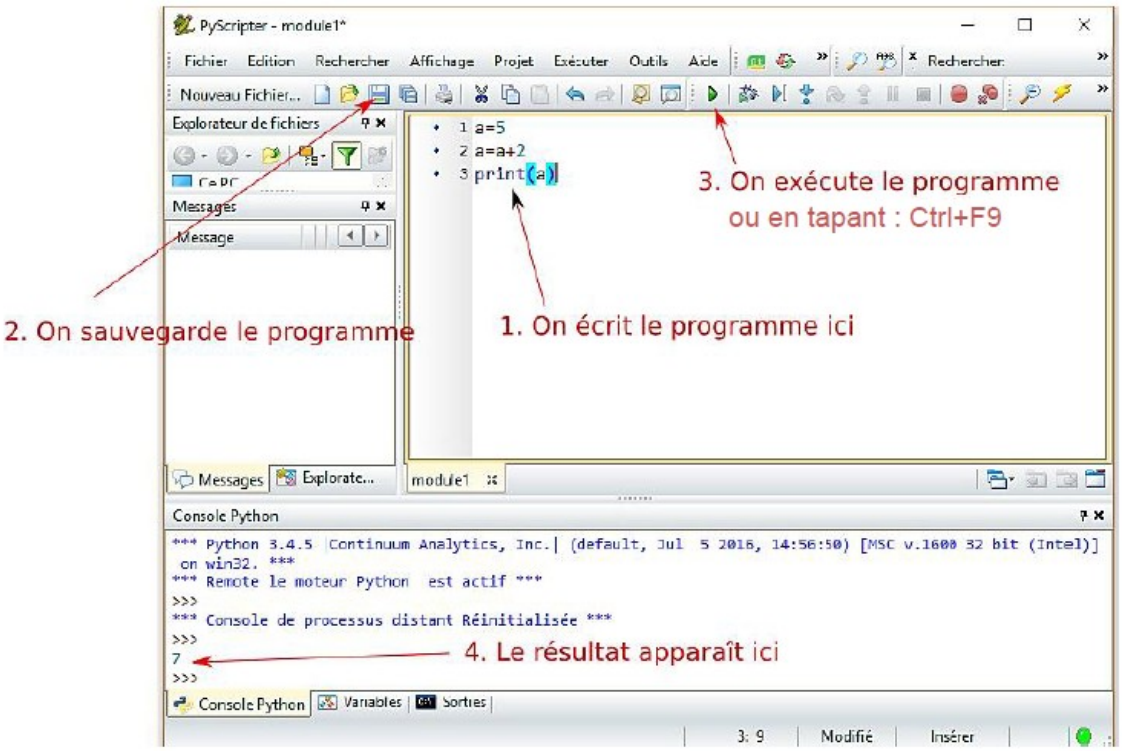
Les bases du langage Python

Python est un langage créé par Guido Van Rossum au début des années 90. Il est sous licence libre GPL, gratuit et fonctionne sur toutes les plateformes (Windows, Linux, OSX . . .). Python a été conçu pour être un langage lisible. Les commentaires sont indiqués par le caractère croisillon (#).

I. Editeur et console

Il existe plusieurs éditeurs Python. Voici une présentation de l'un d'entre eux : EduPython.



Des sites web proposent un éditeur en ligne comme https://trinket.io/python3 ou https://repl.it/languages/python3

II. Variable et entrée / sortie

II.1. Variable

Une **variable** est un emplacement dans la mémoire de l'ordinateur, auquel on donne un **nom** et qui va pouvoir stocker une **donnée**. L'**affectation** permet de donner une valeur à une variable. En Python, l'affectation se fait avec le signe « = ».

Le nom d'une variable peut contenir des lettres, des chiffres (mais pas comme premier caractère) et même des _

```
a = 10
a = a + 5
b = « Salut »
```

On lit : Dans la variable a, stocker le nombre 10
On lit : Dans a, stocker son ancien contenu + 5
On lit : Dans b, stocker le texte « Salut »

L'instruction *type(a)* renvoie le **type de la variable** a. Les plus fréquents sont : int (nombre entier), float (nombre réel), str (chaîne de caractère), bool (uniquement 2 valeurs possibles : vrai/oui ou faux/non). Python est un langage à **typage dynamique** (le type d'une variable dépend de la donnée stockée et peut donc changer après une nouvelle affectation).

Opérations	Interprétation	Exemples de syntaxe	Remarque
+, -, *, /	addition, soustraction, multiplication et division		
a//b	Partie entière de la division de a par b	> 12//11 1	$12 \div 11 \approx 1,0909$
a % b	Reste de la division euclidienne de a par b	> 17 % 3 2	$17 = 3 \times 5 + 2$
int(a)	partie entière	> int(12.123) 12	
divmod(a,b)	Quotient et Reste de la division euclidienne de a par b	> divmod(20,3) (6,2)	$20 = 3 \times 6 + 2$
a**b ou pow(a,b)	a Puissance b	> 2**3 ou pow(2,3) 8	$2^3 = 8$
a**(1/2)	Racine carrée $\sqrt{a}$	> 9**(1/2) 3	$\sqrt{9} = 3$
a**(1/n)	Racine $n^{\text{ième}}$ de a : $\sqrt[n]{a}$	> 27**(1/3) 3	$\sqrt[3]{27} = 3$
abs(x)	Valeur absolue de x : $\lvert x \rvert$	> abs(-5.2) 5.2	$\lvert -5.2 \rvert = 5.2$
round(a,n)	Arrondie de a à 10^{-n} près	> round(2.2563,2) 2.26	

On ne peut pas ajouter une donnée numérique (int ou float) avec une chaîne de caractère.

II.2. Entrée / sortie

***SORTIE** : Pour afficher du texte, on utilise la fonction print en mettant le texte entre guillemets.*

```
print(« Bonjour»)
```
Afficher « Bonjour ».

***ENTRÉE** : Pour poser une question à la personne qui est devant l'ordinateur et récupérer sa réponse, on utilise la fonction input*

```
p = input(« Quel est votre prénom ? »)
print(« Bonjour » , p)
age = int(input(«Choisir un nombre»))
```
Dans la variable p, stocker le prénom donné
Afficher « Bonjour » suivi de ce prénom
Dans age, stocker l'entier correspondant au nombre choisi

III. Test et boucle

III.1. Test

Pour effectuer un **TEST**, on utilise la commande if suivie du test puis du caractère **:**

Les instructions à faire uniquement si le test est vrai doivent ensuite être « indentées », c'est-à-dire décalées à droite grâce à la touche tabulation.

```
nb = int( input(«Choisir un nombre entier.»))
if nb > 5 :
   print(«Il est supérieur à 5.»)
else :
   print(«Inférieur ou égale à 5.»)
```

Dans la variable nb, stocker l'entier choisi.
Si cet entier est supérieur à 5 alors
    Afficher « Il est supérieur à 5. ».
Sinon
    Afficher « Inférieur ou égale à 5. ».

*Le mot clé **elif** est une contraction de **else** et de **if***

Pour tester une égalité, il faut utiliser un double égale.

III.2. Boucle

BOUCLE : La commande for .. *in* .. **:** *permet de récupérer les uns après les autres les éléments de la liste fournie après le in en effectuant à chaque fois les instructions « indentées ».*

```
for i in [2,5,8] :
   print(«Le double de »,i,« est », 2*i)
```

Pour chacun des nombres 2, 5 et 8
    Afficher le nombre et son double

range(début , fin , pas)

range(début , fin , pas) : Génère une liste d'entiers. Les paramètres *début* et *pas* sont optionnels.

- Dans l'intervalle [0 , fin[si un seul paramètre est renseigné.
  L = range(4) va créer la liste [0 , 1 , 2 , 3] de 4 termes, le premier sera L[0] = 0, le dernier L[3] = 3.
- Dans l'intervalle [début ; fin[si 2 paramètres sont renseignés.
  L = range(1 , 5) va créer la liste [1 , 2 , 3 , 4] le premier terme sera L[0] = 1 et le dernier L[3] = 4.
- Dans l'intervalle [début ; fin[mais de *pas* en *pas*, si les 3 paramètres sont renseignés.
  L = range(2 , 9 , 2) va créer la liste [2 , 4 , 6 , 8] .

while condition :

while condition :
Exécute une instruction ou un bloc d'instructions tant que la condition est vérifiée.
La boucle peut donc ne jamais être exécutée si, d'entrée la condition est Fausse (False) ou s'exécuter indéfiniment si la condition est toujours Vraie (True).

```
# Dans l'éditeur PYTHON

a=5
while a>0:
    a=a-1
    print(a)
```

```
# Dans la console PYTHON
4
3
2
1
0
```

Aide-mémoire Python

Variables	
`x = 1 + 2`	déclare une variable `x`, initialisée avec `1+2`
`x = x + 1`	incrémente la variable `x`
`x += 1`	une autre façon de l'écrire
Entrées-sorties	
`print(2 * x)`	affiche la valeur de `2 * x`
`print('texte')`	affiche la chaîne `texte`
`s = input('votre nom :')`	affiche `votre nom :` et attend une saisie au clavier, stockée dans la variable `s`
`n = int(input('entrez n :'))`	attend une saisie au clavier et l'interprète comme un entier, stocké dans la variable `n`
Entiers	
`n // 3`	le quotient de la division euclidienne de `n` par 3
`n % 3`	le reste de la division euclidienne de `n` par 3
`n == 17`	est-ce que `n` est égal à 17 ? (booléen)
`n != 17`	est-ce que `n` est différent de 17 ? (booléen)
Conditionnelle	

`if n > 100:`
`  n = n - 10`
`else:`
`  n = n + 11`
	si `n` est plus grand que 100
	alors lui retrancher 10
	sinon
	lui ajouter 11

Boucle sur les entiers	
`for n in range(0, 100):` `  print(n)`	parcourt tous les entiers de 0 inclus à 100 exclus la variable `n` prend successivement chaque valeur
Boucle tant que	
`while n < 1000:` `  n = 2 * n`	tant que `n` est plus petit que 1000 multiplier `n` par 2
Hasard	
`from random import randint`	on importe la fonction `randint` de la bibliothèque `random`
`randint(0, 10)`	un entier tiré au hasard entre 0 et 10 inclus

Turtle	
`from turtle import *`	on importe toutes les fonctions de la bibliothèque `turtle`
`goto(100,50)`	va au point de coordonnées (100, 50)
`forward(60)`	avance de la distance 60
`backward(40)`	reculedr de la distance 40
`left(45)`	pivote à gauche de 45 degrés
`right(120)`	pivote à droite de 120 degrés
`circle(10, 30)`	trace un arc de cercle de 30 degrés et de rayon 10
`up()`	lève le crayon (ne dessine plus)
`down()`	pose le crayon (dessine de nouveau)
`color(0.2, 0.2, 0.2)`	fixe la couleur, avec des composantes rouge/vert/bleu entre 0 et 1
`begin_fill()`	commence un tracé qui sera rempli
`end_fill()`	remplit ce qui a été tracé depuis le dernier appel à `begin_fill()`