STRUCTURES ALGORITHMIQUES

Table des Matières

1.Structure linéaire.

2.Structures alternatives.

2.1 Structure SI…ALORS…SINON…

3.Structures répétitives (ou itératives).

3.1 Structure FAIRE…JUSQU'À

3.2 Structure TANT QUE…FAIRE

3.3 Structure POUR…FAIRE

4- Choix d'un langage de programmation

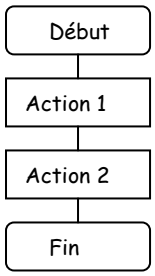
Structures algorithmiques

Un algorithme est composé d'un ensemble de structures ordonnant à un processeur de réaliser dans un ordre précis un nombre de taches élémentaires dans le but de résoudre un problème technique donné.
L'algorithme peut être décrit sous forme graphique (Algorigramme ou Organigramme) ou sous forme littérale (notation algorithmique).

1.Structure linéaire.

On exécute successivement une suite d'action dans l'ordre de leur énoncé.

Algorigramme



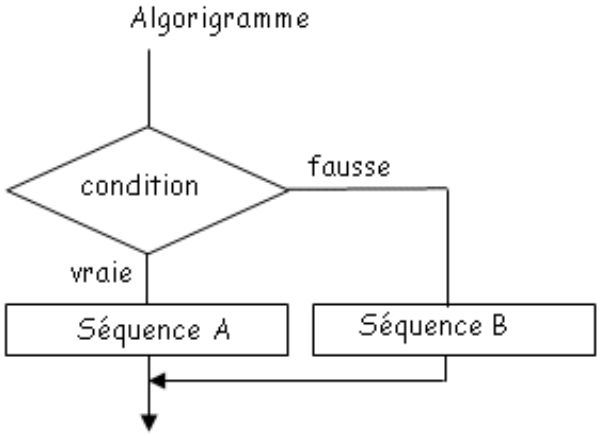
Notation algorithmique

Début
Action 1
Action 2
Fin

2.Structures alternatives.

2.1 Structure SI…ALORS…SINON…

Cette structure offre le choix entre deux séquences s'excluant mutuellement.



Notation algorithmique

Si condition **Alors**
    Séquence A
Sinon
    Séquence B
Fin Si

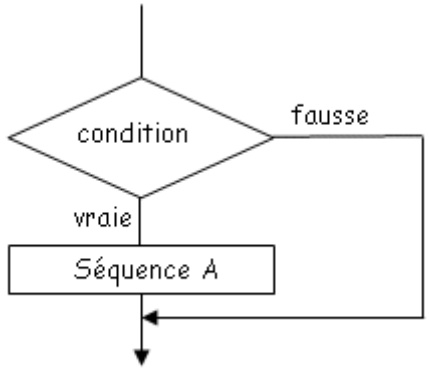
Exemple en langage C

```
If ( condition )
{   Séquence A ; }
else
{   Séquence B ;  }
```

Exemple en Basic11

```
If  ( condition ) then
   Séquence A
Else
   Séquence B
End if
```

La structure peut se limiter à SI…ALORS, si la condition est vrai on exécute la séquence A si elle est fausse on quitte la structure sans exécuter de séquence.



Notation algorithmique

Si condition **Alors**
Séquence A
Fin Si

Exemple en langage C

```
If ( condition )
{   Séquence A  ; }
```

Exemple en Basic11

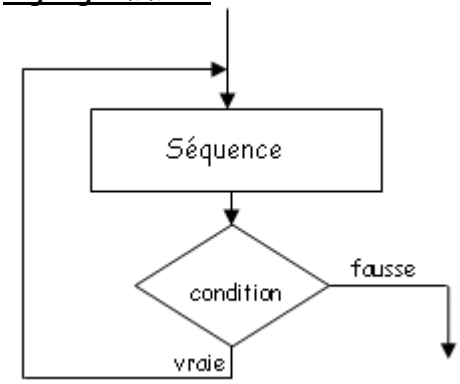
```
If  ( condition ) then
   Séquence A
End if
```

3.Structures répétitives (ou itératives).

3.1 Structure FAIRE…JUSQU'À

La séquence est exécutée au moins une fois, elle est répétée tant qu'elle est vraie.

Algorigramme :



La traduction en algorithme peut se faire de 2 façons :

Notation algorithmique

```
Faire
    Séquence
Tant que condition vraie
```

Notation algorithmique

```
Faire
    Séquence
Jusqu'à condition fausse
```

Exemple en langage C

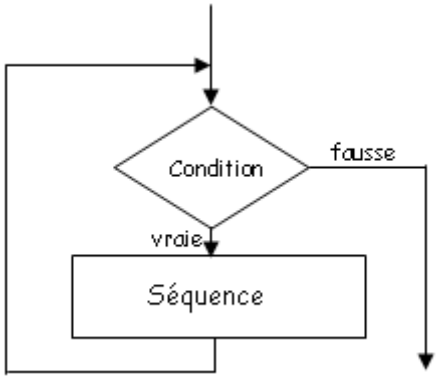
```
Do
  {  Séquence ;     }
While (condition vraie)
```

Exemple en Basic11

```
Do
   Séquence
Loop until  condition fausse
```

3.2 Structure TANT QUE…FAIRE

On teste d'abord la condition la séquence est exécutée tant que la condition est vraie.



Notation algorithmique

```
Tant que condition vraie
    Séquence
Fin tant que
```

Exemple en langage C

```
while (condition)
  {
     Séquence ;
  }
```

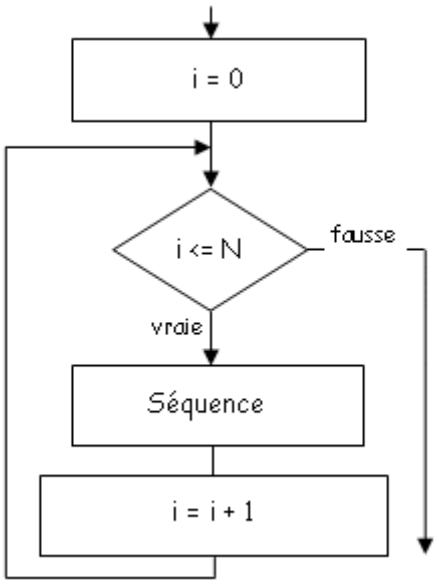
Exemple en Basic11

```
Do while (condition )
   Séquence
Loop
```

3.3 Structure POUR…FAIRE

On connaît le nombre d'itérations

Algorigramme :



Notation algorithmique

Pour i = 0 à N
    Faire Séquence
Fin Pour

Exemple en langage C

```
For (i=0; i<=N; i++)
    {
    Séquence;
    }
```

Exemple en Basic11

```
For i = 0 to N
   Séquence
Next i
```

4- Choix d'un langage de programmation

Tout d'abord, avant de se demander « quel langage vais-utiliser pour réaliser telle application » , il faut se poser la question « comment vais-je résoudre mon problème algorithmique »
Car la difficulté première est d'élaborer le bon algorithme pour résoudre le problème : cela demande de la recherche, beaucoup de réflexion et ce d'autant plus que le problème à résoudre est difficile.
Ensuite, une fois l'algorithme mis en place, on peut passer à la phase « codage », c'est-à-dire à la phase de traduction dans un langage donné : Visual basic, C , assembleur , php, perl, lisp … mais cela ne présente pas de difficulté majeure si ce n'est la connaissance de la syntaxe des différents langages (mais avec un bouquin ou une ressource sur internet, vous pouvez le faire sans être pour autant spécialiste)
Le choix du langage de programmation se fera en fonction de critères pratiques : facilité de codage (bibliothèque d'instructions prédéfinies) , rapidité d'exécution, disponibilité du langage pour le processeur ou le serveur (si vous êtes sur internet par exemple) ou encore affinités du programmeur avec tel ou tel outil.

Souvent le débutant (qui a entendu parler du C comme LE langage mirifique, le plus beau le plus fort et le plus répandu) se dit « je dois apprendre le C, ou le C++ pour devenir un bon programmeur » … c'est aussi stupide que se dire qu'en apprenant la grammaire française vous alliez devenir un grand écrivain de la langue de Molière.

Un bon programmeur est donc avant tout un bon algorithmicien qui saura ensuite exploiter au mieux tel ou tel langage de programmation pour réaliser le programme demandé.
Le php ou le perl est particulièrement pratique pour la programmation d'application sur internet car ils disposent d'une bibliothèque de ressources importantes. Le basic11 (langage structuré et compilé) ou le C sera adapté à la programmation du microcontrôleur 68HC11