Définition 1 *Un **paradigme** est une manière d'implémenter des **concepts** (de programmation) dans un langage de programmation. On parle aussi de **style** de programmation.*

Remarques 2

1. La plupart des langages de programmation modernes implémentent plusieurs paradigmes. Il est donc parfois difficile de séparer les paradigmes au sein d'un langage de programmation, de plus leur implémentation peuvent différer d'un langage à l'autre.
2. L'évolution des langages de programmation est celui d'une ouverture de plus en plus grande vers un public de programmeurs de moins en moins expert. Ceci est possible en créant des langages de plus en plus abstraits et proche des langages « naturels » (humains).

Style impératif

C'est le plus ancien style de programmation. Il est très proche du langage machine. Ces caractéristiques sont :

① ▶ utilisation de variables ;

② ▶ branchements conditionnels, boucles for et while ;

③ ▶ branchements non conditionnels (*goto* et *label*) ;

④ ▶ Les instructions sont exécutées dans l'ordre du code source.

Exemples de langages implémentant ce paradigme

- les langages de script (bash, VBS, CMD,...)
- Fortran, C, Java, Python

Style structurée

Le style structuré est un sous-style du style impératif apparu en 1970 suite à l'article de Dijskra « *GO TO statement considered harmful* ». Ce style privilégie l'utilisation des modules (aussi appelés procédures ou fonctions dans certains langages). La **maintenance** du code en est grandement facilitée. Ce langage privilégie une approche **top-down** c'est-à-dire une hiérarchie dans l'équipe de développement. Les caractéristiques de ce style sont :

① ▶ des bouts de codes autonomes et facilement réutilisables (les modules) ;

② ▶ Les instructions suivent les appels des différents modules ;

③ ▶ Ce style forme un cadre plus rigoureux que les branchements non conditionnels (goto et label. Ces derniers n'existant plus dans les langages modernes).

Exemples de langages :

- Pascal
- C/C++, Java, Python

Style fonctionnel

Le style de programmation fonctionnel applique systématiquement le concept de fonction : tous les calculs sont effectués lors de l'évaluation d'une fonction. C'est un sous-style du style déclaratif. Ce style crée un cadre idéal pour la **preuve de programme**. Les caractéristiques sont :

① ▶ pas d'utilisation de variables ;

② ▶ Les instructions sont exécutées dans l'ordre d'appel des fonctions (qui n'est pas l'ordre du code) ;

③ ▶ Si on définit les **interfaces** des fonctions, ce style permet paralléliser le développement aisément.

- Lisp, Scheme, (JavaScript)
- (O)CAML, Haskwell, (Python),...

Style orienté objet

Pour le style Orienté Objet, le programmeur crée des « objets » qui correspondent aux concepts mis en jeu pour résoudre le problème posé. Chaque objet possède des **propriétés** et des **méthodes** qui permettent de créer et de manipuler l'objet. Tous les objets appartiennent à une **classe** (de référence). Les classes sont hiérarchisées ce qui permet d'obtenir un **héritage** et donc d'éviter de réécrire du code déjà écrit.
Quelques points importants concernant les langages implémentant ce paradigme :

① ▶ le code est interprété par une machine virtuelle (souvent compilé en **byte code**) ;

② ▶ Comme pour le style fonctionnel, l'utilisation des interface permet un développement parallèle aisé.

③ ▶ L'écriture des programmes passe par un phase de modélisation sous forme d'objet. Cette abstraction est un grand avantage pour de gros projets et un désavantage pour de petits.

④ ▶ Les **modificateurs** (quand ils existent) permettent une sécurité des données.

Exemples

- Smalltalk
- C++, Java, Visual Basic .NET, Python, (JavaScript)

Autres styles

Style logique : Prolog
Style déclaratif : SQL
Style concurrent ()