

I Création et manipulation de listes

Exercice 1 (Listes par compréhension)

Pour créer une liste dont on connaît la forme des éléments, on peut utiliser la syntaxe suivante :

```
1 [ element for variable in conteneur ]
```

où **element** est la forme des éléments souhaités (et son expression peut dépendre de **variable**) et la boucle **for** à la même signification que d'habitude : **variable** va prendre toutes les valeurs contenues dans **conteneur** (qui peut être : un **range**, une liste, une chaîne).

1. Décrire les valeurs (ainsi que la longueur de la liste) que contient la liste L après l'exécution de chacun des codes suivants. Vous pouvez répondre par une phrase ou en écrivant la liste complète.

```
1 L = [0 for i in range(100)]
```

2. Même question avec

```
1 L = [i**2 for i in range(1, 11)]
```

3. Idem

```
1 L = [c for c in 'une chaîne']
```

Exercice 2

Donner une instruction python permettant de créer la liste de longueur 100 contenant les nombres de la forme $\frac{k}{50}$ pour $k \in [0, 99]$

Exercice 3 (Création élément par élément)

Il arrive que l'on ne puisse pas calculer explicitement *a priori* chaque élément de la liste que l'on souhaite construire. Dans ce cas on peut utiliser la syntaxe suivante :

```
1 L = [] # on crée d'abord une liste vide
2 for .... # ou une boucle while suivant les cas
3     # calcul d'un élément en plus de la liste
4     L.append(elt) # où elt est l'élément calculé
5     # cette commande ajoute à la fin de la liste L la valeur contenue dans elt
```

On considère le problème suivant : on fixe $u_0 = 1$ puis on pose pour tout $n \in \mathbb{N}$, $u_{n+1} = \frac{u_n}{2} + \frac{1}{u_n}$. On souhaite construire la liste des 15 premières valeurs de $(u_n)_{n \in \mathbb{N}}$ c'est à dire qu'on veut obtenir $L = [u_0, u_1, \dots, u_{14}]$.

1. Le calcul à la main des premières valeurs montre que l'on doit connaître une valeur de (u_n) avant de calculer la suivante : on peut déduire u_1 de u_0 puis u_2 de u_1 . Ainsi, en utilisant une seule variable on peut calculer les valeurs successives de (u_n) .

Algorithme :

- **val** reçoit la valeur de u_0
- Répéter 14 fois : **val** reçoit $\text{val}/2 + 1/\text{val}$

Au départ **val** contient u_0 . Après avoir exécuté une première fois l'instruction à répéter, **val** contient $\frac{u_0}{2} + \frac{1}{u_0}$ c'est à dire u_1 .

Pour la deuxième répétition, le calcul est donc $\frac{u_1}{2} + \frac{1}{u_1}$ et **val** reçoit en fait la valeur de u_2 . Et ainsi de suite.

Écrire le code python correspondant à cet algorithme dans la fonction `calculer14`, présente dans la feuille de script `tp4.py`.

2. Tester le code produit dans la console. La valeur de retour doit être 1.414213562373095.
Copier ensuite le code dans la fonction `liste_valeurs` l'adapter. Cette fonction doit cette fois retourner la liste des 15 premières valeurs de $(u_n)_{n \in \mathbb{N}}$.

II Graphiques

Nous souhaitons maintenant faire tracer des figures simples à python. Nous allons d'abord charger les outils python permettant le dessin.

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

Cette commande charge en mémoire la bibliothèque `matplotlib.pyplot` sous le nom `plt`. Nous pouvons maintenant accéder aux fonctions de dessins définies dans cette bibliothèque. Par exemple, la commande qui nous intéresse ici sera

```
1 plt.plot
```

Il s'agit de la fonction `plot` définie dans cette bibliothèque (quelque part dans le fichier `pyplot.py` on trouve `def plot.....`).

2.1 Utilisation de plot

La fonction `plot` s'utilise (dans sa version la plus simple) de la manière suivante :

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

où `X` et `Y` sont des listes (ou au moins des conteneurs) de même longueur, qui contiennent des nombres.

- `X` est une liste d'abscisses
- `Y` est une liste d'ordonnées

Le tracé s'effectue en plaçant les points données par ces listes de coordonnées, puis en les reliant dans l'ordre.

Exercice 4

Tester (dans la **console**) et comprendre les commandes suivantes (il faut avoir effectué l'import d'abord) :

```
1 X = [-1, 1, -1, 1]
2 Y = [0, 0, 2, 2]
3 plt.plot(X, Y)
```

Exercice 5

Tracer un carré de côté 1 et de centre $O : (0, 0)$.

Exercice 6

D'après le cours de math, les points A_k de coordonnées $(\cos \frac{2k\pi}{7}, \sin \frac{2k\pi}{7})$ pour $k \in \llbracket 0, 6 \rrbracket$ sont les sommets d'un heptagone régulier inscrit dans le cercle trigonométrique.

Les fonctions mathématiques sont disponibles dans la bibliothèque `math`

```
1 import math
2 # on a accès à math.cos et math.sin maintenant
```

Tracer cet heptagone.

Exercice 7

Compléter la fonction `polygone` qui prend comme argument un entier `n` et trace un polygone régulier à n côtés si $n \geq 3$. Cette fonction n'aura pas de valeur de retour, et ne tracera rien lorsque $n < 3$.

2.2 Tracé de courbe représentative

Le principe utilisé pour tracer une courbe représentative d'une fonction f est une application directe des principes précédents :

- on crée tout d'abord une liste d'abscisses "suffisamment rapprochées".
- on crée ensuite la liste de toutes les valeurs de la fonction aux abscisses précédentes, ce seront nos ordonnées.

Exercice 8

On souhaite, pour un n entier naturel non nul donné, découper l'intervalle $[a, b]$ en n intervalles de même longueur.

1. Quelle est la longueur (notée ℓ) de chaque petit intervalle ?
2. On note $x_0 = a$, puis x_1 la deuxième extrémité du premier petit intervalle, puis x_2 la deuxième extrémité du second.... Pour quel indice p a-t-on $x_p = b$? Exprimer les x_i en fonction de a, b, ℓ puis en fonction de a et b uniquement

Exercice 9

On souhaite créer une fonction `abscisses` qui prend comme argument (dans l'ordre) deux nombres a, b et un entier n et retourne la liste des x_i définis dans la question précédente.

1. Quelle sera la valeur de retour de `abscisses(0, 1, 2)` ? et de `abscisses(0, 1, 5)` ? Et si n vaut 1 ? Et si $a > b$?
2. Compléter la fonction dans la feuille de script. On pourra utiliser au choix une liste par compréhension ou une construction pas à pas.
3. Vérifier le comportement de votre fonctions sur les exemples de la question 1.

Exercice 10

Grâce à la fonction précédente, tracer la courbe représentative de \sin sur l'intervalle $[-2\pi, 2\pi]$ en utilisant 401 points. On pourra utiliser la constante `math.pi` déjà définie.

Même question avec $x \mapsto \frac{1}{1+x^2}$, mais sur l'intervalle $[-3, 3]$

Exercice 11

Définir en python une fonction `f` qui prend un nombre `x` comme argument et retourne $\begin{cases} 1 + \sin(x) & \text{si } x \leq 0 \\ e^x & \text{si } x > 0 \end{cases}$.

Tracer ensuite la courbe sur l'intervalle $[-2\pi, 2]$.

III Pour aller plus loin

Exercice 12

On souhaite tracer sur une même figure deux courbes : celle de sin et celle de cos. Le principe est simple : on exécute plusieurs appels à `plt.plot`.

1. Tracer les deux courbes demandées sur $[-2\pi, 2\pi]$ en choisissant le nombre de point de telle manière à obtenir un tracé convenable (tester avec seulement 10 points, puis avec 100).

On pourra au choix : travailler directement dans la console (pas très pratique) ou créer une nouvelle feuille de script que l'on exécute en entier.

2. Il est difficile de savoir quelle courbe est tracée dans ce cas. Pour les identifier, on peut imposer la couleur.

```
1 plt.plot(X, Y, color='red') # trace en rouge, mais ce n'est pas la seule couleur disponible.
2 # Les arguments que l'on passe sous forme argument=valeur sont des arguments
3 # optionnels. Si on ne les donne pas, on obtient quand même un résultat
```

Tracer les courbes précédentes en imposant la couleur de chacune.

3. On peut également légender nos tracés.

```
1 plt.plot(X, Y, label='nom de la courbe')
```

puis après tous les appels à `plt.plot`

```
1 plt.legend() # crée la légende à partir de tous les label donnés.
```

A vous de jouer !

Exercice 13

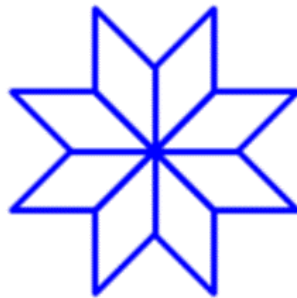
Tracer dans chaque cas l'ensemble des points $M(t)$ (avec $t \in [-2\pi, 2\pi]$) dont les coordonnées sont :

1. $(\cos(t) + \sqrt{8}\cos(\frac{t}{2}), \sin(t))$
2. $(\sin^3(t), \cos(t) - \cos^4(t))$.

Application : on peut ainsi très facilement tracer la trajectoire d'un mobile connaissant l'équation horaire

Exercice 14

Tracer (pourquoi pas en alternant les couleurs ?) une figure s'approchant de



Exercice 15

Idem avec

