

VI - Python

Commentaire

| | | |
|---|--------------------------|--|
| # | Insertion de commentaire | # Cette ligne ne sera pas lue par Python |
|---|--------------------------|--|

I - Structures élémentaires

I.1 - Types élémentaires

Nombres

| | | |
|----|----------------|--------------------------------|
| + | Addition | 3.4 + 2 renvoie 5.4 |
| - | Soustraction | 3.4 - 2 renvoie 1.4 |
| * | Multiplication | 3.4 * 2 renvoie 6.8 |
| / | Division | 3.4/2 renvoie 1.7 |
| ** | Puissance | 3.4**2 renvoie (presque) 11.56 |

Booléens

| | | |
|-------------|-------------------------------|-------------------------|
| True, False | Valeurs booléennes vrai, faux | |
| == | Égal | 4 == 2*2 renvoie True |
| > | Strictement supérieur | 4 > 2 renvoie True |
| < | Strictement inférieur | 4 < 2 renvoie False |
| >= | Supérieur ou égal | 4 >= 2 * 2 renvoie True |
| <= | Inférieur ou égal | 4 <= 2 * 2 renvoie True |

Connecteurs logiques

| | | |
|-----|-------------|---------------------------------------|
| and | Et logique | (3 == 0) and (4 == 2*2) renvoie False |
| or | Ou logique | (3 == 0) or (4 == 2*2) renvoie True |
| not | Non logique | not (3 == 0) renvoie False |

I.2 - Structures de contrôle

| Affectation | | |
|---|---|---|
| = | <p>Affectation</p> <p>Appel du contenu</p> <p>Écrasement du contenu de</p> | <pre>x = 3</pre> <p>stocke la valeur 3 dans la variable nommée x.</p> <pre>2 * x + 3</pre> <p>renvoie 9</p> <pre>x = 25 + 3 * 12</pre> |
| Instruction conditionnelle | | |
| <pre>if c1: i1 elif c2: i2 else: i3</pre> | <p>c1, c2 sont des booléens</p> <p>i1, i2, i3 sont des instructions</p> <p>Attention aux indentations</p> <p>Attention aux deux-points :</p> <p>elif (sinon mais si) et else (sinon) sont optionnels</p> | <pre>x = 20 if x < 20: print("Riri") elif x < 50: print("Fifi") else: print("Loulou")</pre> <p>Affiche Fifi</p> |
| Boucle itérative | | |
| <pre>for element in liste: i1</pre> | <p>Nombre prédéterminé d'itérations</p> <p>i1 est une suite d'instructions</p> <p>Attention aux indentations</p> <p>Attention aux deux-points :</p> <p>Liste peut être une liste [3, 12, 1, 4]</p> <p>un intervalle d'entiers range(a, b)</p> <p>un intervalle de réels np.arange(a, b, pas)</p> <p>np.linspace(a, b, nombre)</p> | <pre>for i in [3, 12, 1, 4]: print(i)</pre> <p>Affiche 3 12 1 4</p> <pre>for i in range(3, 7): print(i)</pre> <p>Affiche 3 4 5 6</p> |
| Boucle conditionnelle | | |
| <pre>while c: i1</pre> | <p>c est un booléen</p> <p>i1 est une suite d'instructions</p> <p>Attention aux indentations</p> <p>Attention aux deux-points :</p> <p>Attention à modifier la condition à chaque passage pour qu'elle devienne fausse</p> | <pre>i = 3 while i < 48: i = 2 * i print(i)</pre> <p>Affiche 6, 12, 24, 48</p> |

Fonctions

| | | |
|---|--|---|
| <pre>def f(x, y): i1 return z</pre> | <p>x, y sont les paramètres formels i1 est une suite d'instructions z est la valeur renvoyée Attention aux deux-points :</p> | <pre>def f(x): y = x**2 + 1 return 3 * y</pre> <p>Affiche f(3) renvoie 30</p> |
|---|--|---|

I.3 - Modules

Importer des modules

| | |
|---|--|
| <pre>from numpy import * log(2)</pre> | <p>Importe toutes les fonctions de numpy Appel sans préciser la provenance</p> |
| <pre>import numpy as np np.log(2)</pre> | <p>Charge le module numpy. Appel en précisant le module d'appartenance.</p> |

I.4 - Numpy - Calculs numériques

Module pour effectuer des calculs numériques :

```
import numpy as np
```

Constantes

| | | |
|--------------|-----------------|--------------------|
| np.e | Constante e | Vaut environ 2.718 |
| np.pi | Constante π | Vaut environ 3.14 |

Fonctions

| | | |
|-----------------|---------------------|--|
| np.exp | Exponentielle | np.exp(1) renvoie environ 2.718 |
| np.log | Logarithme népérien | np.log(1) renvoie 0 |
| np.sqrt | Racine carré | np.sqrt(4) renvoie 2.0 |
| np.abs | Valeur absolue | np.abs(-3) renvoie 3 |
| np.floor | Partie entière | np.floor(3.14) renvoie 3.0 |

Création de tableaux / matrices

| | | |
|-------------------------|--|--|
| np.array | Crée un tableau à partir de la liste des éléments | np.array([[1, 2, 3], [4, 5, 6]]) définit $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ |
| np.zeros((n, p)) | Crée une matrice à n lignes et p colonnes ne contenant que des zéros | np.zeros((2, 3)) définit $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ |

| | | |
|--|--|--|
| <code>np.ones((n, p))</code> | Crée une matrice à n lignes et p colonnes ne contenant que des 1 | <code>np.ones((2, 3))</code> définit $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ |
| <code>np.eye(n)</code> | Crée la matrice identité d'ordre n | <code>np.eye(3)</code> définit $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ |
| <code>np.arange(a, b, pas)</code> | Crée un vecteur ligne d'éléments de a (inclus) à b (exclus) espacés de pas | <code>np.arange(1.2, 2, 0.2)</code> définit (1.2 1.4 1.6 1.8) |
| <code>np.linspace(a, b, nbre)</code> | Crée un vecteur ligne d'éléments régulièrement espacés de a (inclus) à b (inclus) contenant nbre éléments | <code>np.linspace(1.2, 2, 5)</code> définit (1.2 1.4 1.6 1.8 2.) |
| Manipulation de matrices | | |
| <code>t[i][j]</code> | Accède à la ligne i colonnes j de t Numérotation à partir de 0 | <code>t = np.array([[1, 2, 3], [4, 5, 6]])</code> <code>t[1][2]</code> renvoie 6 |
| <code>np.shape</code> | Renvoie le nombre de lignes et le nombre de colonnes | <code>t = np.array([[1, 2, 3], [4, 5, 6]])</code> <code>a, b = np.shape(t)</code> <code>a</code> contient 2, <code>b</code> contient 3 |
| <code>np.reshape</code> | Aplatit puis redimensionne un tableau | <code>a = np.array([[1, 2, 3], [4, 5, 6]])</code> <code>np.reshape(a, (3, 2))</code> renvoie $\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$ |
| Opérations sur les matrices | | |
| | | <code>a = np.array([[1, 2], [3, 4]])</code> <code>b = np.array([[-1, 1], [0, 1]])</code> |
| <code>+</code> | Addition élément par élément | <code>a + b</code> renvoie $\begin{pmatrix} 0 & 3 \\ 3 & 5 \end{pmatrix}$ |
| <code>-</code> | Soustraction élément par élément | <code>a - b</code> renvoie $\begin{pmatrix} 2 & 1 \\ 3 & 3 \end{pmatrix}$ |
| <code>*</code> | Multiplication d'une matrice par un réel | <code>2 * a</code> renvoie $\begin{pmatrix} 2 & 4 \\ 6 & 8 \end{pmatrix}$ |
| <code>np.dot</code> | Produit matriciel | <code>np.dot(a, b)</code> renvoie $\begin{pmatrix} -1 & 3 \\ -3 & 7 \end{pmatrix}$ |
| Les fonctions <code>np.exp</code> , <code>np.sqrt</code> ,... s'effectuent élément par élément | | |

Stastitiques

| | | |
|------------------------|--|---|
| <code>*, /, **</code> | ATTENTION! Opérations élément par élément | <code>b**a</code> renvoie $\begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$ |
| | | <code>t = np.array([[1, 2, 3], [4, 5, 6]])</code> |
| <code>np.sum</code> | Somme des éléments d'un tableau | <code>np.sum(t)</code> renvoie 21 <code>np.sum(t, 0)</code> renvoie $\begin{pmatrix} 5 & 7 & 9 \end{pmatrix}$ <code>np.sum(t, 1)</code> renvoie $\begin{pmatrix} 6 \\ 15 \end{pmatrix}$ |
| <code>np.min</code> | Minimum des éléments d'un tableau | <code>np.min(t)</code> renvoie 1 <code>np.min(t, 0)</code> renvoie $\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$ <code>np.min(t, 1)</code> renvoie $\begin{pmatrix} 1 \\ 4 \end{pmatrix}$ |
| <code>np.max</code> | Maximum des éléments d'un tableau | <code>np.max(t)</code> renvoie 6 <code>np.max(t, 0)</code> renvoie $\begin{pmatrix} 4 & 5 & 6 \end{pmatrix}$ <code>np.max(t, 1)</code> renvoie $\begin{pmatrix} 3 \\ 6 \end{pmatrix}$ |
| <code>np.mean</code> | Moyenne des éléments d'un tableau | <code>np.mean(t)</code> renvoie 3.5 <code>np.mean(t, 0)</code> renvoie $\begin{pmatrix} 2.5 & 3.5 & 4.5 \end{pmatrix}$ <code>np.mean(t, 1)</code> renvoie $\begin{pmatrix} 2 \\ 5 \end{pmatrix}$ |
| <code>np.median</code> | Médiane des éléments d'un tableau | <code>np.median(t)</code> renvoie 3.5 <code>np.median(t, 0)</code> renvoie $\begin{pmatrix} 2.5 & 3.5 & 4.5 \end{pmatrix}$ <code>np.median(t, 1)</code> renvoie $\begin{pmatrix} 2 \\ 5 \end{pmatrix}$ |
| <code>np.var</code> | Variance des éléments d'un tableau | <code>np.var(t)</code> renvoie 2.916... <code>np.var(t, 0)</code> renvoie $\begin{pmatrix} 2.25 & 2.25 & 2.25 \end{pmatrix}$ <code>np.var(t, 1)</code> renvoie $\begin{pmatrix} 0.6... \\ 0.6... \end{pmatrix}$ |
| <code>np.std</code> | Écart-type des éléments d'un tableau | <code>np.std(t)</code> renvoie 1.707... <code>np.std(t, 0)</code> renvoie $\begin{pmatrix} 1.5 & 1.5 & 1.5 \end{pmatrix}$ <code>np.std(t, 1)</code> renvoie $\begin{pmatrix} 0.816... \\ 0.816... \end{pmatrix}$ |
| <code>np.cumsum</code> | Somme cumulée des éléments Aplatit le tableau si nécessaire | <code>np.cumsum(t)</code> renvoie $\begin{pmatrix} 1 & 3 & 6 & 10 & 15 & 21 \end{pmatrix}$ <code>np.cumsum(t, 0)</code> renvoie $\begin{pmatrix} 1 & 2 & 3 \\ 5 & 7 & 9 \end{pmatrix}$ <code>np.cumsum(t, 1)</code> renvoie $\begin{pmatrix} 1 & 3 & 6 \\ 4 & 9 & 15 \end{pmatrix}$ |

I.5 - Pyplot - Graphiques

Module pour effectuer des rendus graphiques :

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

| Tracé | |
|--|--|
| <code>plt.plot(X, Y)</code> | X : liste des abscisses Y : liste des ordonnées Crée le graphique contenant le tracé de la suite de points |
| <code>plt.show()</code> | Montre le graphique |
| Compléments | |
| <code>plt.xlim(xmin, xmax)</code> | xmin : abscisse minimale xmax : abscisse maximale |
| <code>plt.ylim(ymin, ymax)</code> | ymin : ordonnée minimale ymax : ordonnée maximale |
| <code>plt.axis([xmin, xmax, ymin, ymax])</code> | Fixe les abscisses / ordonnées minimales / maximales |
| <code>plt.grid(True)</code> | Affiche le quadrillage |
| <code>plt.grid(False)</code> | Masque le quadrillage |
| <code>plt.legend()</code> | Affiche la légende. |
| Graphiques particuliers | |
| <code>plt.hist(x)</code> | Crée un histogramme avec les valeurs de x Choix des critères automatique ou à préciser avec une option |
| <code>plt.bar(x, hauteur)</code> height liste des hauteurs des barres | x liste des abscisses des barres |
| <code>plt.boxplot</code> | Boîtes à moustaches |

I.6 - Random - Pseudo-alea

Module pour utiliser des nombres pseudo-aléatoires :

```
import numpy.random as rd
```

| Loi uniforme | |
|----------------------------|--|
| <code>rd.rand(n, p)</code> | Renvoie un tableau à n lignes et p colonnes Chaque élément est la réalisation d'une variable aléatoire de loi uniforme sur $[0, 1]$ |

I.7 - Pandas - Panel data - Gestion des données

Module pour manipuler des données :

```
import pandas as pd
```

| Statistiques | |
|--------------|--|
| pd.mean | Moyenne des éléments du tableau par catégorie |
| pd.std | Écart-type des éléments du tableau par catégorie |