Mémento du langage de programmation « Python »

1. Les types de données

Type	Dénomination	Exemple de valeur	Conversion
Booléen	**bool**	true false *(vrai) (faux)*	
Nombre entier	**int**	2020	**int(**x**)**
Nombre entier long	**long**	5**16	
Flottant (Nombre décimaux)	**float**	2.72	**float(**x**)**
Complexe	**complex**	5-2j	
Texte (chaînes de caractères)	**string**	"bienvenu"	**str(**x**)**
Listes (ou tableaux)	**list**	[1,2,'coucou',-3.5]	

2. Les opérations de base sur les nombres

Syntaxe	Opération
x**+**y	Addition de *x* et *y*
x**-**y	Soustraction de *x* et *y*
x*****y	Multiplication de *x* et *y*
x**/**y	Division de *x* par *y*
x **%** y	Reste de la division de *x* par *y*
abs(x**)**	Valeur absolue de *x*
x******y	Puissance x^y
min(a,b**)**	Minimum entre a et b
max(a,b**)**	Maximum entre a et b
round(x**)**	Arrondi x à l'entier le plus proche
round(a,b**)**	Arrondi x avec b chiffres après la virgule

3. Affecter et Interagir avec l'utilisateur

Fonction ou méthode	Description
x **=** 1997	**Affecte** la valeur 1197 à la variable x
x **=** y	**Affecte** la valeur de la variable y à la variable x
x **=** x+1	**Incrémente** la valeur de la variable x de 1 **Attention**  Ici nous n'avons pas une égalité au sens mathématique mais une affectation de variable !
t = **input("**entrez texte**"))**	Demande à l'utilisateur d'entrer un texte et affecte le texte dans t
n = **int(input("**entrez n**"))**	Demande à l'utilisateur d'entrer un nombre entier et affecte la valeur dans n
x = **float(input("**entrez x**"))**	Demande à l'utilisateur d'entrer un nombre décimal et affecte la valeur dans x
print(x**)**	Affiche dans la zone d'exécution la valeur de x
print('BONJOUR**')**	Affiche le texte BONJOUR dans la zone d'exécution

4. Les opérateurs de comparaison

Opérateur	Comparaisons numériques
x **==** y	Est-ce que *x* est égal à *y* ?
x **!=** y	Est-ce que *x* est différent de *y* ?
x **>** y	Est-ce que *x* est strictement supérieur à *y* ?
x **<** y	Est-ce que *x* est strictement inférieur à *y* ?
x **>=** y	Est-ce que *x* est supérieur ou égal à *y* ?
x **<=** y	Est-ce que *x* inférieur ou est égal à *y* ?

Opérateur	Comparaisons d'objets
x **is** y	Est-ce que *x* et *y* représentent le même objet ?

Opérateur	Logique sur des variables de type booléen
`condition1` **`or`** `condition2`	Opérateur « OU » : Retourne `True` si au moins l'une des deux conditions est vérifiée
`condition1` **`and`** `condition2`	Opérateur « ET » : Retourne `True` si les deux conditions sont vérifiée
`not` `x`	Opérateur « CONTRAIRE » : Retourne la valeur booléenne contraire de *x*

5. Les tests de décision

Fonction ou méthode	Description
`if` `condition :` `    instruction`	Teste la condition. Si la condition est vérifiée, exécute la ou les instructions indentées

`if` `condition :`
`    instruction1`
`else :`
`    instruction2`
	Teste la condition. Si la condition est vérifiée, exécute la ou les instructions indentées 1, sinon exécute la ou les instructions indentées 2

`if` `condition1 :`
`    instruction1`
`elif` `condition2 :`
`    instruction2`
`else :`
`    instruction3`
	Teste la condition 1. Si la condition 1 est vérifiée, exécute la ou les instructions indentées 1, sinon teste la condition 2.
	Si la condition 2 est vérifiée, exécute la ou les instructions indentées 2 sinon exécute la ou les instructions indentées 3

6. Les boucles

Fonction ou méthode	Description
`while` `condition :` `    instruction(s)`	Exécute en boucle la ou les instructions indentés tant que la condition est vérifiée
`for` `variable` **`in range(`**`n`**`):`** `    instruction(s)`	Exécute en boucle la ou les instructions indentés pour une variable entière allant **de 0 à n–1**
`for` `variable` **`in range(`**`n,m`**`):`** `    instruction(s)`	Exécute en boucle la ou les instructions indentés pour une variable entière allant **de n à m–1**
`for` `variable` **`in range(`**`n,m,k`**`):`** `    instruction(s)`	Exécute en boucle la ou les instructions indentés pour une variable entière allant **de n à m–1** avec **un pas de k**
`for caractere in` `chaine:` `    instruction(s)`	Exécute en boucle la ou les instructions indentés pour chaque caractère de la chaîne de caractère `chaine`

7. Opérations sur les chaînes de caractères

Soit `str`, `str1` et `str2` trois chaînes de caractères

Fonction ou méthode	Description
`len(``str`**`)`**	Compte le nombre de caractère (espaces compris) dans la chaîne `str`
`str.`**`upper()`**	Met toute la chaîne `str` en majuscule
`str.`**`lower()`**	Met toute la chaîne `str` en minuscule
`str.`**`count(`**`str1`**`)`**	Compte les occurrences de `str1` dans `str`
`str.`**`find(`**`str1`**`)`**	Retourne l'indice (la position) où `str1` apparaît pour la première fois dans `str`
`str.`**`replace(`**`str1,str2`**`)`**	Remplace dans `str1` toutes les sous-chaînes `str1` par `str2`
`str`**`[i]`**	Retourne le caractère d'indice (de position) i dans la chaîne `str`
`str`**`[i:j]`**	Extrait de la chaîne `str` allant de l'indice i (inclus) à l'indice j (exclus)
`str1`**`+`**`str2`	Concatène (ajoute à la suite) les chaînes `str1` et `str2`
`str1`**`in`** `str`	Retourne `True` si `str1` est incluse dans `str` et `False` sinon

8. Opérations sur les listes

Soit `list` , `list1` et `list2` trois listes, `k` un nombre entier et `ob` un objet quelconque.

Fonction ou méthode	Description
`list = [1,2,3,4,5]`	Créer une liste de nombres
`list = ["a","ab","abc"]`	Créer une liste de mots
`list = []`	Créer une liste vide
`len(``list`**`)`**	Compte le nombre d'objets dans `list`
`list.`**`count(`**`ob`**`)`**	Compte le nombre d'occurrences de l'objet `ob` dans `list`
`list.`**`append(`**`ob`**`)`**	Ajoute l'objet `ob` à la fin de `list`
`list.`**`insert(`**`i,ob`**`)`**	Ajoute l'objet `ob` dans `list` à l'indice (position) i
`list.`**`remove(`**`ob`**`)`**	Enlève la première occurrence de `ob` dans `list`
`del` `list[k]`	Efface l'élément d'indice (de position) `k` de `list`
`list1`**`+`**`list2`	Concatène (ajoute à la suite) les listes `list1` et `list2`
`ob`**`in`** `list`	Retourne `True` si `ob` est inclus dans `list` et `False` sinon

9. Module de fonctions mathématiques : from math import *

Permet d'utiliser un grand nombre de fonctions mathématiques non disponibles par défaut dans le langage python. Module à importer en début de programme : **`from math import *`**

Fonction ou méthode	Description
`sqrt(``x`**`)`**	Renvoi $\sqrt{x}$
`pi`	Renvoi la valeur de π
`sin(``x`**`)`** **`cos(`**`x`**`)`** **`tan(`**`x`**`)`**	Renvoi les résultats du sinus, du cosinus et de la tangente
`asin(``x`**`)`** **`acos(`**`x`**`)`** **`atan(`**`x`**`)`**	Renvoi les résultats de l'arc sinus, de l'arc cosinus et de l'arc tangente
`log(``x`**`)`**	Renvoi ln(x) le logarithme népérien de `x`
`log10(``x`**`)`**	Renvoi log(x) le logarithme décimal de `x`
`exp(``x`**`)`**	Renvoi l'exponentielle de `x`
`e`	Renvoi la valeur de e
`hypot(``x,y`**`)`**	Renvoi $\sqrt{x^2+y^2}$
`degrees(``x`**`)`**	Convertit l'angle `x` de radians en degrés
`radians(``x`**`)`**	Convertit l'angle `x` de degrés en radians

10. Module de Dessin : from turtle import *

On peut créer des dessins à l'aide d'un module appelé turtle (tortue en anglais).
Module à importer en début de programme : **`from turtle import *`**

Fonction ou méthode	Description
`color(``"blue"`**`)`**	Détermine la couleur (écrite en anglais) du trait de dessin.
`Screen().bgcolor(``"red"`**`)`**	Détermine la couleur de l'arrière plan
`speed(``10`**`)`**	Règle la vitesse du tracé (avec un nombre entre 1 et 10, 10 étant la plus rapide)
`pensize(``4`**`)`**	Règle l'épaisseur du trait tracé
`forward(``50`**`)`**	Fait avancer la tortue de 50 pixels
`backward(``40`**`)`**	Fait reculer la tortue de 40 pixels
`right(``90`**`)`**	Fait tourner la tortue à 90° dans le sens des aiguilles d'une montre
`left(``120`**`)`**	Fait tourner la tortue à 120° dans le sens contraire des aiguilles d'une montre
`clear()`	Efface l'écran en gardant la position de la tortue
`reset()`	Efface l'écran et réinitialise la position de la tortue
`up()`	Lève le crayon pour déplacer sans tracer
`down()`	Abaisse le crayon
`circle(``30`**`)`**	Trace un cercle de rayon 30 pixels centré sur la tortue
`position()`	Retourne les coordonnées de la tortue

11. Module de Tirages aléatoires : from random import *

Module à importer en début de programme : **`from random import *`**

Fonction ou méthode	Description
`randint(``1,6`**`)`**	Tire au hasard un entier entre 1 et 6
`random()`	Tire au hasard un nombre entre 0 inclus et 1 exclus
`uniforme(``a,b`**`)`**	Tire au hasard un nombre entre `a` et `b`
`choice(``list`**`)`**	Tire au sort un élément dans la liste `list`

12. Module de Graphique : from matplotlib.pyplot import *

On peut créer des graphiques pour tracer des fonctions ou des nuages de points.
Module à importer en début de programme : **`from matplotlib.pyplot import *`**
Soit `list` une liste.

<table>
<tr><th>Fonction ou méthode</th><th>Description</th></tr>
<tr><td><code>axis(</code><code>xmin,xmax,ymin,ymax</code><code>)</code></td><td>Définit les dimensions du repère</td></tr>
<tr><td><code>grid(True)</code> ou <code>grid(False)</code></td><td>Affiche ou désaffiche le quadrillage</td></tr>
<tr><td><code>plot(</code><code>x,y,</code><code>options</code><code>)</code></td><td>Trace les points de coordonnées <code>x</code> et <code>y</code>
<code>options</code> est une chaîne de deux ou trois caractères donnant la couleur et le type de points :
<table>
<tr><th>Couleur</th><th>Style</th></tr>
<tr><td><code>b</code> bleu</td><td><code>-</code> ligne continue</td></tr>
<tr><td><code>g</code> vert</td><td><code>--</code> tirets</td></tr>
<tr><td><code>r</code> rouge</td><td><code>:</code> pointillées</td></tr>
<tr><td><code>c</code> cyan</td><td><code>.</code> des points</td></tr>
<tr><td><code>m</code> magenta</td><td><code>o</code> des billes</td></tr>
<tr><td><code>y</code> jaune</td><td><code>x</code> des croix</td></tr>
<tr><td><code>k</code> noir</td><td><code>v</code> des tria,gles</td></tr>
<tr><td><code>w</code> blanc</td><td><code>-.</code> tirets-points</td></tr>
</table>
Exemple :
<code>plot(</code><code>x,y,</code><code>'ro')</code> dessine des billes rouges</td></tr>
<tr><td><code>linspace(</code><code>xmin,xmax,nbpts</code><code>)</code></td><td>Créer une liste qui donne l'intervalle de variation d'une fonction mathématique et le nombre de points calculés.
Exemple pour tracer la fonction x^2 sur l'intervalle [–2 ; 3] avec 100 points calculés :
<code>x = linspace(-2,3,100)</code>
<code>y = x**2</code>
<code>plot(x,y)</code></td></tr>
<tr><td><code>bar(</code><code>x,y,1</code><code>)</code></td><td>Trace un diagramme en bâtons où la largeur de chaque bâton est de 1 (modifiable).</td></tr>
<tr><td><code>hist(</code><code>list</code><code>,bins=[</code><code>a,b,c,d</code><code>])</code></td><td>Trace un histogramme sur une série de données <code>list</code> dont les bornes des classes sont [a ; b[, [b ;c[et [c ; d[.</td></tr>
<tr><td><code>title(</code><code>texte</code><code>)</code></td><td>Ajoute le titre <code>texte</code> au graphique</td></tr>
<tr><td><code>xlabel(</code><code>texte</code><code>)</code></td><td>Ajoute le titre <code>texte</code> sur l'axe des abscisses</td></tr>
<tr><td><code>ylabel(</code><code>texte</code><code>)</code></td><td>Ajoute le titre <code>texte</code> sur l'axe des ordonnées</td></tr>
<tr><td><code>text(</code><code>x,y,texte</code><code>)</code></td><td>Affiche <code>texte</code> en (<code>x</code> ; <code>y</code>) sur le graphique</td></tr>
<tr><td><code>show()</code></td><td>Affiche le repère et les tracés</td></tr>
<tr><td><code>clf()</code></td><td>Efface et réinitialise la fenêtre graphique.</td></tr>
</table>

13. Créer une fonction

Fonction ou méthode	Description

`def` `nom(p1,p2)`
`    instruction(s)`
`    `**`return`** `résultat`
	Une fonction est un sous-programme qui porte un `nom` et qui peut utiliser plusieurs paramètres (`p1`,`p2`, ...) ou aucun.
	Le mot-clé **`return`** est obligatoire à la fin d'une fonction : il indique le résultat renvoyé par la fonction.
	Le résultat renvoyé par une fonction peut être réutilisé dans un autre programme ou une autre fonction.