I Commandes de manipulation des chaînes

1.1 Listes

Rappels de quelques commandes utiles

```
L = [1, 12, True]  # crée une liste
L = []  # la liste vide
L.append(1)  # ajoute l'élément à la fin de L.
L2 = [2, 3]
L + L2  # concaténation.
L[0]  # accès au premier élément
L[0] = 12  # modification du premier élément
```

1.2 Slice

On peut sélectionner toute une partie d'une liste avec la syntaxe suivante

```
L[1:4]  # retourne la liste  dont les éléments sont les
# éléments de L d'indices range(1, 4)
L[:6]  # commence la sélection au début
L[3:]  # sélectionne jusqu'à la fin
```

II Tracé de courbes

2.1 Principe du tracé en python

La fonction de tracé que nous allons utiliser est dans une bibliothèque qu'il faut charger au préalable.

```
import matplotlib.pyplot as plt
```

Le principe général de fonctionnement est simple. On passe à la fonction de tracé une liste de points (définis par leurs coordonnées) et la fonction se charge d'afficher ces points et de les relier par des lignes droites.

```
plt.plot(X, Y)
```

où X est la liste des abscisses et Y la liste des ordonnées des points à relier.

2.2 Mise en pratique

Exercice 1
Nous allons avoir besoin de créer des listes d'abscisses rapidement. Créer un script python **trace.py** et une fonction `abscisses(a, b, n)` qui retourne une liste de $n+1$ points équirépartis $[a, \dots, b]$. On découpe l'intervalle $[a, b]$ en n segments de longueur égales.
Aide mathématique : si on note l la longueur de ces n segments, que vaut l ? Le deuxième point de la liste est donc $a + l$. Quel est le troisième ?
On donnera deux versions de cette fonction. Une à base de boucle for et de .append, l'autre en utilisant la notion de liste par compréhension.

Exercice 2
Tracer, dans la console, la courbe représentative de sin sur l'intervalle $[-\pi, \pi]$. On prendra une liste d'ordonnées de longueur 100.

Pour varier la présentation, on peut passer d'autres arguments à la fonction plot.

```
plt.plot(X, Y, 'o')  # change le style de tracé
plt.plot(X, Y, color=''cyan'')  # pour imposer la couleur.
```

Exercice 3
Créer (dans la feuille de script) une fonction `trace(f, a, b, n)` qui prend une **fonction** f, deux nombres a, b et un entier naturel non nul n comme paramètre et trace la courbe de f sur l'intervalle $[a, b]$ que l'on a découpé en n segments.

2.2.1 Plusieurs tracés

Exercice 4
Utiliser la fonction précédente pour tracer tan sur $[-\pi, \pi]$. Le résultat fait apparaître des droites qui doivent pas apparaître dans cette courbe.
Pour y remédier, on va utiliser plusieurs appels à `plt.plot`. Définir une variable $epsilon = 10e - 3$. Créer les liste `X1, X2, X3` des abscisses dans $[-\pi, -\frac{pi}{2} - epsilon], [-\frac{pi}{2} + epsilon, \frac{pi}{2} - epsilon], [\frac{pi}{2} + epsilon, \pi]$ de manière à équirepartir les points et en obtenir 100 en tout.
Créer les listes `Y1, Y2, Y3` des ordonnées correspondantes, puis

```
plt.plot(X_1, Y_1, X_2, Y_2, X_3, Y_3)
```

III La bibliothèque numpy

Il se trouve que python dispose d'une bibliothèque très pratique pour effectuer toutes les opérations précédentes.

```
import numpy as np
```

3.1 `linspace`

Exercice 5
Comprendre le fonctionnement de la commande `np.linspace`.
On pourra s'aider de l'aide disponible grâce à la fonction `help`

3.2 Les fonctions numpy

Exercice 6
La bibliothèque numpy contient une version des fonctions mathématiques usuelles

```
np.sin([0, np.pi, np.pi/2])
```

Ces fonctions peuvent d'appliquer à des nombres aussi bien qu'à des listes ou des vecteurs numpy (la version numpy des listes, comme la valeur de retour de `linspace`)
En 3 lignes, dans la console, tracer la courbe représentative de exp sur $[-1, 1]$ avec 150 points.

Exercice 7 (Bonus)
Créer une fonction `vectorise(f)` où f est une fonction numérique et qui retourne une fonction g. g doit être définie pour des listes de valeurs et retourner la liste des valeurs de f correspondante.

IV Fonctions usuelles

4.1 Les solutions d'équations différentielles

Créer une fonction `sol_edl(omega, A, phi, tf)` qui trace la courbe de la fonction $t \mapsto \mathrm{A}\cos(\omega t + phi)$ sur l'intervalle $[0, tf]$.

4.2 Les fonctions polynomiale

On représente la fonction polynomiale $f : x \mapsto a_0 + a_1 x + \cdots + a_n x^n$ par la liste $[a_0, \ldots, a_n]$. Créer une fonction `trace_polynome(L, a, b)` qui trace la courbe représentative de la fonction polynomiale représentée par la liste L sur l'intervalle $[a, b]$

V Algorithmes autour des liste de valeurs

Exercice 8
Créer une fonction `annulation(X, Y)` qui prend une liste d'abscisse et une liste d'ordonnée et retourne la liste des valeurs approchées des éventuels abscisse où la fonction correspondante s'annule.
On pourra considérer, comme pour le tracé python, que les fonctions sont affines par morceaux.

Exercice 9
Créer une fonction `extrema(X, Y)` qui retourne une liste à deux éléments : (x_m, y_m) et $(x_\mathrm{M}, y_\mathrm{M})$ qui sont les coordonnées du minimum (première occurrence) et du maximum (première occurrence) des valeurs de la fonction représentée par X et Y.