

Programmation fonctionnelle

La programmation fonctionnelle est un paradigme de programmation de type déclaratif qui considère le calcul en tant qu'évaluation de fonctions mathématiques.

Comme le changement d'état et la mutation des données ne peuvent pas être représentés par des évaluations de fonctions, la programmation fonctionnelle ne les admet pas, au contraire elle met en avant l'application des fonctions, contrairement au modèle de programmation impérative qui met en avant les changements d'état.

Principes de la programmation fonctionnelle :

- Fonction de première classe
- Fonction d'ordre supérieur
- Fonction pure
- Récursion
- Fonction anonyme
- Objets immuables

Un langage de programmation est dit avoir des **fonctions de première classe** lorsque les fonctions dans ce langage sont traitées comme n'importe quelle autre variable.

Les **fonctions d'ordre supérieur** sont des fonctions qui prennent une ou plusieurs fonctions en entrée et/ou qui renvoient une fonction.

Exemple :

```
def a(f):  
    return lambda x: f(x) + 2  
  
f = lambda x: 2*x  
a(f)(3) # 8
```

Une **fonction pure** est une fonction qui possède les propriétés suivantes :

- Sa valeur de retour est la même pour les mêmes arguments (pas de variation avec des variables statiques locales, des variables non locales, des arguments mutables de type référence ou des flux d'entrée).
- Son évaluation n'a pas d'effets de bord (pas de mutation de variables statiques locales, de variables non locales, d'arguments mutables de type référence ou de flux d'entrée-sortie).

Une **fonction pure** est ainsi un analogue informatique d'une fonction mathématique. Certains auteurs, particulièrement de la communauté des langages impératifs, utilisent le terme "pure" pour désigner les fonctions qui ont uniquement la propriété susmentionnée.

Les boucles ne sont pas autorisées et on utilise la **récursion** à la place. La **récursion terminale**, un cas particulier de récursivité assimilée à une itération, est préféré à la récursion simple. (La récursion terminale n'est pas supportée par Python.)

Une **fonction anonyme**, aussi appelée **lambda** expression ou fonction lambda est une fonction n'ayant pas de nom.

Un **objet immuable**, en programmation orientée objet et fonctionnelle, est un objet dont l'état ne peut pas être modifié après sa création. Ce concept est à contraster avec celui d'objet variable.

A retenir :

Le paradigme de la programmation fonctionnelle comporte plusieurs concepts. Le premier est celui d'un seul calcul qui se fait sans modifier les valeurs déjà construites (variables, structures de données) mais en construisant plutôt de nouvelles valeurs. Le second est celui d'une programmation dans laquelle les fonctions sont des valeurs comme les autres, passés en arguments, renvoyé comme résultats ou encore stockées comme des données.