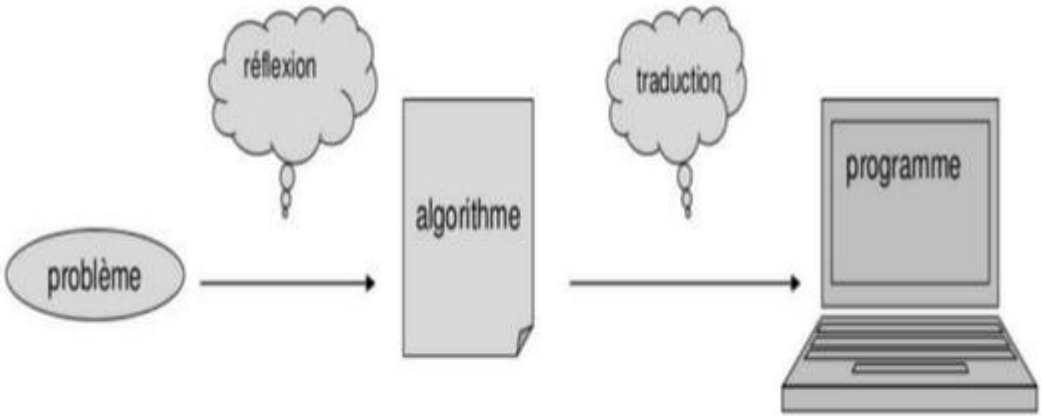

python™

La programmation



Présentation

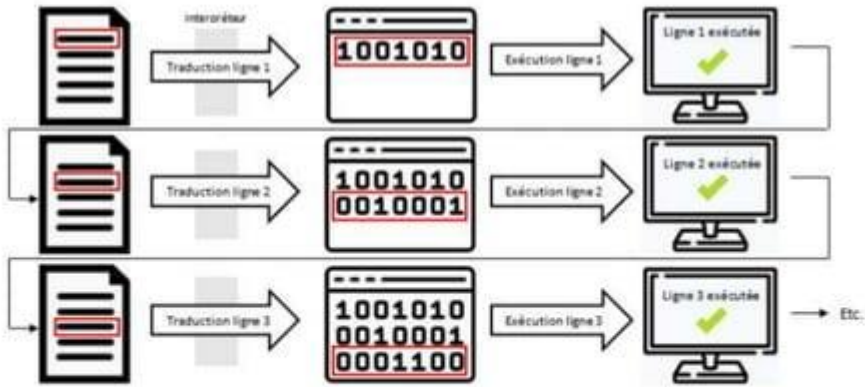
- Python est un langage de programmation qui a été créé en 1989 par Guido Van Rossum Au pays bas

- Le nom Python vient d'un hommage à la série télévisée Monty Python flying circus dont G.V.Rossem est fan

Présentation

Python est un langage interprété

- **Les instructions sont traduites en langage machine (0 et 1) au fur et à mesure de leur lecture**



Variables

- Une **variable** est une **zone mémoire** dans laquelle on stocke une **donnée**
- Chaque variable porte **un nom**
- **La création** d'une variable et son **initialisation** se font en **même temps**
- **age=42** **nom="Bob"**

Variables

- **Le nom d'une variable :**

  - Ne doit contenir que : **(a-z) (A-Z) (0-9) (_)**
  - Doit commencer par : **une lettre (a-z) (A-Z) ou soulignement (_)**
  - Ne doit pas commencer par : **un chiffre (0-9)**
  - Ne doit pas être un un mot réservé de **Python:**

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
sync	elif	if	or	yield

Le nom d'une variable est sensible à la casse;

Nom Nom nOm nOM sont quatre variables différentes

les types de variables

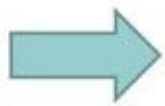
- Le type d'une variable correspond à la nature de celle-ci
- Les principaux types sont:
  - Entier (**int**)
  - Réel (**float**)
  - Logique (**bool**)
  - Chaine de caractères (**string**)
  - Liste (**list**)
  - Dictionnaire (**dict**)
  - Classe (**class**)
  - .....

les types de variables

La fonction type(variable) indique le type d'un' variable

- type (4)----------→int
- type (5.75)----------→float
- type (True)----------→bool
- type('bonjour')----------→str
- type(['singe','chat','chien'])----------→list
- type(([1:'un',2:'deux',3:'troix'])----------→dict
- .....

```
print(type(4))
print(type(5.75))
print(type(True))
print(type('bonjour'))
print(type(['singe','chat','chien']))
print(type({1:'un',2:'deux',3:'trois'}))
```



```
<class 'int'>
<class 'float'>
<class 'bool'>
<class 'str'>
<class 'list'>
<class 'dict'>
```

les types de variables

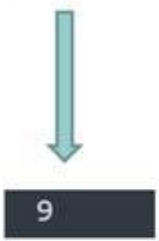
La fonction type(variable) indique le type d'un' variable

Pour Python, la valeur **2** (**nombre entier**) est différente de **2.0** (**réel** ***float***) et est aussi différente de **'2'** (**chaîne de caractères**).

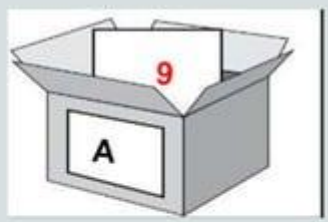
Les opérations sur les numériques

L'affectation

```
#affecter 9 à la variable A
A=9
print(A)
```



RAM

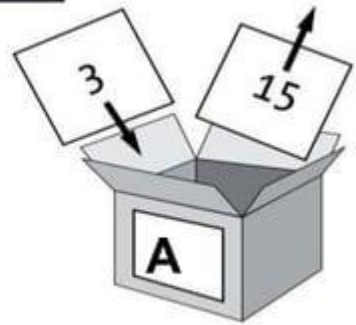


Les opérations sur les numériques

L'affectation : écraser la valeur

```
#affecter 15 à A
A=15
#ecrser la valeur de A par l'affectation de 3
A=3
print(A)
```

3



Les opérations sur les numériques

L'affectation : incrémentation

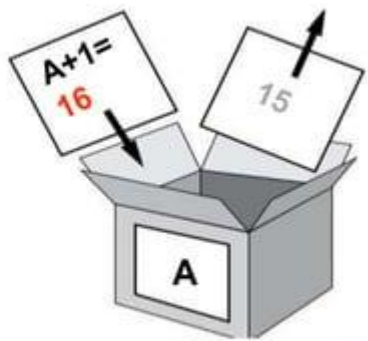
```
#affecter 15 à A
A=15
#augmenter la valeur de A par l'ajout de 1
A=A+1                #on peut aussi écrire     A+=1
print(A)
```

```
A=15

A+=1

print(A)
```

16

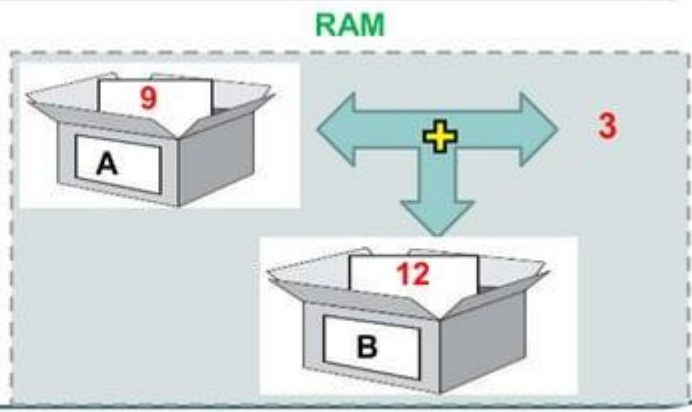


Les opérations sur les numériques

L'addition

```
#affecter 9 à la variable A
A=9
#ajouter 3 à la valeur de a ,puis affecter le résultat à B
B=A+3
print(B)
```

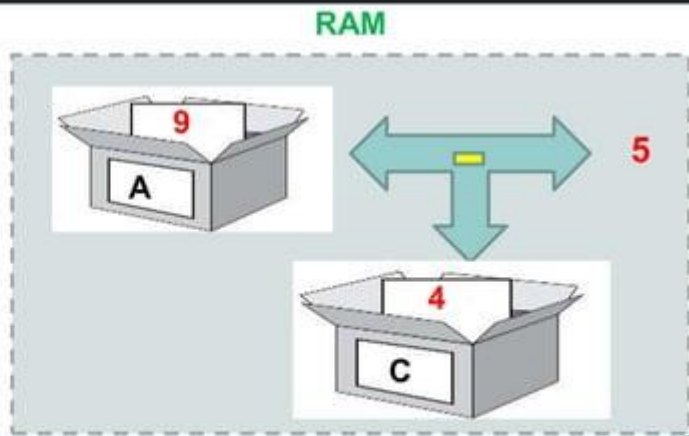
12



Les opérations sur les numériques

La soustraction

```
#affecter 9 à la variable A
A=9
#soustraire 5 de la valeur de A, puis affecter le résultat à C
C=A-5
print(C)
```

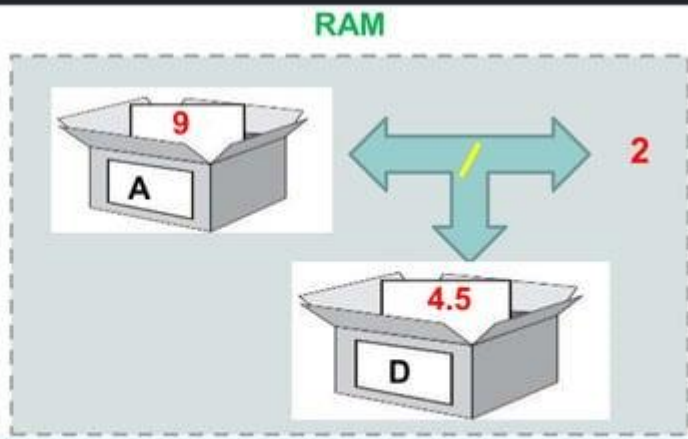


Les opérations sur les numériques

La division réelle

```
#affecter 9 à la variable A
A=9
#diviser la valeur de A par 2 (division réelle) , puis affecter le résultat à D
D=A/2
print(D)
```

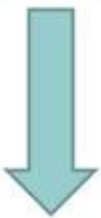
```
4.5
```

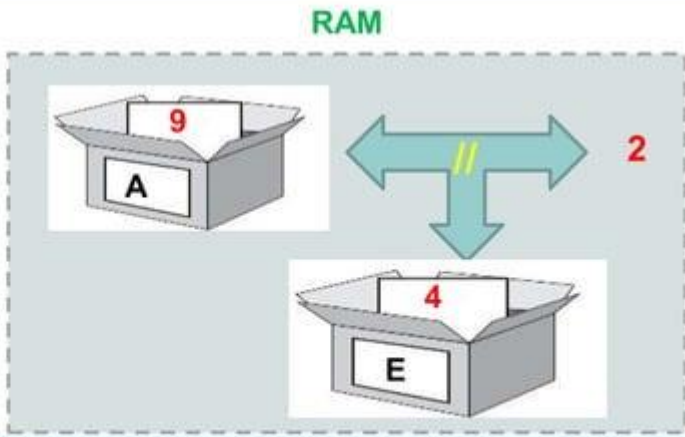


Les opérations sur les numériques

La division Euclidienne

```
#affecter 9 à la variable A
A=9
#diviser  A par 2 (division Euclidienne) puis affecter le résultat à D
E=A//2
print(E)
```



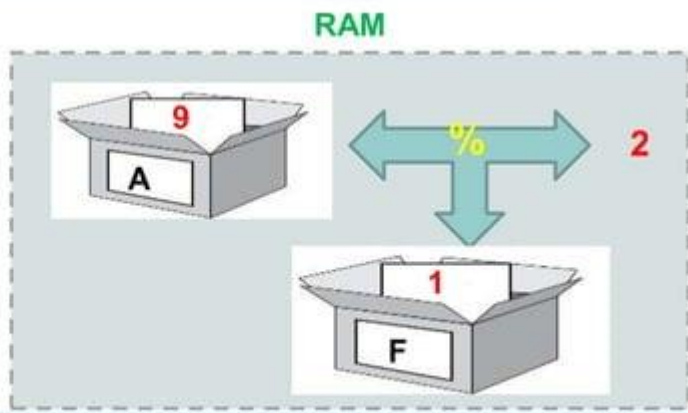


Les opérations sur les numériques

Le reste (modulo)

```
#affecter 9 à la variable A
A=9
#affecter le reste de la division de 9 par 2 à F
F=A%2
print(F)
```

1

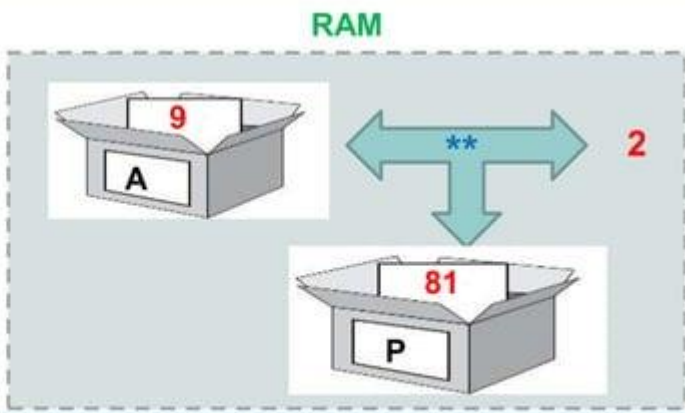


Les opérations sur les numériques

La puissance

```
#affecter 9 à la variable A
A=9
#elever la valeur de A à la puissance 2, puis affecter le résultat à P
P=A**2
print(P)
```

81



Chaîne de caractères

C'est une suite de caractères mis entre :

- **guillemets simples :** `'...'`
- **guillemets doubles :** `"..."`
- **Triples guillemets** `'''...'''` ou `"""..."""` **pour mettre une chaîne de caractères sur plusieurs lignes**

```
nom='Alami chami'
prenom="Mohamed Amine"
classe='Tronc commun 9'
adresse='''
quartier Narjiss rue La Roche N° B44
Casablanca  - Maroc -
'''
```

Chaîne de caractères

On peut mettre des guillemets simples ou doubles à l'intérieur d'une chaine comme ceci:

```
message="j'ai dit que:'Python est mon langage préféré' "
```

```
message='Le langage "Python" porte le nom de Monty Python'
```

```
message='Le langage \'Python\' porte le nom de Monty Python'
```

```
message="Le langage \"Python\" porte le nom de Monty Python"
```

Chaîne de caractères

Concaténer des chaines de caractères

```
'py'+'thon'
```

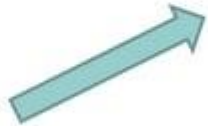
```
'python'
```

```
nom_complet='jamal'+'alami'
print(nom_complet)
```

```
prenom='jamal'
nom='alami'
nom_complet=prenom+nom
print(nom_complet)
```

```
jamalalami
```

```
nom='alami'
nom_complet='jamal'+nom
print(nom_complet)
```



Chaîne de caractères

Changer la casse d'une chaine de caractères:

```
nom='guido van rossum'
print(nom.title())
```

```
Guido Van Rossum
```

```
nom='guido van rossum'
print(nom.upper())
```

```
GUIDO VAN ROSSUM
```

```
nom='GUIDO VAN ROSSUM'
print(nom.lower())
```

```
guido van rossum
```

Lecture/Ecriture

- **Lecture:**
  - **Syntaxe:**

    input()
  - **Exemple:**

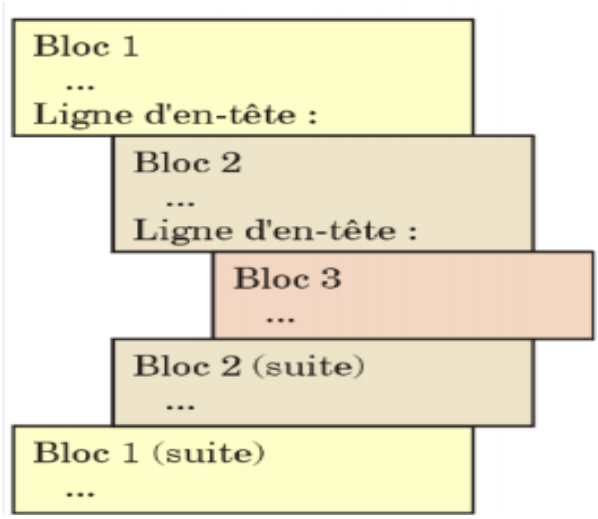
    X=float(input(″donner un nombre réel: ″))

- **Ecriture:**
  - **Syntaxe:**

    print(″message ″)

Indentation

- Python utilise l'indentation du code avec des caractères blancs plutôt que des mots clés (debut / fin)



Choix

- **Syntaxe:**

```
if conditions :
    blocs d'instructions
```

- **Exemple:**

```
a = -150
        if a <0:
            print ('a est négatif')
```

Choix

- **Choix double:**
  - **Syntaxe:**

```
if conditions :
        blocs d'instructions 1
else:
        blocs d'instructions 2
```

  - **Exemple:**

```
if a <0:
        print ('a est négatif')
else:
        print('a est positif ')
```

Choix

- **Choix Multiple:**
  - **Syntaxe:**

```
if conditions1 :
        blocs d'instructions 1
elif condiotions2 :
        blocs d'instructions 2
elif conditions I :
...
else:
    autres instructions
```

choix

- **Exemple de Choix Multiple:**

```
a = 10.
 if a > 0:
     print( 'a est strictement positif' )
     if a >= 10:
             print( 'a est un nombre')
        else:
             print( 'a est un chiffre')
     a += 1
elif a is not 0:
     print( 'a est strictement négatif')
else:
     print( 'a est nul' )
```

boucles

- **While:**
  - **syntaxe:**

```
while conditions:
        instructions
```

  - **Instructions particuliers:**

break : sort de la boucle
continue : remonte au début de la boucle,
pass : ne fait rien,

boucles

- **While: exemple: y est-il premier ?**

```
nombre = input("Écris un nombre entier positif : ")
nombre = int(nombre)
i = 2
while i < nombre and nombre % i != 0:
        i = i + 1
        if i == nombre:
                print("Le nombre", nombre, "est premier ! Fantastique !")
        else:
                print("Ce n'est pas un nombre premier.")
```

boucles

- For:
  - Syntaxe:

```
for cible in séquence d'objets:
        bloc instructions
```

- range:
  - Syntaxe

```
range(start,stop+1,step)
```

  - Exemple:

range(6) donne la séquence(liste) [0,1,2,3,4,5]
range(1,6)----->[1,2,3,4,5]
range(1,6,3)----->[1,4]

Boucles

- For: Exemple:

```
prod = 1
for p in range(1, 10):
        prod *= p
        print(prod)
```

- **Exécution:**

1
2
6
24
120
720
5040
40320
362880

Liste
Afficher une liste

```
#création d'une liste d'animaux
animaux=['singe','lion','tigre','souris','girafe']
#afficher une liste
print(animaux)
```

```
['singe', 'lion', 'tigre', 'souris', 'girafe']
```

Liste
Accéder à un élément

- pour accéder à un élément de la liste on le référence par son indice (index)

```
animaux=['singe','tigre','lion','girafe']
```

indices	0	1	2	3
indices	-4	-3	-2	-1

```
print(animaux[0])
print(animaux[2])
print(animaux[-3])
```

→ singe

→ lion

→ lion

Liste
Récupérer l'indice d'un élément

```
1   animaux=['singe','tigre','lion','girafe','tigre']
2
3   print("l'indice de 'tigre' est :",animaux.index('tigre'))
4
5   print("l'indice de 'girafe' est :",animaux.index('girafe'))
```

```
l'indice de 'tigre' est : 1
l'indice de 'girafe' est : 3
```

Liste

Modifier un élément de la liste

```
1   animaux=['singe','tigre','lion','girafe','tigre']
2
3   animaux[3]='gazelle'
4
5   print(animaux)
```

```
['singe', 'tigre', 'lion', 'gazelle', 'tigre']
```

Liste
Ajouter un élément à la liste

La fonction append() permet d'jouter un éléments à la fin de la liste

```
1   notes=[10,12,15,17,13,20]
2
3   notes.append(14) #ajout de la note 14 à la liste
4
5   print(notes)
```

```
[10, 12, 15, 17, 13, 20, 14]
```

Liste
Insérer un élément à la liste

La fonction insert (indice , élément) permet d'insérer un élément à une position donnée

```
1   eleves=['sami','siham','sara','yahia','hicham']
2
3   eleves.insert(3,'kamal')
4
5   print(eleves)
```

```
['sami', 'siham', 'sara', 'kamal', 'yahia', 'hicham']
```

Liste
Insérer un élément à la liste

La fonction insert (indice , élément) permet d'insérer un élément à une position donnée

```
1   eleves=['sami','siham','sara','yahia','hicham']
2
3   eleves.insert(3,'kamal')
4
5   print(eleves)
```

```
['sami', 'siham', 'sara', 'kamal', 'yahia', 'hicham']
```

Liste
supprimer un élément par son indice

La fonction pop (indice) permet de supprimer un élément par son indice.

```
1   eleves=['sami','siham','sara','yahia','hicham']
2
3   eleves.pop(3) #supprime l'element "yahia"
4
5   print(eleves)
```

```
['sami', 'siham', 'sara', 'hicham']
```

Liste
supprimer le dernier élément

pop() : supprime le dernier élément,

```
1   eleves=['sami','siham','sara','yahia','hicham']
2
3   eleves.pop()    #supprime l'element "hicham"
4
5   print(eleves)
```

```
['sami', 'siham', 'sara', 'yahia']
```

Liste
supprimer un élément par sa valeur

remove(x) : Supprime de la liste le premier élément dont la valeur est égale à *x*

```
1   eleves=['sami','siham','sara','yahia','hicham']
2
3   eleves.remove('sara')    #supprime l'element "hicham"
4
5   print(eleves)
```

```
['sami', 'siham', 'yahia', 'hicham']
```

Liste
supprimer une sous liste

del: supprime permet de supprimer une sous liste

```
1   nombres=[7,4,-2,-3,-8,9,16]
2
3   del nombres[2:5]    #supprime [-2,-3,-8]
4
5   print(nombres)
```

```
[7, 4, 9, 16]
```

Liste
vider une liste

```
1   nombres=[7,4,-2,-3,-8,9,16]
2
3   del nombres[:]    #supprime [7,4,-2,-3,-8,9,16]
4
5   print(nombres)
```

```
[]
```