Unidade 3: Iniciação à programação em Python | **Lição 3: Programação e Recursividade**

Nesta terceira lição da Unidade 3, aplicará o conhecimento sobre algoritmos e a linguagem Python para programar com recursividade.

Objetivos:

- Aplicar uma função em linguagem Python.
- Implementar programação com recursividade.

Cálculo do Máximo Divisor Comum (Método iterativo)

Para calcular o máximo divisor comum entre dois números naturais **a** e **b**, **mdc(a,b)**, utilizaremos o algoritmo de Euclides. Note-se que consideraremos a > b.

Procederemos da seguinte forma:

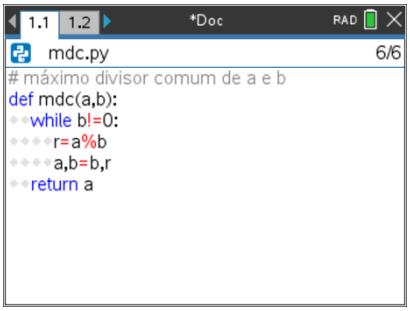
- Efetuamos a divisão euclidiana de **a** por **b**. Designamos o resto por **r** (não utilizamos o quociente).
- De seguida trocamos **a** por **b** e **b** por **r**.
- Enquanto o resto for diferente de 0, repetimos o processo.

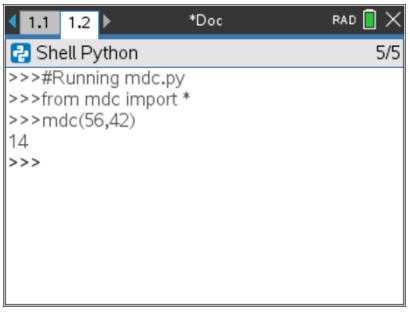
Após um determinado número de iterações, obteremos resto 0.

Desta forma, obtemos o **mdc(a,b)** que será o **último resto não nulo** do processo recursivo anterior.

IMPLEMENTAÇÃO DO PROGRAMA:

- Crie um novo programa, designando-o por **mdc**.
- Crie uma função **mdc(a,b)** teclando menu, depois **4: Planos integrados** e por fim **1: Funções**.
- O sinal ≠ representa-se em linguagem Python por **!=** , que se acede através do menu **4: Planos integrados** depois **3: Ops** ou pelo atalho ctrl + = da unidade portátil.
- Note que a atribuição **a,b=b,r** no programa permite o ganho de uma linha de código, já que corresponde às atribuições **a=b** e **b=r**.
- Execute o programa com diferentes pares de números. Verifique a correção dos resultados obtidos.





MAIS ALÉM:

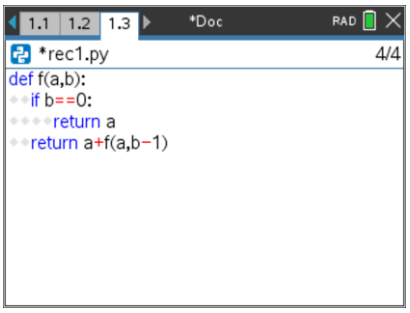
Programação com recursividade

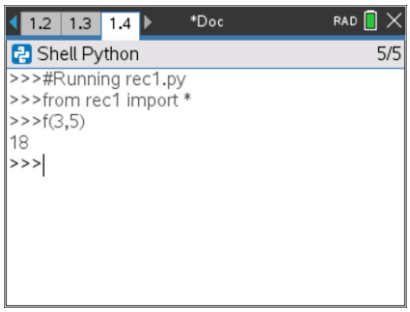
Um algoritmo diz-se recursivo quando em algum momento se chama a si próprio.

A recursividade pode ter muitas vantagens num algoritmo. Em primeiro lugar, pode resolver problemas geralmente insolúveis com a utilização apenas de ciclos **FOR** ou **WHILE**. Pode ainda tornar um algoritmo mais legível e curto, mas sobretudo permite, em alguns casos, uma grande economia de tempo.

Um primeiro exemplo de recursividade:

- Crie um novo programa com o nome **rec1**.
- Escreva o programa que se encontra no ecrã ao lado.
- Quais serão os primeiros casos (valores de a e b) a aplicar este programa? Que resultado obterá?
  (Lembre-se de que a e b são inteiros positivos com $a > b$)
- O que garante, nesta função recursiva, que o programa vai parar?
- Considere a=5 e b=3, efetue manualmente, com papel e lápis, todos os procedimentos desta função. Que resultado obteve?
- Execute, agora, o programa, usando o atalho ctrl + R, e no interpretador execute a função ***f***.
- Explore, comparando resultados, situações como $f(a,b)$, $f(b,a)$, $f(a,b+c)$ e $f(a,b)+f(a,c)$. Que igualdades lhe sugerem?
- O que traduz o resultado da função $f(a,b)$ com a e b naturais?





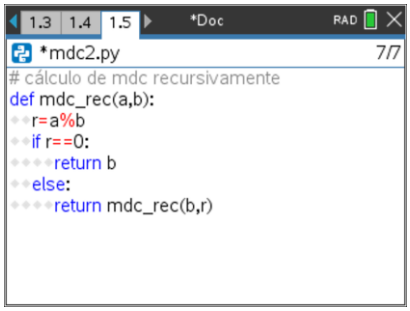
Um cálculo do mdc recursivamente

O cálculo do máximo divisor comum de dois números naturais ***a*** e ***b*** utiliza sempre o algoritmo de Euclides.

Sendo ***r*** o resto e ***q*** o quociente da divisão euclidiana de ***a*** por ***b*** teremos que:

$a = b*q + r, \quad r < b.$

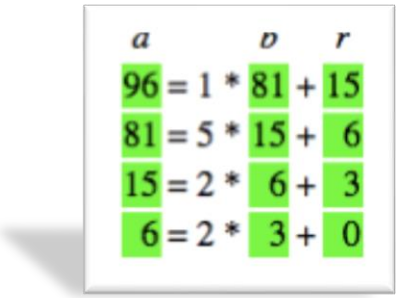
Qualquer divisor comum de ***a*** e ***b*** é também divisor de ***r***, sendo $r = a - b*q$, e reciprocamente qualquer divisor comum de ***b*** e ***r*** divide $a = b*q + r$.



Assim, o cálculo do mdc(a,b) é reduzido ao cálculo do mdc(b,r), e podemos começar de novo, sem receio, um ciclo recursivo, pois os sucessivos restos constituem uma sequência estritamente decrescente. O último resto não nulo é o mdc pretendido.

Por exemplo, para ***a***=96 e ***b***=81, os cálculos sucessivos obtidos pela função **mdc_rec()** do programa anterior são os seguintes:

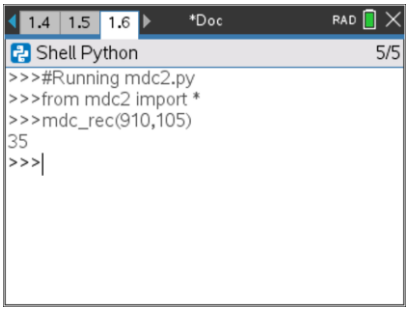
- mdc_rec(96,81)
  r=15
- mdc_rec(81,15)
  r=6
- mdc_rec(15,6)
  r=3
- mdc_rec(6,3)
  r=0
- **3**



Portanto, o mdc(96,81)=3 .

- Execute o programa, premindo ctrl + R, e utiliza a função **mdc_rec()** com diferentes pares de números naturais.

- Explore propriedades da função máximo divisor comum recorrendo ao cálculo, através da função **mdc_rec()**, de um número significativo de casos que lhe permitam formular conjeturas!



IMPORTANTE:

Para evitar ciclos viciosos, sem fim, uma função recursiva deve sempre incluir um caso particular em que o resultado é calculado diretamente, ou seja, sem chamada recursiva; deve-se também garantir que este caso particular seja sempre atingido no final.