

MEMENTO PYTHON

LES INSTRUCTIONS CONDITIONNELLES ET LES BOUCLES

```
if conditions1 :  
    instructions1  
elif conditions2 :  
    instructions2  
else :  
    instructions3
```

elif est la contraction de else if

$a < 10$ renvoie *True* si $a < 10$, *False* sinon
 $a > 10$ renvoie *True* si $a > 10$, *False* sinon
 $a \leq 10$ renvoie *True* si $a \leq 10$, *False* sinon
 $a \geq 10$ renvoie *True* si $a \geq 10$, *False* sinon

```
for variable in seq :  
    instructions
```

seq est une liste
(voir la fonction `range` dans OPÉRATIONS SUR LES LISTES)

```
while conditions :  
    instructions
```

while signifie tant que
Dès que les conditions ne sont plus vraies, la boucle s'arrête

LES FONCTIONS (PYTHON)

```
def joli(prenom) :  
    txt=prenom+' est joli(e)'  
    return txt
```

joli('johan') renverra la phrase 'johan est joli(e)'
joli('amelie') renverra la phrase 'amelie est joli(e)'

```
def g(x) :  
    if x<3 :  
        return 4x+5  
    else :  
        return x+2
```

Cette fonction renvoie l'image d'un nombre par la fonction g définie par $g(x) = \begin{cases} 4x+5 & \text{si } x < 3 \\ x+2 & \text{si } x \geq 3 \end{cases}$.
 $g(2)$ renverra donc 13 et $g(4)$ renverra 6

LES BIBLIOTHÈQUES DE PYTHON

```
from bibliothèque import fonction
```

importe la *fonction*

```
import bibliothèque as nom
```

importe la *bibliothèque* et la nomme *nom*
il suffit alors d'écrire *nom.fonction*

```
help('bibliothèque')
```

affiche l'aide de *bibliothèque*

```
from bibliothèque import *
```

importe toutes les fonctions de *bibliothèque*
mais ceci est déconseillé car certaines fonctions sont communes à plusieurs bibliothèques et ne donnent pas les mêmes résultats, comme *pow*

VARIABLES ET OPÉRATIONS

bibliothèque	fonction	explications
	input() input(message)	Affiche message et invite l'utilisateur à rentrer une valeur
	print(objet)	Affiche objet print(2+3) affiche 5 print("Hello world !") affiche la chaîne Hello world !
	a*b	Multiplication de a par b 2*3 renvoie 6 2*1/2 renvoie 1.0
	x**y ou pow(x,y)	Renvoie x ^y 2**3 renvoie 8 2**0,5 renvoie 1.4142135623730951 2**60 renvoie 1152921504606846976
	pow(x,y,m)	Renvoie x ^y modulo m pow(2,10,3) renvoie 1 car 2 ¹⁰ =1 024=3x341+1
math	pow(x,y)	Renvoie x ^y (résultat de type float) pow(2,3) renvoie 8.0 pow(2,60) renvoie 1.152921504606847e+18 Remarque : pow(2,10,3) n'existe pas → il faut alors écrire pow(2,10)%3 pour obtenir 1.0
	a/b	Quotient de a par b 1/3 renvoie 0.3333333333333333 4/2 renvoie 2.0
	a//b	Quotient de la division euclidienne de a par b
	a%b	Reste de la division euclidienne de a par b 14%3 renvoie 2 car 14=3x4+2 14%1.1 renvoie 0.7999999999999999 (au lieu de 0.8) 14%1.4 renvoie 8.881784197001252e-16 (au lieu de 0)
math	sqrt(x)	\sqrt{x} sqrt(2) renvoie 1.4142135623730951
	max(a,b,c)	Maximum de (a,b,c)
	min(a,b,c)	Minimum de (a,b,c)
	eval(expression)	eval("a") renvoie la valeur contenue dans la variable a
math	isnan(x)	Renvoie <i>True</i> si x n'est pas un nombre, et <i>False</i> sinon
	int(x)	Troncature de x int(12.958) renvoie l'entier 12 int(-2.57) renvoie l'entier -2
	int(chaîne)	Convertir chaîne en entier int("54") renvoie 54
	float(chaîne)	Convertir chaîne en nombre float("54.27") renvoie 54.27

Sur les nombres en virgule flottante (float), je vous conseille vivement de lire [cet article](#).
Faire très attention : par exemple 0.1+0.2 ne renvoie pas 0.3.

bibliothèque	fonction	explications
	round(x)	Arrondi entier de x round(12.958) renvoie l'entier 13 round(-2.57) renvoie l'entier -3
	round(x,nb)	Arrondi de x avec une précision de nb décimales round(12.958,2) renvoie 12.96 Attention : à cause des nombres en virgule flottante, certains résultats renvoyés peuvent être faux. Comme round(2.675,2) qui renvoie 2.67 au lieu de 2.68. Voir cet article .
	abs(x)	Valeur absolue de x
	str(x)	Renvoie une chaîne formée avec le nombre x str(25) renvoie "25"
	isinstance(objet,class)	Renvoie <i>True</i> si objet est du type class, et <i>False</i> sinon. isinstance(2,int) renvoie <i>True</i> car 2 est un entier isinstance(2.54,float) renvoie <i>True</i> car 2.54 est un réel
math	pi	La constante π à la précision disponible
math	ceil(x)	Arrondi supérieur de x (ceil = plafond) ceil(12,958) renvoie l'entier 13 ceil(-2,57) renvoie l'entier -2
math	floor(x)	Arrondi inférieur de x (floor = plancher) floor(12,958) renvoie l'entier 12 floor(-2,57) renvoie l'entier -3
math	cos(x), sin(x), tan(x)	Fonctions trigonométriques (mesures en radians)
math	degrees(x)	Convertit l'angle x de radians en degrés
math	radians(x)	Convertit l'angle x de degrés en radians
math	exp(x), ln(x)	Fonctions exp et ln
math	factorial(x)	Factorielle x!
math	log(x) ou log(x,e)	Renvoie ln(x)
random	randint(a,b)	Entier pseudo-aléatoire compris entre a et b inclus
random	random()	Nombre (float) pseudo-aléatoire dans [0;1[
random	uniform(a,b)	Renvoie un nombre compris entre a et b inclus
	lambda car : exp	Crée la fonction qui à car associe exp f=lambda x : x**2 crée la fonction f qui à x associe son carré et donc f(3) renvoie 9

LES LISTES

bibliothèque	fonction	explications
	<code>[]</code>	Crée une liste vide
	<code>seq*nb</code>	Agrandit la liste <code>seq</code> avec elle-même <code>[0,1]*3</code> renvoie <code>[0,1,0,1,0,1]</code> <code>[0]*5</code> renvoie <code>[0,0,0,0,0]</code>
	<code>seq[i]</code>	Renvoie le i-ème élément de la liste <code>seq</code> Si <code>l=['a','b','d','c']</code> alors <code>l[1]</code> renvoie 'b' Si <code>l='mathematiques'</code> alors <code>l[5]</code> renvoie 'm'
	<code>seq[i:j]</code>	Renvoie la liste contenant les éléments de la liste <code>seq</code> entre la position <code>i</code> (inclus) et la position <code>j</code> (exclu) Si <code>l=['a','b','d','c']</code> alors <code>l[1:3]</code> renvoie ['b','d'] Si <code>l='mathematiques'</code> alors <code>l[1:5]</code> renvoie 'athe'
	<code>seq[:]</code>	Crée une nouvelle liste identique à la liste <code>seq</code> Pour copier une liste <code>seq</code> , il faut donc écrire <code>seq2=seq[:]</code>
	<code>seq[-i:]</code>	Renvoie les <code>i</code> derniers éléments de la liste <code>seq</code> Si <code>l=['a','b','d','c']</code> alors <code>l[-2:]</code> renvoie ['d','c'] Si <code>l='mathematiques'</code> alors <code>l[-3:]</code> renvoie 'ues'
	<code>seq[:i]</code>	Renvoie les <code>i</code> premiers éléments de la liste <code>seq</code> Si <code>l=['a','b','d','c']</code> alors <code>l[:2]</code> renvoie ['a','b'] Si <code>l='mathematiques'</code> alors <code>l[:3]</code> renvoie 'mat'
numpy	<code>multiply(seq1,nb)</code>	Multiplie la liste <code>seq</code> par le nombre <code>nb</code>
	<code>len(seq)</code>	Renvoie le nombre d'éléments de la liste <code>seq</code>
	<code>range(a)</code>	Renvoie la liste de 0 à <code>a - 1</code> <code>range(3)</code> renvoie <code>[0,1,2]</code>
	<code>range(a,b)</code>	Renvoie la liste de <code>a</code> à <code>b - 1</code> <code>range(3,6)</code> renvoie <code>[3,4,5]</code>
	<code>range(a,b,c)</code>	Renvoie la liste de <code>a</code> à <code>b - 1</code> avec un pas de <code>c</code> <code>range(1,10,2)</code> renvoie <code>[2,4,6,8,10]</code>
<code>xrange(a)</code> peut être utilisé à la place de <code>range(a)</code> si <code>a</code> est trop grand (voir https://docs.python.org/2/library/functions.html#xrange)		
	<code>seq.append(elt)</code>	Rajoute l'élément <code>elt</code> à la liste <code>seq</code> Si <code>l=['a','b','d','c']</code> alors <code>l.append('e')</code> renvoie ['a','b','d','c','e']
numpy	<code>append(seq1,seq2)</code>	Renvoie la liste formée de <code>seq1</code> suivie de <code>seq2</code> <code>append([0,1],[3,4])</code> renvoie <code>[0,1,3,4]</code>
	<code>sorted(seq)</code>	Renvoie la liste <code>seq</code> triée dans l'ordre croissant (aussi bien pour des nombres que des chaînes) <code>sorted([1,25,8,12])</code> renvoie <code>[1,8,12,25]</code> <code>sorted(['a','d','c'])</code> renvoie ['a','c','d'] <code>sorted(['1','25','8','12'])</code> renvoie ['1','12','25','8']
	<code>sorted(seq,reverse=True)</code>	Renvoie une nouvelle liste <code>seq</code> triée dans l'ordre décroissant
	<code>seq.reverse()</code>	Trie la liste <code>seq</code> dans l'ordre décroissant
	<code>sum(seq)</code>	Renvoie la somme des valeurs de la liste <code>seq</code>
	<code>seq.count(car)</code>	Renvoie le nombre d'occurrences de <code>car</code> dans la liste <code>seq</code>

LES LISTES (SUITE)

bibliothèque	fonction	explications
	<code>enumerate(seq,start)</code>	<code>enumerate(["mauvais","moyen","bon"],0)</code> renvoie la liste [(0,"mauvais"),(1,"moyen"),(2,"bon")]
	<code>val in seq</code>	Renvoie <i>True</i> si la valeur <code>val</code> est dans la liste <code>seq</code> , <i>False</i> sinon
	<code>del seq[pos]</code>	Dans la liste <code>seq</code> , supprimer l'item à la position <code>pos</code>
numpy	<code>delete(seq,pos)</code>	Renvoie une nouvelle liste <code>seq</code> sans la valeur à la position <code>pos</code> <code>delete([0,1,2,3],1)</code> renvoie [0,2,3]
numpy	<code>insert(seq,elt,pos)</code>	Liste <code>seq</code> dans laquelle on a rajouté <code>elt</code> en position <code>pos</code> <code>insert([5,6,8,9],1,4)</code> renvoie [5,4,6,8,9]
numpy	<code>add(seq1,seq2)</code>	Additionne les deux listes <code>seq1</code> et <code>seq2</code> (de même taille) <code>add([1,5],[2,6])</code> renvoie [3,11]
numpy	<code>unique(seq)</code>	Renvoie la liste <code>seq</code> sans aucune répétition <code>unique([1,2,1,3,1,4])</code> renvoie [1,2,3,4]
numpy	<code>linspace(start,stop,num)</code>	Renvoie la liste de <code>start</code> à <code>stop</code> , avec <code>num</code> nombres <code>linspace(2,3,5)</code> renvoie [2.,2.25,2.5,2.75,3.] <code>linspace(2,3,5,endpoint=False)</code> renvoie [2.,2.2,2.4,2.6,2.8]
random	<code>choice(seq)</code>	Renvoie un élément aléatoire de la liste <code>seq</code> <code>choice([1,2,3])</code> renvoie 1, 2 ou 3
random	<code>shuffle(seq)</code>	Renvoie une liste formée par un pseudo-mélange de la liste <code>seq</code>
random	<code>sample(seq,k)</code>	Renvoie une liste pseudo-aléatoire de longueur <code>k</code> à partir de la liste <code>seq</code>

CHAÎNES DE CARACTÈRES

bibliothèque	fonction	explications
	<code>str.lower()</code>	Met la chaîne <code>str</code> en minuscules
	<code>str.upper()</code>	Met la chaîne <code>str</code> en majuscules
	<code>str.replace(car1,car2)</code>	Dans la chaîne <code>str</code> , remplace le caractère <code>car1</code> par <code>car2</code>
	<code>str.split(car)</code>	Renvoie une liste contenant les mots de la chaîne <code>str</code> séparés par le caractère <code>car</code> <code>"vive les maths".split('e')</code> renvoie ['viv', ' l', 's maths'] <code>"vive les maths".split(' ')</code> renvoie ['vive', ' les', 'maths']
	<code>str.count(car)</code>	Nombre de fois que la chaîne <code>str</code> contient le caractère <code>car</code> <code>"mathematiques".count('a')</code> renvoie 2
	<code>str.find(car)</code>	Renvoie la position dans la chaîne <code>str</code> où se trouve le caractère <code>car</code> (et renvoie -1 si <code>car</code> n'apparaît pas) <code>"mathematiques".find('a')</code> renvoie 1

```
for car in chaine :
    instructions
```

Fait prendre à la variable `car` toutes les valeurs des caractères de la chaîne `chaine`

```
liste = ["a","d","m"]
for val in enumerate(liste):
    print val
```

Affiche : (0, 'a')
(1, 'd')
(2, 'm')

GRAPHIQUES

Principales fonctions

bibliothèque	fonction	explications
matplotlib.pyplot	axis[v1,v2,v3,v4]	Définit les dimensions du repère
matplotlib.pyplot	grid()	Affiche le quadrillage
matplotlib.pyplot	plot(seq1,seq2)	seq1 est la liste des abscisses seq2 est la liste des ordonnées plot(seq1,seq2) trace les points de coordonnées associées
matplotlib.pyplot	show()	Permet l'affichage de la courbe

Un exemple :

```
import matplotlib.pyplot as plt
from numpy import linspace

x_list,y_list=[],[]
for x in linspace(-10,10,100000):
    x_list.append(x)
    y_list.append(x**2-3*x+x-1)

plt.plot(x_list,y_list)
plt.grid()
plt.savefig("fonction_poly.png")
plt.show()
```

