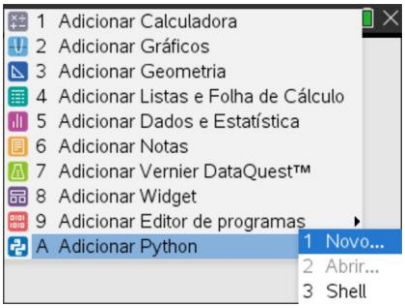
```
x=int(input("Indica o 1.° termo da sucessão: "))
seq=[]
def collatz_sequencia(x):
    seq = [x]
    if x < 1:
      return []
    while x > 1:
      if x % 2 == 0:
         x = x / 2
      else:
         x = 3 * x + 1
        seq.append(int(x))
    return seq
print(collatz_sequencia(x))
```

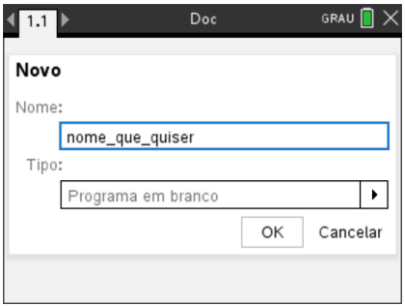
Editar Python na TI-nspire CX II-T

Ligue a sua calculadora e crie um novo documento.
Escolha uma página de *Python*:

menu A **Adicionar Python** → 1 **Novo.**

Coloque um nome à sua escolha, de seguida, prime em **OK**.
Abre-se uma página vazia, que é o editor de *Python* da calculadora/tecnologia TI-Nspire CX II-T, onde deve escrever o código.

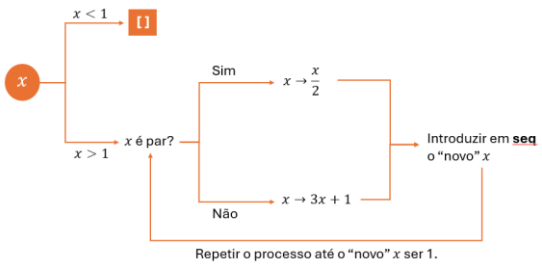


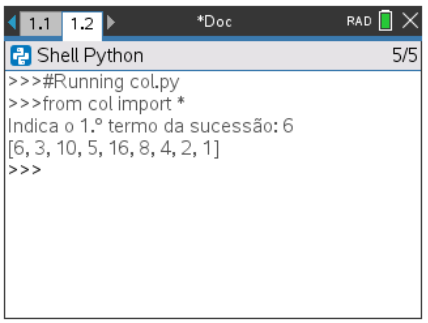


1. A partir de um número natural dado, como simular a sequência de Collatz?

Pretende-se um programa que, depois de executado, apresente uma lista com os termos de uma sequência finita cujo último termo desta é 1.

A lei de formação desta sequência é a seguinte: se o número for par, divide-se por dois, caso seja ímpar, multiplica-se por três e soma-se um, e assim sucessivamente até chegar no número um. O algoritmo para a formação da sequência segue-se no esquema ao lado.





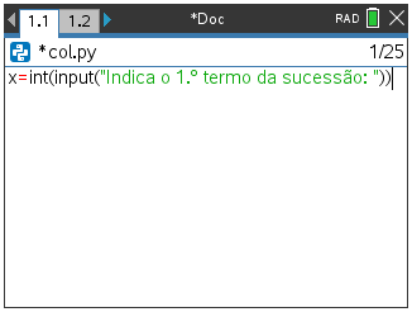
Por exemplo, a sequência de Collatz cujo primeiro elemento é o 6 é $[6,10,5,16,8,4,2,1]$:

$$6 \xrightarrow{\div 2} 3 \xrightarrow{\times 3+1} 10 \xrightarrow{\div 2} 5 \xrightarrow{\times 3+1} 16 \xrightarrow{\div 2} 8 \xrightarrow{\div 2} 4 \xrightarrow{\div 2} 2 \xrightarrow{\div 2} 1$$

I. Pretende-se fazer um programa que, inicialmente, pergunte, ao utilizador qual o primeiro termo da sequência, **x**, e que, no final, devolva a sequência de Collatz para o primeiro termo recebido.

A função que permite receber um dado por parte do utilizador é **input()** e para que este seja atribuído numa variável, à qual deve ser dado um nome, tem de ser identificado que se trata de um valor numérico decimal, com **float()**.

```
x=float(input("Indica o 1.º termo da sequência:"))
```



Para obter a função **input()**, pode escrever com o teclado ou, a partir do menu:

menu → 4 **Planos integrados** → 6 **I/O** → 2 **input()**

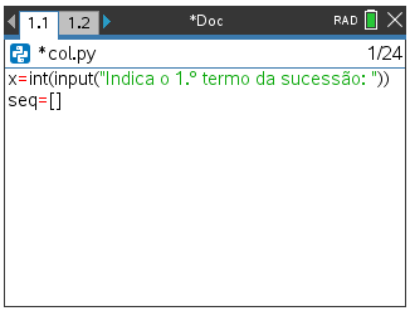
Já para obter a função **float()**, pode escrever com o teclado ou, a partir do menu:

menu → 4 **Planos integrados** → 5 **Tipo** → 1 **int() /** 2 **float()**

(Para escrever palavras, letras ou texto, que não sejam variáveis, tem de colocar aspas " ".)

II. A sequência de Collatz é uma sequência onde os termos seguintes dependem dos anteriores. Estes termos, ao longo da construção da sequência, vão sendo armazenados numa lista, inicialmente, vazia que será chamada de ***seq***. Portanto, a linha de código seguinte a ser acrescentada é:

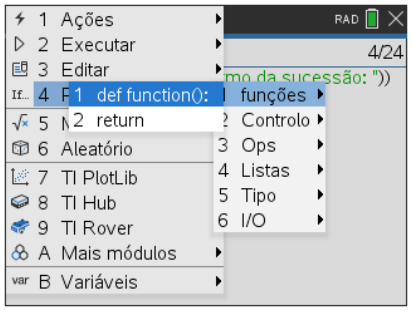
```
seq=[ ]
```



III. De seguida, segue-se a construção de uma "função" que recebe o valor do primeiro termo, constrói a respetiva sequência de Collatz e, por fim, a devolve em forma de lista. Esta "função" é semelhante ao que noutras linguagens de programação se chama de sub-rotina, ou seja, uma secção de código que é ativada a partir das linhas de código do
programa subsequentes. Esta estrutura é bastante útil, por exemplo, quando no programa se precisa de fazer algumas vezes as ações que nela estão previstas.

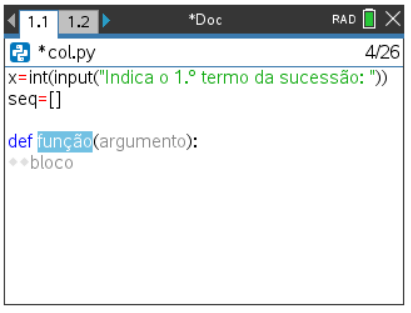
Para obter esta estrutura, caso conheça a sintaxe, poderá escrever com o teclado, mas pode também obter através do menu:

menu → 4 **Planos integrados** → 1 **funções** → 1 **def function():**

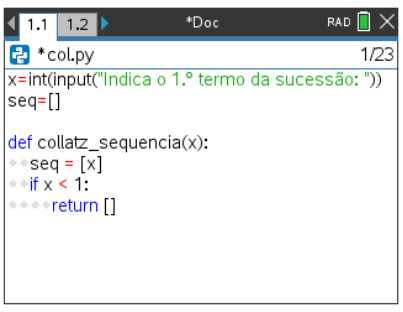


Obtida a estrutura, é agora necessário preencher os campos em falta.

Com base no esquema do algoritmo apresentado no início, a função, designada por **collatz_sequencia**, terá de argumento **x**, o qual, sendo o primeiro termo, será logo inserido na lista, anteriormente, vazia.
Se o valor introduzido pelo utilizador for inferior a 1, o que a função retorna uma lista vazia, dado que a sequência é de números naturais com a exceção do 0.
Assim, começa-se a completar o corpo da função, colocando-se as seguintes linhas de código:




```
def collatz_sequencia(x):      # o argumento da função é o valor x
    seq=[x]                    # a sequência com o termo indicado pelo utilizador
    if x<1:                    # se x for inferior a 1
        return [ ]             # retorna lista vazia
```



IV. Se o valor introduzido for maior que 1 então, **enquanto** ele for maior que 1, o programa verifica se **x** é um número par e se o for, o seu valor passa para metade, caso contrário o seu valor é multiplicado por 3 e é somado 1.
Como este processo é executado uma certa condição acontece então, passando agora para linguagem *Python*, é necessário utilizar uma estrutura de controlo, **while...**, que pode escrever com o teclado ou obter no menu:

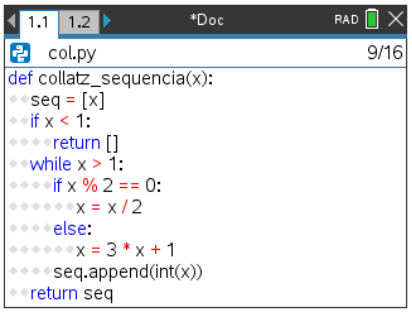
menu → 4 **Planos integrados** → 2 **Controlo** → 8 **while..**

Por fim, à medida que este processo é feito, os valores serão adicionados à lista **seq**.
Esta adição dos termos à lista, em linguagem *Python* é permitido através do comando **.append()**, que se pode escrever com o teclado ou obter no menu:

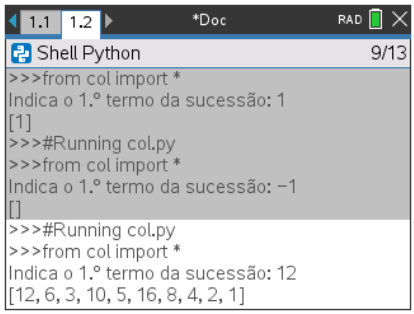
menu → 4 **Planos integrados** → 4 **Listas** → 6 **.append()**

Então, tem-se, assim, a função completa:

```
def collatz_sequencia(x):          # o argumento da função é o valor x
    seq=[x]                        # a sequência com o termo indicado pelo utilizador
    if x<1:                        # se x for inferior a 1
        return [ ]                 # retorna lista vazia
    while x>1:                     # enquanto x é maior que 1:
        if x%2==0:                 # se x é par:
            x=x/2                  # então x←x/2
        else:                      # caso contrário:
            x=3*x+1                # x←3x+1
        seq.append(int(x))         # acréscimo do termo à sequência seq
    return seq                     # retornar a sequência completa
```



V. Escrito o programa, falta executá-lo. Pode utilizar-se uma instrução do menu (menu 2 1), mas é claramente mais simples utilizar um atalho, uma combinação de teclas (ctrl + R).
O resultado aparece numa nova página destinada a mostrar o resultado da execução do programa, ***Shell Python***, na qual também e podem fazer operações e programas, mas que não permanecerão gravados após o fecho da aplicação.
Para voltar ao editor de *Python*, deve utilizar o touchpad, seja com o toque do dedo e o botão central ( ) depois de sobrepor o cursor ao número da página, ou com as teclas laterais, premindo previamente ctrl, para abrir a página anterior ou posterior.



Algumas ideias sobre programação, relacionadas com o contexto