

# Soluções dos Problemas



FCUL Rally Pro 2018

# 1) somar e subtrair alternadamente

Percorrer a lista e saber se estamos num índice ímpar ou par



## 2) quantos números maiores que o anterior

Em cada ciclo mantemos dois números para determinar se a condição é verificada



### 3) Pares de somas pares

Pesquisar elemento a elemento nas duas listas (a variável *indice* como índice comum) e contabilizar apenas os pares de elementos que nos interessam.

```
definição de contador para 0
definição de indice para 1
repetir até indice > tamanho de Lista de valores 2
  faça
    definir item1 para na lista Lista de valores 1 obter # indice
    definir item2 para na lista Lista de valores 2 obter # indice
    se item1 + item2 é par
      faça
        definir contador para contador + 1
      definir indice para indice + 1
imprime contador
```

The image shows a Scratch script designed to count the number of pairs of elements from two lists whose sum is even. The script starts by initializing a 'contador' (counter) to 0 and an 'indice' (index) to 1. It then enters a loop that repeats until the 'indice' is greater than the 'tamanho de' (size of) 'Lista de valores 2'. Inside this loop, it performs the following steps: 1) Retrieve 'item1' from 'Lista de valores 1' at the current 'indice'. 2) Retrieve 'item2' from 'Lista de valores 2' at the current 'indice'. 3) Check if the sum of 'item1' and 'item2' is even ('é par'). 4) If the sum is even, increment the 'contador' by 1. 5) Increment the 'indice' by 1 to move to the next element in the lists. Finally, the script prints the value of 'contador'.

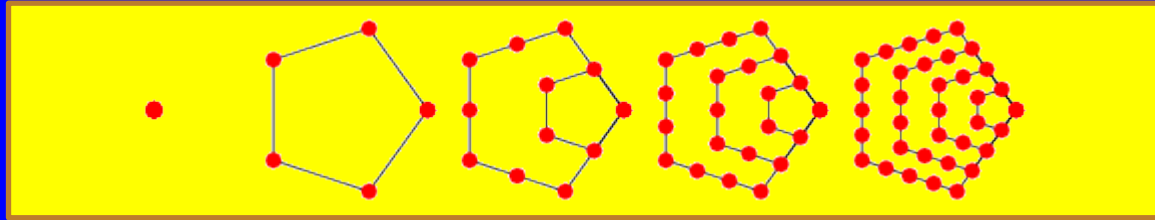
## 4) Triplos Pitagóricas

Usamos três ciclos tendo a atenção que as guardas dos ciclos ajudem a satisfazer a condição  $a < b < c$ , tornando o algoritmo mais eficiente



## 5) Números pentagonais

A dificuldade deste problema era perceber qual a expressão numérica que produz o n-ésimo número pentagonal  $P_n$

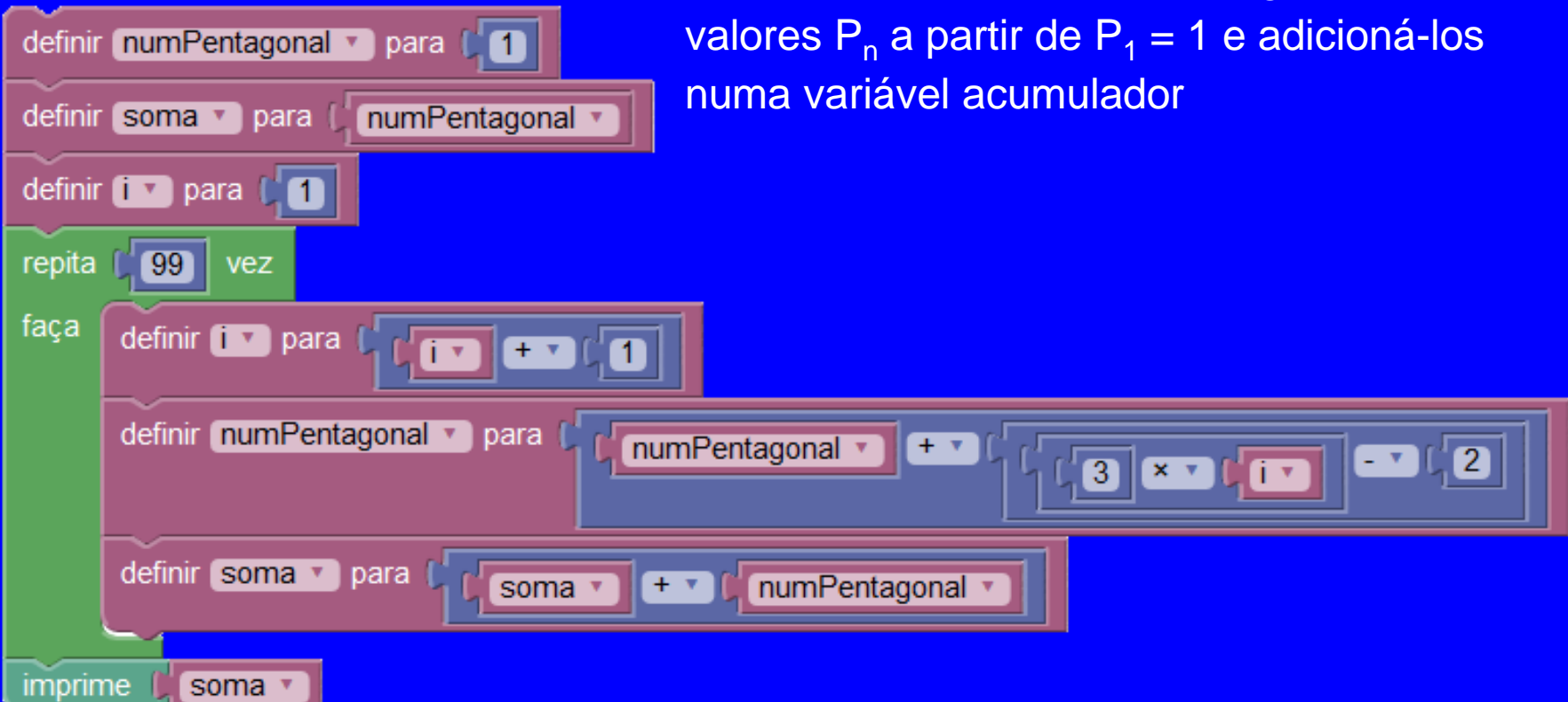


A figura correspondente a um número pentagonal adiciona 3 novas arestas à figura anterior.

Se for o  $i$ -ésimo pentágono, essas três arestas têm  $i$  pontos cada, mas dois desses pontos são vértices partilhados por duas arestas, logo têm-se de descontar, ie,  $P_i = P_{i-1} + 3i - 2$

## 5) Números pentagonais

Tendo a expressão, é preciso gerar os 100 valores  $P_n$  a partir de  $P_1 = 1$  e adicioná-los numa variável acumulador



## 6) Ladrilhos

Este problema tem a ver com a representação ternária de números.

Se os blocos fossem potências de 10, seria fácil ver que para ladrilhar, por exemplo, 123, seriam precisos 6 ladrilhos, um de 100, dois de 10, três de 1.

Como não temos um conversor de decimal para ternário, podemos usar uma abordagem egoísta (greedy algorithm) e ir ladrilhando com as maiores potências, subtraindo a distância que sobra até que esta seja zero.

Por exemplo, para 123 seria:

- $123 - 1 \times 3^4 = 123 - 81 = 42$
- $42 - 1 \times 3^3 = 42 - 27 = 15$
- $15 - 1 \times 3^2 = 15 - 9 = 6$
- $6 - 2 \times 3^1 = 0$

Ou seja,  $123_{10} = 11120_3$

precisaríamos de 5 ladrilhos



## 6) Ladrilhos

```
definir blocos para [criar lista com [531441] [177147] [59049] [19683] [6561] [2187] [729] [243] [81] [27] [9] [3] [1]]
definir distancia para [1234567]
definir numBlocos para [0]
definir blocos_indice para [1]
repita até [distancia ≤ 0]
faça
  repita até [distancia ≥ na lista blocos obter # blocos_indice]
  faça
    definir blocos_indice para [blocos_indice + 1]
  definir distancia para [distancia - na lista blocos obter # blocos_indice]
  definir numBlocos para [numBlocos + 1]
imprime numBlocos
```

# 7) Créditos e débitos

A técnica é ir guardando, no intervalo corrente, o número seguinte se a soma total não for zero.

Para cada novo intervalo corrente, guardamo-lo como resposta se este for o maior até agora encontrado.

```
definição de função somaCorrente para 0
definição de função resposta para 0
definição de função indice para 1
repetir tamanho de Lista de valores 1 vez
  faça
    definir elemento para na lista Lista de valores 1 obter # indice
    se somaCorrente + elemento >= 0
      faça
        definir somaCorrente para somaCorrente + elemento
        se somaCorrente > resposta
          fazer definir resposta para somaCorrente
      senão
        definir somaCorrente para 0
    definir indice para indice + 1
  imprimir resposta
```

Se o intervalo corrente ficar negativo, deixa de nos interessar e recomeçamos um novo intervalo corrente com o número positivo seguinte.