INF1221 : PROGRAMMATION PYTHON

AMOUZOU Grâce Dorcas Akpéné / EPL / 2024-2025 M

BEFOR WE START

Démarrez l'interpréteur Python :

- ★ dans l’invite de commandes
- ★ dans Pycharm
- ★ dans Spyder

SÉANCE 4 - PARTIE 1: APPROFONDISSEMENT

COPIE DE LISTES 1/2

- ★ liste1 = list(range(5))
- ★ liste1
- ★ liste2 = liste1
- ★ liste2
- ★ liste1[3] = -50
- ★ liste1
- ★ liste2

Cela vient du fait que Python a effectué la copie de liste par référence. Les deux listes pointent vers le même objet dans la mémoire.

COPIE DE LISTES 2/2

Pour contrer le problème, utiliser la fonction list() ou la méthode .copy()

- ★ liste3 = list(liste1)
- ★ liste4 = liste1.copy()

Cette astuce ne fonctionne que pour des listes à une dimension (c'est-à-dire pour des listes qui ne contiennent que des éléments de type simple comme des entiers, des floats, des chaînes de caractères et des booléens), mais pas pour des listes de listes.

BOUCLE FOR AVEC LES LISTES

- animaux = ["girafe", "tigre", "singe", "souris"]
- for animal in animaux:
- ... print(animal)

La variable animal est appelée **variable d'itération**, elle prend successivement les différentes valeurs de la list *animaux* à chaque **itération** (ou tour) de boucle.

Exercice :

Voici les notes d'un étudiant [14, 9, 6, 8, 12]. Calculez la moyenne de ces notes. Utilisez l'écriture formatée pour afficher la valeur de la moyenne avec deux décimales.

SÉANCE 4 - PARTIE 2: FONCTIONS

DÉFINITION

Pour définir une fonction, Python utilise le mot-clé def. Si on souhaite que la fonction renvoie quelque chose, il faut utiliser le mot-clé return.

- def carre(x):
- ... return x**2
-
- print(carre(4))

Notez qu'une fonction ne prend pas forcément un argument et ne renvoie pas forcément une valeur. Notez qu'une fonction ne prend pas forcément un argument et ne renvoie pas forcément une valeur.

PASSAGE D'ARGUMENTS

Une particularité des fonctions en Python est que vous n'êtes pas obligé de préciser le type des arguments que vous lui passez.

- def multiplier(x, y):
- ... return x*y
-
- multiplier(2, 3)
- multiplier(3.1415, 5.23)
- multiplier("to", 2)
- multiplier([1,3], 2)

PASSAGE D'ARGUMENTS 1/2

Les **arguments positionnels** sont strictement obligatoires à préciser lors de l'appel de la fonction. De plus, il est nécessaire de respecter le même ordre lors de l'appel que dans la définition de la fonction.

Mais il est aussi possible de passer un ou plusieurs argument(s) de manière facultative et de leur attribuer une valeur par défaut (**argument par mot-clé**).

- def fct(x=1):
- ... return x
-
- fct()
- fct(5)

PASSAGE D'ARGUMENTS 2/2

- def fct(a, b, x=0, y=0, z=0):
- ... return a, b, x, y, z
-
- fct(1, 1)
- fct(1, 1, z=5)
- fct(1, 1, z=5, y=32)
-
- fct(z=0)

RENVOI DE RÉSULTAT

Un énorme avantage en Python est que les fonctions sont capables de renvoyer plusieurs objets à la fois.

- def carre_cube(x):
- ... return x**2, x**3
-
- carre_cube(2)
- z1, z2 = carre_cube2(3)

VARIABLES LOCALES ET VARIABLES GLOBALES

Une variable est dite **locale** lorsqu'elle est créée dans une fonction. Elle n'existera et ne sera visible que lors de l'exécution de ladite fonction.

Une variable est dite globale lorsqu'elle est créée dans le programme principal. Elle sera visible partout dans le programme.

BEFOR WE START

Démarrez l’un des environnements suivants :

- ★ Pycharm
- ★ Spyder
- ★ Jupyter notebook

EXERCICES

- ★ Ecrire une fonction qui renvoie le carré d'un nombre.
- ★ Ecrire une fonction qui renvoie la factorielle d'un nombre.
- ★ Ecrire une fonction qui renvoie un nombre élevé à la puissance d'un second.
- ★ Ecrire une fonction qui détermine si un nombre est premier ou non.

SÉANCE 4 :
END