Les fonctions

Objectifs :

- Comprendre l'interêt des fonctions ;
- Apprendre à définir et utiliser des **fonctions** en Python.
- Comprendre l'importance des **paramètres** et des **valeurs de retour**.
- Savoir organiser un programme en utilisant des fonctions pour plus de **lisibilité** et de **réutilisabilité**.

Table des matières

1. Pourquoi les fonctions sont importantes ?....1
2. Syntaxe des fonctions....1
3. Exemples simples....2
4. Points à retenir....2
5. Exercices pratiques....2
   a. Exercice 1 : Fonction de somme....2
   b. Exercice 2 : Fonction de comparaison....2
6. Défis supplémentaires....3
   a. Défi 1 : Tester un entier pair....3
   b. Défi 2 : Convertisseur d'unités....3
   c. Défi 3 : Triangle valide....3
   d. Défi 4 : Calcul de la valeur de mon livret A....3
7. Pour aller plus loin, arguments nommés et valeurs par défaut....3
8. Comment obtenir des informations sur une fonction ? help()....3

1. Pourquoi les fonctions sont importantes ?

Les fonctions permettent de diviser un programme en petites parties réutilisables. Cela permet de :

1. **Réduire la répétition de code** : au lieu d'écrire plusieurs fois le même code, on peut l'encapsuler dans une fonction.
2. **Faciliter la lecture** : un programme est plus lisible lorsqu'il est divisé en tâches.
3. **Tester plus facilement** : tester et valider des petites fonctions indépendantes est plus simple que de tester un grand programme d'un seul bloc.

2. Syntaxe des fonctions

Voici comment créer une fonction simple en Python :

```
def nom_de_la_fonction(parametre1, parametre2):
    # instructions
    return resultat
```

- `def` : mot-clé pour définir une fonction.
- `nom_de_la_fonction` : nom donné à la fonction.
- `parametre1, parametre2` : paramètres que la fonction prend en entrée.
- `return` : permet de renvoyer un résultat à la fin de la fonction.

3. Exemples simples

Exemple 1 : Fonction sans paramètre (et sans grand intérêt...)

```
def dire_bonjour():
    print("Bonjour !")

dire_bonjour()  # Appel de la fonction
```

Exemple 2 : Fonction avec des paramètres

```
def addition(a, b):
    return a + b

resultat = addition(5, 3)
print(resultat)  # Affiche 8
```

Exemple 3 : Fonction avec plusieurs retours

```
def carre_et_cube(x):
    carre = x ** 2
    cube = x ** 3
    return carre, cube

carre, cube = carre_et_cube(3)
print("Carré:", carre)  # Affiche 9
print("Cube:", cube)  # Affiche 27
```

4. Points à retenir

1. **Appel d'une fonction** : utiliser son nom suivi de parenthèses.
2. **Paramètres** : variables passées à la fonction lors de l'appel.
3. **Valeur de retour** : la fonction peut renvoyer une valeur avec `return`.
4. Penser à nommer de manière explicite la fonction et ses paramètres.

5. Exercices pratiques

a. Exercice 1 : Fonction de somme

Créer une fonction `somme(a, b, c)` qui prend trois nombres en paramètres et renvoie leur somme.

b. Exercice 2 : Fonction de comparaison

Créer une fonction `comparaison(a, b)` qui renvoie :

- `"a est plus grand"` si `a > b`,
- `"b est plus grand"` si `b > a`,
- `"a et b sont égaux"` si `a == b`.

6. Défis supplémentaires

a. Défi 1 : Tester un entier pair

Créer une fonction `est_pair(n)` qui renvoie 1 si le nombre est pair, `0` sinon.

b. Défi 2 : Convertisseur d'unités

Créer une fonction `celsius_vers_fahrenheit(c)` qui convertit une température Celsius en Fahrenheit (formule : `F = C * 9/5 + 32`).

c. Défi 3 : Triangle valide

Créer une fonction `est_triangle(a, b, c)` qui vérifie si trois longueurs peuvent former un triangle (la somme de deux côtés doit être supérieure au troisième).

d. Défi 4 : Calcul de la valeur de mon livret A

Je décide d'ouvrir un livret A qui me rapporte 1,5 % par an. Si je fournis la somme que je compte déposer, est-il possible qu'une fonction me calcule et m'affiche la somme que j'aurai sur mon livret dans 1, 2, …. 10 ans.

En fait je peux économiser et j'envisage donc de verser chaque année une somme sur mon livret A. Si à l'ouverture de mon livret je verse une somme X et que j'ajoute à ce que j'ai sur mon livret chaque année une somme Y, est-il possible d'écrire une fonction qui me calcule la somme dont je disposerai sur mon livret dans 1,2, ….10 ans ?

On peut aussi ajouter le taux par an en paramètre pour comparer le livret A à un autre placement

7. Pour aller plus loin, arguments nommés et valeurs par défaut

Pour certaines fonctions il est intéressant de définir des valeurs par défauts. Dans ce cas les paramètres avec des valeurs par défaut doivent être placés après les paramètres sans valeur par défaut. Exemple :

```
def repeter(texte, n=3) :
    for i in range(n) :
        print(texte)
```

`repeter("bonjour")`
`bonjour`
`bonjour`
`bonjour`
	`repeter("bonjour",5)`
	`bonjour`
	`bonjour`
	`bonjour`
	`bonjour`
	`bonjour`

8. Comment obtenir des informations sur une fonction ? help()

Taper help(nom) dans un interpréteur python. Essayer help(range) ou help(random.randint)