



Unidade 5: Utilização da biblioteca TI System

Lição 3: Exibição e temporização

Nesta terceira lição da unidade 5, poderá aprender como utilizar as opções de exibição e de temporização da biblioteca **TI System** do Python.

Objetivos:

- Compreender o funcionamento das instruções **disp...**
- Utilizar estas instruções num programa, para além dos outros módulos.

Nas primeiras duas lições desta unidade aprendeu como importar/exportar listas de dados e a trabalhar com uma equação de regressão.

A biblioteca TI System tem outras opções eventualmente úteis, que nos propomos descobrir aqui.

1: A instrução clear_history

- Iniciar um novo programa com o nome U5L3.
- Importar o módulo **TI System**.
- Escrever um programa que defina uma tabela de uma função.

$$x \rightarrow 3x^2 - \frac{1}{x+1} + 2$$

- Lembrar o programa U5L3 e executar. Deverá obter o ecrã ao lado.

```

1 from ti_system import *
2 recall_value("name")
3 store_value("name", value)
4 recall_list("name")
5 store_list("name", list)
6 eval_function("name", value)
7 get_platform()
8 get_key()
9 get_mouse()
A while get_key() != "esc":

```

```

*U5L3.py
from ti_system import *
x=[]
y=[]
def f(n):
    return 3*n**2-1/(n+1)+2
def tab(n):
    for i in range(n):
        x.append(i)
        y.append(round(f(i),1))

```

```

Shell Python
>>>#Running U5L3.py
>>>from U5L3 import *
>>>f(3)
28.75
>>>tab(5)
>>>x
[0, 1, 2, 3, 4]
>>>y
[1.0, 4.5, 13.7, 28.8, 49.8]
>>>|

```





A instrução `clear_history` limpa a tela e coloca o cursor para início no topo do ecrã. Obtém assim uma nova tela, sem os resultados de execuções anteriores.

```

1.1 1.2 1.3 *Documento3 RAD 2/10
*U5L3.py
from ti_system import *
clear_history()
x=[]
y=[]
def f(n):
    return 3*n**2-1/(n+1)+2
def tab(n):
    for i in range(n):
        x.append(i)
        y.append(round(f(i),1))

```

Assim, a execução do programa torna-se mais legível.

```

1.1 1.2 1.3 *Documento3 RAD 9/9
Shell Python
>>>f(5)
76.83333333333334
>>>tab(10)
>>>x
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>>y
[1.0, 4.5, 13.7, 28.8, 49.8, 76.8, 109.9, 148.9, 193.9, 244.9]
>>>

```

2: A instrução `eval_function()`

- Inserir uma página de Calculadora e definir a função:

$$g: x \rightarrow \sin(x)$$

- A instrução `eval_function()` permitirá a utilização num programa em Python de uma instrução previamente definida noutra aplicação da TI-Nspire™ (Calculadora, Gráficos, Listas e Folha de Cálculo...).

```

1.1 1.2 1.3 *Documento3 RAD Efectuado
g(x):=sin(x)

```

Retome o programa e modifique-o como no ecrã para poder tabelar a função g : e depois representar graficamente a nuvem de pontos correspondente ao intervalo $[0 ; 2\pi]$.

A instrução `eval_function` encontra-se no menu **TI System**, mas atenção à sintaxe.

`eval_function("name",value)`: o argumento name será aqui **g** e não $g(x)$

```

1 from ti_system import *
2 recall_value("name")
3 store_value("name",value)
4 recall_list("name")
5 store_list("name",list)
6 eval_function("name",value)
7 get_platform()
8 get_key()
9 get_mouse()
A while get_key() != "esc":

```



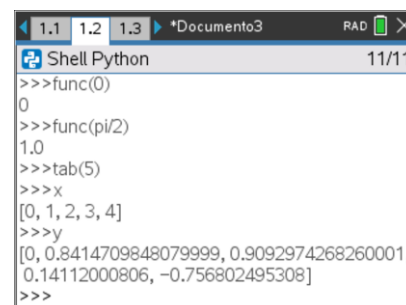
- Completar e modificar o programa anterior.



```

1.1 1.2 1.3 *Documento3 RAD
U5L2.py 8/21
from ti_system import *
from math import *
import ti_plotlib as plt
clear_history()
x=[]
y=[]
def func(n):
    return eval_function("g",n)
def tab(n):
    for i in range(n):
        x.append(i)
        y.append(func(i))
    
```

- Executar o programa. Atenção, a unidade é, por defeito, o radiano.
- Veja as opções da biblioteca Maths para uma expressão da medida dos ângulos em graus ou uma conversão. (Unidade 1, Lição 1).



```

1.1 1.2 1.3 *Documento3 RAD
Shell Python 11/11
>>>func(0)
0
>>>func(pi/2)
1.0
>>>tab(5)
>>>x
[0, 1, 2, 3, 4]
>>>y
[0, 0.8414709848079999, 0.9092974268260001,
0.14112000806, -0.756802495308]
>>>
    
```

- Modificar novamente o programa para obter as coordenadas de n pontos de abcissa no intervalo $[0 ; 2\pi]$. Pode visitar aqui o que aprendeu na unidade 4, lição 2.

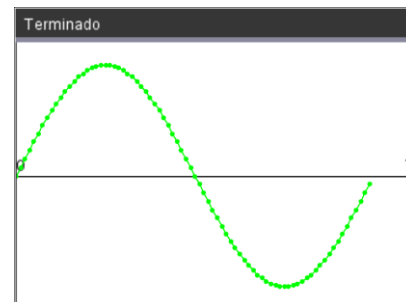


```

1.1 1.2 1.3 *Documento3 RAD
U5L2.py 21/21
for i in range(n):
    x.append(i*2*pi/n)
    y.append(func(i*2*pi/n))
# Representação gráfica
def graf():
    plt.cls()
    plt.window(0,7,-1.2,1.2)
    plt.axes("on")
    plt.color(0,255,0)
    plt.plot(x,y,"o")
    
```

- Terminar o programa, incluindo uma função **graf()**.

- Verifique o funcionamento do programa.



NOTA:

A instrução **recall_value("name")** permitirá também colocar numa variável de um programa em Python o conteúdo de uma variável utilizada noutra aplicação da TI-Nspire™.