Méthodes numériques

Formulaire Python

I Exemples de base

```
# tout ce qui est après le symbole # est un commentaire, non pris en compte par l'ordinateur

a = 2      # pour affecter la valeur 2 à la variable a
b = 3*a    # ainsi b vaut 3*2=6
c = a**3   # le symbole ** sert à exprimer une puissance, ici a puissance 3 (ce qui fait 8)

print(a)                  # affiche la valeur de a
print("a vaut" + str(a)) # affiche "a vaut" suivi de la valeur de a
                          # (str sert à convertir le nombre a en chaine de caractères)
```

II Boucle for

Une boucle for sert à répéter un bloc de code plusieurs fois.

```
N = 10

for i in range(N):  # la boucle se répète N fois (ici 10 fois), et i vaut successivement 0,1,...,N-1
    print(i)        # par exemple ceci affiche 0, puis 1, puis 2, ... puis 9.
```

Parfois on n'exploite pas la valeur de `i` :

```
N = 10

for i in range(N):   # la boucle se répète N fois (ici 10 fois), et i vaut successivement 0,1,...,N-1
    print("bonjour") # ceci va afficher à chaque fois bonjour, donc 10 fois.
```

Enfin, ce qui est dans la boucle for, donc ce qui est répété, est la partie du code qui est décalée (on dit "indentée") :

```
N = 10

for i in range(N):
    print("bonjour") # ceci fait partie de la boucle
    print("hello")   # ceci aussi

print("hi")          # pas décalé donc n'appartient pas à la boucle (et n'est pas répété).
```

III Listes

Une liste est une succession d'éléments, par exemple `[1, 5, 2.2, 10]`.

- Créer une liste :

  ```
  L = [1, 5, 2.2, 10]  # crée une liste qui s'appelle L
  ```

  On peut aussi créer une liste vide à l'aide de `[]`. Par exemple `L=[]`.

- On accède à l'élément numéro `i` avec `L[i]` où `L` est le nom de la liste.

  La numérotation commence à 0.

  Ci-dessus, `L[0]` vaut 1, `L[1]` vaut 5, `L[2]` vaut 2.2, `L[3]` vaut 10, `L[4]` n'existe pas.

- On peut ajouter un élément à une liste avec la commande `append` :

  ```
  L = [1, 5, 2.2, 10]  # crée une liste qui s'appelle L
  L.append(42)         # ajoute l'élément 42 à la fin de la liste
  print(L)             # va afficher [1, 5, 2.2, 10, 42]
  ```

- La longueur (ou nombre d'éléments) d'une liste s'obtient avec `len(L)`.

  Ci-dessus (après l'ajout de 42), `len(L)` vaut 5.

- On peut parcourir une liste avec une boucle for. Par exemple si on veut afficher un par un les éléments de `L` :

```
L = [1, 5, 2.2, 10]
for i in range(len(L)):  # ici len(L) vaut 4 donc i va de 0 à 4-1=3
    print(L[i])
```

- Pour calculer la moyenne et l'écart-type d'une liste, on peut utiliser des fonctions toutes faites de la librairie `numpy` :

```
import numpy as np   # utilisation de la bibliothèque numpy, qu'on appelle np

L = [1, 5, 2.2, 10]     # crée une liste qui s'appelle L

moy = np.mean(L)        # moy contient alors la valeur moyenne de la liste
u   = np.std(L,ddof=1) # u contient alors l'écart-type de la liste
```

  Plus précisément, en notant $N = \text{len(L)}$:

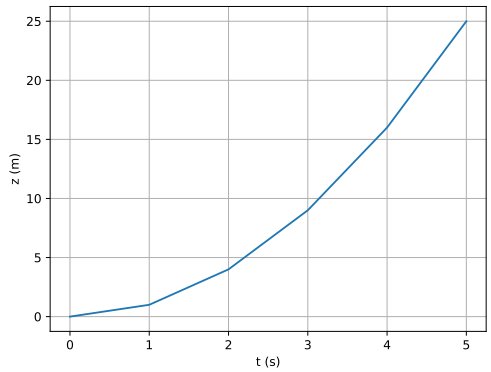
  - `np.moy(L)` calcule $\frac{1}{N}\sum_{i=0}^{N-1} \text{L[i]}$
  - `np.std(L,ddof=1)` calcule $\sqrt{\frac{1}{N-1}\sum_{i=1}^{N-1})(\text{L[i]} - \text{np.moy(L)})^2}$. (std signifie "standard deviation", écart-type en anglais)
  - Rq : si on tape juste `np.std(L)`, alors on calcule $\sqrt{\frac{1}{N}\sum_{i=1}^{N-1})(\text{L[i]} - \text{np.moy(L)})^2}$, ce qui est quasiment pareil.

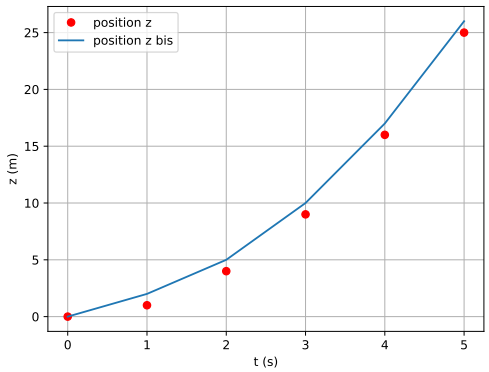
IV Graphiques

Exemple minimal

```
import matplotlib.pyplot as plt  # on importe la bibliothèque graphique, on l'appelle plt

plt.figure(1)                # crée la figure 1

liste_t = [0,1,2,3,4,5]     # exemple de deux listes : une des instants t et l'autre
liste_z = [0,1,4,9,16,25]   # des positions z.

plt.plot(liste_t, liste_z) # on trace liste_z (en ordonnée) en fonction de liste_t (en abscisse).
plt.xlabel('t (s)')        # légende de l'axe des abscisses
plt.ylabel('z (m)')        # légende de l'axe des ordonnées
plt.grid()                 # ajoute une grille
plt.show()                 # affiche le graphe
```

On obtient la figure ci-dessous à gauche. Pour la figure de droite, cf paragraphe "raffinements".





Raffinements

```
plt.plot(liste_t, liste_z, 'o', color='red', label='position z')

liste_z_bis = [0,2,5,10,17,26]
plt.plot(liste_t, liste_z_bis, label='position z bis')  # on peut rajouter une seconde courbe

plt.legend(loc='best')                          # écrit et positionne les "label"
plt.title('position z en fonction du temps t')  # ajoute un titre
```

Ci-dessus `color` définit la couleur. `'o'` indique qu'on trace des points. (Autres symboles : `'-'` pour une ligne tiretée, `'-.'`, `':'`, etc...)