Função `fatias_nulas()`

Uma fatia de uma lista de números é **nula** se ela tem comprimento pelo menos um e a soma dos seus números é zero.
Por exemplo, `lst[1:5]` e `lst[6:7]` são fatias nulas de `lst = [ 6, 1, -2, -3, 4, 15, 0, -2]`.

O que você deve fazer

Escreva a função `fatias_nulas()` como especificada abaixo.

```
def fatias_nulas( lst ):
    '''(list) -> int
    RECEBE uma lista `lst`.
    RETORNA o número de fatias nulas de `lst`.
    '''
```

Exemplos

```
    In [2]: fatias_nulas([1, -2, 3])
    Out[2]: 0

    In [3]: fatias_nulas([1, -2, 3, 0])
    Out[3]: 1

    In [4]: fatias_nulas([1, -2, 3, 0, -1])
    Out[4]: 2

    In [5]: fatias_nulas([-2, 1, -2, 3, 0, -1])
    Out[5]: 4

    In [6]: fatias_nulas([0, 0, 0])
    Out[6]: 6
```

O que você deve entregar

Você deve entregar um arquivo com o nome `nulas.py` contendo a sua solução. Você deve incluir quaisquer outras funções auxiliares que implementar.

Deposite sua solução, mesmo que **incompleta (!)** ou com **erros (!)** ANTES de esgotar o prazo de 60 minutos.

Triângulos

Números $a \geq b \geq c > 0$ são comprimentos dos lados de um triângulo se $a < b + c$.

Suponha que a, b e c são comprimentos dos lados de um triângulo, $a \geq b \geq c > 0$. Quanto aos **ângulos** diz-se que um triângulo é

- *retângulo* se $a^2 = b^2 + c^2$;
- *obtusângulo* se $a^2 > b^2 + c^2$; ou
- *acutângulo* se $a^2 < b^2 + c^2$.

Quanto aos **lados** diz-se que um triângulo é

- *equilátero* se os três lados têm o mesmo comprimento;
- *isósceles* se exatamente dois lados têm o mesmo comprimento; ou
- *escaleno* se os três lados têm comprimentos diferentes.

O que você deve fazer

Escreva uma função `main()` que LÊ três números *inteiros positivos*, **não** necessariamente ordenados, e IMPRIME uma mensagem indicando se eles são ou não comprimentos dos lados de um triângulo (`triângulo` ou `não triângulo`).

No caso deles serem comprimentos dos lados de um triângulo o programa **deve** ainda IMPRIMIR mensagens indicando a classificação do triângulo quanto aos *ângulos* (`retângulo`, `obtusângulo` ou `acutângulo`) e quanto aos *lados* (`equilátero`, `isósceles` ou `escaleno`).

Exemplos de execução

```
Digite 3 inteiros positivos: 3 2 4
triângulo
obtusângulo
escaleno

Digite 3 inteiros positivos: 5 3 4
triângulo
retângulo
escaleno

Digite 3 inteiros positivos: 3 3 3
triângulo
acutângulo
equilátero

Digite 3 inteiros positivos: 3 4 3
triângulo
acutângulo
isósceles

Digite 3 inteiros positivos: 5 2 3
não triângulo
```

O que você deve entregar

Você deve entregar um arquivo com o nome `triangulo.py` contendo a sua solução. Sua solução deve se comportar **exatamente** da mesma forma como nos exemplos. Não se preocupe com entradas inválidas. Você deve incluir quaisquer outras funções auxiliares que desejar implementar.

Deposite sua solução, mesmo que **incompleta (!)** ou com **erros (!)** ANTES de esgotar o prazo de 60 minutos.