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.
Son **intérêt** est d'être codé dans un langage informatique afin qu'une machine (ordinateur, calculatrice, etc.) puisse l'exécuter rapidement et efficacement.
Les **trois phases** d'un algorithme sont, dans l'ordre :
(a) l'entrée des données
(b) le traitement des données
(c) la sortie des résultats

UNE VARIABLE

Lors de l'exécution d'un algorithme, on va avoir besoin de stocker des données, voire des résultats. Pour cela, on utilise des **variables**. On attribue un nom à chaque variable.
Instructions de base sur des variables

- La **saisie** : on demande à l'utilisateur de l'algorithme de donner une valeur à la variable
- l'**affectation** : le concepteur de l'algorithme donne une valeur à la variable. Cette valeur peut-être le résultat d'un calcul
- l'**affichage** : on affiche la valeur de la variable.

L'AFFECTATIONS

Attention à ne pas confondre une **variable** et sa **valeur** :
Une variable est une case mémoire, repérée par une lettre.
Sa valeur est le nombre (ou le mot …) qu'elle contient provisoirement.

On peut modifier plusieurs fois les valeurs que l'on place dans cette case. Attention, à chaque fois, la valeur précédente est effacée.

L'INSTRUCTION CONDITIONNELLE

La résolution des certains problèmes nécessite la mise en place d'un **test** pour savoir si l'on doit effectuer une tâche.

Si *la condition est remplie*
alors
on effectue la tâche,
sinon
on effectue (éventuellement) une autre tâche.

LA BOUCLE « POUR »

Lorsque l'on doit répéter un nombre de fois connu à l'avance la même tâche, on utilise une **boucle itérative** de la forme « **Pour.. allant de... à** ».
La variable utilisée dans la boucle est appelée **compteur**. À chaque passage dans la boucle, sa valeur est automatiquement augmentée de 1.

Pour *variable* ***allant de*** *valeur_depart* ***à*** *valeur_fin*
faire
tâche 1 tâche 2 ...
Fin pour

LA BOUCLE « TANT QUE »

Lorsque le nombre de répétitions n'est pas connu à l'avance, il peut dépendre d'une condition, le traitement est répété tant que la condition est vraie. Lorsqu'elle est fausse, on sort de la boucle. On parle de boucle conditionnelle.

Tant que *condition*
faire
tâche 1 tâche 2
Fin Tant que