ALGORITHMIQUE · CHAPITRE 19 · FONCTIONS

Définir et appeler une fonction

Définissez et appelez une première fonction. Reliez paramètre, argument, corps et résultat avec double_value en pseudo-code, Python et MATLAB.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Décrire une sous-tâche par son entrée et son résultat.
- Distinguer définition et exécution d'une fonction.
- Suivre plusieurs appels sans recopier le calcul.

Prérequis. Savoir lire des affectations, conditions et boucles simples.

Pourquoi donner un nom à une opération ?

Vous devez doubler plusieurs valeurs dans des endroits différents d'un programme. Recopier chaque fois le calcul fonctionne, mais disperser la règle rend ses modifications plus difficiles à suivre. Une **fonction** rassemble une méthode sous un nom : `double_value`.

Définir puis appeler

La **définition** décrit ce que la fonction fera lorsqu'on lui transmettra une valeur. L'**appel** demande effectivement d'exécuter ces instructions avec une valeur donnée. Lire une définition ne signifie pas que tous les calculs de son corps viennent d'être exécutés. L'entrée et le résultat forment un petit contrat : pour un nombre x, renvoyer 2 × x.

Paramètre, argument, résultat et appelant

Un **paramètre** est un nom qui reçoit une donnée dans la définition d'une fonction. Dans `double_value(value)`, ce nom est `value`. Un **argument** est la donnée transmise lors d'un appel : dans `double_value(7)`, l'argument vaut 7. Cette distinction concerne le rôle du nom et de la valeur, pas leur orthographe.

Le **corps** de la fonction contient ses instructions. Le **résultat retourné** est la valeur rendue disponible à l'endroit de l'appel. L'**appelant** est la partie du programme qui demande cette exécution. Dans `result = double_value(7)`, l'appelant reçoit 14 et l'affecte à `result`.

Le pseudo-code utilise `FUNCTION` et `END FUNCTION` pour délimiter la définition, puis `RETURN` pour renvoyer une valeur. Python utilise `def` pour définir la fonction. En MATLAB, `function result = double_value(value)` annonce explicitement le nom de sortie et le paramètre.

Enregistrez d'abord la fonction MATLAB dans `double_value.m`, puis son bloc d'appel dans `use_double_value.m`, dans le même dossier. La fenêtre de commande permet de lancer le script d'appel. La sortie de la fonction n'apparaît pas par magie dans n'importe quel script : l'appel et l'affectation décrivent le transfert du résultat.

Une première fonction

Pseudo-code

```
// example.pseudo
// double_value
FUNCTION double_value(value)
    RETURN value * 2
END FUNCTION
```

Python

```python
# double_value.py
def double_value(value):
    return value * 2

result = double_value(7)
print(result)  # 14
```

MATLAB

```matlab
% double_value.m
function result = double_value(value)
    result = value * 2;
end
```

En Python, `return` transmet la valeur calculée à l'appelant. En MATLAB, le nom situé à gauche du signe égal dans la déclaration de fonction désigne la sortie : ici `result`. Enregistrez cette version MATLAB dans `double_value.m` avant de lancer les appels suivants.

MATLAB

```matlab
% use_double_value.m
result = double_value(7);
disp(result); % 14
```

Suivre un appel complet

De l'appel au résultat

Étape	Ce qui se passe
Call double_value(7)	Le paramètre value reçoit 7
Corps de la fonction	Le résultat calculé est 14

Étape	Ce qui se passe
Return à l’appelant	La valeur 14 devient disponible pour l’affectation

Le nom `value` à l'intérieur de la fonction n'impose pas le nom de la variable utilisée à l'extérieur. L'appel `double_value(7)` transmet une valeur, pas une phrase à interpréter ni le nom d'une instruction à copier.

Réutiliser la règle

Prévoyez les résultats des appels avec 0, 3 et −2 : respectivement 0, 6 et −4. Puis appelez la fonction deux fois avec la même entrée. Cette version ne modifie aucun état extérieur : ses résultats sont identiques pour ces appels. Il ne faut pas en déduire que toutes les fonctions possibles ont cette propriété.

Délimiter une sous-tâche

<details>
<summary>▼ Que calcule double_value(double_value(3)) ?</summary>

L'appel intérieur produit d'abord 6. Cette valeur devient l'entrée de l'appel extérieur, qui renvoie 12. On ne double pas deux fois simultanément : il faut suivre l'ordre des dépendances.
</details>

<details>
<summary>▼ Faut-il afficher le résultat dans la fonction pour pouvoir l'utiliser ?</summary>

Non. Une valeur retournée peut être utilisée dans un autre calcul ou affichée ensuite. Séparer calcul et affichage permet de réutiliser la fonction dans plusieurs contextes.
</details>

Donner un sens au nom

Une fonction facilite la lecture si son nom et son contrat disent ce qu'elle fait. Le chapitre suivant distingue précisément paramètres, arguments et résultat, puis montre pourquoi afficher une valeur n'est pas la même chose que la retourner.

À retenir - fiche mémo

Définition

Elle décrit le traitement nommé ; l’appel demande son exécution.

Paramètre / argument

Le paramètre est le nom dans la définition ; l’argument est la donnée transmise.

Retour

Une valeur retournée est disponible chez l’appelant pour un autre calcul.

Pseudo-code

```
// example.pseudo
// double_value
FUNCTION double_value(value)
    RETURN value * 2
END FUNCTION
```

Point de vigilance. Afficher un résultat ne le retourne pas automatiquement.

Consulter la cheatsheet : Fonctions · **Télécharger la fiche du thème en PDF**

Version web : https://www.clearcode.fr/courses/programming/algorithmic/functions