

# Soluções dos Problemas



FCUL Rally Pro 2016

# 1) A soma dos números ímpares da lista 1

É necessário percorrer a lista e somar - numa nova variável - apenas os elementos que satisfazem a condição de ser ímpar:



## 2) O segundo menor número da lista 1

Um exemplo que utiliza as funcionalidades do ambiente: criar uma nova lista sem o menor valor, e calcular o menor valor desta nova lista.

```
definir menor para menor de uma lista Lista de valores 1
definir idx_menor para na lista Lista de valores 1 encontre a primeira ocorrência do item menor
definir lista2 para Lista de valores 1
na lista lista2 definir # idx_menor como maior de uma lista Lista de valores 1
imprime menor de uma lista lista2
```

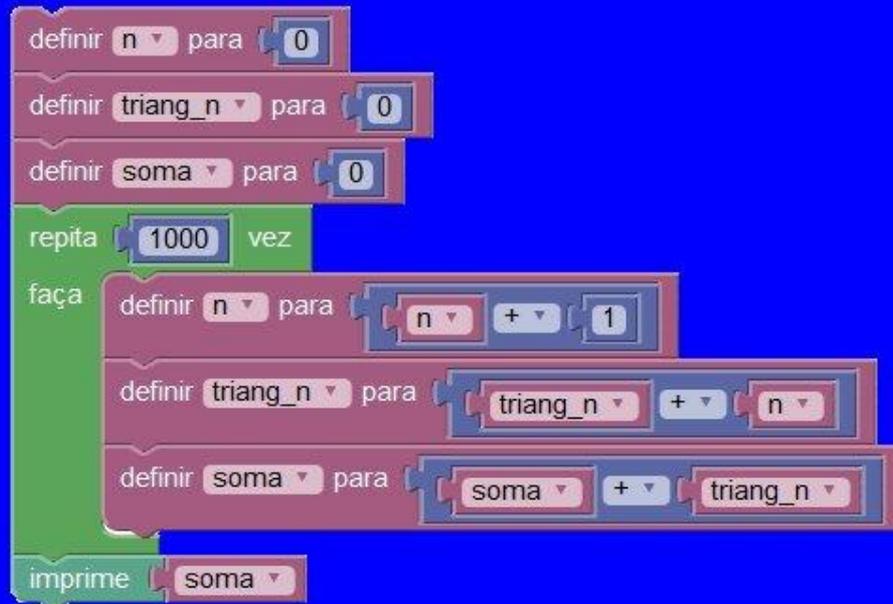
### 3) A soma dos elementos na mesma posição

Pesquisar sincronizadamente, elemento a elemento, nas duas listas (uso da variável *i* como índice comum) e somar apenas os elementos iguais.



## 4) Soma de números triangulares

O  $n$ -ésimo  $n^{\circ}$  triangular é  $1+2+\dots+n$ . Podemos calcular o  $n^{\circ}$  triangular seguinte a partir do anterior ( $\text{triang\_n} = \text{triang\_n} + n$ ). Assim, define-se um ciclo que calcula o próximo número e o soma de imediato (variável soma).

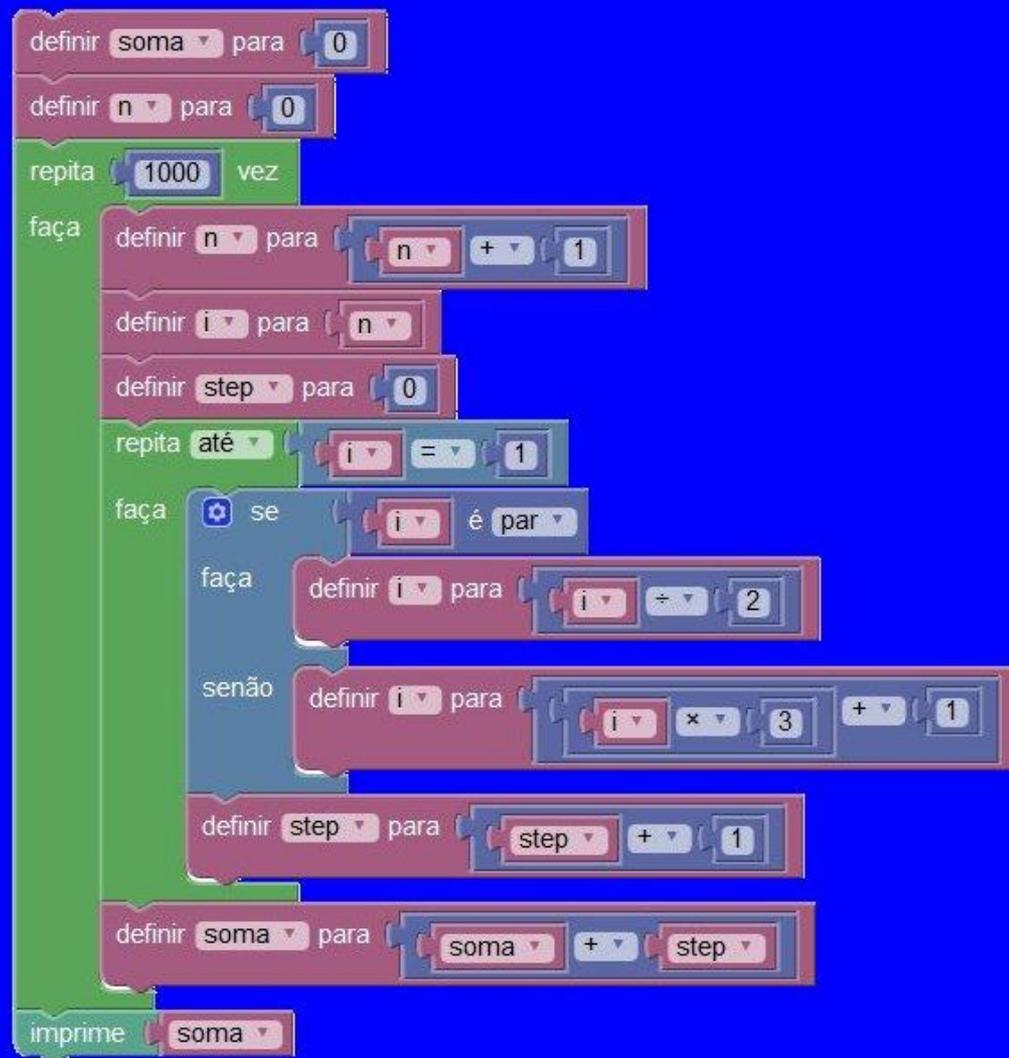


## 5) A Conj. de Collatz

Dado um natural  $n$ , dividir por dois se for par, triplicar e somar 1 se for ímpar. Com este novo número repetir o processo.

A **conjectura de Collatz** diz-nos que este algoritmo termina sempre para qualquer  $n$  inicial.

O algoritmo seguinte calcula a soma do total de passos dados quando se aplica o algoritmo aos primeiros mil números naturais.



## 6) Sequência de Tribonacci

Seja uma sequência numérica dada por estas regras:

$$\text{trib}(1) = \text{trib}(2) = \text{trib}(3) = 1$$

$$\text{trib}(n) = \text{trib}(n-3) + \text{trib}(n-2) + \text{trib}(n-1); \text{ para } n > 3$$

Calcular a soma dos primeiros 60 números desta sequência.

```
definição de trib para criar lista com o item 0 repetido 60 vezes
na lista trib definir # 1 como 1
na lista trib definir # 2 como 1
na lista trib definir # 3 como 1
definir i para 4
repita 57 vez
faça
  definir sum para na lista trib obter # i - 1
  definir sum para sum + na lista trib obter # i - 2
  definir sum para sum + na lista trib obter # i - 3
na lista trib definir # i como sum
definir i para i + 1
imprime soma da lista trib
```

The image shows a Scratch script to calculate the sum of the first 60 Tribonacci numbers. The script starts by defining a list named 'tribs' with 60 zeros. It then sets the first three elements of the list to 1. A loop repeats 57 times, starting from index 4. In each iteration, it calculates the sum of the three previous elements (indices i-1, i-2, and i-3) and stores it in a variable 'sum'. This sum is then stored in the 'tribs' list at index 'i'. Finally, the index 'i' is incremented by 1. After the loop, the sum of all elements in the 'tribs' list is printed.

# 7) Jogo de Bachet

Este problema requer guardar as soluções intermédias para acelerar o cálculo. Usamos uma lista para esse efeito.

Esta técnica é conhecida por programação dinâmica.

```
definiir n para 1000
definiir bachet para criar lista com o item 0 repetido n + 1 vezes
na lista bachet definir # 1 como 2
definiir moves para criar lista com 1, 3, 6
definiir pedras para 1
repeita n vez
  faça
    na lista bachet definir # pedras + 1 como 2
    para cada item move na lista moves
      faça
        se pedras >= move
          faça
            se
              na lista bachet obter # pedras + 1 - move = 2
              faça
                na lista bachet definir # pedras + 1 como 1
            definir pedras para pedras + 1
  imprime soma da lista bachet
```