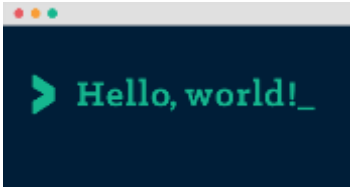
Programmation Python

Python est un langage puissant, à la fois facile à apprendre et riche en possibilités. Dès l'instant où vous l'installez sur votre ordinateur, vous disposez de nombreuses fonctionnalités intégrées au langage que nous allons découvrir

➢ Le programme

Communiquer avec un ordinateur demande un langage particulier : un langage de programmation (scratch, python, java etc…). Il utilise un ensemble de mots et de règles appelés « instruction » et « syntaxe » et forme des programmes informatiques.

Un **programme** est un ensemble d'opérations destinées à être effectuées par l'ordinateur. Une instruction correspond à une action.



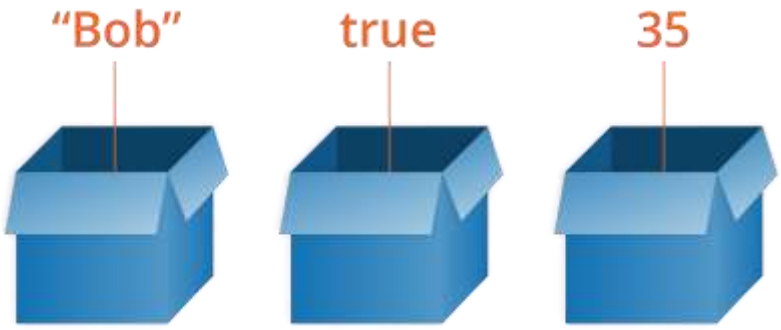
Exemple : **print** est une instruction qui permet d'afficher un message

```
print(« Hello world »)
```

➢ Les variables

Pour écrire un programme, il faut enregistrer les données nécessaires au traitement dans des espaces mémoires. Les variables désignent des emplacements de stockage. Dans un programme, elles sont repérées par des noms et prennent des valeurs qui peuvent évoluer au cours du temps

Une **variable** stocke une valeur dans un espace de la mémoire de l'ordinateur. Elle est désignée par un nom.



Des valeurs sont affectées aux variables pendant l'exécution du programme. En Python, on affecte une valeur à une variable de différents types comme :

- Des nombres entiers (**int**)
- Des nombres décimaux, appelés aussi « flottant » (**float**)
- Des textes, appelés « chaînes de caractères » ou « string » (**str**).

```
Entier1=2

Flottant=3.147

Entier2=int(2,13)
```

➢ Les instructions élémentaires

1. Les opérateurs sur les variables

A chaque variable est associé un ensemble d'opérations qui dépendent de son type (opération mathématique sur les nombres, longueur des chaînes de caractères, etc…)

Un **opérateur** permet d'effectuer des opérations ou d'agir sur les variables.

Remarque : les opérations sur les variables respectent les règles de priorité mathématiques.

Exemple : quelques opérateurs :

```
a = 4
b = 2
c = "Bonjour"
d  = "le monde"
somme=a+b
produit=a*b
cube=a**3
division=a/b
longueur=len(c)
phrase = c+d : concaténation de chaines de caractères
```

Le nom d'une variable est composé des lettres d'a à z, de A à Z, et des chiffres 0 à 9, mais il ne doit pas commencer par un chiffre.

Les lettres accentuées, les cédilles, les espaces, les caractères spéciaux tels que $, #, @, etc. sont interdits, à l'exception du caractère _ (souligné). Le tiret (-) est bien sûr interdit puisqu'il correspond aussi à la soustraction.

La casse est significative : toto et Toto sont des variables différentes !

Python compte 33 mots réservés qui ne peuvent pas non plus être utilisés comme noms de variable (ils sont utilisés par le langage lui-même) :

and	as	Assert	break	class	continue	def	del	yield.
elif	else	Except	False	finally	for	from	global	
if	import	In	is	lambda	None	nonlocal	not	
or	pass	Raise	return	True	try	while	with	

2. Les instructions d'entrée et de sortie

Une instruction d'entrée permet à un programme de lire des valeurs saisies au clavier par l'utilisateur. Une instruction de sortie affiche les valeurs de variables à l'écran.

En Python, l'instruction d'entrée « **input** » permet d'affecter la valeur saisie dans une variable. L'instruction de sortie « **print** » permet d'afficher à l'écran la valeur des variables.

Attention : l'instruction « **input** » permet d'obtenir des chaînes de caractères. Si on veut saisir un entier, il faut alors utiliser en plus l'instruction « **int** »

```
nombre=int(input(« nombre ? »)
```

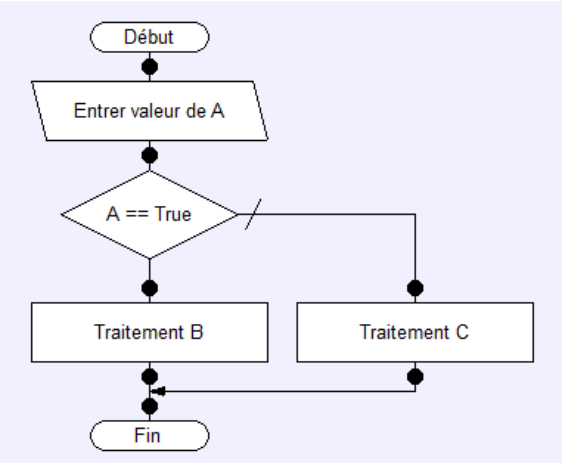
L'instruction print ne peut afficher que des variables de même type :

```
print ( a* b)    # affichera 8
print( a + c)  #affichera TypeError: unsupported operand type(s) for +: 'int' and 'str'
print ( str(a) +c ) # affichera 4Bonjour soit la concaténation des chaines a et c
```

➢ Les conditions

Un programme doit souvent prendre en compte différents cas et d'adapter à des situations. Il est alors nécessaire d'écrire des conditions.

Une **condition** est une expression logique dont le résultat est soit « **vrai** » soit « **faux** »




```
Programme nom_du_programme ;

##FORWARDCOMMENT##

DEBUTPROG
    Entrer valeur de A ;
    SI ( A == True ) ALORS
        Traitement B ;
    SINON
        Traitement C ;
    FINSI
FINPROG
```

Une condition est construite à l'aide d'opérateurs de comparaison :

- L'opérateur « égale » noté ==
- L'opérateur « différent de » noté != ou <>
- Les opérateurs « inférieur à » ou « supérieur à » notés < et >
- Les opérateurs « inférieur ou égal à » ou « supérieur ou égal à » notés <= et >=

Lorsque la situation à tester est plus compliquée, il est possible de combiner plusieurs conditions grâce aux opérateurs logiques :

- « **and** » qui signifie « et »
- « **or** » qui signifie « ou »
- « **not** » qui signifie « non »
  Exemple : une condition qui vérifie qu'une distance est inférieure à 40m et qu'un accès wifi est autorisé :

```
distance<40 and acces == « autorisé »
```

Syntaxe python

Une condition doit être suivie par deux points

Les instructions à exécuter si la condition est réalisée sont indentées, c'est-à-dire décalées vers la droite avec une tabulation.

➢ Les instructions conditionnelles

1. La structure conditionnelle « if »

Suivant la valeur d'une condition (vraie ou fausse), le programme choisit les actions à réaliser. On parle de structures conditionnelles.

La structure conditionnelle « **if** » permet d'exécuter un bloc d'instruction lorsqu'une condition est vérifiée.

Exemple : programme qui affiche l'autorisation d'accès d'un membre à un groupe privé.

```
if membre == « oui » :
      print(« accès autorisé »)
```

2. La structure conditionnelle « if…else »

La structure conditionnelle « **if…else** » permet d'exécuter un bloc d'instructions lorsqu'une condition est vérifié et un autre bloc lorsqu'elle ne l'est pas

Exemple : On affiche maintenant l'interdiction au groupe en cas de réponse fausse :

```
if membre == « oui » :
      print(« Accès autorisé »)
else :
      print(« Accès refusé »)
```

3. La structure conditionnelle « if…elif…else »

La structure conditionnelle « **if…elif…else** » permet de gérer plusieurs conditions. Si une condition n'est pas validée, la suivant est étudié. En Python, « **elif** » est contraction de **else** if qui signifie « *sinon si* »

Exemple : Programme qui affiche les préférences de confidentialité d'un groupe

```
if statut == « secret » :
      print(« seuls les membres voient le groupe »)
elif statut == « fermé » :
      print(« tout le monde voit le groupe mais pas les publications »)
else :
      print(« tout le monde voit le groupe et les publications »)
```

4. Les principaux opérateurs Logiques

Symbole	Signification littérale	Symbole	Signification littérale
<	Strictement inférieur à	!=	Différent de
>	Strictement supérieur à		
<=	Inférieur ou égal à	not	Non
>=	Supérieur ou égal à	and	Et (variante &)
= =	Egal à	or	Ou (variante |)