Python

Structure de données de base

Yannick CHISTEL

Lycée Dumont d'Urville - CAEN

Octobre 2020

Présentation

Une fonction rassemble sous un même nom une séquence d'instructions qui seront exécutées lors de l'appel de la fonction.

La fonction est introduite par le mot clé **def** suivi du nom de la fonction, suivi de parenthèses et les deux points qui imposeront une indentation.

- **def** nom de la fonction() :

Renvoyer un résultat

Une fonction peut **renvoyer** ou retourner une valeur. Le mot-clé utilisé en Python pour renvoyer une valeur est **return**.

- **def** nom de la fonction() :
  instructions
  return valeur

Exemple

On veut afficher un carré de 5 colonnes et 5 lignes rempli par la lettre X.

La fonction **dessinecarre()** renvoie une chaine de caractères (type string) qui affichera le carré avec la commande **print**.

```
def dessinecarre():
    motif=""
    for _ in range(5):
        for _ in range(5):
            motif=motif+"X"
        motif=motif+"\n"
    return motif

d=dessinecarre()
print(d)
```

```
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
```

L'appel de la fonction **dessinecarre** provoquera l'exécution de toutes les instructions contenues dans la fonction et renvoie une chaine de caractères contenant des X et des retours à la ligne.

Fonction avec 1 paramètre

La fonction précédente **dessinecarre** réalise des carrés de côté 5. On peut modifier cette valeur en transmettant un **paramètre** à la fonction. La syntaxe est la suivante :

- **def** nom de la fonction(**paramètre**) :
  instructions

Exemple

```
# La fonction dessinecarre avec un paramètre

def dessinecarre(n):
    motif=""
    for _ in range(n):
        for _ in range(n):
            motif=motif+"X"
        motif=motif+"\n"
    return motif


d=dessinecarre(8)
print(d)
```

```
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
XXXXXXXX
```

L'appel de la fonction **dessinecarre(8)** réalise un carré de 8 lignes et 8 colonnes.

Fonction avec plusieurs paramètres

On peut passer plusieurs paramètres à une fonction selon la syntaxe suivante :

- **def** nom de la fonction(**paramètre1,paramètre2,...**) :
  instructions

Exemple

```
def dessinecarre(n,c):
    motif=""
    for _ in range(n):
        for _ in range(n):
            motif=motif+c
        motif=motif+"\n"
    return motif


d=dessinecarre(10,"O")
print(d)
```

```
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
OOOOOOOOOO
```

L'appel de la fonction **dessinecarre(10,"O")** réalise un carré de 10 lignes et 10 colonnes avec la lettre O.

Variable locale à une fonction

Une fonction peut utiliser des variables nécessaires à son fonctionnement, pour réaliser des calculs intermédiaires. Ces variables sont dites locales à la fonction.
Il n'est pas possible d'accéder à la valeur d'une variable locale en dehors de la fonction.

Exemple

Fonction qui calcule et renvoie la moyenne des nombres passés en paramètres.

- **def** moyenne(a,b,c) :
  s=a+b+c
  return s/3

Attention

La fonction moyenne utilise une variable locale **s**. Cette variable n'est pas accessible sauf dans la fonction. Si on veut afficher la valeur de s avec **print(s)**, une erreur est affichée !

Interruption d'une fonction

Il est possible d'arrêter l'exécution d'une fonction en renvoyant une valeur booléenne (True, False). Le mot clé **return** est présent à plusieurs endroits dans le code de la fonction.

Exemple

La fonction ci-dessous contient une instruction print. L'affichage ne se réalisera pas cas la fonction renvoie un booléen avant l'instruction print.

```
def est_pair(n):
    if n%2==0:
        return True
    else:
        return False
    print("le nombre est pair")

est_pair(5)
```

Dès qu'une condition est vérifiée, l'exécution de la fonction est arrêtée et la valeur booléenne est renvoyée.