X - 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

`=`	Affectation
	
	Appel du contenu
	Écrasement du contenu de
		`x = 3`
		stocke la valeur 3 dans la variable nommée `x`.
		`2 * x + 3` renvoie `9`
		`x = 25 + 3 * 12`

Instruction conditionnelle

`if c1:`
`  i1`
`elif c2:`
`  i2`
`else:`
`  i3`
	`c1`, `c2` sont des booléens
	`i1`, `i2`, `i3` sont des instructions
	Attention aux indentations
	Attention aux deux-points :
	`elif` (sinon mais si) et `else` (sinon) sont optionnels
		`x = 20`
		`if x < 20:`
		`  print("Riri")`
		`elif x < 50:`
		`  print("Fifi")`
		`else:`
		`  print("Loulou")`
		Affiche `Fifi`

Boucle itérative

`for element in liste:`
`  i1`
	Nombre prédéterminé
	`i1` est une suite
	Attention aux
	Attention aux
	Liste peut être
		d'itérations
		d'instructions
		indentations
		deux-points :
		une liste `[3, 12, 1, 4]`
		un intervalle d'entiers `range(a, b)`
		un intervalle de réels `np.arange(a, b, pas)`
		`np.linspace(a, b, nombre)`
			`for i in [3, 12, 1, 4]:`
			`  print(i)`
			Affiche `3 12 1 4`
			`for i in range(3, 7):`
			`  print(i)`
			Affiche `3 4 5 6`

Boucle conditionnelle

`while c:`
`  i1`
	`c` est un booléen
	`i1` est une suite d'instructions
	Attention aux indentations
	Attention aux deux-points :
	Attention à modifier la condition à chaque passage pour qu'elle devienne fausse
		`i = 3`
		`while i < 48:`
		`  i = 2 * i`
		`  print(i)`
		Affiche `6`, `12`, `24`, `48`

Fonctions

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

I.3 - Modules

Importer des modules

`from numpy import *` `log(2)`	Importe toutes les fonctions de `numpy` Appel sans préciser la provenance
`import numpy as np` `np.log(2)`	Charge le module `numpy`. Appel en précisant le module d'appartenance.

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 $\begin{pmatrix}0&0&0\\0&0&0\end{pmatrix}$

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

Stastitiques

`*`, `/`, `**`	**ATTENTION !** Opérations élément par élément	`b**a` renvoie $\begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}$
		`t = np.array([[1, 2, 3], [4, 5, 6]])`
`np.sum`	Somme des éléments d'un tableau	`np.sum(t)` renvoie 21 `np.sum(t, 0)` renvoie $\begin{pmatrix} 5 & 7 & 9 \end{pmatrix}$ `np.sum(t, 1)` renvoie $\begin{pmatrix} 6 \\ 15 \end{pmatrix}$
`np.min`	Minimum des éléments d'un tableau	`np.min(t)` renvoie 1 `np.min(t, 0)` renvoie $\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$ `np.min(t, 1)` renvoie $\begin{pmatrix} 1 \\ 4 \end{pmatrix}$
`np.max`	Maximum des éléments d'un tableau	`np.max(t)` renvoie 6 `np.max(t, 0)` renvoie $\begin{pmatrix} 4 & 5 & 6 \end{pmatrix}$ `np.max(t, 1)` renvoie $\begin{pmatrix} 3 \\ 6 \end{pmatrix}$
`np.mean`	Moyenne des éléments d'un tableau	`np.mean(t)` renvoie 3.5 `np.mean(t, 0)` renvoie $\begin{pmatrix} 2.5 & 3.5 & 4.5 \end{pmatrix}$ `np.mean(t, 1)` renvoie $\begin{pmatrix} 2 \\ 5 \end{pmatrix}$
`np.median`	Médiane des éléments d'un tableau	`np.median(t)` renvoie 3.5 `np.median(t, 0)` renvoie $\begin{pmatrix} 2.5 & 3.5 & 4.5 \end{pmatrix}$ `np.median(t, 1)` renvoie $\begin{pmatrix} 2 \\ 5 \end{pmatrix}$
`np.var`	Variance des éléments d'un tableau	`np.var(t)` renvoie $2.916\ldots$ `np.var(t, 0)` renvoie $\begin{pmatrix} 2.25 & 2.25 & 2.25 \end{pmatrix}$ `np.var(t, 1)` renvoie $\begin{pmatrix} 0.6\ldots \\ 0.6\ldots \end{pmatrix}$
`np.std`	Écart-type des éléments d'un tableau	`np.std(t)` renvoie $1.707\ldots$ `np.std(t, 0)` renvoie $\begin{pmatrix} 1.5 & 1.5 & 1.5 \end{pmatrix}$ `np.std(t, 1)` renvoie $\begin{pmatrix} 0.816\ldots \\ 0.816\ldots \end{pmatrix}$
`np.cumsum`	Somme cumulée des éléments Aplatit le tableau si nécessaire	`np.cumsum(t)` renvoie $\begin{pmatrix} 1 & 3 & 6 & 10 & 15 & 21 \end{pmatrix}$ `np.cumsum(t, 0)` renvoie $\begin{pmatrix} 1 & 2 & 3 \\ 5 & 7 & 9 \end{pmatrix}$ `np.cumsum(t, 1)` 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	
`plt.plot(X, Y)`	`X` : liste des abscisses `Y` : liste des ordonnées Crée le graphique contenant le tracé de la suite de points
`plt.show()`	Montre le graphique
Compléments	
`plt.xlim(xmin, xmax)`	`xmin` : abscisse minimale `xmax` : abscisse maximale
`plt.ylim(ymin, ymax)`	`ymin` : ordonnée minimale `ymax` : ordonnée maximale
`plt.axis([xmin, xmax, ymin, ymax])`	Fixe les abscisses / ordonnées minimales / maximales
`plt.grid(True)` `plt.grid(False)`	Affiche le quadrillage Masque le quadrillage
`plt.legend()`	Affiche la légende.
Graphiques particuliers	
`plt.hist(x)`	Crée un histogramme avec les valeurs de `x` Choix des critères automatique ou à préciser avec une option
`plt.bar(x, hauteur)` `height` liste des hauteurs des barres	`x` liste des abscisses des barres
`plt.boxplot`	Boîtes à moustaches

I.6 - Random - Pseudo-alea

Module pour utiliser des nombres pseudo-aléatoires :
`import numpy.random as rd`

Loi uniforme	
`rd.rand(n, p)`	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