Université Moulay Ismail
Faculté des Sciences et Techniques
Département d'Informatique
Errachidia

Année universitaire : 2020/2021
Filière MIP (S1) : Module I111

Chapitre II: Les éléments de base d'un algorithme

1. Généralités sur l'Algorithmique

L'algorithmique est un terme d'origine arabe, hommage à Al Khawarizmi(780-850) auteur d'un ouvrage décrivant des méthodes de calculs algébriques.
Un algorithme est une méthode de résolution de problème énoncée sous la forme d'une série d'opérations à effectuer. La mise en œuvre de l'algorithme consiste en l'écriture de ces opérations dans un langage de programmation et constitue alors la brique de base d'un programme informatique.

- Une recette de cuisine est un algorithme!
- Le mode d'emploi d'un magnétoscope est aussi un algorithme!

Un algorithme, c'est une suite d'instructions, qui une fois exécutée correctement, conduit à un résultat donné.
Pour fonctionner, un algorithme doit donc contenir uniquement des instructions compréhensibles par celui qui devra l'exécuter (l'ordinateur).
L'ADN, qui est en quelque sorte le programme génétique, l'algorithme à la base de construction des êtres vivants, est une chaîne construite à partir de quatre éléments invariables. Ce n'est que le nombre de ces éléments, et l'ordre dans lequel ils sont arrangés, qui vont déterminer si on obtient une puce.
Les ordinateurs eux-mêmes ne sont fondamentalement capables d'exécuter que quatre opérations logiques :

1 **l'affectation de variables**
2 **la lecture / écriture**
3 **les tests**
4 **les boucles**

Un algorithme informatique se ramène donc toujours au bout du compte à la combinaison de ces quatre petites briques de base. Il peut y en avoir quelques-unes, quelques dizaines, et jusqu'à plusieurs centaines de milliers dans certains programmes.
La taille d'un algorithme ne conditionne pas en soi sa complexité: de longs algorithmes peuvent être finalement assez simples, et de petits algorithmes peuvent être très compliqués.
L'informatique est la science du traitement automatique de l'information. Pour cela il faut:

1. modéliser cette information,
2. définir à l'aide d'un formalisme strict les traitements dont elle fera l'objet.
3. et enfin traduire ces traitements dans un langage compréhensible par un ordinateur.

Les deux premiers points concernent l'algorithmique, alors que le dernier point relève de ce que l'on nomme la programmation.
L'écriture d'un programme consiste généralement à implanter une méthode de résolution déjà connue et souvent conçue indépendamment d'une machine pour fonctionner aussi bien sur toutes les machines ou presque. Ainsi, ce n'est pas le programme mais la méthode qu'il faut étudier pour comprendre comment traiter le problème. Le terme algorithme est employé en informatique pour

décrire une méthode de résolution de problème programmable sur machine. Les algorithmes sont la « matière » de l'informatique et sont l'un des centres d'intérêt de la plupart, sinon la totalité, des domaines de cette science.

2. Application en informatique

Lorsqu'il s'agit de résoudre des problèmes à l'aide de l'outil informatique, on doit suivre les étapes suivantes :

a) Analyse du problème

Dans cette phase, il faut faire l'inventaire et la description des données connues et des résultats cherchés, identifier les relations entre elles, puis décrire les traitements nécessaires à effectuer.

→ **Algorithme**.

b) Traduction de l'algorithme

L'algorithme élaboré doit être écrit dans un langage compréhensible par la machine, on parle alors de programme, et le code utilisé pour l'écrire sur ordinateur est appelé **langage de programmation**.

Le langage de programmation permet d'écrire dans un langage compréhensible par la machine les opérations que l'ordinateur doit effectuer.

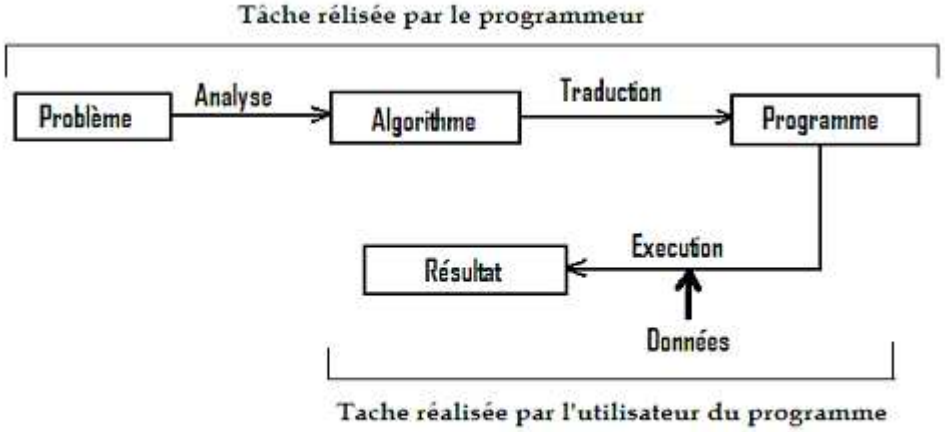


Schéma de résolution informatique d'un problème

3. Notion de programme

a) Le code source

Le code source est le texte du programme en langage de haut niveau (C ou Python par exemple). Si l'on dispose du code source d'un programme, il est toujours possible d'étudier son fonctionnement et de le modifier en cas de besoin.

La traduction en binaire du code source génère un programme exécutable par l'ordinateur, Ce travail est réalisé par un programme spécialisé appelé **compilateur**.

b) Le compilateur

Le compilateur examine les instructions écrites par le programmeur et les transforme en langage binaire, compréhensible par le processeur. Un texte écrit dans un langage doit être compilé à l'aide d'un compilateur approprié à ce langage précis.

Exemple : Un texte écrit en langage C doit être compilé par un compilateur C.

Remarques :

– Il est impossible de générer le code source à partir du programme exécutable.

– Un logiciel "Open Source" est un programme pour lequel on dispose du code source.

– Un logiciel "fermé" est un programme qui est la propriété d'une personne ou d'une société qui ne souhaite pas diffuser le code source.

c) L'interpréteur

Dans certains langages, le code source n'est pas préalablement traduit en langage machine par un compilateur. Dans ce cas, la transformation en langage machine se fait au moment de l'exécution du programme : un interpréteur traduit le programme, ligne par ligne.

Exemple : Le langage Python est un langage interprété.

Il est nécessaire de disposer d'un interpréteur approprié pour chaque langage utilisé. Un programme écrit en langage Python doit être traité par un interpréteur Python.

4. Notion de données

Les algorithmes agissent sur des données, qui peuvent varier ou rester constantes.

a) Les variables

Une variable est le nom d'un espace mémoire utilisé pour mémoriser une valeur pour une utilisation ultérieure. Elle est caractérisée par :

- **Son nom** : Appelé aussi identificateur, il est formé d'une suite de lettres, de chiffres et de traits de soulignement _, dont le premier caractère est obligatoirement une lettre. Pour des raisons de clarté, il est vivement recommandé d'utiliser des noms significatifs.
- **Son type** : il caractérise les valeurs que peut prendre cette donnée ainsi que les actions autorisées sur celle-ci.
  Par exemple, une donnée numérique entière ne peut pas recevoir une valeur réelle.
- **Sa valeur** : c'est la valeur stockée dans la variable à un moment donné.

Exemples :

1. Identificateur : Age
   Type : Entier
   Valeur : 28
2. Identificateur : Salaire
   Type : Réel
   Valeur : 8760,50

Syntaxe de déclaration :

Identificateur : Type

Exemple :

Age : Entier
Longueur, hauteur : Réel

b) Les constantes

Une constante est une donnée dont la valeur est précisée au début de l'algorithme et qui ne varie pas durant le déroulement de celui-ci.

Exemples :

1. Pi est une constante utilisée pour le calcul de l'aire d'un disque et le périmètre d'un cercle.
2. Le Taux de la Valeur Ajoutée (TVA) est fixé, en général, à 20% du prix d'un produit.

Syntaxe :

Constante nom_constante = valeur

Exemple :

Constante Pi ← 3.14, TVA ← 0.20

c) Les types simples de données

Le tableau suivant présente les types simples ainsi que leur description :

TYPE	DESCRIPTION
ENTIER	Une donnée de ce type prend ses valeurs dans l'intervalle [-32768, 32767]
REEL	Elle prend ses valeurs dans l'échelle $2,9 * 10^{-39}$ à $1,7 * 10^{+38}$ avec 11 chiffres après la virgule.
LOGIQUE	Elle peut prendre la valeur **Vrai** ou **Faux**.
CARACTÈRE	Les valeurs de ce type contiennent des caractères alphabétiques ou numériques.
CHAINE DE CARACTÈRES	Elle est utilisée pour manipuler des chaines de caractères représentants des mots ou des phrases.

5. Les instructions simples

a) L'affectation

L'opération d'affectation consiste à attribuer une valeur à une variable. Elle est représentée par une flèche orientée à gauche : ←

Exemples :

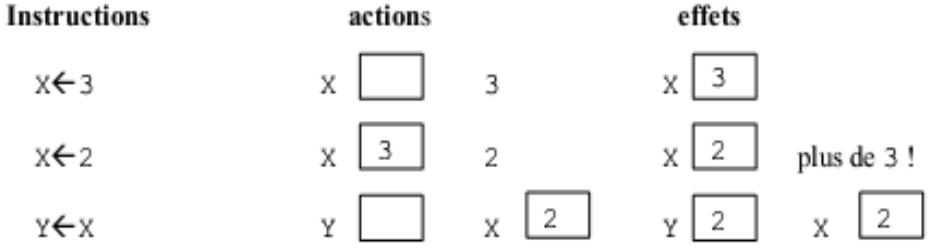
- A ←3 (A doit être de type ENTIER ou REEL).
- B ← 4 * A + 3,12 (B doit être de type REEL).
- Test ← A<B (Test doit être de type LOGIQUE).
- C ← 'Z' (C doit être de type CARACTÈRE).

Exemple :
X←3
Signifie mettre la valeur 3 dans la case identifiée par X. A l'exécution de cette instruction, la valeur 3 est rangée en X (nom de la variable).
On peut représenter la variable X par une boite ou case, et quand elle prend la valeur 3, la valeur 3 est dans la case X :

X 

X 

On remarque qu'une variable ne peut contenir à un instant donné qu'une seule valeur.
Utilisations :
Voici quelques effets déclenchées par l'utilisation de l'affectation (←) :



b) Les opérateurs arithmétiques et logiques

- Les opérateurs arithmétiques sont les suivants : + (Addition), - (soustraction), * (Multiplication), et / (Division).
- Les opérateurs de comparaison usuels sont : = (égal), <> (différent), < (inférieur), > (supérieur), <= (inférieur ou égal) et >= (supérieur ou égal).

Exemples : Soit Test une variable de type LOGIQUE et A et B des ENTIERS

Test ← (A > B) Test ← A = B

- Autres opérateurs logiques binaires : soit **Exp1** et **Exp2** deux expressions logiques
  - ✓ **ET** : Le résultat de (Exp1 ET Exp2) est vrai si les deux expressions logiques simples sont vraies.
  - ✓ **OU** : Le résultat de (Exp1 OU Exp2) est vrai si l'une des expressions logiques simples est vraie.
  - ✓ **NON** : Le résultat de (NON Exp1) est vrai si l'expression logique simple est fausse et vice versa.
- **Les tables de vérité**

A	B	A et B	A ou B	Non A
VRAI	VRAI	VRAI	VRAI	FAUX
VRAI	FAUX	FAUX	VRAI	FAUX
FAUX	VRAI	FAUX	VRAI	VRAI
FAUX	FAUX	FAUX	FAUX	VRAI

Exemple :

Soit **Test** une variable de type LOGIQUE et **A**, **B** et **C** des ENTIERS

Test ← (A ET B)
Test ← (A ET B) OU C
Test ← NON C

c) La division entière

- La fonction DIV permet de donner le résultat de la division entière d'un nombre par un autre.

Exemple : 7 DIV 2 = 3

- La fonction MOD (se lit Modulo), permet de donner le reste de la division entière d'un entier par un autre.

Exemple : 7 MOD 2 = 1

d) Lecture d'une donnée

C'est l'action qui permet à l'utilisateur de fournir à l'algorithme les valeurs de données variables.

Syntaxe : LIRE (V1, V2, … Vn)

Où V1, V2,… et Vn sont des variables pas forcément de même type.

Lors de l'exécution de l'action LIRE, l'algorithme attend que l'utilisateur fournisse, à partir du clavier, les valeurs respectives V1, V2, … et Vn.

Exemples :

LIRE (N)
LIRE (A, B, C)

e) Ecriture d'une donnée

C'est l'algorithme qui permet à l'algorithme d'afficher pour son utilisateur des messages ou des valeurs sur un périphérique de sortie (l'écran par exemple).

Syntaxe : ECRIRE (Val1, Val2,… Valn)

Où Val1, Val2,… Valn sont des constantes, des variables, des expressions, ou des chaines de caractères.

Exemples :

ECRIRE ('Donner le rayon du cercle')

ECRIRE ('La surface du cercle est :', Pi*R*R)

6. Enchaînement séquentiel – Notion de bloc

On dit que des actions s'enchaînent séquentiellement, quand la fin d'une action déclenche l'exécution de l'action suivante. On appellera bloc d'actions un ensemble d'actions enchaînées séquentiellement.

Notation :

```
Début
      Action 1
      Action 2
        ……
      Action N
Fin
```

Bloc d'actions

L'exécution de ce bloc ne peut commencer que par la première action et ne peut se terminer que par la dernière.

7. Structure d'un algorithme

Un algorithme se compose des parties suivantes :

- Un **en-tête** constitué du mot **Algorithme**, suivi d'un **nom** identifiant l'algorithme.
- Un **bloc de déclaration des données**, divisé en deux parties :
  - ✓ La **partie des variables** contiendra la déclaration des données de nature Variable.
  - ✓ La **partie des constantes** contiendra la déclaration des données de nature constante.
- Un **Bloc** qui englobe toutes les instructions constituant l'algorithme.

Exemple :

On se propose d'écrire un algorithme qui calcule la surface d'un disque.

```
Algorithme  Surface_Disque                          } En-tête de l'algorithme
Constantes  Pi ← 3.14                               }
Variables   R, S : REEL                             } Bloc de déclaration des données
DEBUT                                               }
        ECRIRE ('Donnez le rayon d'un disque :')    }
        LIRE (R)                                    }
        S ← Pi * R * R                              } Bloc d'instruction
        ECRIRE ('La surface est :', S)              }
FIN
```