Iniciação ao Menu Python - Casio fx-CG50

1. O Pensamento Computacional nas AE de Matemática do Ensino Secundário

"Os aspetos comuns entre o Pensamento Matemático e o Pensamento Computacional, bem como a relevância atual do Pensamento Computacional na ciência e na sociedade, justificam que o currículo de Matemática valorize esta abordagem conceptual na resolução de problemas. As Aprendizagens Essenciais de Matemática A promovem o desenvolvimento de práticas como a abstração, a decomposição, o reconhecimento de padrões, a análise e definição de algoritmos, bem como a aquisição de hábitos de depuração e otimização dos processos envolvidos na atividade matemática. Deste modo, a aposta no Pensamento Computacional revela a aproximação do currículo às recomendações internacionais, designadamente em relatórios da União Europeia, e também o alinhamento com o currículo de Matemática do Ensino Básico, favorecendo o desenvolvimento desta capacidade de forma integrada, coerente e progressiva."

(Fonte: DGE. Disponível em https://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/mat_a_10_-_vf.pdf)

Abstração - Extrair a informação essencial de um problema.

Decomposição - Estruturar a resolução de problemas por etapas de menor complexidade de modo a reduzir a dificuldade do problema.

Reconhecimento de padrões - Reconhecer ou identificar padrões e regularidades no processo de resolução de problemas e aplicá-los em outros problemas semelhantes.

Algoritmia - Desenvolver um procedimento (algoritmo) passo a passo para solucionar o problema, nomeadamente recorrendo à tecnologia.

Depuração - Procurar e corrigir erros, testar, refinar e otimizar uma dada resolução.

(Fonte: DGE. Disponível em https://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/3_ciclo/ae_mat_7.o_ano.pdf)

2. Uma linguagem de programação - PYTHON

"As atividades de programação devem ser integradas com uma complexidade progressiva, sendo relevantes para o desenvolvimento de processos algorítmicos, de um pensamento estruturado e do raciocínio lógico, proporcionando um vasto campo de aplicação da Matemática e envolvendo genuinamente a formulação e a resolução de problemas, além de promover o desenvolvimento do pensamento computacional."

(Fonte: DGE. Disponível em https://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/mat_a_10_-_vf.pdf)

"O Python é uma linguagem de programação de alto nível, interpretada, para programação de âmbito geral. Criada por Guido van Rossum e inicialmente lançada em 1991, o Python tem uma filosofia de design que dá ênfase à legibilidade do código, notavelmente usando espaços em branco com significado. O Python possui construções que permitem uma programação clara em pequena e grande escala.

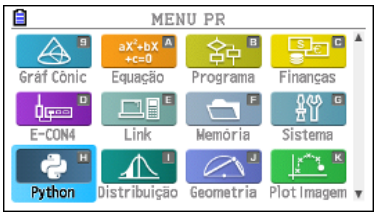
O Python deixa-o trabalhar rapidamente e integrar sistemas de forma mais eficiente.

O Python é utilizado por milhares de pessoas em todo o mundo."

(Fonte: Python Portugal. Disponível em https://python.pt/acerca/)

3. A calculadora gráfica Casio fx-CG50 e o seu emulador

3.1. Menu Python

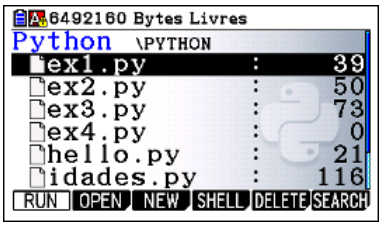


O modo Python oferece um ambiente de tempo de execução para a linguagem de programação Python. Permite criar, salvar, editar e executar arquivos Python.

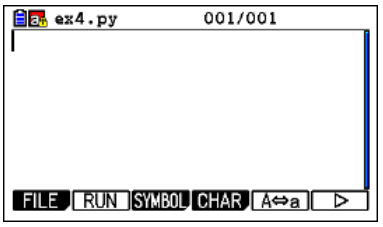
O modo Python é compatível com a versão 1.9.4 do MicroPython, que foi adaptada para ser executada nesta calculadora. Em termos gerais, o MicroPython é diferente do Python executado em computadores. Além disso, o modo Python não é compatível com todas as funções, comandos, módulos e bibliotecas do MicroPython.

O ambiente do menu Python é constituído por três áreas:

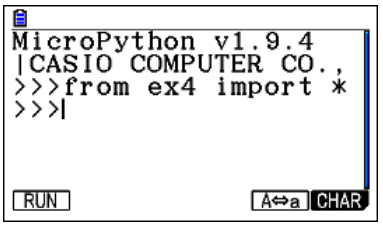
- a Lista de arquivos, em que são exibidos os nomes dos programas com a extensão *.py;
- o Editor, onde se escreve o script (sequência de instruções executadas pelo programa);
- o Shell, onde se podem escrever expressões em linguagem Python e executar scripts.



Lista de arquivos



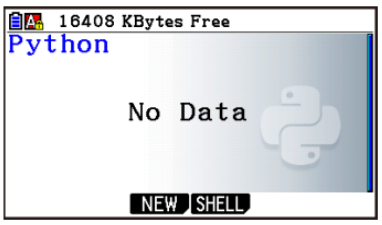
Editor



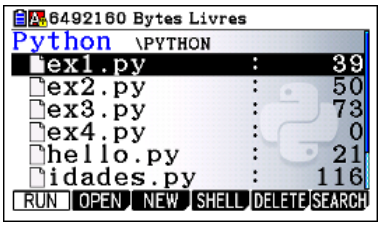
Shell

3.2. Lista de arquivos

O primeiro elemento exibido ao selecionar o modo Python no menu principal é o ecrã da lista de arquivos.

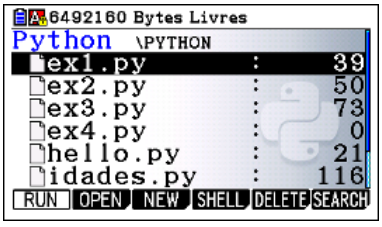


Quando não há arquivo *.py (nem pastas)



Quando não há arquivo *.py (ou pastas)

As opções disponíveis na lista de arquivo são:

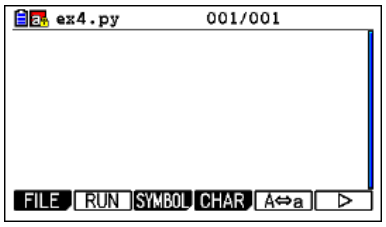


RUN	Executar um arquivo presente na lista
OPEN	Abrir um arquivo
NEW	Criar um novo arquivo
SHEL	Aceder ao Shell
DELETE	Apagar um arquivo
SEARCH	Pesquisar um arquivo na lista

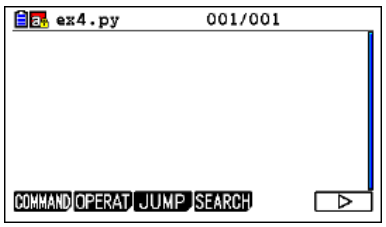
3.3. Editor

Um programa é um arquivo que contém um algoritmo (sequência de instruções), codificado usando a linguagem de programação Python de modo a poder ser executado pela calculadora. O Editor é o ambiente onde se escreve o guião do programa (script).

As opções disponíveis no Editor são:

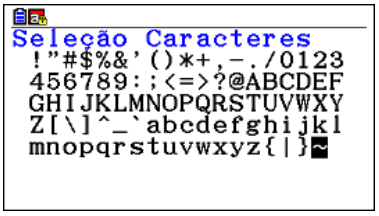


FILE	Guardar o arquivo
RUN	Executar o script no Shell
SYMBOL	Exibir símbolos: , () [] : ; # ’ ” \ _
CHAR	Exibir símbolos, operadores, números e letras
A⇔ a	Alternar entre maiúsculas e minúsculas
▷	Ver mais funções

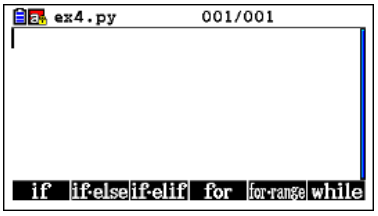


COMMAND	Abrir estruturas de condição e de repetição
OPERAT	Exibir operadores: = != > < % | ^ & ~
JUMP	Colocar o cursor numa linha específica
SEARCH	Procurar texto
▷	Ver mais funções

As opções dos separadores “CHAR” e “COMMAND” são:



CHAR

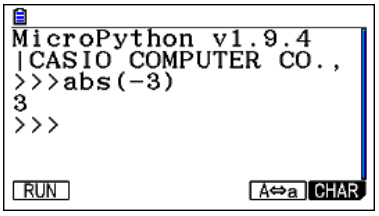


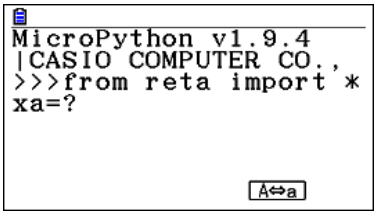
COMMAND

3.4. Shell

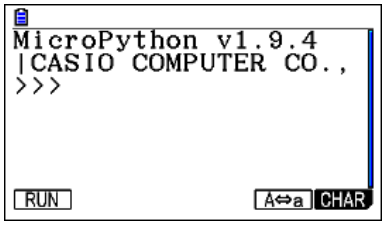
O Shell pode ser usado de duas formas:

- para inserir diretamente expressões e comandos e obter os seus resultados, usando a sintaxe da linguagem Python;
- para executar scripts.





As opções disponíveis no Shell são:

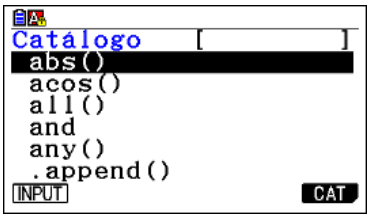


RUN	
A⇔ a	Alternar entre maiúsculas e minúsculas
CHAR	Exibir símbolos, operadores, números e letras

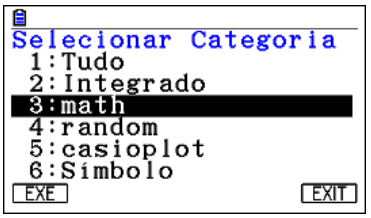
3.5. Inserir texto e comandos

Existem três formas de inserir texto e comandos no menu Python:

- usando o teclado da calculadora/emulador (pressione [ALPHA] para inserir um caractere alfabético assinalado a vermelho no teclado ou [SHIFT] [ALPHA] para permanecer em modo alfabético);
- usando as funções do Editor (SYMBOL, CHAR, COMMAND, etc.);
- usando o catálogo ([SHIFT] [4]).



Catálogo



Categorias do catálogo

Os operadores aritméticos da linguagem Python são:

Operação	Tecla	Descrição
$x + y$	[+]	Adição
$x - y$	[−]	Subtração
$x * y$	[×]	Multiplicação
x / y	[÷]	Divisão
$- x$	[−]	Simétrico de x
$x ** n$	[^] ou [×] [×]	Potenciação (x elevado a n)
$x // y$	[÷] [÷]	Divisão inteira de x por n
$x \% y$	[F4] (CHAR)	Resto da divisão inteira de x por y

Outras expressões, tais como π e $\sqrt{\ }$, requerem a importação do módulo “math” (from math import $*$), disponível no catálogo.

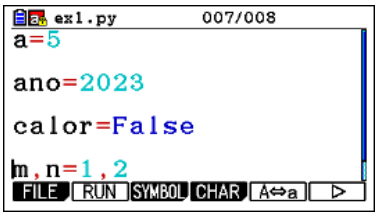
3.6. Variáveis e dados

➟ O Python pode manipular informações de diferentes tipos:

- inteiro (int) – número sem parte decimal;
- real (float) – número com parte decimal;
- lógico (bool) que apenas pode assumir dois valores, True (verdadeiro) e False (falso);
- cadeia de caracteres (str) – sequência de caracteres entre aspas;
- lista (list) – sequência de elementos.

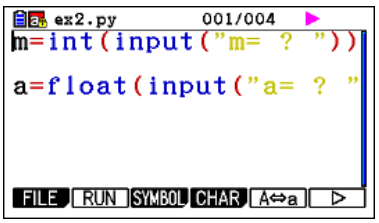
As variáveis são utilizadas para armazenar valores em memória.

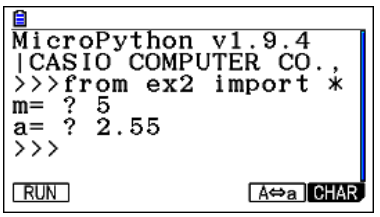
Para atribuir um valor a uma variável, utiliza-se o sinal igual (=).



➟ A entrada de dados (do Shell para o Editor) efetua-se com a função *input()*. Esta função importa dados do tipo cadeia de caracteres (*str*) que podem ser convertidos em números (*int* ou *float*).

Exemplo:

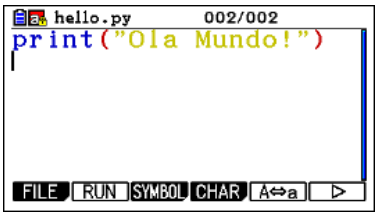


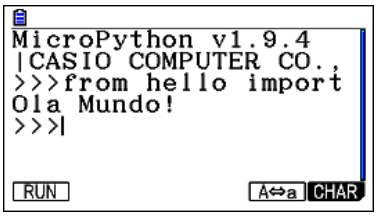


- Pedir um número inteiro (*int*) que será atribuído à variável *m*;
- Pedir um número real (*float*) que será atribuído à variável *a*.

➟ A saída de dados (do Editor para o Shell) efetua-se com a função *print()*.

Exemplo:





3.7. Funções

Uma *função* é uma estrutura que agrupa um conjunto de instruções.

Para definir uma função, usa-se a palavra reservada *def*.

A sintaxe das funções é definida por três partes:

- o nome (separado da palavra *def* por um espaço);
- o(s) argumento(s) (escritos entre parênteses e separados por vírgulas);
- o corpo da função (conjunto de instruções).

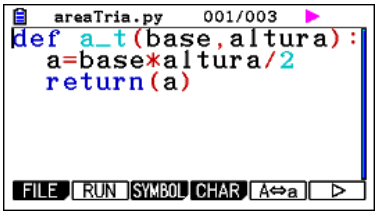
```
def nome(argumento1,argumento2,...):
    corpo da função
```
indentação

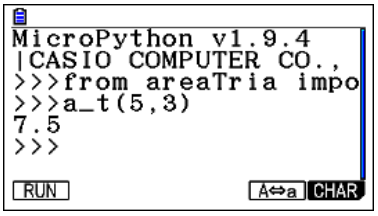
A utilização dos dois pontos (:) no final da primeira linha faz com que o corpo da função fique indentado (com um avanço à esquerda), nas linhas seguintes.

A indentação (ou paragrafação) é obrigatória em Python.

No corpo da função, pode ser utilizada a instrução *return()* para que a função devolva um valor.

Exemplo (Área do triângulo):





As funções servem essencialmente para criar procedimentos que são executados várias vezes ao longo do programa (a função pode ser invocada em qualquer parte do script, evitando assim a repetição de todas as linhas do seu corpo).

3.8. Estruturas de condição

Uma expressão cujo valor é do tipo lógico chama-se uma condição. As condições são criadas com operações relacionais que permitem comparar dois valores.

Os operadores relacionais da linguagem Python são:

Operação	Tecla	Descrição
$x == y$	SHIFT • SHIFT •	Igual
$x != y$	F4 (CHAR)	Diferente
$x > y$	F4 (CHAR)	Maior que
$x < y$	F4 (CHAR)	Menor que
$x >= y$	F4 (CHAR)	Maior ou igual que
$x <= y$	F4 (CHAR)	Menor ou igual que

Os operadores relacionais também estão disponíveis pressionando F6 (▷) F2 (OPERAT).

Nota: Não confundir “x = 1” (o valor 1 é atribuído à variável x) com “x == 1” (condição verdadeira se x for igual a 1 e falsa se x for diferente de 1).

As condições podem ser combinadas através das seguintes operações lógicas:

Operação	Descrição
$p\ and\ q$	Conjunção (e)
$p\ or\ q$	Disjunção (ou)
$not\ p$	Negação (não)

Os operadores lógicos estão disponíveis no catálogo (SHIFT 4).

➞ IF - ELSE

A estrutura condicional *if – else* permite a seleção entre duas alternativas.

A sintaxe é a seguinte:

```
if condição :
    instruções 1
else :
    instruções 2
```

A condição começa por ser avaliada: se o seu valor lógico for True, são executas apenas as instruções 1; se o valor lógico for False, são executas apenas as instruções 2.

A utilização dos dois pontos (:) no final da primeira linha faz com que as instruções fiquem indentadas, com dois espaços para a direita.

➞ IF - ELIF - ELSE

A estrutura condicional *if – elif - else* permite a seleção entre três ou mais alternativas.

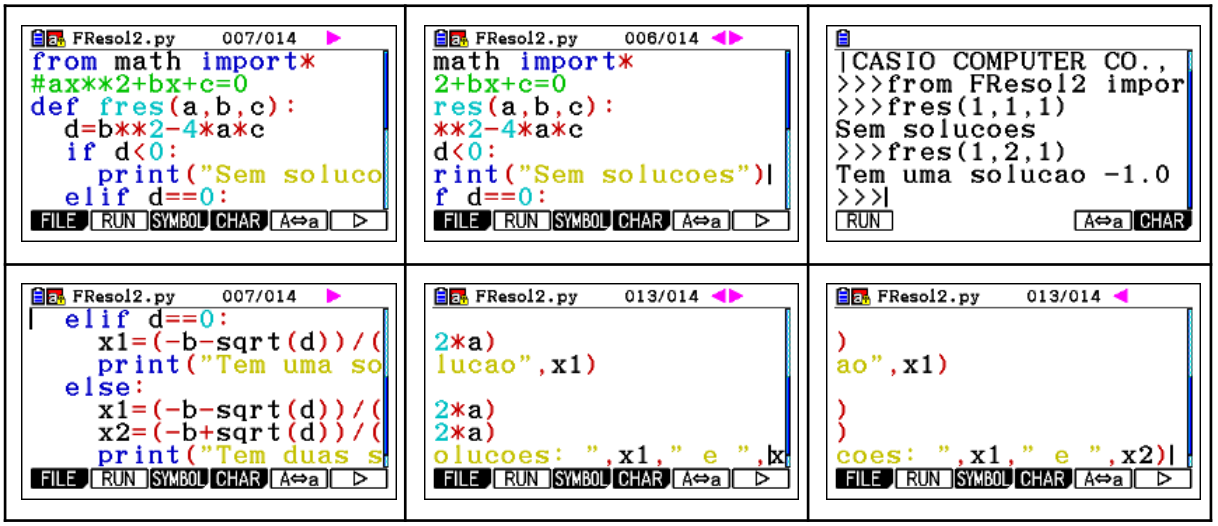
A sintaxe é a seguinte:

```
if condição :
    instruções 1
elif condição 2 :
    instruções 2
else :
    instruções 3
```

Exemplo (Fórmula resolvente de equações do 2.º grau):

Nota: É possível inserir comentários num script, após o símbolo # (disponível no separador CHAR). Um comentário é uma frase em linguagem natural que aumenta a facilidade de leitura das instruções.

Os comentários (exibidos numa cor diferente) são ignorados na execução do programa.



3.9. Estrutura de repetição (FOR i IN RANGE)

Em programação, uma sequência de instruções executada repetitivamente é um ciclo.

Um ciclo é constituído por uma instrução inicial, que controla a sua execução, e por um conjunto de instruções designado por corpo do ciclo.

A instrução for *i in range* permite repetir um bloco de instruções, um número predeterminado de vezes.

A sintaxe é a seguinte:

```
for i in range(a,b) :
    corpo do ciclo
```

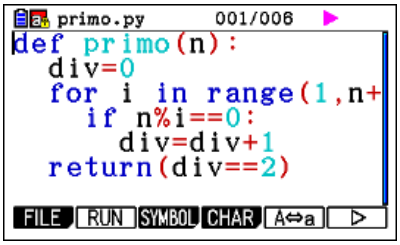
A utilização dos dois pontos (:) no final da primeira linha faz com que as instruções do corpo do ciclo for fiquem indentadas (com um avanço à esquerda).

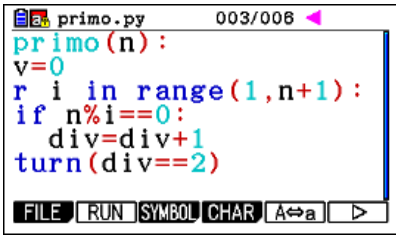
Nesta estrutura, a variável i percorre um conjunto de valores especificado por (a,b):

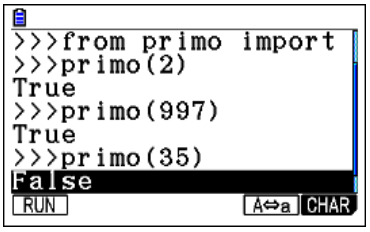
range(a,b) designa os números inteiros *i* tais que $a \leq i < b$.

Exemplo (Números primos):

"Um número n é primo se tiver dois e somente dois divisores"







3.10. Mensagens de erro

Se um programa não for executado da forma esperada, a causa pode ser um erro (bug) no script.

Neste caso, é exibida uma mensagem de erro no Shell, em vermelho, com indicação da linha onde está o erro e do tipo de erro (por exemplo, um erro de sintaxe).

Use o cursor (◀ ▶) para consultar os detalhes da mensagem de erro.

Pressione EXIT regressar ao Editor e corrigir o erro.