

Chapitre II : Informatisation d'un problème

II.1 Informatique : est un domaine d'activité scientifique, technique, et industriel concernant le traitement automatique de l'information numérique au moyen des ordinateurs.

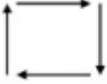




II.2 Ordinateur : machine automatique de traitement de l'information, obéissant à des programmes formées par des suites d'opérations arithmétiques et logiques.

II.3 Langage algorithmique : est un langage générique permettant de traiter des problèmes par concaténation d'instructions élémentaires. Il est à la base de tous les langages de programmation. Le langage algorithmique exprime les instructions résolvant un problème donné indépendamment des particularités d'un langage de programmation.

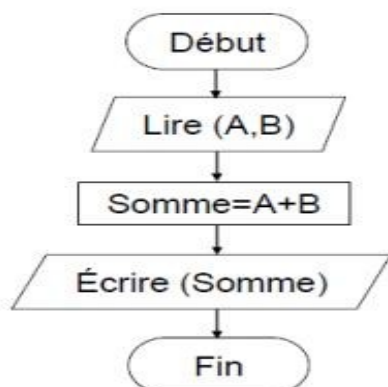
En générale on utilise deux types de notations pour représenter les algorithmes:

- L'organigramme
- Le pseudo-code

II.3.a L'organigramme : est un schéma fonctionnel qui présente les différentes parties d'un algorithme les unes à la suite des autres en utilisant des symboles graphiques pour visualiser l'exécution de l'algorithme et le cheminement des données.

Noms	Symbole	Définition
<i>Flèches</i>		Elles indiquent le sens du traitement (haut, bas, gauche, droite).
<i>Début / Fin</i>		Ce symbole indique le début ou la fin de l'organigramme
<i>Entrée / Sortie</i>		Ce symbole indique les données d'entrées et de sorties
<i>Boite de traitement</i>		Elle indique un traitement spécifique qui peut être exécuté
<i>Boite de décision (Test)</i>		Elle permet d'envoyer le traitement sur un chemin ou sur un autre, selon le résultat du test

Exemple : Somme de deux nombres A et B



L'écriture d'un algorithme avec un organigramme est rapidement abandonnée. La lisibilité devient rapidement difficile pour les algorithmes assez long.

II.3.b Le pseudo-code : est purement conventionnel, aucune machine n'est censée le reconnaître.
Exemple : Somme de deux nombres A et B

Algorithme *somme_de_deux_nombres*

Variables A, B, somme en entier

Début

Lire A

Lire B

somme \leftarrow A+B

Ecrire "La somme de A et B est : ", somme

Fin

II.4 Langage de programmation: est une notion conventionnelle destinée à formuler des algorithmes et produire des programmes informatiques qui les appliquent . Il permet de donner des ordres (instructions) à la machine. A chaque instruction correspond une action du processeur. L'intérêt est d'écrire des programmes (suite consécutive d'instructions) destinées à effectuer une tâche donnée.

Un langage de programmation est mis en œuvre par un traducteur automatique : compilateur ou interprète.

II.4.a Compilateur : est un programme informatique qui transforme dans un premier temps un code source écrit dans un langage de programmation donnée en un code cible qui pourra être directement exécuté par un ordinateur, à savoir un programme en langage machine .

II.4.b Langage machine : suite de bits qui est interprétée par le processeur d'un ordinateur exécutant un programme informatique. C'est le langage natif d'un processeur, c'est à dire le seul qu'il puisse traiter . Il est composé d'instructions et de données à traiter codées en binaire.

- Un bit (binary digit) = 0 ou 1 (2 états électriques)
- Une combinaison de 8 bits = 1 octets \rightarrow possibilités qui permettent de coder tous les caractères alphabétiques, numériques, et symboles tels que ?,*&,...
- Le code ASCII (American Standard Code for Information Interchange) donne les correspondances entre les caractères alphanumériques et leur représentation binaire, Ex : A = 01000001, ? = 00111111
- Les opérateurs logiques et arithmétiques de base (addition, multiplication, ...) sont effectuées en binaire

Il existe plusieurs types de langages de programmations on parle de paradigme de programmation : les langages impératifs (procéduraux) : suite des instructions à réaliser dans l'ordre d'exécution.

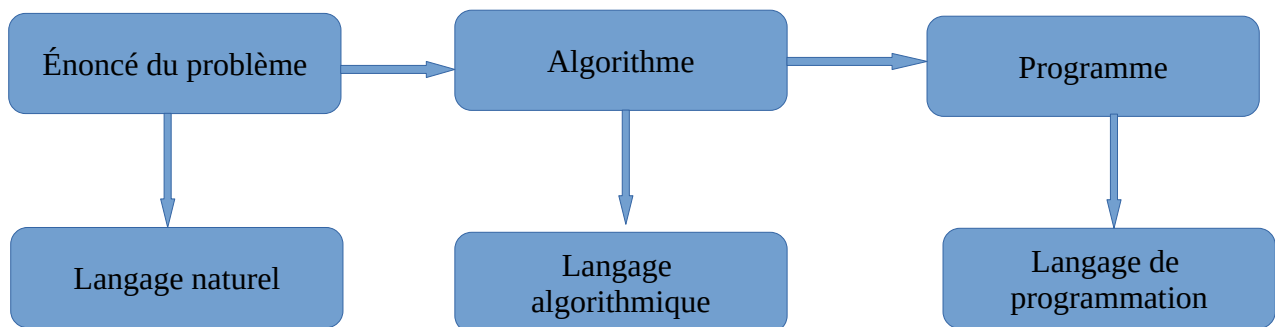
Exemple : Fortran, Cobol, Pascal, C,

les langages déclaratifs : règles de calcul et vérification de propriété sans spécification d'ordre :

fonctionnels : les règles sont exprimés par des fonctions. Exemple : Lisp, Caml

logique : les règles sont exprimées par des prédicats logiques. Exemple : Prolog

les langages orientés objets : modéliser par entités ayant des propriétés et des interactions possibles. Exemple : C++, java, python ...

II.5 Étapes de réalisation d'un programme :*Fig II.1 étapes de réalisation d'un programme*

La réalisation de programmes passe par l'écriture d'algorithmes. D'où l'intérêt de l'algorithmique.

II.6 Les différents éléments d'un algorithme sont :

- **Données** : ce qui doit être donné à l'algorithme
- **Résultats** : ce que doit produire l'algorithme
- **L'algorithme** : les grandes étapes des traitements et calculs

II.7 Structure d'un algorithme :

Nom de l'algorithme // partie en-tête qui précise le nom de l'algorithme

Déclaration des variables // partie déclaration des variables

Début // partie traitement des données

Algorithme (définir les actions à suivre pour résoudre un problème donné)

Fin

Exemple :

Écrire un algorithme qui permet d'afficher « Bonjour ».

Algorithme algo-bonjour

Début

écrire (' 'bonjour' ');

Fin