Mémo Python 3

Liste des mots clés ou mots réservés

and	del	for	is	raise
as	elif	from	lambda	return
assert	else	global	not	try
break	except	if	or	while
class	exec	import	pass	with
continue	finally	in	print	yield
def				

Listes des opérateurs

Opérations	**Symboles**	**Exemples**
addition	*+*	2 + 5 donne 7
soustraction	-	8 - 2 donne 6
multiplication	*	*6 * 7* donne 42
exponentiation (puissance)	**	*5 ** 3* donne 125
division	/	7 / 2 donne 3.5
reste de division entière	%	7 % 3 donne 1
quotient de division entière	//	7 // 3 donne 2

8 opérateurs de comparaisons

x == y	x est égal à y (test d'égalité à distinguer de l'affectation)
x !=y	x est différent de y
x < y	x est strictement inférieur à y
x > y	x est strictement supérieur à y
x <= y	x est inférieur ou égal à y
x >= y	x est supérieur ou égal à y
x is y	Test d'égalité d'objet
x in y	x appartient à y

Type de données

Type de données	Type Python	Exemple
Entier	Integer (Int)	25 ou -25
Réel ou décimal	Float	22.75
Chaîne de caractères	String (Str)	Toto ou '4'
Tableau	Liste (List)	[1 ,2.5, 8, 'Toto']

Instructions conditionnelles

Langage algorithmique	Python
DEBUT **SI** condition1 **ALORS** # instruction 1 **SINON** # instruction 2 **FIN_SI** **FIN**	**if** condition1: # instruction 1 **else:** # instruction 2
Pour des tests multiples: **DEBUT** **SI** condition1 **ALORS** # instruction 1 **SINON SI** condition 2 **ALORS** # instruction 2 **SINON** **#** instruction 3 **FIN_SI** **FIN**	**if** condition1: # instruction 1 **elif** condition 2 : # instruction 2 **else:** # instruction 3

Boucle non bornée ou boucle while

Langage algorithmique	Python
DEBUT **TANT_QUE** condition **FAIRE** # instructions **FIN_TANT_QUE** **FIN**	**while condition :** **# Bloc d'instructions**

Boucle bornée ou boucle for

Pseudo Langage	Syntaxe en python
POUR i **ALLANT DE** debut à fin par pas de FAIRE Instruction **FIN_POUR**	For i in range(debut,fin,pas): #instruction

Attention, fin est exclu. Le pas est égal à 1 par défaut.

Exemple :

```
For i in range (1,4) :
        print(i)
```

Cela affiche :

1
2
3

Les bibliothèques scientifiques utiles

Les **bibliothèques** sont des ensembles de scripts (que nous appellerons modules) rédigés par un tiers, mettant à disposition de nouveaux outils qui n'étaient pas présents. Par exemple, la bibliothèque math mettra à disposition un outil « sqrt » qui permettra de calculer en une instruction la racine carrée d'un nombre.

La bibliothèque math

Instructions	Explication
import math	A écrire au début du programme Permet d'importer et d'utiliser les objets de la bibliothèque math
math.sin(x) math.cos(x) math.tan(x) math.asin(y) math.acos(y) math.atan(y)	Permet d'utiliser les fonctions trigonométriques cosinus, sinus, tangente, arc sinus, arc cosinus et arc tangente. Attention, les valeurs de x pour les fonctions sinus, tangente et cosinus sont nécessairement **en radians**.
math.exp(x)	Permet d'obtenir l'exponentielle de x : e^x
math.log(x, n) math.log10(x) math.log2(x)	Permet d'obtenir le logarithme de x en base n. Par défaut math.log(x) est en base e. Permet d'obtenir les logarithmes en base 10 ou 2 de x (plus précis).
Math.e math.pi math.inf et - math.inf	Permet d'utiliser les constantes e et π Permet de travailler avec $+\infty$ et $-\infty$
math.sqrt(x)	Renvoie la racine carrée de x
math.hypot(x, y)	Renvoie la norme euclidienne $\sqrt{x^2 + y^2}$
math.degrees(x) math.radians(x)	Renvoie la conversion de l'angle en degrés (x étant en radians). Renvoie la conversion de l'angle en radians (x étant en degrés).

La bibliothèque matplot

La bibliothèque matplotlib est utile pour la représentation, en 2D ou 3D, de fonctions mathématiques avancées ou d'ensembles de valeurs numériques. Elle possède de nombreux modules mais celui qui nous intéresse dans le cadre d'un enseignement au lycée est pyplot.

Instructions	Explication
import matplotlib.pyplot as plt	Permet d'importer la bibliothèque pyplot et d'utiliser son alias plt
plt.plot(x, y)	Trace la courbe $y = (x)$. Paramètres optionnels : color : "black", "red", "blue", … modifie la couleur du tracé linewidth : "1", "2", "3" … modifie l'épaisseur du tracé linestyle :"-", "--", ":", "." modifie l'apparence du tracé marker : "+", "x", ".", "o", "v" modifie la forme des points marqués label : "courbe"1", "courbe2", … indique une légende pour chaque courbe. L'instruction plt.legend() doit figurer dans le script. *Remarque :* Il est possible de condenser les paramètres : plt.plot(x, y, "b:x") permet d'obtenir une ligne pointillée bleue avec des points en croix
plt.show()	Affiche les courbes mises en mémoire avec plt.plot(). Si plusieurs courbes étaient en traitement, elles sont superposées.
plt.grid()	Rend visible le quadrillage sur la figure
plt.legend()	Permet d'afficher les labels en légende lorsqu'ils ont été indiqués en paramètre label dans plt.plot()
plt.xlabel("votre texte", fontsize=12) plt.ylabel("votre texte", fontsize=12)	Affiche le "texte" en abscisse ou en ordonnée et précise sa taille.
plt.title("texte", fontsize=12)	Affiche "texte" comme titre de courbe et indique sa taille
plt.axis([x0, x1, y0, y1])	Limite l'affichage en abscisse sur $[x_0, x_1]$ et en ordonnées sur $[y_{0,1}]$ plt.axis("tight") permet de gérer le cadrage automatiquement plt.axis("equal") permet d'obtenir un repère orthonormé.
plt.xlim(a,b) plt.ylim(a,b)	Remplace l'instruction plt.axis() en indiquant l'intervalle $[a, b]$ utilisé pour le tracé en abscisses ou en ordonnées.
plt.text(x, y, "texte")	Permet d'afficher un texte sur la figure à la position (x, y)
plt.savefig("name")	Permet de sauvegarder la figure obtenue sous le nom "name".
plt.arrow(x, y, dx, dy)	Trace une flèche ayant comme point de départ (x, y) et allant jusqu'au point (x+dx, y+dy). Paramètres optionnels (mais indispensables en méca…) : color : "black", "red", "blue", etc. head_length : longueur de la tête de flèche, par défaut 0.0045 head_width : épaisseur de la tête de flèche, par défaut 0.003$

Exemple :

```
import matplotlib.pyplot as plt

x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
plt.plot(x, y, "+")
plt.arrow(x[3], y[3], 1, 4, color="black", head_width=0.2)
plt.show()
```

La bibliothèque numpy

La bibliothèque numpy est une bibliothèque permettant un traitement numérique rapide et efficace des données scientifiques. Elle donne accès à de nouveaux types de tableaux (multidimensionnels), notamment le type **array**, pratiques pour la gestion de vecteurs, le calcul matriciel et le travail sur les polynômes.

Instructions	Explication
import numpy as np	Permet d'importer la bibliothèque numpy et d'utiliser l'alias np
T = np.array([[a1, a2], [b1, b2], [c1, c2]])	Permet de construire un tableau T de type array en le définissant ligne par ligne : $\begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \\ c_1 & c_2 \end{pmatrix}$
T = np.array(L) **L = list(T)**	**Permet de transformer immédiatement une list L en tableau T de type array.** **Inversement, transforme un tableau de type array en liste.** **Remarque : on peut garder le même nom. L'instruction T=list(T) change la liste T en un tableau de même nom.**
np.linspace(a, b, n)	**Crée un tableau de n valeurs entre a et b (incluses), réparties de façon homogène.**
np.arange(a, b, h)	Crée un tableau de valeurs entre a et b (exclue), espacées d'un pas de h
T.size	Renvoie le nombre d'éléments contenus dans le tableau T
np.zeros(n)	Crée le tableau de taille n ne contenant que des 0
np.zeros([n,p])	Crée la matrice de taille $n \times p$ ne contenant que des 0
p = np.poly1d([an, …,a1, a0])	Affecte à la variable p le polynôme $(x) = a_n x^n + \cdots + a_2 x^2 + a_0$
np.roots([an, …,a1, a0]) *ou* np.roots(p)	Renvoie les racines du polynôme $(x) = a_n x^n + \cdots + a_2 x^2 + a_0 = 0$
np.polyfit(x, y, n)	**Renvoie un tableau contenant les coefficients du polynôme d'ordre n modélisant au mieux la fonction y = f(x).**
np.sqrt, np.cos, np.sin, np.tan, np.exp, np.log, np.pi, np.e, …	Les fonctions et les constantes mathématiques usuelles … **Ces fonctions (numpy) acceptent également des tableaux (list ou array) en argument**, *par exemple np.sqrt(L) pour calculer les racines carrées de tous les éléments de la liste L* !

Module	Fonction	explication
random	randint(a,b)	renvoie un entier choisi aléatoirement entre a et b inclus
	random()	renvoie un flottant choisi aléatoirement entre 0 (inclus) et 1 (exclu) (dans [0,1[)
	uniform(a,b)	renvoie un flottant choisi aléatoirement entre a et b inclus

Liens utiles :

https://pixees.fr/informatiquelycee/python_ISN_a1.html

https://lecluseo.scenari-community.org/co/Initiation_Python.html

https://pyspc.readthedocs.io/fr/latest/05-bases/index.html

Activités pour la seconde : https://pyspc.readthedocs.io/fr/latest/07-activites/02-seconde/index.html

Activités pour la 1ère : https://pyspc.readthedocs.io/fr/latest/07-activites/01-premiere/index.html

Activités pour la terminale: https://pyspc.readthedocs.io/fr/latest/07-activites/00-terminale/index.html

https://physique-chimie.enseigne.ac-lyon.fr/spip/spip.php?article1108

http://www.python-simple.com/python-langage/operations.php

Editeur de Python en ligne

- http://www.brython.info/tests/editor.html?lang=fr : mais pas de possibilité d'enregistrer son programme. (sauf copie dans un éditeur de texte)
- https://repl.it/languages/python3 : on peut enregistrer son code mais il faut créer un compte.