Chapitre II : Algorithme

1. Définitions

1.1.Algorithme :

Un algorithme est une suite finie d'opérations élémentaires, à appliquer dans un ordre déterminé, à des données. Sa réalisation permet de résoudre un problème donné.

1.2.Programme

Séquences d'instructions et de données enregistrées sur un support et susceptibles d'être traitée par un ordinateur.
Le programme est la traduction d'un algorithme dans un Langage de programmation qui impose une syntaxe rigoureuse

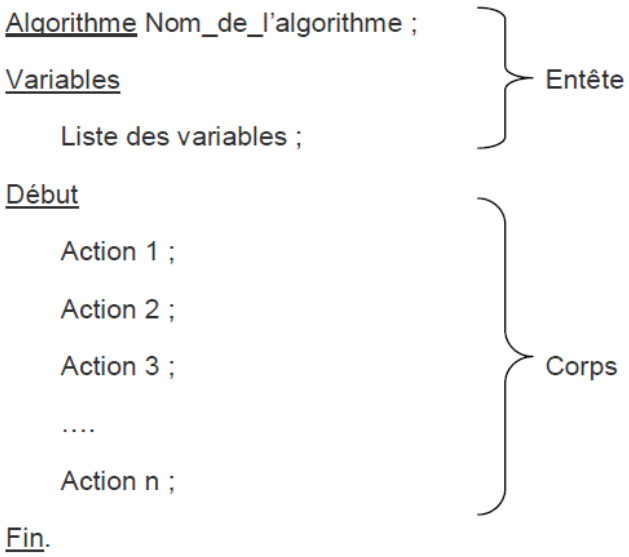
1.3. Langage de programmation

Le langage de programmation est l'intermédiaire entre l'humain et la machine, il permet d'écrire dans un langage proche de la machine mais intelligible par l'humain les opérations que l'ordinateur doit effectuer. Ainsi, étant donné que le langage de programmation est destiné à l'ordinateur, il doit donc respecter une syntaxe stricte.
C'est un ensemble de vocabulaire (lettres, chiffres, symboles, caractère spéciaux,..) et un ensemble de règles (syntaxe ou grammaire) qui permet de réunir les éléments du vocabulaire pour former des phrases (ligne de programme) correctes

2. Caractéristiques d'un algorithme

L'algorithme est un moyen pour le programmeur de présenter son approche du problème à d'autres personnes. En effet, un algorithme est l'énoncé dans un langage bien défini d'une suite d'opérations permettant de répondre au problème. Un algorithme doit donc être :

- ***lisible:*** l'algorithme doit être compréhensible même par un non-informaticien.
- ***de haut niveau:*** l'algorithme doit pouvoir être traduit en n'importe quel langage de programmation, il ne doit donc pas faire appel à des notions techniques relatives à un programme particulier ou bien à un système d'exploitation donné
- ***précis:*** chaque élément de l'algorithme ne doit pas porter à confusion, il est donc important de lever toute ambiguïté
- ***concis:*** un algorithme ne doit pas dépasser une page. Si c'est le cas, il faut décomposer le problème en plusieurs sous-problèmes
- ***structuré:*** un algorithme doit être composé de différentes parties facilement identifiables à savoir : une entête où apparaissent le nom de l'algorithme ainsi que les déclarations des objets manipulés puis le corps de l'algorithme qui commence par "Début" et se termine par "Fin. Entre ces deux mots clés se trouvent les actions ordonnées à exécuter par la machine.



3. Définition d'une variable et ses caractéristiques

Dans un programme informatique, on est toujours besoin de stocker provisoirement des valeurs. Il peut s'agir de données stockées sur un support (disque dur, CD, diskette..) ou fournies par l'utilisateur (frappées au clavier). Il peut aussi s'agir de résultats obtenus par le programme, intermédiaires ou définitifs. Ces données peuvent être des nombres, du texte, etc. donc quand on a besoin de stocker une information au cours d'un programme, on doit utiliser une ***variable***.

Une variable est une entité qui contient une information. Elle possède :

- Un nom, on parle d'identifiant ;
- Un type qui caractérise l'ensemble des valeurs que peut prendre la variable :
  - ✓ Nombre Entier
  - ✓ Nombre flottant (Réel)
  - ✓ Booléen (uniquement valeurs *Vrai* ou *Faux*)
  - ✓ Caractère (alphabétique, numérique)
  - ✓ Chaine de caractères (mot ou phrase)

A un type donné, correspond un ensemble d'opérations définies pour ce type.

4. Opérateur, opérande et expression

- Un **opérateur** est un symbole d'opération qui permet d'agir sur des variables ou de faire des "calculs"
- Un **opérande** est une entité (variable, constante ou expression) utilisée par un opérateur
- Une **expression** est une combinaison d'opérateur(s) et d'opérande(s), elle est évaluée durant l'exécution de l'algorithme, et possède une valeur (son interprétation) et un type

  Exemple :

  dans a + b

  a est l'opérande gauche

  + est l'opérateur

  b est l'opérande droite

  a + b est appelé une expression

Si par exemple a vaut 2 et b 3, l'expression a + b vaut 5
Si par exemple a et b sont des entiers, l'expression a + b est un entier

4.1. Les opérateurs

4.1.1. Opérateurs numériques :

Ce sont les quatre opérations arithmétiques tout ce qu'il y a de classique.
+ : addition
- : soustraction
* : multiplication
/ : Division
Avec en plus pour les entiers **div** et **mod**, qui permettent respectivement de calculer une division entière et le reste de cette division.
Mentionnons également le ^ qui signifie « puissance ». **45 au carré** s'écrira donc **45 ^ 2**.
Enfin, on a le droit d'utiliser les parenthèses, avec les mêmes règles qu'en mathématiques. La multiplication et la division ont « naturellement » priorité sur l'addition et la soustraction. Les parenthèses ne sont ainsi utiles que pour modifier cette priorité naturelle.

4.1.2. Opérateurs logiques (ou booléens) :

Il s'agit du ET, du OU, du NON et du OuExclusif XOR.

5. Actions de base sur les variables :

Dans un programme, les données sont manipulées via des variables

- une variable est une case mémoire
- une variable est désignée par un nom (identifiant)
- une variable a un type de donnée (implicite dans certains langages)
- une variable contient une valeur du type et cette valeur peut varier

Cycle de vie d'une variable :

- déclaration de la variable (nom et type)
- affectations de valeurs à la variable
- suppression de la variable (souvent automatique)
-

5.1. Déclaration d'une variable

La déclaration de variables est très simple, elle se fait de la façon suivante:

variable *Nom_de_ma_variable: type*

Nom_autre_variable: autre_type

5.1.1. Les identificateurs

On appel identificateurs les noms des variables (et des fonctions). Ils sont composés d'une suite de lettres ***non accentuées*** et de chiffres. Un identificateur doit être significatif pour faciliter la compréhension du programme. Le premier caractère doit être une lettre. Le symbole '_' est aussi considéré comme une lettre. L'ensemble des symboles utilisables est donc: {0,1,2,...,9,A,B,...,Z,_,a,b,...,z}

REM :

- En langage C, Le premier caractère doit être une lettre (ou le symbole '_'.
- C distingue les majuscules et les minuscules, ainsi:
- **'Nom_de_variable'** est différent de **'nom_de_variable'**
- La longueur des identificateurs n'est pas limitée, mais C distingue 'seulement' les 31 premiers caractères.

5.1.2. Les types de variables

On algorithmique les types de données de base sont : l'entier, le réel, le booléen, le caractère et la chaine de caractère. On langage C nous pouvons résumé ces types avec leurs spécificités comme suit :

définition	*description*	*domaine min*	*domaine max*	*nombre d'octets*
char	caractère	-128	127	1
short	entier court	-32768	32767	2
int	entier standard	-32768	32767	2
long	entier long	-2147483648	2147483647	4
unsigned char	caractère	0	255	1
unsigned short	entier court	0	65535	2
unsigned int	entier standard	0	65535	2
unsigned long	entier long	0	4294967295	4

définition	*précision*	*mantisse*	*domaine min*	*domaine max*	*nombre d'octets*
float	simple	6	$3.4 * 10^{-38}$	$3.4 * 10^{38}$	4
double	double	15	$1.7 * 10^{-308}$	$1.7 * 10^{308}$	8
long double	suppl.	19	$3.4 * 10^{-4932}$	$1.1 * 10^{4932}$	10

REM : En C il n'existe pas de type spécial pour variables booléennes. Tous les types de variables numériques peuvent être utilisés pour exprimer des opérations logiques

5.2. La lecture

L'ordre LIRE permet à l'ordinateur d'acquérir des données à partir de l'utilisateur, dans des cases mémoire bien définies (qui sont les variables déclarées).

Rappel

Les variables sont des cases mémoire, supposées contenir un type de données, nommées par le nom de variable.

Syntaxe

LIRE (variable1, [[variable2] …])

Remarques :

1. La saisie se fait uniquement dans des variables. Ce sont les cases (cellules) qui pourront accueillir les données correspondantes.

2. La donnée à introduire doit être de même type que la variable réceptrice.

3. en langage C on utilise la fonction **scanf** (dont le nom vient de l'anglais scan formatted) qui est une fonction de la bibliothèque standard du langage C. Déclarée dans l'entête `<stdio.h>`, cette fonction peut être utilisée pour la saisie de données formatées, qu'il s'agisse de lettres, de chiffres ou de chaînes de caractères.

Exemple ***: int x ;*** on déclare un entier x

Scanf (x) ; on lit la valeur de X

5.2. L'écriture

Cette action permet de communiquer un résultat ou un message sur écran ou sur imprimante pour l'utilisateur.

Syntaxe

ECRIRE (paramètre1 [[,paramètre2]…])

Paramètre = variable | expression | constante

Constante = nombre | message

Exemples

ECRIRE (" La valeur de 3*2 est égale à ", 3*2)

ECRIRE (" La moyenne est = ", MOY)

Remarque : en langage C, La fonction **printf** est utilisée pour transférer du texte, des valeurs de variables ou des résultats d'expressions vers le fichier de sortie standard *stdout* (par défaut l'écran).

5.3. L'affectation

C'est l'action de charger une valeur dans une variable. Cette valeur peut elle-même être une variable, le résultat d'une expression arithmétique ou logique ou une constante.

Syntaxe

Affectation d'une valeur à une variable : **Nom-variable ← valeur ;**
Affectation d'une variable à une variable : **Nom-variable1 ← Nom-variable2 ;**
Affectation du résultat de calcul à une variable :
Nom-variable ← nom-variable1 opérateur nom-variable2 ;

Ou encore l'affectation du résultat de plusieurs calcul à une variable :
Nom-variable ← nom-variable1 opérateur1 nom-variable2 … opérateur-n nom-variable n ;
Exemple
La résolution d'une équation de second degré se fait par calcul du discriminant delta :
delta ← b*b – 4*a*c ;
Dans le cas de delta > 0 la machine réalise le calcul des deux solutions X_1 et X_2 en exécutant les instructions :

$X_1 \leftarrow -b + \sqrt{delta} / 2*a$;

$X_2 \leftarrow -b - \sqrt{delta} / 2*a$;

Remarque

- L'affectation ne vide pas la variable émettrice (à notre droite) de sa valeur. Par contre, le contenu de la variable réceptrice est écrasé.
- En langage C , l'opérateur d'affectation simple est "=" il signifie "affecter à".

Syntaxe :

<Destination> = <Source>; Où <Source> et <Destination> doivent être de même nature et où <Source> est une expression et <Destination> une variable ;

Exemple

int x, y ;

x=5 ; //x=5

y=3 ; //y=3

x=y ; //x=3 (valeur de y)