

« Ce n'est pas en faisant un programme de 1200 lignes de code que l'on sait programmer »

La programmation, ce n'est pas uniquement le fait de saisir son clavier pour créer le tout dernier programme révolutionnaire mais c'est aussi l'action de prendre une feuille de papier et un crayon pour déterminer un cahier des charges et mettre en évidence des algorithmes de programmation. L'objectif des séances à venir est de vous faire acquérir ces réflexes.

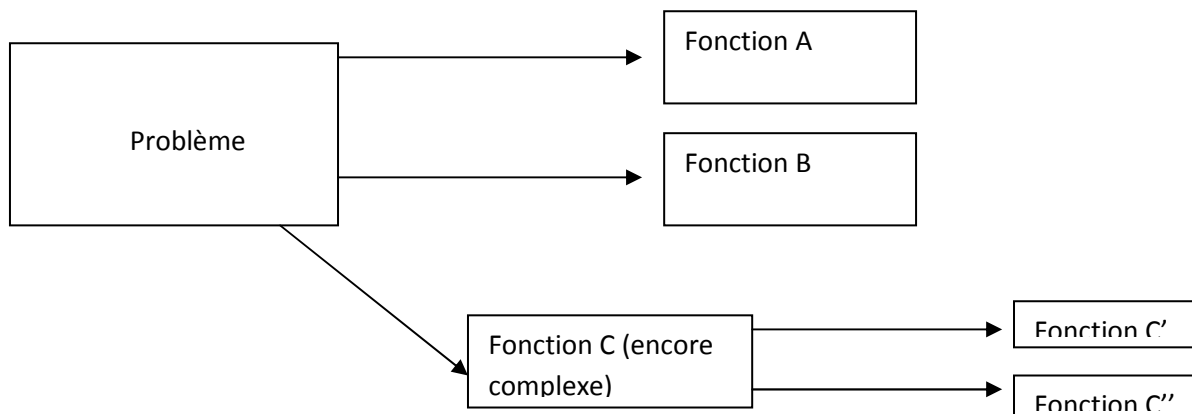
I- La création d'un cahier des charges

a. Un cahier des charges, késako ?

Il s'agit d'un plan de création de votre programme où vous devez mettre à plat toutes vos idées afin de les structurer. C'est également ici que vous incluez les contraintes qui ont pu vous être imposées. Attention ! Le cahier des charges doit être complet et le plus clair possible pour que, lorsque vous commencerez à programmer, vous ne partiez pas dans toutes les directions.

b. L'analyse générale du problème

Avant de commencer à programmer, il faut que le problème nous paraisse le plus simple possible. Il faut donc faire ce que l'on appelle : une analyse fonctionnelle. Cette méthode consiste à découper un problème sous forme de fonctions simple. Si une fonction paraît encore trop compliquée, on la redécoupe en sous-fonctions :



Le redécoupage de fonction s'appelle : Conception structurée descendante. Ceci est valable pour la programmation événementielle mais pas pour la programmation orientée objet.

c. L'analyse détaillée

Il s'agit d'écrire son programme sur papier dans un langage humain, également appelé « Pseudo-code ».

Exemple : faire une suite arithmétique de raison $r=2$ et de premier terme $U_0 = 8$ et de dernier terme $U_{10} = 28$.

Début de la boucle

pour i allant de 1 à 10

$U_i = U_0 + i * 2$

fin de l'occurrence

Fin de la boucle

d. Codage du pseudo-code

C'est seulement à partir de cette étape que l'on commence à écrire du code.

Exemple pour la suite arithmétique $U(n) \begin{cases} U_0 = 8 \\ r = 2 \end{cases} \forall n \in \mathbb{N}, n \leq 10$

```
Dim n As Integer
```

```
Dim I As Integer
```

```
For I = 1 To 10
```

```
    n = 8 + I * 2
```

```
Next
```

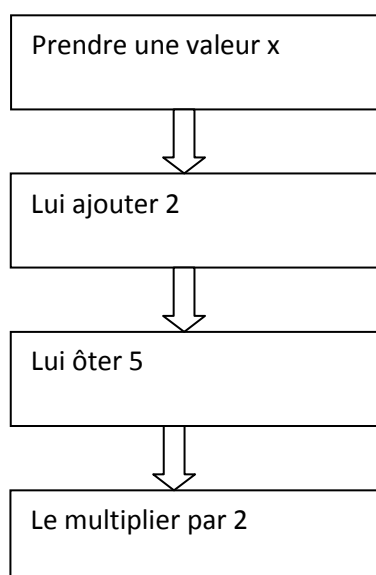
e. Test du programme

II- Les algorithmes (Cours réalisé avec l'appui de <http://plasserre.developpez.com/v1-4.htm>)

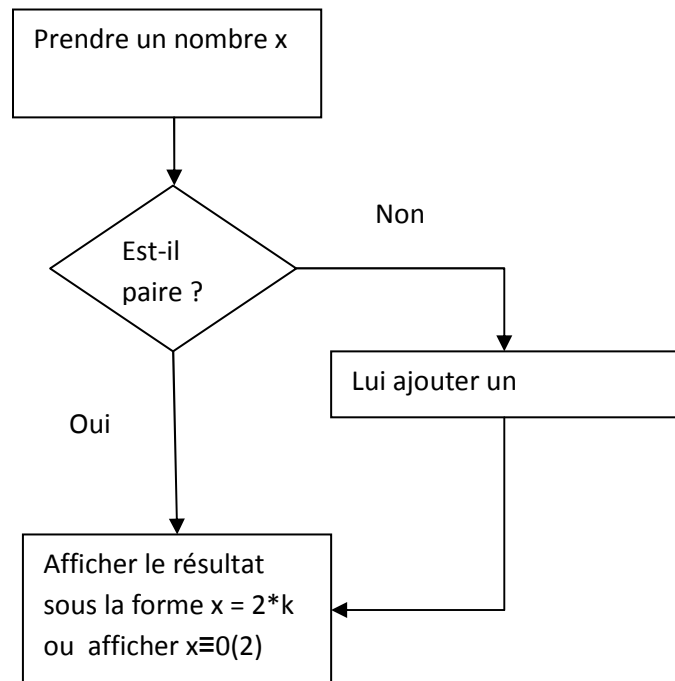
« L'algorithme est une succession de tests, décisions et actions dans le but de décrire le comportement d'une entité (objet, programme, personne) ». Définition du Dicodunet.

Les algorithmes offrent trois possibilités de conception :

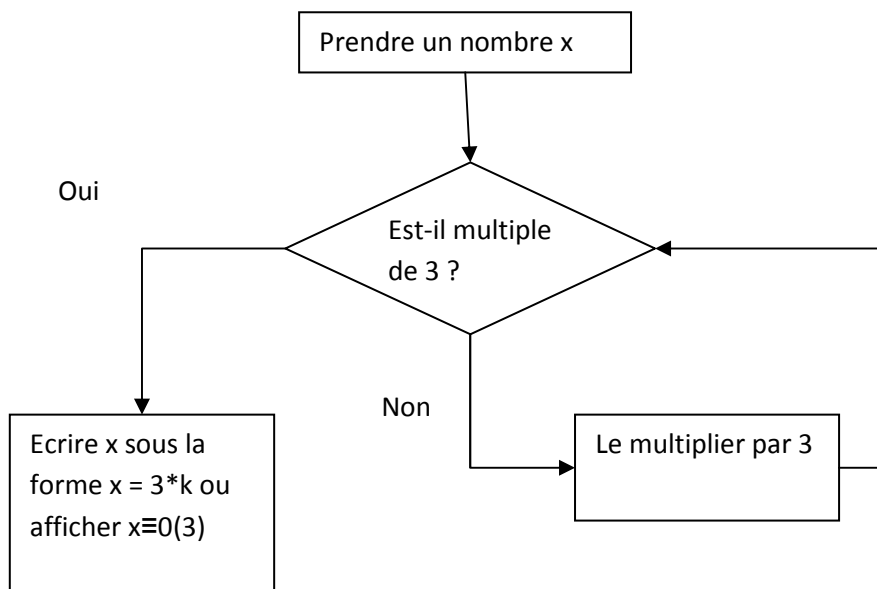
- La séquence : les actions sont effectuées les unes après les autres



- Le choix : Les éléments ont un traitement différent selon une condition initiale



- La répétition : Le programme effectue une boucle jusqu'à ce qu'une condition soit rempli



ATTENTION ! La représentation d'algorithmes n'est pas standardisée. Des différences peuvent apparaître entre votre méthode de faire et celle d'un autre. Il en va de même pour le pseudo-code. Le principal étant que votre algorithme puisse être compris à la fois par vous et par quelqu'un d'autre.

Par ce fait, des principes généraux doivent être adoptés par les concepteurs (il s'agit des trois possibilités de conceptions précédentes).

III- Notions théoriques avancées de programmation

a- Les variables

Une variable : On rappelle que les variables sont des informations TEMPORAIRES stockées dans la RAM (ou mémoire vive) de votre ordinateur. Elles sont appelées variables car leur contenu peut changer durant l'exécution du programme. Elles fonctionnent sur le principe des variables mathématiques (ex : On pose $x=3 \rightarrow \text{Dim } x \text{ as Integer} = 3$).

Une variable est désignée par son type. Ce type renseigne l'ordinateur sur la nature de l'information à stocker. Il existe différents types :

Type numérique :

'Entier' (Integer en VB), 'réel'.. (Single en VB) Exemple d'un entier:
123

Type alphanumérique :

'Caractère' (Char en VB) contient 1 caractère. Exemple d'un caractère:
"a" (avec des guillemets)

'Chaîne de caractères', (String en VB), contient plusieurs caractères.
Exemple: "toto" (avec des guillemets)

Autres Type :

Booléen (Boolean en VB) ne peut contenir que True ou False ('Vrai' ou 'Faux').

Objet. (Object en VB)

Monétaire (Décimal en VB)

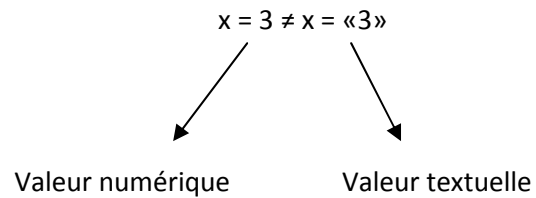
Date (Date en VB)

A partir des types précédents on peut créer des **types complexes (ou structurés)** :

Les tableaux (Array).

Les Collections.

L'affectation : Avant d'affecter, **IL EST INDISPENSABLE DE DECLARER AU PREALABLE LA VARIABLE** par la commande **Dim nom as type**. Les valeurs sont ensuite affectées par le signe = (ex : x=3). Les données numériques sont affectées sans guillemets autour de la valeur à enregistrer alors que les valeurs textuelles sont entourées de guillemets :



Lorsque vous déclarez une variable et lui affectez une valeur initiale, c'est une **initialisation**. Lorsque vous utilisez une valeur donnée directement dans le code, cette valeur est appelée **littéral**.

Les variables sont dynamiques : leur valeur peut être modifiée. Parallèlement aux variables, il existe des constantes qui sont des enregistrements fixes (une fois une valeur fixée, on ne peut plus la changer durant l'exécution du programme). Ces constantes fonctionnent presque comme les variables et on a besoin d'être **déclarées**. En VB.NET, on déclare une constante par **Const nom as type**.