



Unité 5 : Utiliser la librairie ti_system

Compétence 3 : Affichage et temporisation

Dans cette troisième leçon de l'unité 5, vous allez découvrir comment utiliser les options d'affichage et de « temporisation » de la librairie Python **TI System**.

Objectifs :

- Comprendre le fonctionnement des instructions **disp...**
- Utiliser ces instructions dans un script en complément de celles des autres modules.

Lors des leçons 1 et 2 de cette unité, vous avez appris à importer-exporter des listes de données puis à travailler sur une équation de régression.

La librairie TI System comporte d'autres options parfois utiles que l'on se propose, que l'on se propose de découvrir ici.

1 : L'instruction clear_history

- Commencer un nouveau script et le nommer U5SB3.
- Importer le module **TI System**.

```

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":

```

- Écrire un script permettant de tabuler une fonction.

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

```

U5SB3.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))

```

- Rappeler votre script U5SB3 et demander son exécution. Vous devriez obtenir l'écran ci-contre.

```

Shell Python
>>>#Running U5SB3.py
>>>from U5SB3 import *
>>>f(3)
28.75
>>>tab(5)
>>>x
[0, 1, 2, 3, 4]
>>>y
[1.0, 4.5, 13.7, 28.8, 49.8]
>>>

```



10 Minutes de Code

TI - NSPIRE™ CX II & TI - PYTHON

L'instruction `clear_history` nettoie l'écran et place le prompt de la console en haut de l'écran. Vous obtenez ainsi au sein d'un programme un écran débarrassé de ses précédents affichages.



```
1.1 1.2 1.3 *Classeur RAD 1/10
*U5SB3.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))
```

Ainsi l'exécution du script est rendue plus lisible.

```
1.1 1.2 1.3 *Classeur 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 : L'instruction `eval_function()`

- Insérer une page de calculs `ctrl` `I` et définir la fonction :

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

- L'instruction `eval_function()` va permettre d'utiliser dans un script Python, une instruction préalablement définie au sein d'une autre application de la TI-Nspire™ (Calculs, Graphiques, Tableur&Listes...).

```
1.1 1.2 1.3 *Classeur RAD Terminé
g(x):=sin(x)
```

Reprendre le script et le modifier comme sur l'écran ci-contre de façon à pouvoir tabuler la fonction g : puis de représenter graphiquement le nuage de points correspondant dans l'intervalle $[0 ; 2\pi]$.

L'instruction `eval_function` se trouve dans le menu **TI System**, mais attention à sa syntaxe.

`eval_function(« name »,value)` : l'argument name sera ici **g** et non pas $g(x)$

```
1 Actions 3/12
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":
B clear_history()
C get_time_ms()
```





10 Minutes de Code

TI - NSPIRE™ CX II & TI - PYTHON

UNITE 5 : COMPETENCE 3

NOTES DU PROFESSEUR

- Compléter en modifiant le script précédent.

```

1.1 1.2 1.3 *Classeur RAD 1/15
U5SB3.py
from ti_system import *
from math import *
import tiplotlib as plt
clear_history()
x=[]
y=[]
def fonction(n):
    return eval_function("g",n)

def tab(n):
    for i in range(n):
        x.append(i)
        y.append(fonction(i))

```

- Exécuter le script. Attention, l'unité par défaut est le radian.
- Voir les options de la librairie Maths pour une expression des mesures des angles en degrés ou une conversion. (Unité 1, compétence 1).

```

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

```

- Modifier à nouveau le script afin de calculer les coordonnées de n points dans l'intervalle $[0 ; 2\pi]$. Vous pouvez réinvestir ici ce que vous avez appris à l'unité 4, compétence 2.

```

1.1 1.2 1.3 *Classeur RAD 17/23
*U5SB3.py
def tab(n):
    for i in range(n):
        x.append(i*2*pi/n)
        y.append(fonction(i*2*pi/n))
# Représentation graphique
def graphe():
    plt.cls()
    plt.window(0,7,-1.2,1.2)
    plt.axes("on")
    plt.color(0,255,0)
    plt.plot(x,y,"o")

```

- Terminer le script en incluant une fonction **graphe()**.

- Vérifier le fonctionnement du script.

Remarque : L'instruction **recall_value(« name »)** permettra de la même manière de réinvestir dans une variable d'un script Python, le contenu d'une variable utilisée dans une autre application de la TI-Nspire™.

