ALGORITHMIQUE ET STRUCTURES DE DONNEES
بنية المعطيات و الخوارزميات

1- Introduction :

La résolution automatique d'un problème s'effectue en plusieurs étapes :

(A) Etape d'analyse : permet de comprendre le problème en répondant aux 3 questions suivantes :

1. Quelles sont les entrées (les **données**) ?
2. Quelles sont les méthodes (**traitements** ou **opérations**) qui permettent d'aboutir aux résultats recherchés à partir des données qu'on a ?
3. Quelles sont les sorties (Les **résultats** attendus) ?

Exemples : Calcul de la surface et du périmètre d'un carré de longueur de coté c.
Calcul de la moyenne du module INF1 pour un étudiant.

(B) Après la réponse aux questions, on passe à l'étape de **l'écriture de l'Algorithme**, qui comprend trois blocs (étapes) :

1. **Lecture** des **données**.
2. Les **opérations** (traitements).
3. **L'écriture** des sorties (**résultats**).

(C) Traduire l'algorithme en programme, écrit dans un **langage de programmation évolué.**
Exemples de langages de programmation connus : Fortran, Pascal, Java, Matlab, C, C++, Delphi..

(D) Edition : On introduit le programme dans l'ordinateur (saisie au clavier).
E) Compilation et exécution du programme.

2. Définition de l'algorithme :

Le terme Algorithme vient du nom du mathématicien Abou Jaafar Mohamed Ibn Moussa **Al Khawarizmi**.
Un Algorithme est une suite ordonnée d'instructions qui permettent la résolution d'un problème. L'ordre des instructions est important (exécution séquentielle).
Instruction : Action agissant sur un ou plusieurs objets (données).
En programmation, il existe plusieurs types d'actions et d'objets. Les objets ont des types (entier, réel, ...).
Dans ce chapitre, on abordera les instructions, les objets et les types de base.

3. LES OBJETS DE BASE : Les objets de base (simples) manipulés dans un algorithme sont :

- Les constantes.
- Les variables.
- Les expressions.

3.1. Les constantes

Une constante est un objet qui a une valeur qui ne change pas tout au long de l'exécution du programme :

- Un nom fixe.
- Un type fixe.
- Une valeur fixe.

Exemple : Pi = 3.14 Nom : Pi ; Type : Réel ; Valeur : 3,14.
N= $6.023 * 10^{23}$: nombre d'Avogadro
C= $3*10^{8}$: célérité de la lumière.

3.2. Les variables

Une variable est un objet dont la valeur peut être modifiée tout au long de l'exécution du programme.
- Un nom fixe.
- Un type fixe.
- Une valeur variable.

Exemple : La Note d'un examen.
Nom variable : Note ; **Type variable**: Réel ; **Valeurs de la variable** : 12 10.25 18.5
Remarque :
1- Puisque la valeur d'une variable peut changer plusieurs fois dans un programme, l'ordinateur réserve une place dans la mémoire centrale (MC) pour chaque variable. Une variable est donc un espace mémoire qui va contenir des données au fur et à mesure que le programme avance dans son exécution.
Cependant, à un instant donné, une variable ne peut contenir qu'une seule valeur.
2- Les **noms** des **constantes** et des **variables** sont des **identificateurs**.

Comment choisir un identificateur :
 Un identificateur choisi par le programmeur est formé de lettres (A, B,……, Z; a, b,…, z) et des chiffres (0,1,……,9), exemple : X, Y, Somme, Note_Eval, et ne doit pas contenir un caractère spécial (>, <, !, ?, +,-,…) ni un espace.
 Un identificateur doit commencer par une lettre ; Exemple : A1, B2, Note1, Note_2,… ;
 Un identificateur doit être significatif ;
 Un identificateur doit être différent d'un mot clé ou réservé (utilisé par un langage de programmation : begin, end, program, if, else…)

3.3. Les expressions : Elles désignent le calcul arithmétique ou logique.

Une expression est formée de : Constantes, variables, opérateurs et de parenthèses.
Exemple : (3 * x) + 2 / (5 * x) - 3

A- Les expressions arithmétiques : X **OP** Y

- X,Y : Constantes ou variables .
- Op : + , - , * , / .

Exemples d'expressions arithmétiques : 2* X +Y,

B- Les expressions Logiques :

Une expression logique (booléenne) comporte des opérandes et des opérateurs de comparaison (>, <, ≥, ≤, =, ≠) ou bien des opérateurs booléens (et / ou/ non).

1- Les expressions Logiques simples : E1 **OP** E2
- E1, E2 : Expressions Arithmétiques.
- OP : >, <, ≥, ≤, =, ≠. **Exemple** : X > (Y*3)

2- Les expressions Logiques complexes : E1 **OP** E2
- E1, E2 : expressions Logiques simples.
- OP : ET, OU, NON (Opérateurs Logiques). **Exemple** : (X+ Y>4) et (Y>0)

N.B : On ne peut pas écrire (0 ≤ note ≤ 20) dans un algorithme, ça s'écrit : (0 ≤ note) et (note ≤ 20).

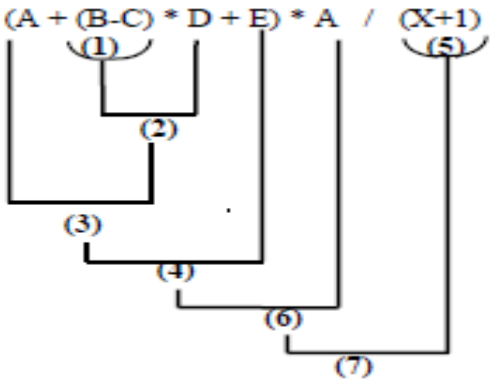
Les tables de vérité :
Soit X et Y deux variables de type booléen : true = vrai = 1 , false = faux = 0 .

X	Non (X)
Vrai	Faux
Faux	Vrai

X	Y	X et Y	X ou Y
Vrai	Vrai	Vrai	Vrai
Vrai	Faux	Faux	Vrai
Faux	Vrai	Faux	Vrai
False	Faux	Faux	Faux

Question : Quelle est la valeur de l'expression :
1+2*3-4 = ? (5, 3, -1, -4) ?
Pour aboutir aux résultats attendus de l'ordinateur, on doit apprendre sa façon de faire les choses ; à savoir sa façon d'évaluer (calculer) les expressions, pour se faire voici les opérateurs par lesquels l'ordinateur commence les calculs :
Les priorités des opérateurs :
1) Les fonctions (on va les voir plus tard).
2) Les parenthèses.
3) Non, - (le moins unitaire).
4) *, /, ET.
5) +, -, OU.
6) >, ≥, <, ≤, =, ≠.
7) En cas d'égalité des priorités, on commence par l'opérateur le plus à gauche.

Exemple :

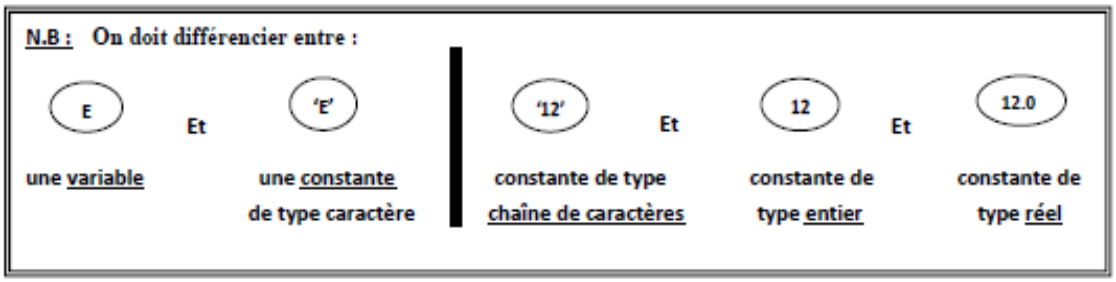


4. LES TYPES DE BASE:

Un type contient l'ensemble de valeurs que peut prendre une variable ; cet ensemble doit être fini.
Un type est caractérisé par :
- ses valeurs
- les opérations qui peuvent s'effectuer sur des variables ayant ce type.

1)- Le type entier : Un objet de type entier désigne une quantité indivisible (en mathématiques : **Z**, l'ensemble des entiers).
Exemple : Nombre de personnes, nombre de boites,…
Un entier est représenté sur 16 bits (2 Octets).
2)- Le type réel : Un objet est réel si sa valeur est fractionnaire (en mathématiques: **R,** l'ensemble des
réels).
Exemple : - Valeur de π (3,14).
- Une surface.
- Une moyenne.
- Un prix.
Un réel est représenté sur 4 ou 6 octets.
3)- Le type caractère : Un objet de type caractère est:
- Une lettre (A,….., Z ,a,…..,z) ;
- Un chiffre (0, 1 , ….. ,9) ;
- Un caractère spécial (+, -, *, /, ' ', ?, !, ...).

Remarque : Une constante de type caractère est mise entre apostrophes.



4)- Le type chaîne de caractères:
Un objet de type chaîne caractères est une suite de caractères; Chaque caractère sera codé sur un octet.
Exemple : '1ERE ANNEE': c'est une **constante** de type **chaîne de caractères**, elle sera représentée sur 10
octets ; Le blanc (**espace**) est **un caractère**.

5)- Le type booléen (logique):
Un objet de type booléen peut prendre uniquement deux valeurs : VRAI ou FAUX.

5. LES INSTRUCTIONS DE BASE: Ce sont les instructions qui manipulent les objets de base.
1)- L'instruction de saisie ou de lecture (lire) : elle permet d'attribuer une valeur à une variable et la stocker dans sa zone mémoire lors de l'exécution du programme; cette valeur provient de l'extérieur de l'ordinateur (clavier).

Syntaxe :

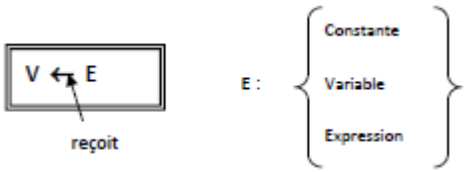
```
Lire (V)
ou
Lire (V1, V2,.....,Vn)
```

V : variable
Vi : variables

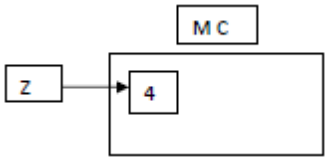
Autres syntaxes possible : SAISIR, ENTRER

Remarque : D'après la définition :
a) Il est interdit de lire une constante ; Si x=3 déclarée comme constante, lire(x) est impossible et lire ('Bonjour') est impossible.
b) Il est interdit de lire une expression arithmétique ou logique; Lire(a+b) est impossible.

2) L'instruction d'affectation (⟵) : est l'instruction qui permet de stocker une valeur dans une variable.
Syntaxe :



Exemple :
Z ⟵ (x + y)*2 veut dire : calculer la valeur de l'expression (x+y)*2 puis donner le résultat à la variable Z.



Si x= 1, et y = 1, alors :

Exemples d'actions d'affectation correctes:
a1←b
b ← 3*5
y ← 2.5
Exemples d'actions d'affectation incorrectes:
a+1← 3; (a+1 n'est pas un identificateur de variable)
3 ← a ; (On ne peut pas attribuer une valeur à une constante, 3 est une constante et non une varible) si on veut le contraire on écrit : a ← **3**
a←**b**←**c**; (on ne peut pas combiner, ce sont deux instructions d'affectation)
Correction :
{a ← **b**
b ← c}

Règle : Le type de la **valeur** doit être du même type de celui de la **variable**.

Exceptions à la règle précédente :
1. On peut affecter une valeur de type **entier** à une variable de type **réel (Z ⊂ R)**, mais le contraire n'est pas vérifié.
2. On peut affecter une valeur de type **caractère** à une variable de type **chaîne de caractères**, mais le contraire n'est pas vérifié.

Remarque : L'instruction d'affectation et de lecture permettent d'attribuer des valeurs à des **variables** et non à des constantes.

3)- L'instruction d'affichage ou d'écriture (écrire) : Elle permet à l'ordinateur d'afficher à l'écran la valeur d'un objet.

Syntaxe :

Ecrire (E)
ou
Ecrire (E1, E2,....., En)

E i: { Constante | Variable | Expression }

- Une constante est affichée telle qu'elle est ;
- Une variable est remplacée par ça valeur ;
- Une expression est évaluée et le résultat est affiché.

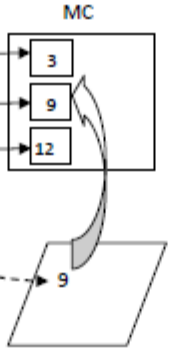
Autres syntaxes possible : AFFICHER, SORTIR

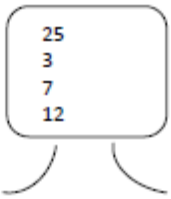
Exemple 1:

```
Lire(A)
Lire(B)
S ← A + B
Ecrire('La somme est égale à ',s)
```

Exemple 2: Travaillez l'exemple suivant :

```
a ← 1+2*3-4
Ecrire(25)
Ecrire(a)
Ecrire(2*a+1)
lire(b)
c ← a + b
Ecrire(c)
```

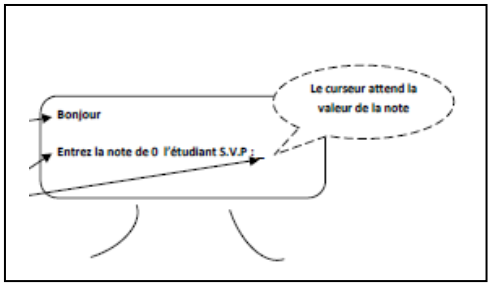




Écriture d'un message

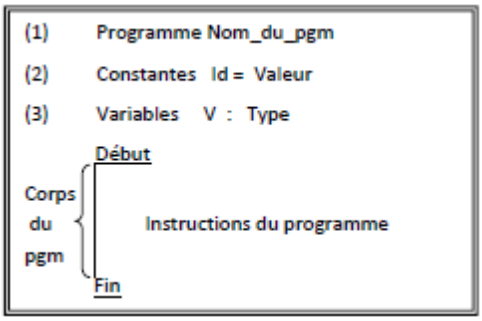
Un message dans un programme est une constante de type chaîne de caractères, pour faciliter son utilisation. Toute instruction de lecture doit être précédée par un message pour informer l'utilisateur (qui ignore le contenu de votre programme) de la donnée qu'il va faire entrer (la valeur de la variable utilisée par l'instruction lire qui suit le message).

Exemple :
Ecrire ('Bonjour')
Ecrire ('entrez la note de l'étudiant SVP :')
Lire (note)



Remarque : Une **expression** est un **objet** et **non une instruction**, c'est pour ça qu'une expression ne peut pas apparaître dans un programme toute seule (elle doit être manipulée par une instruction d'affectation ou d'écriture).

6. Formalisme d'un algorithme: par convention nous utiliserons les notations suivantes pour écrire nos programmes :



(1) L'en-tête du programme : le nom du programme qui est un identificateur choisi par le programmeur.
(2) La définition des constantes : donner les **noms** et les **valeurs** de toutes les constantes du programme.
(3) La déclaration des variables : donner les **noms** et les **types** de toutes les variables du programme.

Exemple : Somme : entier
Moyenne : réel
Nom : chaîne de caractère.

Règle : toute variable doit être déclarée avant d'être utilisée dans le programme parce que la machine a besoin de connaître la taille de données à réserver dans la MC.

Exemple 1: écrire un algorithme qui calcule la moyenne d'un étudiant au module INF1, en passant par l'étape d'analyse.