

TP N 2 Variables, type de données et opérations

Manipulation de type de données fondamentales

Une variable est une zone de la mémoire de l'ordinateur réservée pour le stockage des valeurs.

Par exemple :

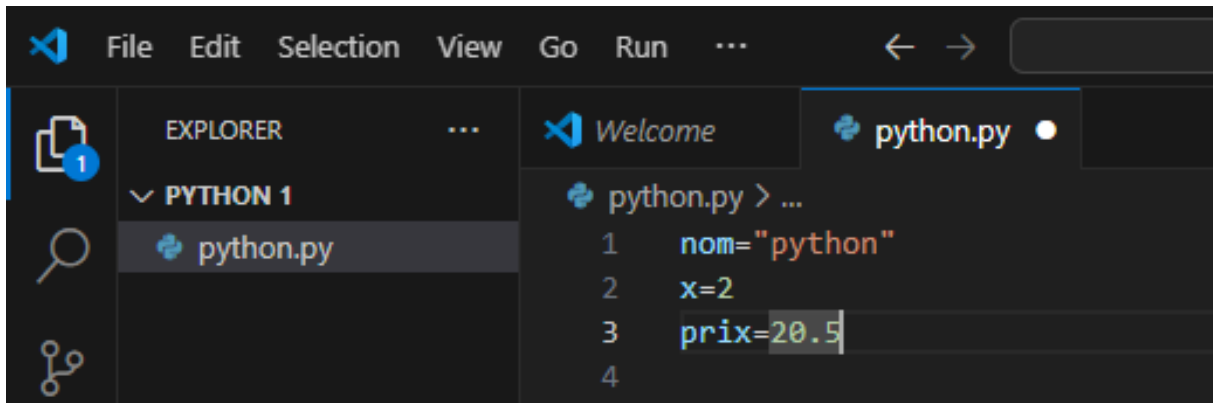


Figure 1 : Les variables en python.

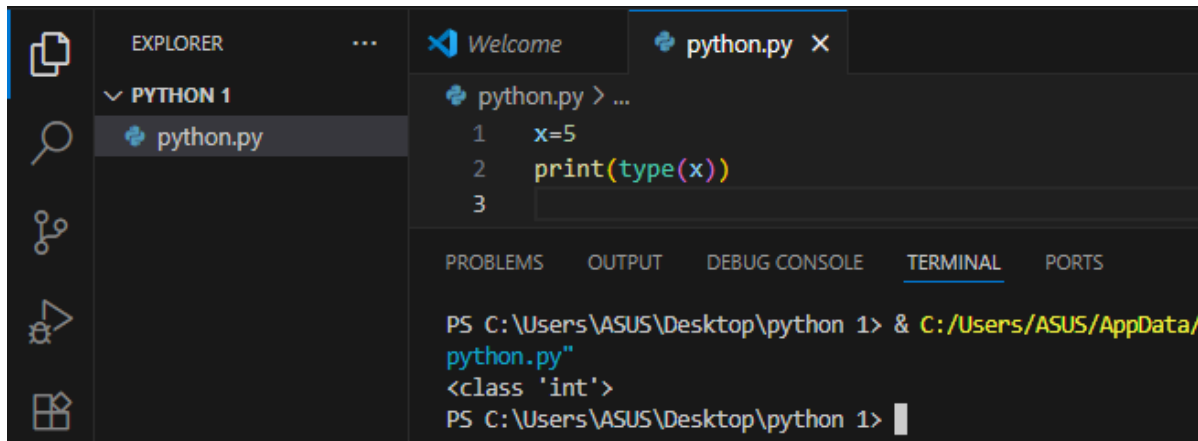
Nom, x et prix sont des variables.

Les types de variables en python

En python on trouve plusieurs types standards, le tableau suivant présente les différents types :

Type		Exemple	Description
Numérique	int	20 ou -3	Pour les nombres entiers (positif ou négatif).
	float	3.14	Pour les nombres décimaux à virgule flottante.
	complex	3+5i	Pour les nombres complexes.
Texte	str	"bonjour"	Une chaîne de caractères (texte) qui doit être entre deux guillemets.
Booléens	bool	True ou False	Logique (vrai ou faux).

Exemple



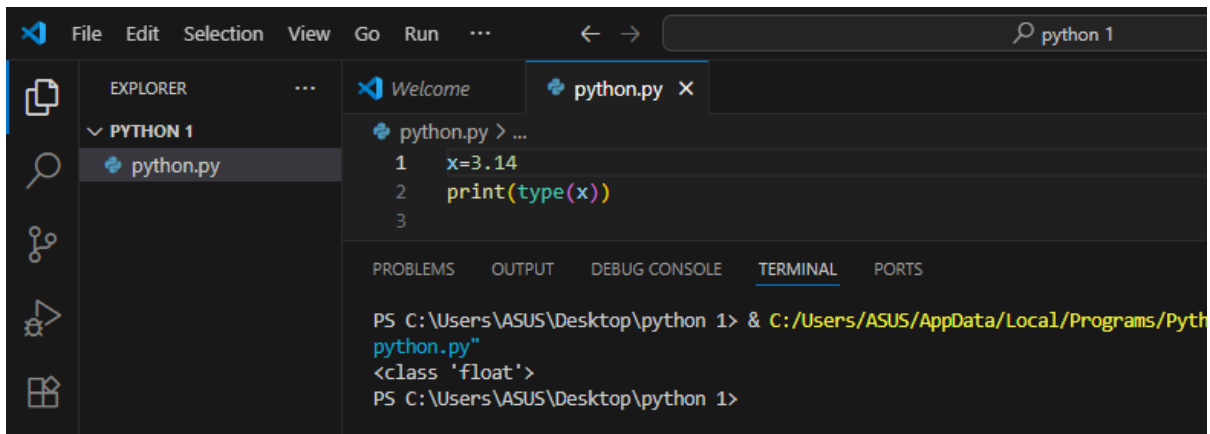
The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a file named 'python.py' under a folder 'PYTHON 1'. The main editor window displays the following code:

```
1 x=5
2 print(type(x))
3
```

The TERMINAL panel at the bottom shows the command prompt output:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39-64/python.py
<class 'int'>
PS C:\Users\ASUS\Desktop\python 1>
```

a- Variable de type numérique (entier)



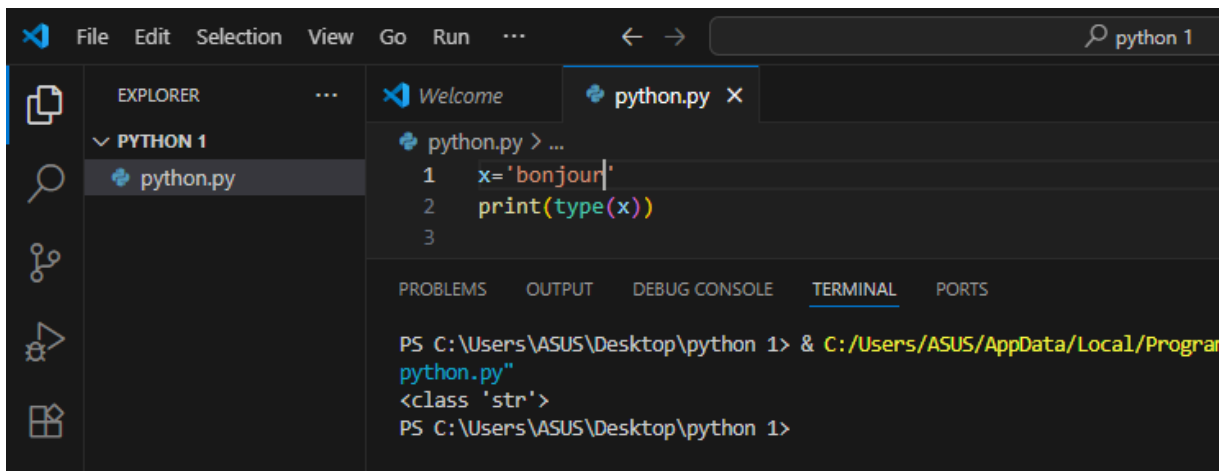
The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a file named 'python.py' under a folder 'PYTHON 1'. The main editor window displays the following code:

```
1 x=3.14
2 print(type(x))
3
```

The TERMINAL panel at the bottom shows the command prompt output:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39-64/python.py
<class 'float'>
PS C:\Users\ASUS\Desktop\python 1>
```

b- Variable de type numérique (décimale)



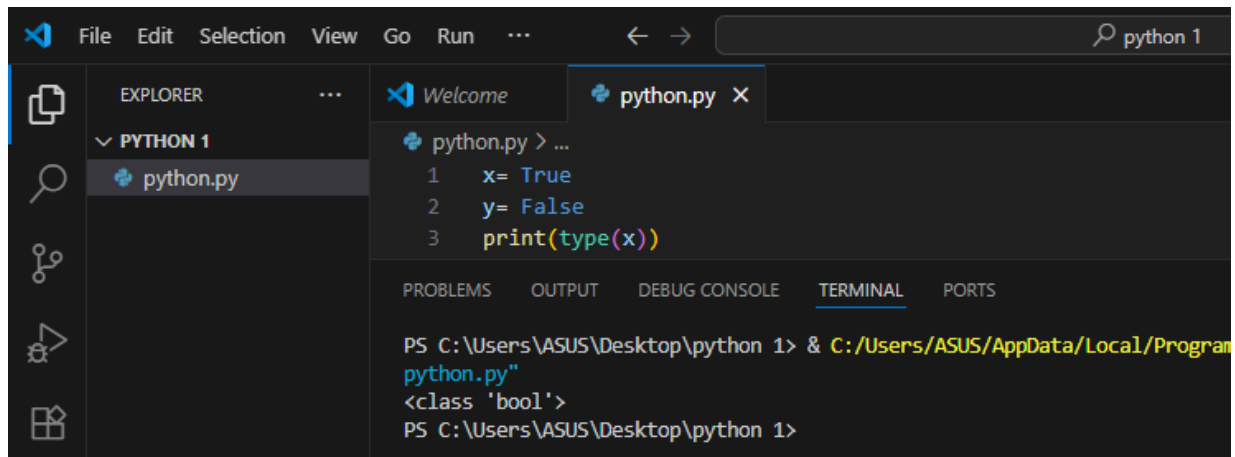
The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a file named 'python.py' under a folder 'PYTHON 1'. The main editor window displays the following code:

```
1 x='bonjour'
2 print(type(x))
3
```

The TERMINAL panel at the bottom shows the command prompt output:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python39-64/python.py
<class 'str'>
PS C:\Users\ASUS\Desktop\python 1>
```

c- Variable de type texte



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a file named 'python.py' under a folder 'PYTHON 1'. The Editor pane shows the following code in 'python.py':

```
1 x= True
2 y= False
3 print(type(x))
```

The TERMINAL pane at the bottom shows the command prompt output:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
<class 'bool'>
PS C:\Users\ASUS\Desktop\python 1>
```

d- Variable de type logique

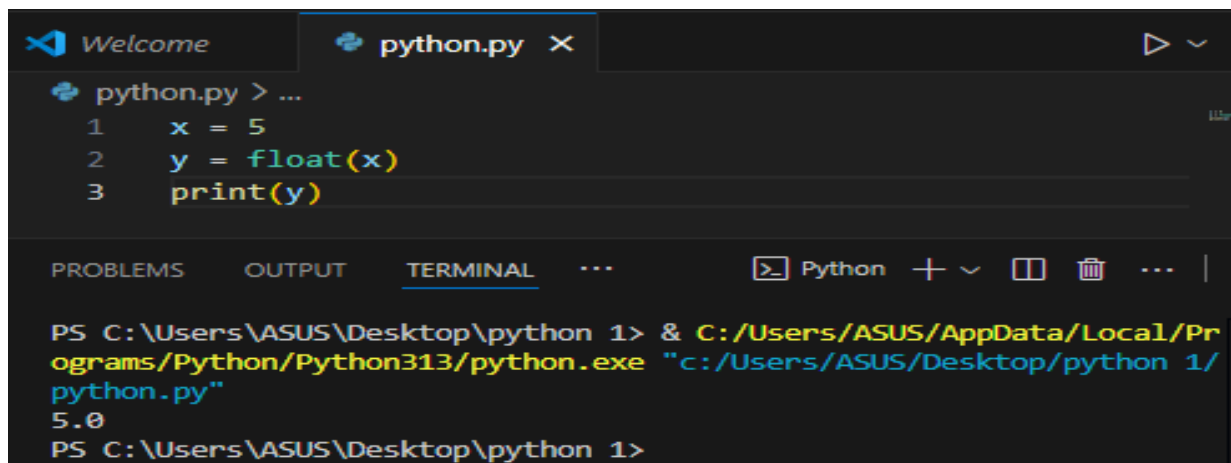
Figure 2 : Exemple sur les différents variables en python.

Conversion entre les types des données

La conversion de types consiste à transformer une donnée d'un type vers un autre, par exemple convertir une chaîne (str) en entier (int), ou un entier en nombre décimal (float).

Exemple

1. convertir un entier en décimal ?



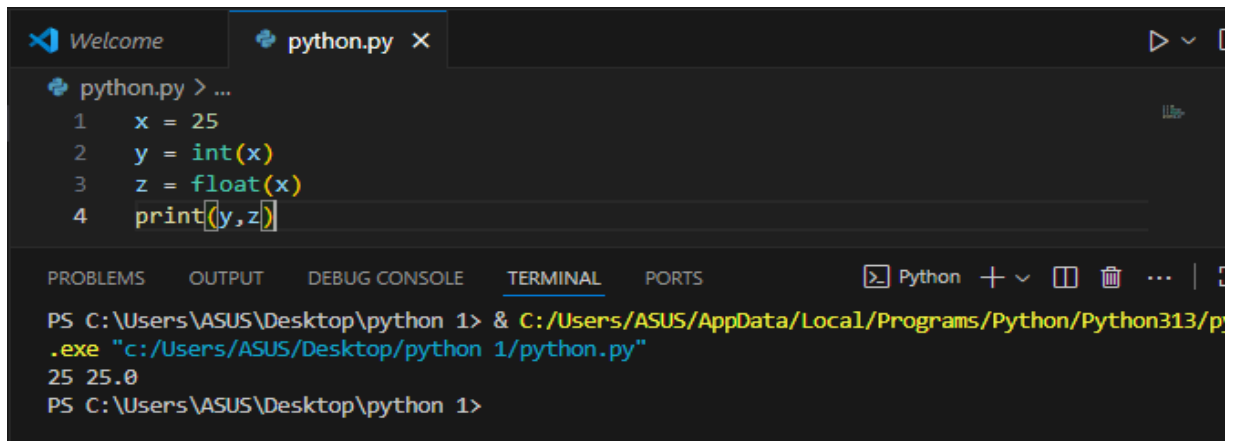
The screenshot shows the Visual Studio Code interface. The Editor pane shows the following code in 'python.py':

```
1 x = 5
2 y = float(x)
3 print(y)
```

The TERMINAL pane at the bottom shows the command prompt output:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
5.0
PS C:\Users\ASUS\Desktop\python 1>
```

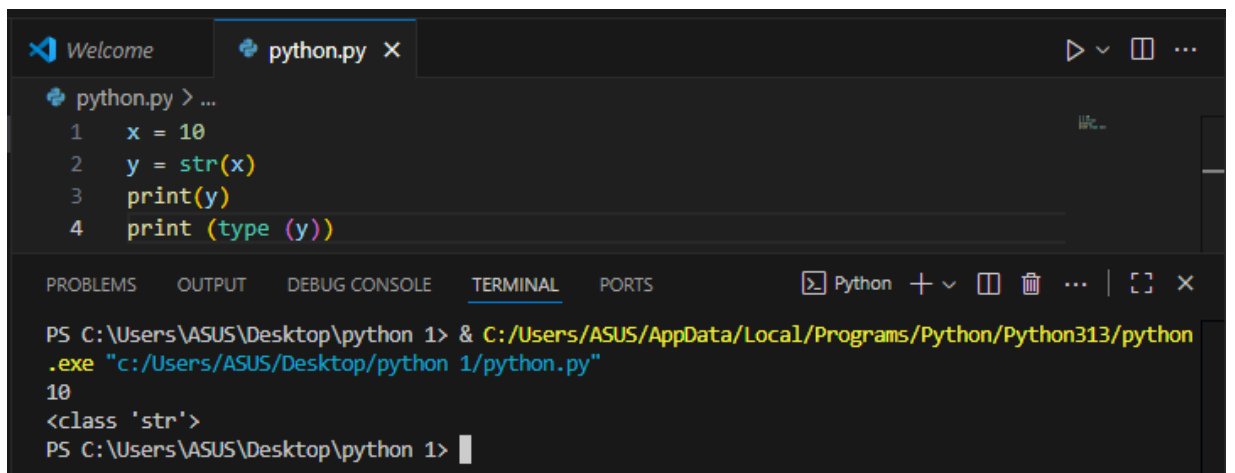
2. convertir une chaine en entier en décimal ?



```
python.py > ...
1 x = 25
2 y = int(x)
3 z = float(x)
4 print(y,z)
```

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
25 25.0
PS C:\Users\ASUS\Desktop\python 1>
```

3. convertir un nombre en chaine de caractères ?



```
python.py > ...
1 x = 10
2 y = str(x)
3 print(y)
4 print (type (y))
```

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
10
<class 'str'>
PS C:\Users\ASUS\Desktop\python 1>
```

Figure 3 : Exemple sur conversion entre les types des données en python.

Opérations arithmétiques et priorités

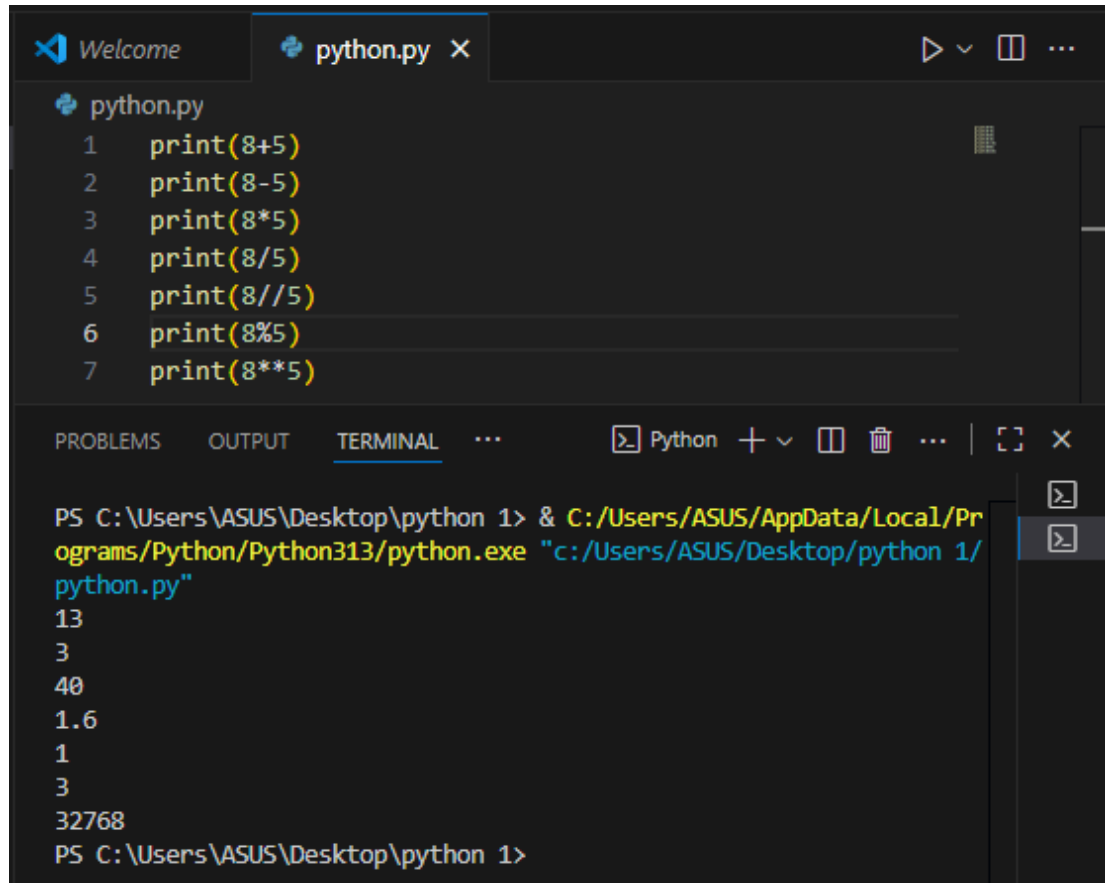
En python comme les autres langages, les opérateurs arithmétiques de base et leurs priorités suivent des règles proches de mathématique classique.

Les opérations arithmétiques de base

Le tableau suivant résume les opérations de base :

Opérations	Symbole	En python	Affichage
Addition	(+)	print(8+5)	13
Soustraction	(-)	print(8-5)	3
Multiplication	(*)	print(8*5)	40

Division	(/)	print(8/5)	1.6
Division entière	(//)	print(8//5)	1 (quotient entier)
Modulo	(%)	print(8%5)	3 (reste de la division entière)
Exposant	(**)	print(8**5)	32768



```

python.py
1  print(8+5)
2  print(8-5)
3  print(8*5)
4  print(8/5)
5  print(8//5)
6  print(8%5)
7  print(8**5)

TERMINAL
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
13
3
40
1.6
1
3
32768
PS C:\Users\ASUS\Desktop\python 1>

```

Figure 4 : Exemple sur Les opérations arithmétiques de base en python.

Priorités des opérateurs

L'ordre des opérations suit la règle PEMDAS comme sera présente dans l'organigramme :

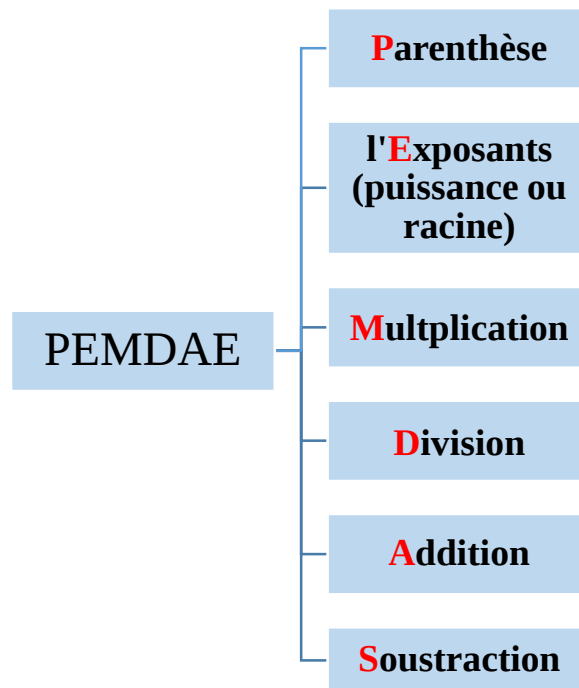


Figure 5 : Priorité des opérations en python.

- Quand deux opérateurs possèdent une priorité identique, les calculs s'effectuent suivant un ordre de gauche à droite. Cela signifie que l'opérateur situé à gauche est évalué en premier
- Les parenthèses permettent de modifier l'ordre naturel et de regrouper les opérations que l'on souhaite effectuer en premier.

Exemple

The screenshot shows a Python IDE with a file named 'python.py' open. The code in the editor consists of five lines, each with a print statement and a comment explaining the result based on operator precedence:

```
1 print(2 + 3 * 4)      # Résultat: 14 (multiplication avant addition)
2 print((2 + 3) * 4)    # Résultat: 20 (parenthèses avant multiplication)
3 print(5 ** 2)         # Résultat: 25 (exponentiation)
4 print(10 // 3)        # Résultat: 3 (division entière)
5 print(10 % 3)         # Résultat: 1 (reste de la divis)
```

Below the editor, the 'TERMINAL' tab is active, showing the command prompt output of running the script:

```
PS C:\Users\ASUS\Desktop\python 1> & C:/Users/ASUS/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ASUS/Desktop/python 1/python.py"
14
20
25
3
1
PS C:\Users\ASUS\Desktop\python 1>
```

Figure 6 : Exemple numérique sur les priorités des opérations en python.

Référence

1. <https://www.datacamp.com/fr/tutorial/python-operators-tutorial>
2. https://www.pythonize.ir/fr/lecons/la_priorit%C3%A9/#Bottom