

# AS TECH

## LES ALGORITHMES

### PLAN DE LA PRÉSENTATION

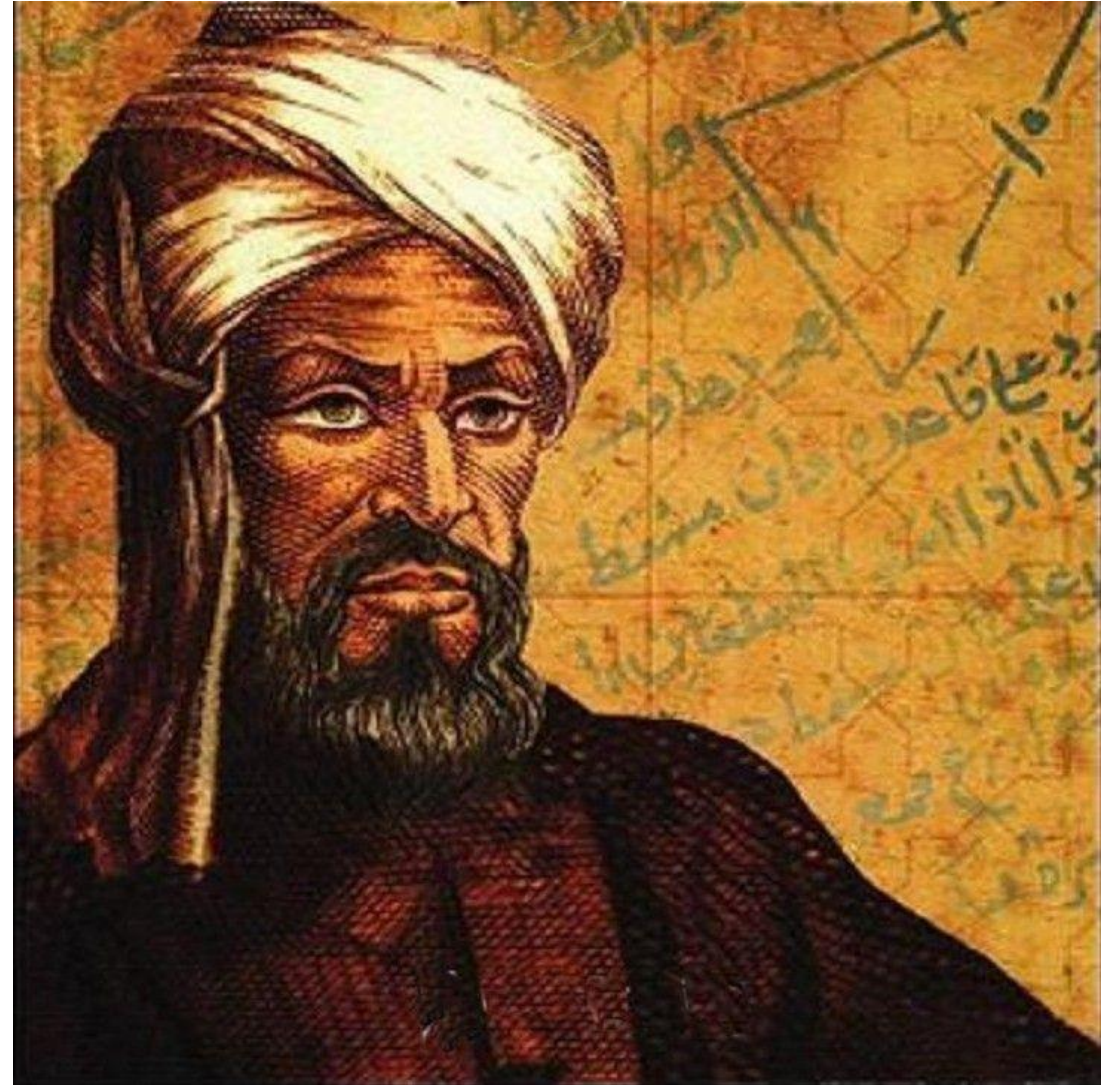
1. ORIGINE DE L'ARITHMITIQUE
2. LIRE ET ÉCRIRE UN MESSAGE
3. LES OPÉRATIONS ARITHMÉTIQUES ET LOGIQUES
4. LES VARIABLES ET LEURS TYPES
5. LES STRUCTURES CONDITIONNELLES
6. LES BOUCLES OU STRUCTURES ITÉRATIVES
7. LES TABLEAUX
8. LES PROCÉDURES ET LES FONCTIONS

```
attachEvent("onreadystatechange",H),e.attachE
boolean Number String Function Array Date RegE
_={};function F(e){var t=_[e]={};return b.ea
t[1])===!1&&e.stopOnFalse){r=!1;break}n=!1,u&
?o=u.length:r&&(s=t,c(r))}return this},remove
ction(){return u=[],this},disable:function()
re:function(){return p.fireWith(this,argument
ending",r={state:function(){return n},always:
romise)?e.promise().done(n.resolve).fail(n.re
id(function(){n=s},t[1^e][2].disable,t[2][2].
=0,n=h.call(arguments),r=n.length,i=1==r|e&
(r),l=Array(r);r>t;t++)n[t]&&b.isFunction(n[t
/><table></table><a href='/a'>a</a><input typ
/TagName("input")[0],r.style.cssText="top:1px
test(r.getAttribute("style")),hrefNormalized:
```

# ORIGINE DE L'ALGORITHMIQUE

Le mot algorithme vient du mathématicien Al Khwarizmi qui au IX siècle écrivit le premier ouvrage systématique donnant des solutions aux équations linéaires et quadratiques.

L'algorithme c'est une suite d'étapes permettant d'obtenir un résultat à partir des éléments fournis.



# LIRE ET ÉCRIRE UN MESSAGE

Imaginons un peu ce programme:

Variable A Numérique

Début

$A \leftarrow 12^2$

Fin

# **Lecture(lire)**

C'est une instruction qui Permet à l'utilisateur de rentrer des valeurs au clavier pour qu'elles soient utilisées par le programme.

# **Ecriture(écrire ou afficher)**

C'est une instruction qui permet au programme de communiquer les valeurs à l'utilisateur en les affichant à l'écran

## LES INSTRUCTIONS LIRE ET ÉCRIRE

Ecrire "Entrez votre nom : «

Lire NomFamille

# LES OPÉRATIONS ARITHMÉTIQUES ET LOGIQUES

**L'arithmétique est une branche des mathématiques qui étudie les propriétés et les règles de calcul entre les nombres. Elle traite, entre autres, des opérations traditionnelles telles que l'addition, la soustraction, la multiplication et la division.**

**Le composant essentiel de l'UC qui s'occupe de cet traitement est UAL (Unité Arithmétique Logique).**

## **1 . Opérations arithmétiques :**

- **Les opérations arithmétiques sont des calculs effectués sur des nombres pour obtenir un résultat.**
- **- Addition (+) : Ajouter deux nombres. Exemple :  $5 + 3 = 8$**
- **- Soustraction (-) : Soustraire un nombre d'un autre. Exemple :  $7 - 2 = 5$**
- **- Multiplication (\*) : Multiplier deux nombres. Exemple :  $4 * 6 = 24$**
- **- Division (/) : Diviser un nombre par un autre. Exemple :  $8 / 2 = 4$**
- **- Modulo (%) : Trouver le reste d'une division. Exemple :  $9 \% 4 = 1$**

**Applications : Calculs, graphismes, gestion de la mémoire, jeux vidéo, etc.**

- **2. Opérations logiques :**
- **Les opérations logiques manipulent des valeurs booléennes (Vrai ou Faux) et sont essentielles pour les décisions en informatique.**
- **- ET (AND) : Retourne Vrai si les deux valeurs sont vraies. Exemple : Vrai AND Faux = Faux**
- **- OU (OR) : Retourne Vrai si l'un des deux est vrai. Exemple : Vrai OR Faux = Vrai**
- **- NON (NOT) : Inverse la valeur. Exemple : NOT Vrai = Faux**
- **- XOR (OU exclusif) : Retourne Vrai si les valeurs sont différentes. Exemple : Vrai XOR Faux = Vrai**

**Applications : Contrôle des conditions, boucles, tests d'erreurs, portes logiques en électronique.**



### **3. Comparaison :**

- Opérations arithmétiques : Manipulent des nombres.**
- Opérations logiques : Manipulent des valeurs booléennes (Vrai/Faux).**

# LES VARIABLES ET LEURS TYPES

**Dans un programme informatique, on a souvent besoin de manipuler des valeurs. Pour stocker ces valeurs on fait appel aux *variables*.**

**Une variable est une entité dont la valeur peut changer, c'est à dire qu'au sein du même algorithme une même variable peut changer de valeurs de nombreuses fois.**

- Les types de variables
- Les booléens:
- Les chaînes de caractères:
- Les entiers:
- Les réels



# LES STRUCTURES CONDITIONNELLES

**Une condition est une expression booléenne elle est soit vraie ou fausse. On distingue trois types de condition: la condition à un choix (réduite) la condition à deux choix(complexe) et la condition à choix multiple(choix selon)**

**a-la condition à un choix(réduite)**

**Ici le bloc d'instruction est exécuté si seulement la condition est vraie.**

- **B-la condition à deux choix(complexe)**

**Ici si la condition est vraie le bloc d'instruction 1 est exécuté . si non le bloc d' instruction 2 est exécuté.**

- **C-la condition à choix multiple(choix selon)**

**Elle permet en fonction de plusieurs conditions de types booléen(=) d'effectuer des instruction(action) différent suivant les valeur d'une seul variable.**

**VE: Variable ou expression**

**Si ve vaut une valeur entre val1 ou val2 alors le bloc d'instruction seras exécuté et les autre seront ignorés.**

**Si ve n'a aucun valeur entre val1 et val2 alors le bloc d' instruction m seras exécuté**

- Dans un algorithme on a souvent besoin de répéter un même bloc d'instructions plusieurs fois. Au lieu d'effectuer cette répétition manuellement on utilise les *structures itératives*.  
En algorithmique on dénombre généralement 3 structures itératives à savoir,

- *la boucle Tant Que, la boucle Pour et la boucle Répéter.*
- *La structure Tant Que:*
- dite aussi *boucle Tant Que*, exécute un même bloc d'instructions tant que la condition spécifiée dans l'algorithme est vraie. Une fois cette condition là devient fausse, alors on quitte la boucle pour poursuivre l'exécution du reste du traitement.

- **La boucle Pour:**
- ***· L'initiation du compteur de la boucle***
- ***· Spécification de la condition qui permet de quitter la boucle***
- ***· Modification de la valeur du compteur de la boucle (par incrémentation ou décrémentation par exemple)***
- **permet de déclarer ces trois instructions dans une même ligne et de manière très simple. En revanche, la boucle Pour est utile quand on connaît à l'avance le nombre d'itérations à exécuter.**

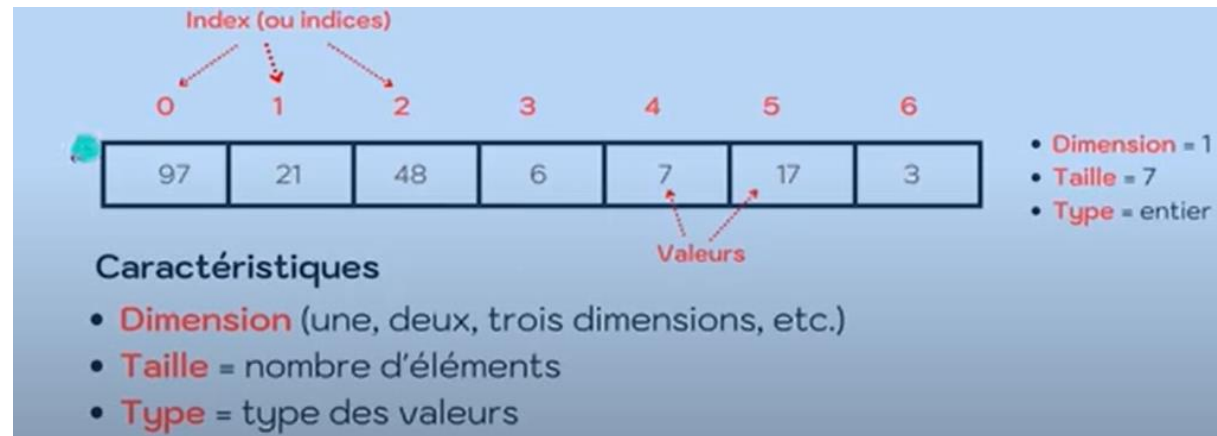


- **La boucle Répéter:**
- **La boucle Répéter quant à elle est aussi une structure itérative, mais sa première itération est toujours exécutée quelque soit la condition de sortie de la boucle. En effet, la vérification de la condition n'est faite qu'après avoir exécuté le bloc d'instructions.**

# LES TABLEAUX

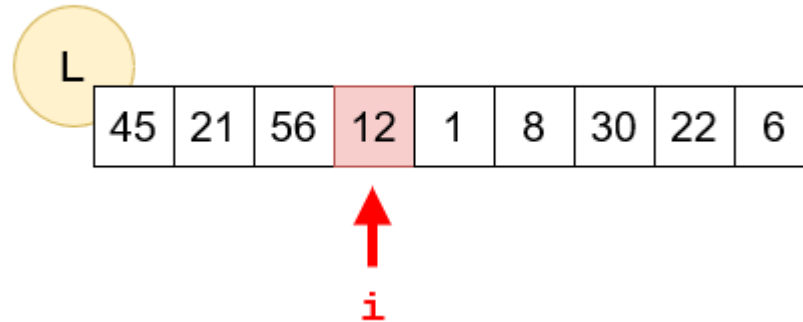
- **Un tableau: c'est une structure de données qui permet de stocker plusieurs valeurs d'un même type les uns à la suite des autres, auxquels on peut accéder efficacement par leur position, ou indice.**

- les caractéristiques d'un tableau



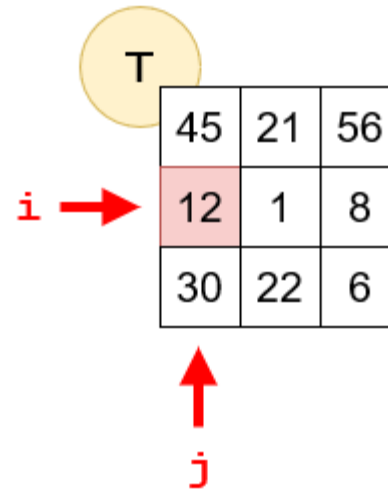
- Un tableau à 1 dimension est un tableau dont la position de chaque élément est accessible directement par son indice.

Tableau à une dimension :



- Un tableau à 2 dimensions est un tableau dont la position de chaque élément est donnée par un couple de 2 indices.

Tableau à deux dimensions :



The diagram illustrates a 2D array named T, represented by a yellow circle. The array is a 3x3 grid of numbers. The first row contains 45, 21, and 56. The second row contains 12, 1, and 8. The third row contains 30, 22, and 6. A red arrow labeled 'i' points to the second row, and a red arrow labeled 'j' points to the first column. The cell containing the number 12 is highlighted in light red, indicating it is the element at the intersection of index i=1 and index j=0.

45	21	56
12	1	8
30	22	6

# LES PROCÉDURES ET LES FONCTIONS

- **Les Procédures**
- **Une procédure est une suite d'instructions décrivant une action simple ou composée, à**
- **laquelle on donne un nom, qui devient lui-même en quelque sorte un sous-programme.**
- **La déclaration d'une procédure se fait comme suit :**

PROCEDURE <nom\_de\_la\_procedure> [ (liste des paramètres : type) ]

**Var** : liste des variables

**DEBUT** {corps de la procédure}

<LISTE DES ACTIONS> {la liste ne doit pas être vide}

**FINPROCEDURE**

- **Les Fonctions**
- **Une fonction est une suite ordonnée d'instructions qui retourne une valeur (bloc d'instructions nommé et paramétré).**
- **La déclaration d'une procédure se fait comme suit :**



**FONCTION** <nom\_de\_la\_fonction> [ (liste des paramètres : type) ] : **type\_fonction**

**Var** : liste des variables

**DEBUT** {corps de la fonction}

<LISTE DES ACTIONS> {la liste ne doit pas être vide}

<nom\_de\_la\_fonction> ← résultat\_des\_calculs

**FINFONCTION**

**MERCI POUR VOTRE ATTENTION.**  
**BIENVENUE AUX QUESTIONS**