
(* Display the original sums with the correct length *)
```

```
originalSums
```

```
{3, 6}
```

```
{1, 0, 0, 2, -2, 10, -10, 4, 4, -4, 4, 0, -8, 0, 6, -2, 0, 4, -4, 0, \  
-2, -2, 8, -6, 0, 4, -4, 8, -6, 10, -6, -6, 14, -16, 4, -4, 8, 0, -8, \  
22, -12, -10, 6, 2, 12, -12, 10, -14, 12, -10, 2, -2, -6, 4, 10, -10, \  
-4, 12, 4, -14, 0, 2}
```