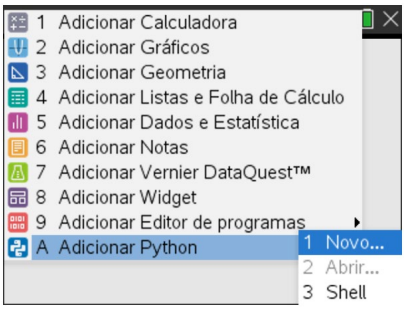
```
def bissecao(f,a,b,erro):
  fa = f(a)
  fb = f(b)
  if fa*fb > 0:
     print("A função tem imagens com o mesmo sinal nos extremos
     deste intervalo")
     exit(0)
  while b-a > 2*erro:
     c = (a+b)/2
     fc = f(c)
     if fa*fc < 0:
       b, fb = c, fc
     elif fa*fc > 0:
       a, fa = c, fc
     else:
       return c
     return c
def f(x):
  return x**3-5*x**2+2*x-3
print(bissecao(f,0,5,0.000001))
```

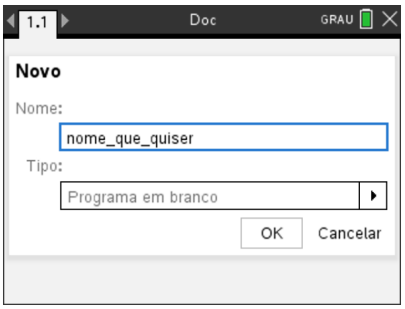
Editar Python na TI-nspire CX II-T

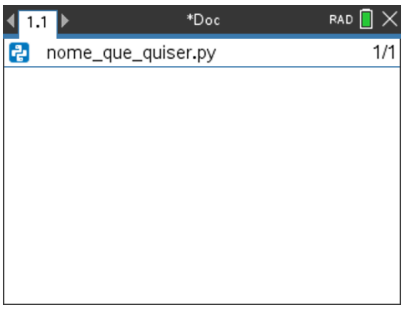
Ligue a sua calculadora e crie um novo documento.
Escolha uma página de *Python*:

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. Dada uma função, como calcular a aproximação de uma raiz num determinado intervalo através do método da Bisseção?

Aqui temos uma forma alternativa ao que foi apresentado em 3 - FUNC - RAE - 1 - p30, em que se evita o cálculo de `f` várias vezes nos mesmos pontos. Para tal usam-se variáveis para armazenar resultados intermédios. Vale a pena recordar que o método da bisseção é um método que é útil para encontrar um valor aproximado do zero de uma função com um erro máximo admissível, definido à partida, que funciona de forma fundamentada no Teorema de Bolzano-Cauchy, sendo necessário:

- a expressão algébrica da função (referenciada com `f`);
- o intervalo inicial que contenha o zero (designe-se por `[a,b]`);
- o erro máximo tolerado para a estimativa do zero.

O funcionamento deste método baseia-se na divisão do intervalo ao meio (o intervalo é bissetado) e escolher aquele onde o Teorema de Bolzano-Cauchy garanta a existência do zero. O processo repete-se quantas vezes for necessário até que a amplitude do intervale chegue a um valor inferior ao erro admissível **ou**, caso a estimativa seja feita com os sucessivos valores centrais do intervalo, até que a **amplitude seja inferior ao dobro do erro admissível**.

No exemplo do programa apresentado nas Aprendizagens Essenciais, a função **f** é $f(x) = x^3 - 5x^2 + 2x - 3$. Pretende-se encontrar uma estimativa do zero da função com um erro máximo de $0{,}000001$, efetuando bisseções sucessivas de intervalos de números reais, iniciando com o intervalo $[0,5]$.

Antes de se apresentar a construção do programa *Python*, com a tecnologia TI-Nspire CX II-T, observe-se uma ideia do método, que favorece a compreensão do algoritmo.

Em linguagem natural, pode escrever-se o seguinte:

Variáveis: a função `f`, os valores `a`, `b` e o erro máximo, `erro`

Início

Atribuição do valor `f(a)` à variável, nomeada de, `fa`

Atribuição do valor `f(b)` à variável, nomeada de, `fb`

Se `fa*fb>0`:

então escreve "A função tem imagens com o mesmo sinal nos extremos do intervalo" e termina imediatamente o programa

Caso contrário, enquanto o erro não for maior que o erro máximo indicado:

Definição do ponto médio `c=(a+b)/2`

Atribuição do valor `f(c)` à variável, nomeada de, `fc`

Se `fa*fc<0` :

Então, o extremo máximo de valor `b` passará a ser o valor `c` e, consequentemente, o valor `fb` passará a ser o valor `fc`

Já se `fa*fc>0` :

Então, o extremo máximo de valor `a` passará a ser o valor `c` e, consequentemente, o valor `fa` passará a ser o valor `fc`

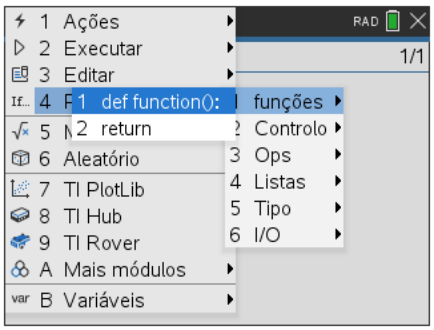
Caso contrário:

o valor pretendido é o `c` e devolve o valor final `c`

Devolve o valor `c`

Fim

I. De modo a possibilitar a aplicação do método da Bisseção a várias funções, em diferentes intervalos e com distintas precisões, define-se uma função denominada **bissecao**. Esta função recebe como argumentos a função a analisar, os extremos do intervalo considerado e o erro máximo admissível, devolvendo uma aproximação do zero da função nesse intervalo.



Esta "função" é semelhante ao que noutras linguagens de programação se chama de subrotina, 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. A ideia desta estrutura é oportuna num contexto de funções matemáticas e, por isso, surge aqui como exemplo de um passo mais à frente em relação à simples utilização de uma estrutura condicional.

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

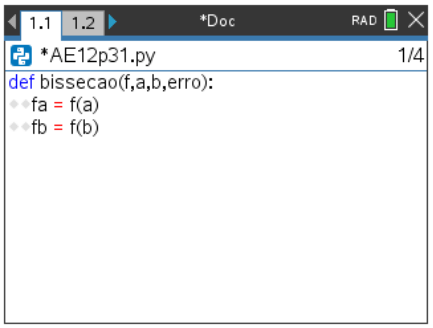
menu 4 **Planos integrados** → 1 **funções** → 1 **def function():**

Obtida a estrutura, é agora necessário preencher os campos em falta.
A função, chamada de **bissecao**, é uma função que recebe quatro argumentos:

```
def bissecao(f,a,b,erro):
    bloco
```

II. Como referido anteriormente, para evitar a "chamada" repetidamente da função **f**, é usado variáveis para armazenar resultados intermédios. Deste modo, é atribuído à variável **fa** o valor de **f(a)** e à variável **fb** valor de **f(b.**
Para tal, escreve-se as seguintes linhas de código:

```
    fa=f(a)
    fb=f(b)
```

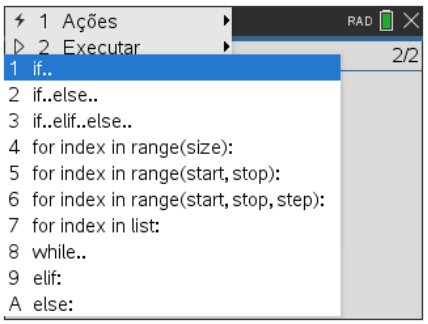


III. Perante o intervalo inicial $[a, b]$, se $f(a) \times f(b) < 0$, garante-se a existência do zero neste intervalo. Caso contrário, não se pode garantir tal existência.
Para colocar esta condição no intervalo inicial, em linguagem *Python*, é necessário utilizar uma estrutura condicional, **if**, que pode escrever com o teclado ou obter no menu:

menu → 4 **Planos integrados** → 2 **Controlo** → 1 **if..**

Deste modo, tem-se de acrescentar as seguintes linhas de código, dentro da função:

```
if f(a)*f(b)>0:
    bloco
```



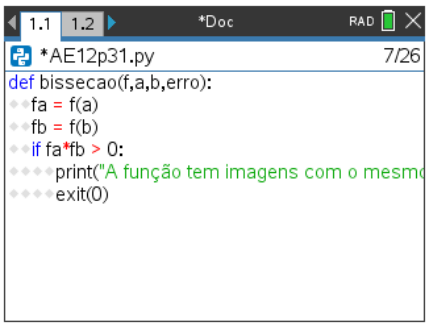
Se este último cenário acontecer, o programa irá parar e mostrará, na página Shell Python, uma mensagem acerca da razão de ter parado. Para tal é necessário colocar as instruções que permitem a observação ou a impressão da informação pretendida, através da função **print(),** a qual pode ser escrita diretamente no editor, com o teclado, ou obtida no menu:

menu → 4 **Planos integrados** → 5 **Tipo** → 6 **I/O** → 1 **print()**

E, posteriormente, coloca-se a mensagem, sempre entre aspas, dentro da função:

```
print("A função tem imagens com o mesmo sinal nos extremos deste
intervalo")
```

Dado o sucedido, o programa deve terminar imediatamente. Esta instrução é feita através do comando **exit(0)**, a qual deve ser inserida, com o teclado, logo após ser apresentada a mensagem.

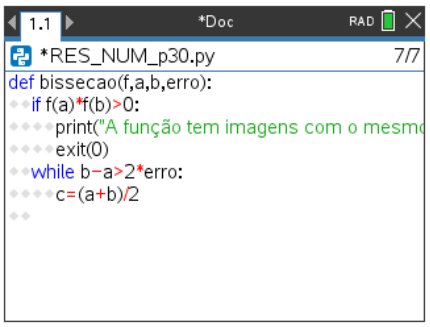


IV. Caso o intervalo inicial seja admissível para continuar o método, começa-se a bisseção do intervalo inicial, utilizando-se o valor central c do intervalo em questão. Para além disso, enquanto o erro máximo não for alcançado, o programa avança. Para este processo, em linguagem *Python*, é necessário utilizar uma estrutura de repetição do tipo, **while…**, que pode escrever com o teclado ou obter no menu:

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

De seguida, coloca-se a seguinte linha de código:

```
while b-a>2*erro:
    bloco
```



A cada iteração, o intervalo é bissetado pelo respetivo valor médio c, que faz com que surja dois subintervalos de igual amplitude.

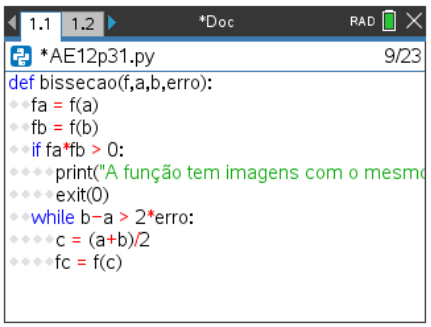
Para tal, dentro da estrutura de controlo, insere-se a seguinte linha de código:

```
c=(a+b)/2
```

Para além disso, é atribuído à variável `fc` o valor de `f(c)` .

Assim, escreve-se, de seguida, a seguinte linha de código:

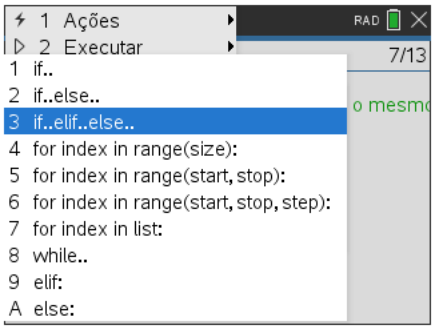
```
    fc=f(c)
```



V. Como do intervalo inicial, surgiram dois subintervalos: $[a, c]$ e $[c, b]$, em que é necessário identificar em qual ocorre mudança de sinal, pois é nesse subintervalo que se encontra o zero.

Deste modo:

- **se** $f(a) \times f(c) < 0$, então o zero pertence ao intervalo $[a, c]$ e este passa a ser o novo intervalo onde se vai avaliar a existência de zero na iteração seguinte.
  Assim, o extremo máximo, b, passa a ser c.
  Nesse caso, a variável b passa a valer o valor de c e, consequentemente, a variável fb passa a valer fc.
- **já se** $f(a) \times f(c) > 0$, então o zero pertence ao intervalo $[c, b]$ e este passa a ser o novo intervalo onde se vai avaliar a existência de raiz na iteração seguinte.
  Assim, o extremo mínimo, a, passa a ser c.
  Nesse caso, a variável a passa a valer o valor de c e, consequentemente, a variável fa passa a valer fc.
- **caso contrário**, ou seja, $f(a) \times f(c) = 0$, então $f(c) \neq 0$ e, portanto, o valor central c é exatamente um zero da função.

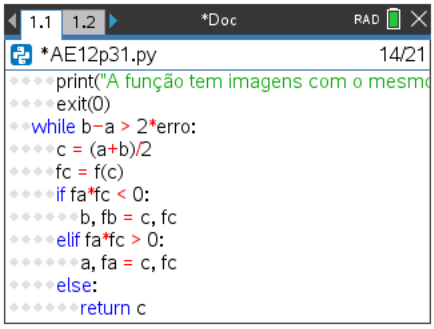


Esta linha de condições, pode ser escrita, utilizando a estrutura condicional, **if..elif..else..**, que pode escrever com o teclado ou obter no menu:

menu → 4 **Planos integrados** → 2 **Controlo** → 1 **if..**

Deste modo, tem-se de acrescentar as seguintes linhas de código, dentro, também, da estrutura de condição:

```
if fa*fc<0:
    b, fb = c, fc
elif fa*fc>0:
    a, fa = c, fc
else:
    return c
```



Nota:

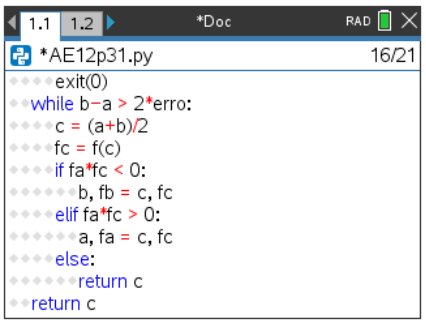
Quando se escreve a linha de código:

```
b, fb = c, fc
```

está-se a fazer uma **dupla atribuição**. Isto é, faz-se duas atribuições ao mesmo tempo: a variável **b** passa a valer **c** e a variável **fb** passa a valer **fc.** Ao invés de b=c e fb=fc, escreve-se tudo em linha.

VI. Quando o ciclo **while..** termina, isto é, quando a amplitude do intervalo já é suficientemente pequena para garantir que o erro máximo seja inferior ao erro pretendido, o valor devolvido é o último valor central calculado. Esse valor central será o zero se nenhuma das condições anteriores se veerificar, pelo que é necessário incluir uma última instrução para devolver o valor c, que deverá aparecer fora da estrutura while, esta com o resultado da execução da função:

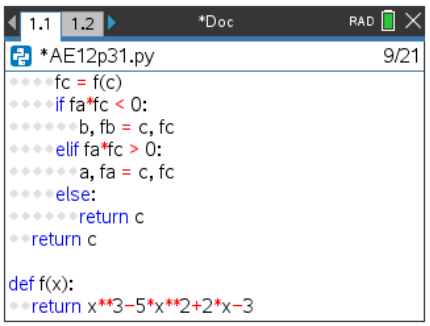
return c



VII. Depois de definida a função bissecao, define-se a função matemática que se pretende estudar. A função em questão é $f(x) = x^3 - 5x^2 + 2x - 3$.
Assim, de modo análogo ao que foi feito anteriormente, deve-se escrever as seguintes linhas de código:

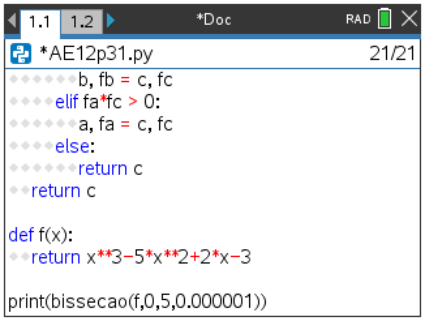
def f(x):

return x3−5*x**2+2*x−3**

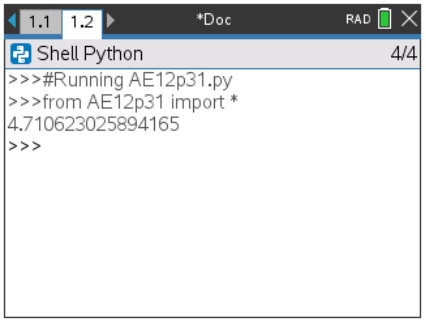


VIII. Para se concluir o programa é necessário colocar as instruções que permitem a observação ou a impressão da informação pretendida, na página Shell Python, que se abre automaticamente logo após a execução do programa.
A linha de código, a colocar agora, é a que se segue:

print(bissecao(f,0,5,0.00001))



IX. 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*, onde poderá alterar os dados de entrada, por exemplo, há mais do que um procedimento à escolha, baseados no botão do touchpad. Pode fazer deslocar o cursor com o dedo até o sobrepor ao retângulo com a designação da página, 1.1, neste caso, e premir o touchpad na parte central (). Pode também utilizar os botões laterais do touchpad após premir a tecla ctrl. Neste caso, ao premir o botão lateral esquerdo, vai para a página anterior, a do editor. Pode voltar à página de *Shell Python* utilizando o mesmo tipo de procedimento.
Na parte superior do ecrã apenas se pode observar a designação e 3 páginas consecutivas, pelo que se o documento tiver mais páginas terá de conjugar os dois procedimentos referidos ou simplesmente o que recorre às teclas laterais do touchpad.

Algumas ideias sobre programação, relacionadas com o contexto