



Unité 5 : Utiliser la bibliothèque TI System

Compétence 1 : Travailler sur des données

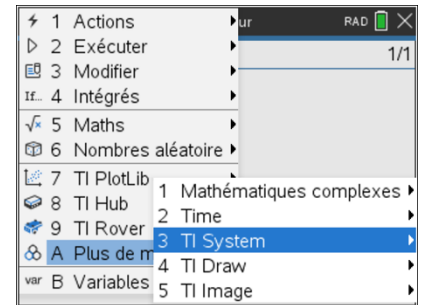
Dans cette première leçon de l'unité 2, vous allez découvrir comment utiliser la bibliothèque **Ti System** pour importer ou exporter des listes dans un script Python.

Objectifs :

- Importer-exporter des listes.
- Réinvestir les notions de l'unité 4 sur les représentations graphiques.

La bibliothèque ou module **Ti System** utilisée seule ou en complément des autres, permet de communiquer (dans les deux sens) avec la calculatrice graphique.

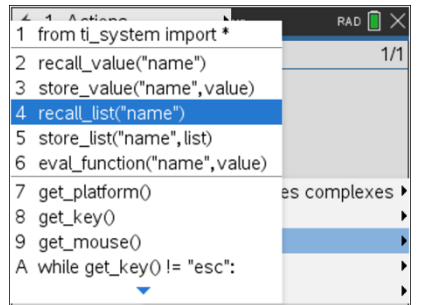
Pour charger cette bibliothèque, appuyer sur la touche **menu** puis **A Plus de modules** et enfin **3 TI System**.



Dans cette leçon, nous allons concentrer notre attention sur l'utilisation des instructions :

4 : var=recall_list(« nom ») et **5 : store_list(« nom »,var)**

Les autres options de cette bibliothèque seront abordées dans les autres leçons de cette unité.

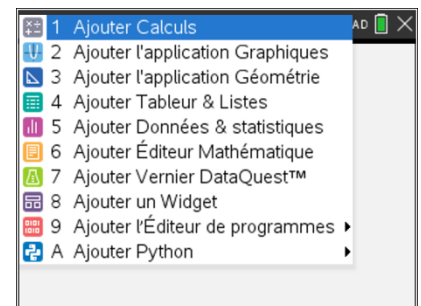


1 : Importer des données depuis la calculatrice.

- a) Création de deux listes.

Dans un premier temps nous allons simplement créer deux listes contenant les données.

- Créer une nouvelle page de calculs **ctrl** **I** puis choisir **1 Ajouter Calculs**.
- Le même travail est réalisable à partir de l'application **Tableur&Listes**.





10 Minutes de Code

TI - NSPIRE™ CX II & TI - PYTHON

UNITE 5 : COMPETENCE 1

NOTES DU PROFESSEUR

Vous allez créer dans la liste I_1 une suite de nombre entre 0 et 12 avec un pas de 0.2 .

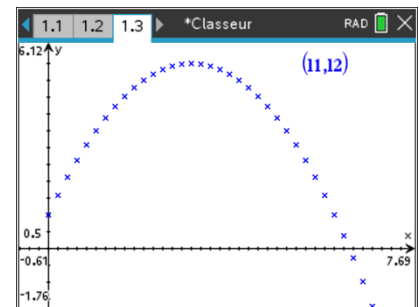
Puis une liste I_2 correspondant à la liste des images de x (parcourant la liste I_1 par la fonction f définie par : $x \mapsto -x^2/2 + 3x + 1$

```

1.1 1.2 1.3 *Classeur RAD
I1:=seq(x,x,0,12,0.2)
{0,0.2,0.4,0.6,0.8,1,1.2,1.4,1.6,1.8,2,2.2,2.4}
I2:=seq(-x^2/2+3*x+1,x,0,12,0.2)
{1,1.58,2.12,2.62,3.08,3.5,3.88,4.22,4.52,4.78}

```

- Insérer une nouvelle page « **Application graphique** ».
- Représenter graphiquement le nuage de points (I_1, I_2).
- Régler les paramètres de la fenêtre graphique tels que : $X_{min} = -1.2$; $X_{max} = 8$; $Y_{min} = -0.5$ et enfin $Y_{max} = 6.5$.



- b) Import des données dans un script Python.
- Commencer un nouveau script et le nommer U5SB1.
 - A partir de **menu** importer la bibliothèque **TI System**.
 - Créer deux variables listes (vides) **absc** et **ordo**.
 - Importer les bibliothèques **TI System** et **TI PlotLib** (il n'y a pas d'ordre particulier à respecter).
 - Créer une variable **absc** puis, à partir des options de la bibliothèque **TI System**, choisir l'option **4** : **var=recall_list**(« **nom** »). Comme les abscisses sont dans la liste I_1 , le champ « nom » est complété par le nom complet de la liste.
 - Créer une autre variable **ordo** et procéder de la même manière avec la liste I_2 .

```

1.1 1.2 1.3 *Classeur RAD 2/6
U5SB1.py
from ti_system import *
import ti_plotlib as plt
absc=[]
ordo=[]
absc=recall_list("I1")
ordo=recall_list("I2")

```

Conseil à l'enseignant : La création de listes vides `absc=[]` et `ordo=[]` n'est pas indispensable, car elles seront créés lors du rappel des listes I_1 et I_2 . Cependant il est préférable de conserver les bonnes habitudes apprises lorsque nous ne faisons pas appel au module **TI System** qui nécessitent cette création préalable.





- Exécuter le script, puis vérifier le contenu de vos variables **absc** et **ordo** en appuyant sur la touche `var`.
- Rappeler ensuite le nom de la « liste » **absc**, puis valider `enter`.
- Procéder de la même façon avec la liste des ordonnées (**ordo**).

```

1.1 1.2 1.3 *Classeur RAD 12/12
Shell Python
>>>from U5SB1 import *
>>>absc
[0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4, 2.6, 2.8, 3.0, 3.2, 3.4, 3.6, 3.8, 4.0, 4.2, 4.4, 4.6, 4.8, 5.0, 5.2, 5.4, 5.6, 5.8, 6.0, 6.2, 6.4, 6.6, 6.8, 7.0, 7.2, 7.4, 7.6, 7.8, 8.0, 8.199999999999999, 8.4, 8.6, 8.800000000000001, 9.0, 9.199999999999999, 9.4, 9.6, 9.800000000000001, 10.0, 10.2, 10.4, 10.6, 10.8, 11.0, 11.2, 11.4, 11.6, 11.8, 12.0]
>>>

```

c) Représentation graphique.

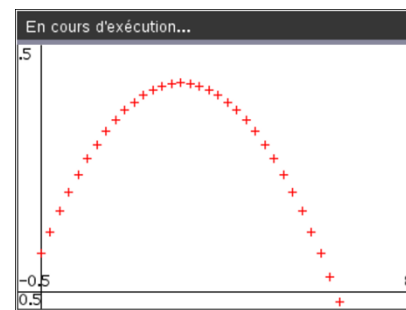
Paramétrer votre représentation graphique comme proposé sur l'écran ci-contre.

```

1.1 1.2 1.3 *Classeur RAD 13/13
*U5SB1.py
absc=[]
ordo=[]
absc=recall_list("l1")
ordo=recall_list("l2")
# Représentation graphique
plt.cls()
plt.window(-0.5,8,-0.5,6.5)
plt.axes("on")
plt.color(255,0,0)
plt.scatter(absc,ordo,"+")
plt.show_plot()

```

Exécuter votre script `ctrl R`.



2 : Exporter des données

Créer un nouveau script et le nommer U5SB11.

Conseil à l'enseignant : Placer le curseur à la fin d'une ligne et valider. L'ordre d'écriture de l'importation des modules est sans importance.

- Créer une fonction nommée **data(a,b,n)**.
- Vous allez créer deux listes de données, à représenter sous forme d'un nuage de points, comportant des valeurs dans un intervalle **[a ; b]**, calculées avec un pas **n**.
- Dans la liste **y**, on calcule la racine carrée des valeurs de la liste **x**.
- Pour créer ces listes de données, nous allons construire une boucle fermée après avoir bien entendu créé deux listes vides.

```


1.2 1.3 1.4 *Classeur RAD 11/13
U5SB11.py
from ti_system import *
def data(a,b,n):
    x=[]
    y=[]
    for i in range(a,b,n):
        x.append(i)
        y.append(sqrt(x[i]))
    store_list("lx",x)
    store_list("ly",y)

```

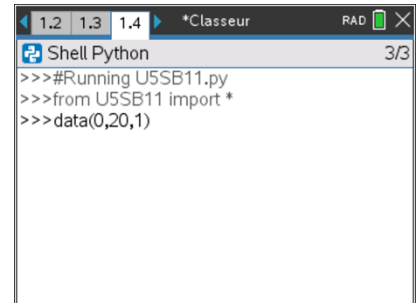




Conseil à l'enseignant : La création de deux listes vides évite le renvoi d'un message d'erreur lors de l'exécution du script.

Remarque : Attention à l'indentation, les instructions **store_list** n'ont pas à être dans la boucle. Utiliser  afin de supprimer un niveau de décalage.

- Exécuter votre script. On prend ici 21 valeurs de 0 à 20 par pas de 1
- Sortir de l'environnement Python et afficher la représentation graphique de vos listes **lx** et **ly** (nuage de points dans l'application graphique par exemple).



```
1.2 1.3 1.4 *Classeur RAD 3/3
Shell Python
>>>#Running U5SB11.py
>>>from U5SB11 import *
>>>data(0,20,1)
```

Conseil à l'enseignant : l'export vers les listes de la calculatrice sera particulièrement intéressant pour représenter des données acquises par l'intermédiaire de capteurs avec le Microcontrôleur **TI-Innovator™** & **Robot TI-Innovator™ Hub**.

