

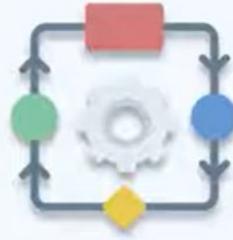
# Plan de cours



Instructions  
de base



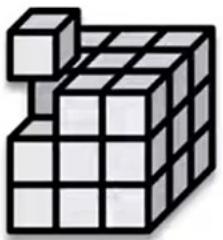
Structures  
conditionnelles



Structures  
répétitives



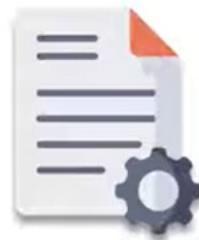
Fonctions  
et modules



Structures  
de données



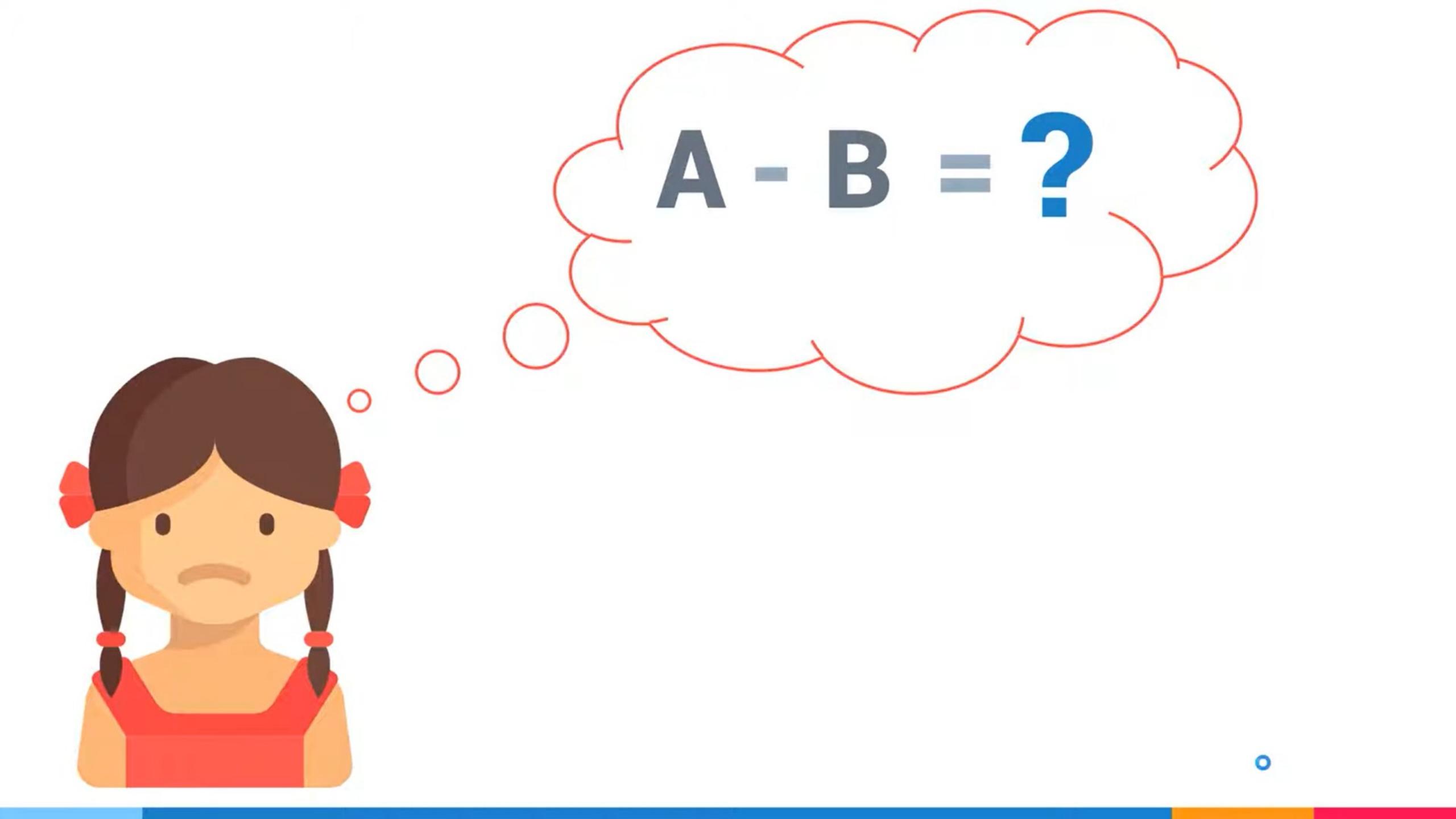
Chaine  
de caractères



Gestion  
des fichiers



Concepts  
avancés


$$A - B = ?$$

# Programme soustraction

```
def soustraction( A , B ):  
    C = A - B  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A , B )
```

# Programme soustraction

```
def soustraction(A, B):  
    C = A - B  
    print("A - B = ", C)  
  
A = float(input("Saisir la valeur de A : "))  
B = float(input("Saisir la valeur de B : "))  
soustraction(A, B)
```

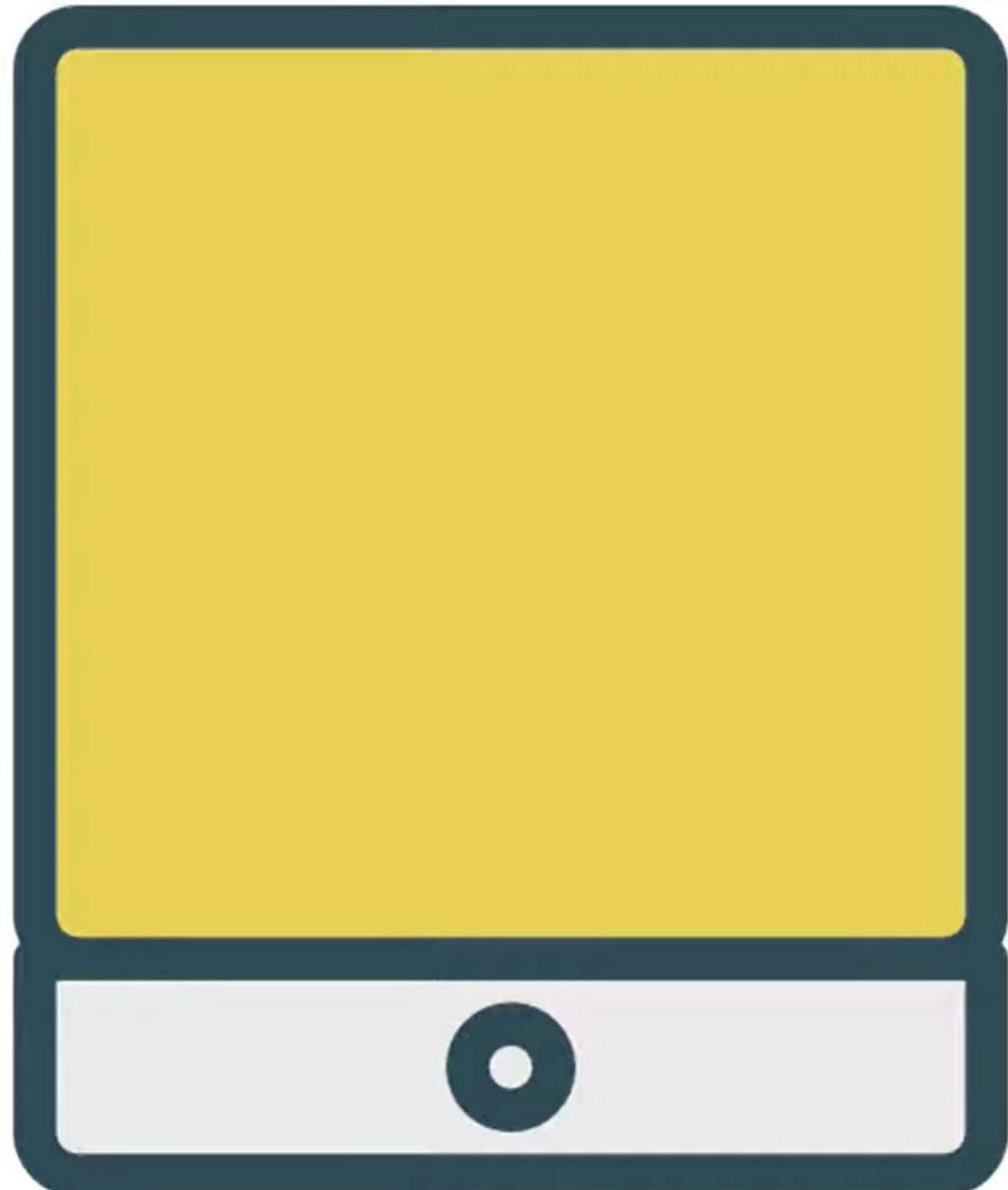
Saisir la valeur de A : 6

Saisir la valeur de B : 2

A - B = 4

# Programme soustraction

```
def soustraction( A , B ):  
  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( B , A )
```



# Programme soustraction

```
def soustraction(A, B):  
    C = A - B  
    print("A - B =", C)  
  
A = float(input("Saisir la valeur de A :"))  
  
B = float(input("Saisir la valeur de B :"))  
  
soustraction(B, A)
```

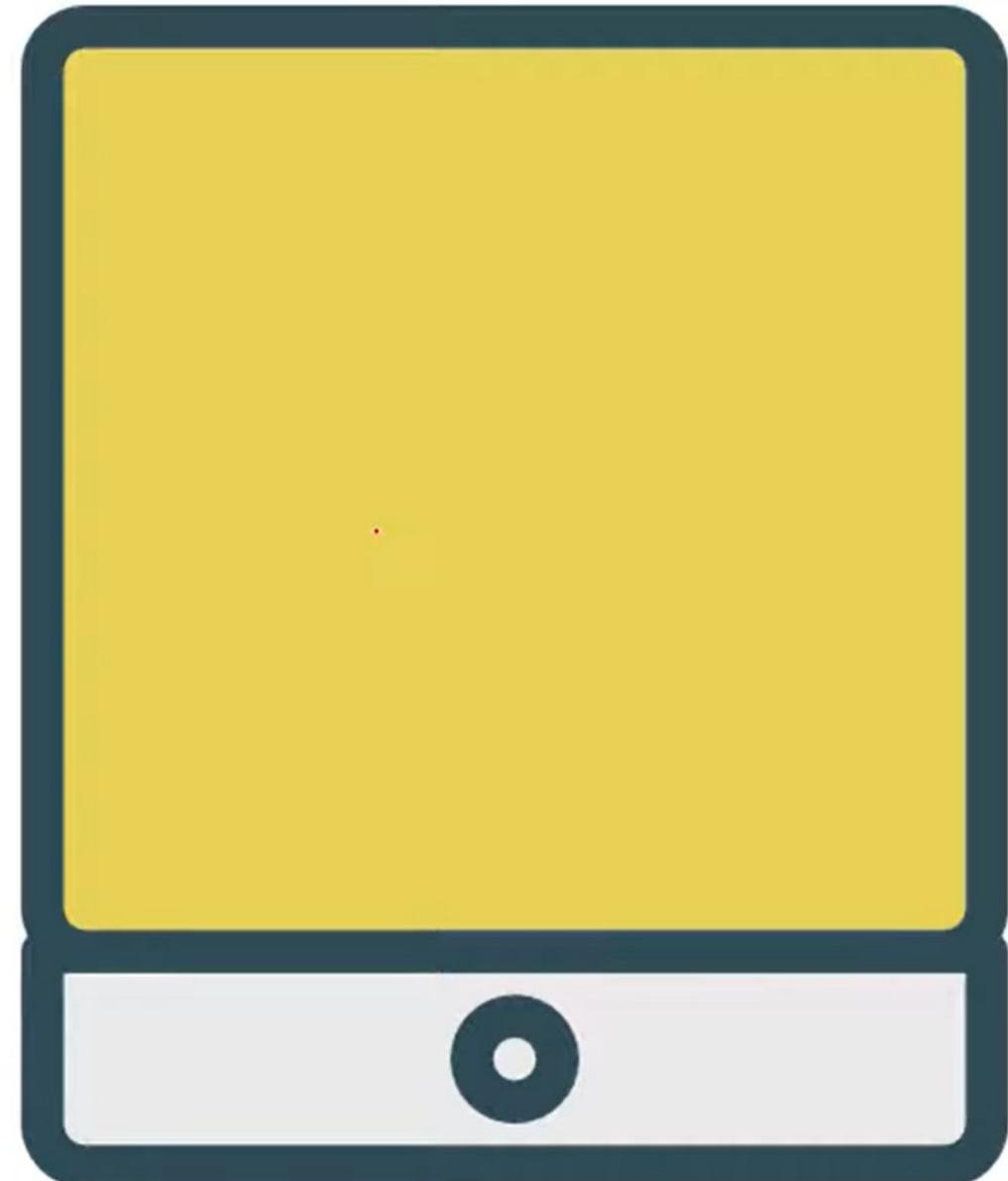
Saisir la valeur de A : 6

Saisir la valeur de B : 2

A - B = -4

# Programme soustraction

```
def soustraction( A , B ):  
  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A )
```



```
1 def soustraction(A, B):
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction(A)
```

⚠ 1 ⚠ 4 ✅ 4

run scratch\_1 ×



Entrez B : 3

Traceback (most recent call last): ⚡ Explain with AI

File "C:\Users\LENOVO\AppData\Roaming\JetBrains\PyCharmCE2025.2\scratches\scratch\_1.py", line 8, in <module>  
 soustraction(A)

TypeError: soustraction() missing 1 required positional argument: 'B'

# Programme soustraction

```
def soustraction( A , B ):  
    C = A - B  
    .  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( )
```

Saisir la valeur de A : 6

Saisir la valeur de B : 2

Error: soustraction() missing 2 required positional arguments:  
'A' and 'B'

scratch\_1.py ×

```
1 def soustraction(A, B):
2     résultat = A - B
3     print(f"{A} - {B} = {résultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction()
```

⚠ 2 ⚠ 4 ✘ 4 ⌂ ⌃

run scratch\_1 ×

Entrez B : 3

Traceback (most recent call last): Explain with AI

File "C:\Users\LENOVO\AppData\Roaming\JetBrains\PyCharmCE2025.2\scratches\scratch\_1.py", line 8, in  
 soustraction()

TypeError: soustraction() missing 2 required positional arguments: 'A' and 'B'

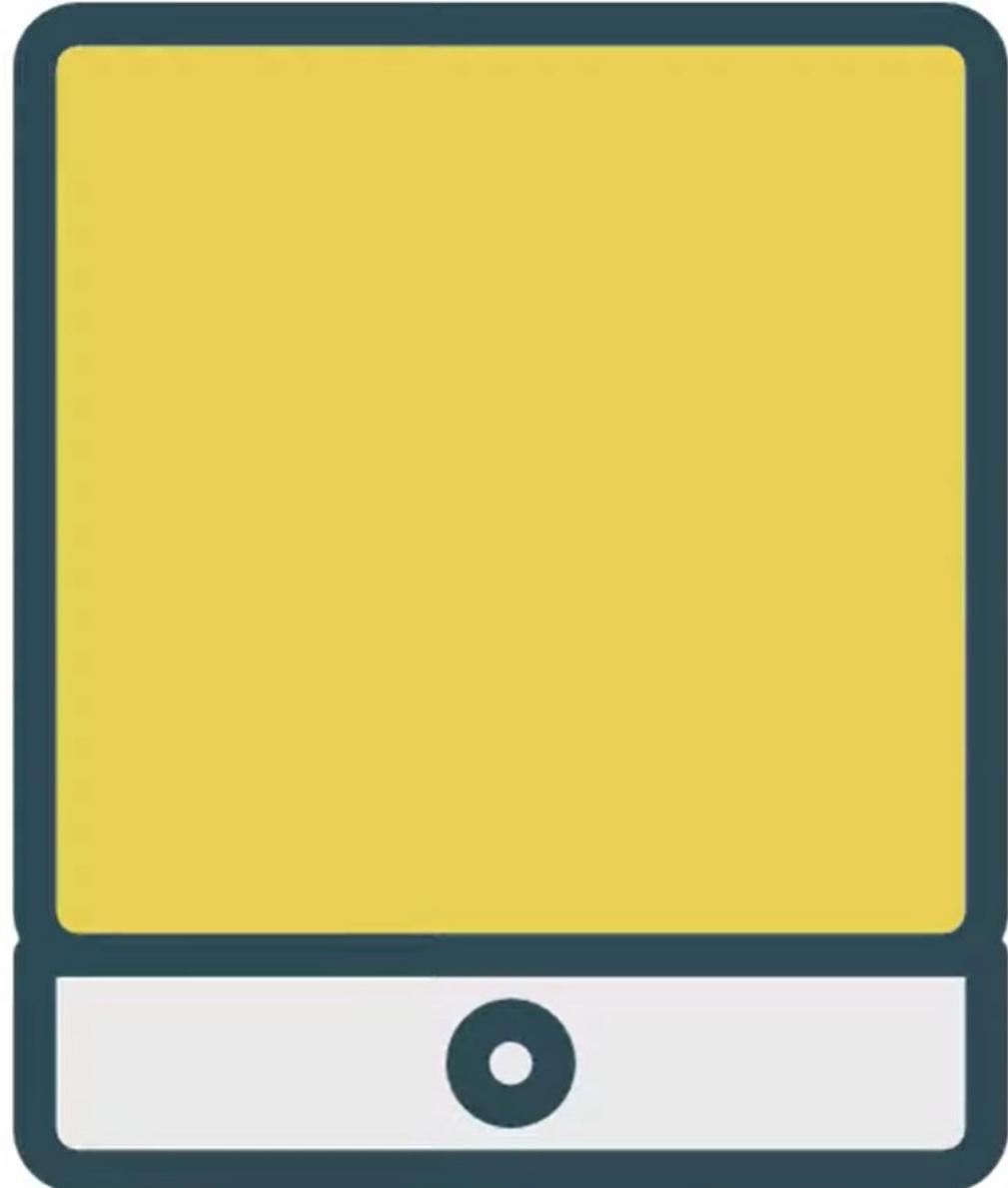
# Typage des paramètres

Vous pouvez appeler une fonction en utilisant l'un des types d'arguments formels suivants:

- Arguments requis.
- **Arguments par défaut**
- **Arguments avec étiquettes (mot-clé)**

# Arguments par défaut

```
def soustraction( A , B = 0 ):  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A , B )
```



```
1 def soustraction(A , B = 0) :
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction(A, B)
```

scratch\_1 ×

⋮

```
D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming\Python\Python310\site-packages\IPython\core\displayhook.py:51: UserWarning: To show the output of a code cell, use the IPython display() function or one of its variants like display.HTML(). See https://ipython.readthedocs.io/en/stable/api/generated/IPython.display.html for more information.
  warnings.warn(_displayhook_deprecation_message, UserWarning)
Entrez A : 6
Entrez B : 2
6 - 2 = 4
```

```
1 def soustraction(A , B = 0) :
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction(A)
```

Run scratch\_1 ×

⟳ | :

```
D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming\Je
Entrez A : 2
Entrez B : 4
2 - 0 = 2
```

# Arguments par défaut

```
def soustraction( A , B = 0 ):  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A )
```

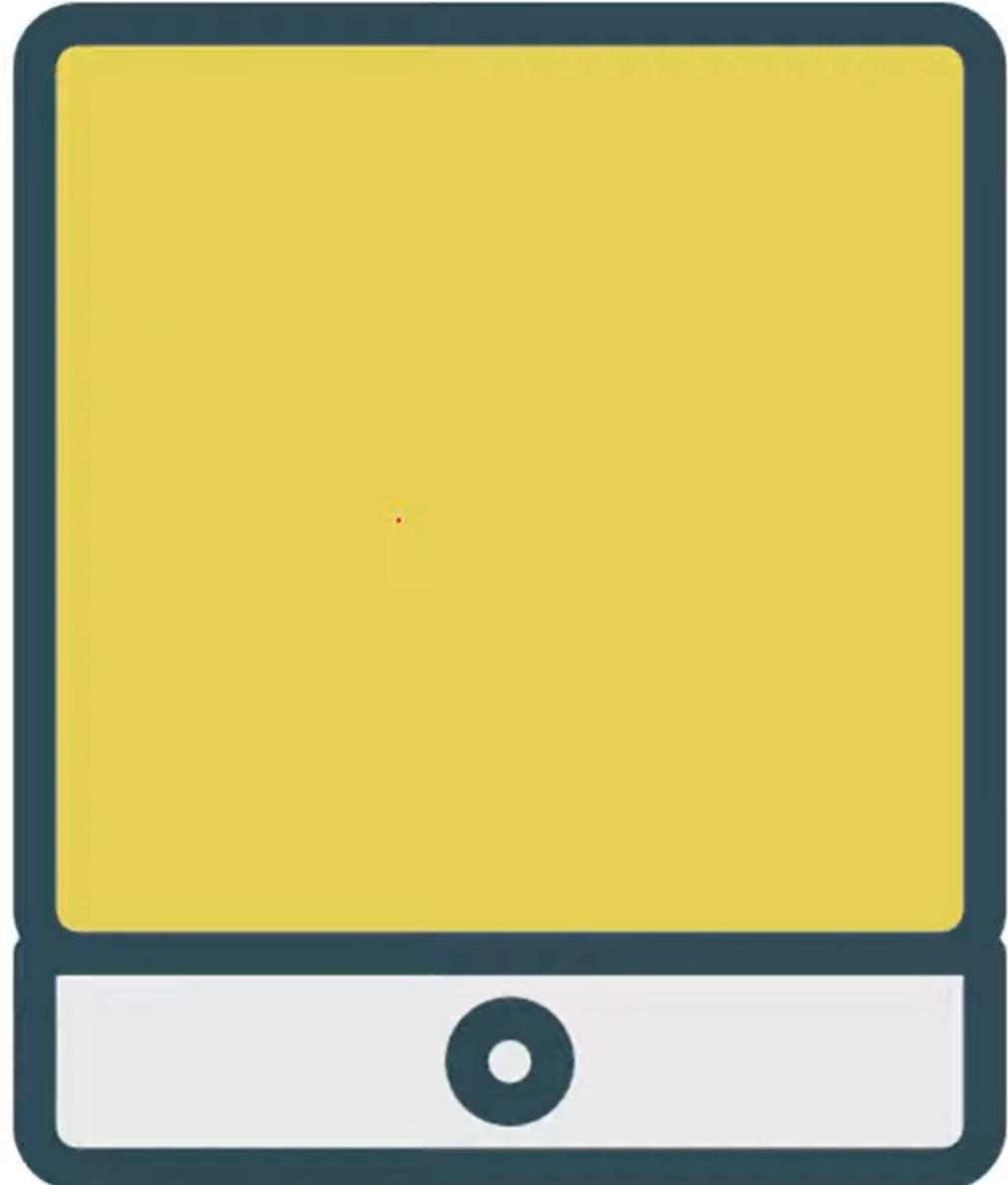
Saisir la valeur de A : 20

Saisir la valeur de B : 2

A - B = 20

# Arguments par défaut

```
def soustraction( A = 1 , B = 0 ):  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction()
```



```
1 def soustraction(A = 4, B = 2) :
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5     A = int(input("Entrez A : "))
6     B = int(input("Entrez B : "))
7
8     soustraction()
```

run scratch\_1 ×



D:\PyCharmProjects\Scratch\Scratch\scratch.pyw C:\Users\LEONNOU\Anaconda\Python\Scripts\

Entrez A : 45

Entrez B : 4

4 - 2 = 2

Process finished with exit code 0

```
def soustraction(A = 4, B = 2) :  
    resultat = A - B  
    print(f"{A} - {B} = {resultat}")
```

```
A = int(input("Entrez A : "))  
B = int(input("Entrez B : "))
```

```
soustraction(A, B)
```

scratch\_1 x

:

D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming\JetBrains\PyCh

Entrez A : 2

Entrez B : 2

2 - 2 = 0

# Arguments avec étiquettes (mot-clé)

```
def soustraction( A , B ):  
  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A = 10 , B = 5 )
```



```
1 def soustraction(A, B) :
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction(A = 10, B = 5)
```

Run

scratch\_1 ×



D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming\JetBrains\PyCh

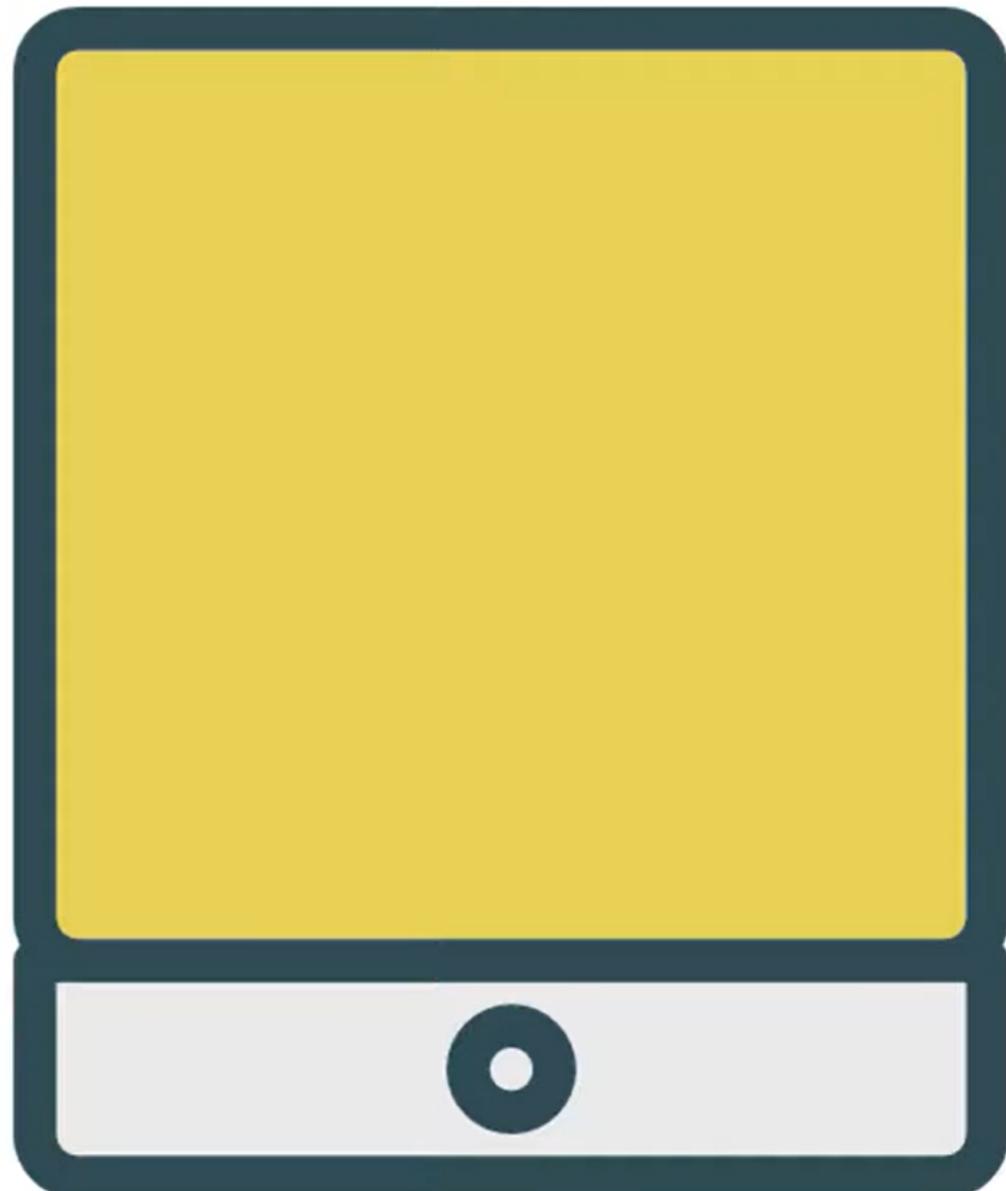
Entrez A : 6

Entrez B : 5

10 - 5 = 5

# Arguments avec étiquettes (mot-clé)

```
def soustraction( A , B ):  
  
    C = A - B  
  
    print( " A - B = " , C )  
  
soustraction( B = 5 , A = 10 )
```



scratch\_1.py ×

```
1 def soustraction(A, B) :
2     resultat = A - B
3     print(f"{A} - {B} = {resultat}")
4
5 A = int(input("Entrez A : "))
6 B = int(input("Entrez B : "))
7
8 soustraction(B = 10, A = 5)
```

⚠ 4 ✅ 4 ^

run scratch\_1 ×

```
D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming\Scratch\scratch_1.py
Entrez A : 2
Entrez B : 3
5 - 10 = -5
```

# Arguments avec étiquettes (mot-clé)

```
def soustraction( A , B ):  
    C = A - B  
  
    print ( " A - B = " , C )  
  
A = float ( input ( " Saisir la valeur de A : " ))  
  
B = float ( input ( " Saisir la valeur de B : " ))  
  
soustraction( A = 10 , B = 5 )
```

~~Saisir la valeur de A :~~  6

~~Saisir la valeur de B :~~  4

A - B = 5

# Arguments requis

Les arguments requis sont les arguments passés à une fonction dans **l'ordre de position correct**. Ici, le nombre d'arguments dans l'appel de fonction doit correspondre exactement à la définition de la fonction.

# Arguments requis

Les arguments requis sont les arguments passés à une fonction dans **l'ordre de position correct**. Ici, le nombre d'arguments dans l'appel de fonction doit correspondre exactement à la définition de la fonction.

Exemple :

```
def message( nom , titre ) :  
    print( " Bonjour " , titre , nom )  
  
message( " Ali " , " M. " )  
  
message( " Mme. " , " Amal " )  
  
message( " Anas " )
```

# Arguments requis

Les arguments requis sont les arguments passés à une fonction dans **l'ordre de position correct**. Ici, le nombre d'arguments dans l'appel de fonction doit correspondre exactement à la définition de la fonction.

Exemple :

<code>def message( nom , titre ):</code>	
<code>print( " Bonjour " , titre , nom )</code>	
<code>message( " Ali " , " M. " )</code>	<i>Bonjour M. Ali</i>
<code>message( " Mme. " , " Amal " )</code>	<i>Bonjour Amal Mme.</i>
<code>message( " Anas " )</code>	<i>Erreur</i>

# Arguments par défaut

Un argument par défaut est un argument qui prend une **valeur par défaut** si une valeur **n'est pas fournie** dans l'**appel** de fonction pour cet argument.

scratch\_1.py ×

```
1 def Salutation(nom, titre = 'Dev.') :  
2     print(f"BONJOUR {titre} {nom}")  
3  
4 Salutation('Kossi')
```

⚠1 ✅1 ^

Run scratch\_1 ×



```
D:\pythonProject1\.venv\Scripts\python.exe C:\Users\LENOVO\AppData\Roaming
```

```
BONJOUR Dev. Kossi
```

# Arguments par défaut

Un argument par défaut est un argument qui prend une **valeur par défaut** si une valeur **n'est pas fournie** dans l'**appel** de fonction pour cet argument.

Exemple :

```
def message( nom , titre =" M. "):  
    print( " Bonjour " , titre , nom )
```

message( " Ali " ) → *Bonjour M. Ali*

message( " Amal " , " Mme. " ) → *Bonjour Mme. Amal*

message( " Anas " , " M. " ) → *Bonjour M. Anas*

# Arguments avec étiquettes (mot-clé)

Les arguments avec étiquettes sont liés aux **appels** de fonction.

Lorsque nous appelons des fonctions de cette manière, **l'ordre (position)** des arguments peut être **modifié**.

Exemple :

```
def message ( nom , titre ):  
    print ( " Bonjour " , titre , nom )
```

```
message( titre = " M. " , nom = " Ali " )
```

Bonjour M. Ali

```
message( nom = " Amal " , titre = " Mme. " )
```

Bonjour Mme. Amal

```
message( " Sara " , " Mlle. " )
```

Bonjour Mlle. Sara

# Exercice 1

Nous considérons la fonction suivante :

```
def afficher( a , b ):  
    print( a , b )
```

Pour chacun des appels de fonction ci-contre, prédire si l'appel est valide ou non.

Appel	Valide / Non valide
afficher()	
afficher( 8 , 4 )	
afficher( 85 )	
afficher( a = 3 , 2 )	
afficher( 2 , a = 3 )	
afficher( 2 , 3 , 3 )	
afficher( b = 6 , 8 )	
afficher( a = 8 , b = 6 )	
afficher( b = 6 , a = 8 )	
afficher( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	
afficher ( 8 , 4 )	
afficher ( 85 )	
afficher ( a = 3 , 2 )	
afficher ( 2 , a = 3 )	
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	
afficher ( 85 )	
afficher ( a = 3 , 2 )	
afficher ( 2 , a = 3 )	
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	
afficher ( a = 3 , 2 )	
afficher ( 2 , a = 3 )	
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	
afficher ( 2 , a = 3 )	
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	Non
afficher ( b = 6 , 8 )	
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	Non
afficher ( b = 6 , 8 )	Non
afficher ( a = 8 , b = 6 )	
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	Non
afficher ( b = 6 , 8 )	Non
afficher ( a = 8 , b = 6 )	Oui
afficher ( b = 6 , a = 8 )	
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	Non
afficher ( b = 6 , 8 )	Non
afficher ( a = 8 , b = 6 )	Oui
afficher ( b = 6 , a = 8 )	Oui
afficher ( 2 , 1 , b = 9 )	

# Exercice 1

```
def afficher ( a , b ):  
    print ( a , b )
```

Appel	Valide / Non valide
afficher ()	Non
afficher ( 8 , 4 )	Oui
afficher ( 85 )	Non
afficher ( a = 3 , 2 )	Non
afficher ( 2 , a = 3 )	Non
afficher ( 2 , 3 , 3 )	Non
afficher ( b = 6 , 8 )	Non
afficher ( a = 8 , b = 6 )	Oui
afficher ( b = 6 , a = 8 )	Oui
afficher ( 2 , 1 , b = 9 )	Non

# Exercice 2

Nous considérons la fonction suivante :

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Pour chacun des appels de fonction ci-contre, prédire si l'appel est valide ou non.

Appel	Valide / Non valide
afficher ( 8 )	
afficher ( a = 4 )	
afficher ( 8 , 5 )	
afficher ( 8 , 5 , c = 2 )	
afficher ( 8 , 5 , 2 )	
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	
afficher ( 8 , 5 )	
afficher ( 8 , 5 , c = 2 )	
afficher ( 8 , 5 , 2 )	
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	
afficher ( 8 , 5 , c = 2 )	
afficher ( 8 , 5 , 2 )	
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	
afficher ( 8 , 5 , 2 )	
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    .  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	Non
afficher ( c = 7 , a = 1 )	
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    .  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	Non
afficher ( c = 7 , a = 1 )	Oui
afficher ( 0 , 7 , e = 1 )	
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	Non
afficher ( c = 7 , a = 1 )	Oui
afficher ( 0 , 7 , e = 1 )	Non
afficher ( 4 , c = 4 )	
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	Non
afficher ( c = 7 , a = 1 )	Oui
afficher ( 0 , 7 , e = 1 )	Non
afficher ( 4 , c = 4 )	Oui
afficher ( 0 , 7 , b = 1 )	

## Exercice 2

```
def afficher ( a , b=5, c="N" ):  
    print ( a , b , c )
```

Appel	Valide / Non valide
afficher ( 8 )	Oui
afficher ( a = 4 )	Oui
afficher ( 8 , 5 )	Oui
afficher ( 8 , 5 , c = 2 )	Oui
afficher ( 8 , 5 , 2 )	Oui
afficher ( 2 , 3 , 3 , 1 )	Non
afficher ( c = 6 , 8 )	Non
afficher ( c = 7 , a = 1 )	Oui
afficher ( 0 , 7 , e = 1 )	Non
afficher ( 4 , c = 4 )	Oui
afficher ( 0 , 7 , b = 1 )	Non