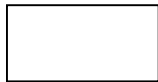


8 Organigrammes et algorithmes.

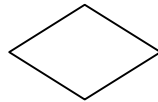
8.1 Organigrammes.

Bien qu'inusité de nos jours, les organigrammes permettent d'avoir une approche visuelle que ne fournissent pas les algorithmes. Un organigramme se définit comme une représentation graphique des structures de contrôle.

Nous ne verrons que les deux symboles suivants :



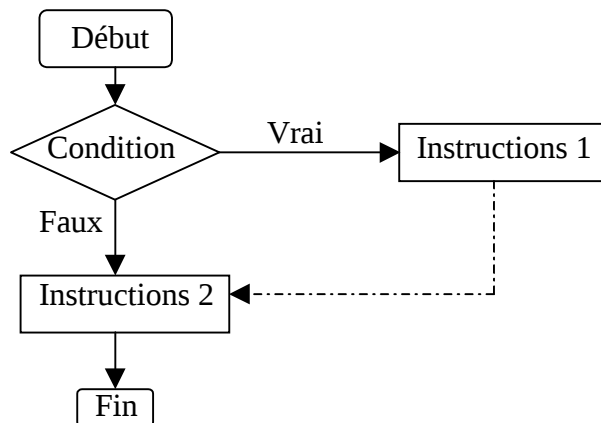
Le rectangle qui représente un bloc d'instructions.



Le losange qui représente une condition : Si Alors Sinon ...

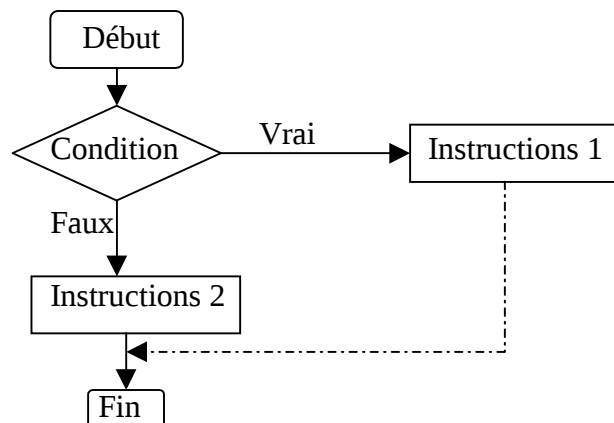
8.1.1 If (Condition) Alors Instructions 1

```
if (Condition)
{
    Instructions 1;
}
Instructions 2;
```

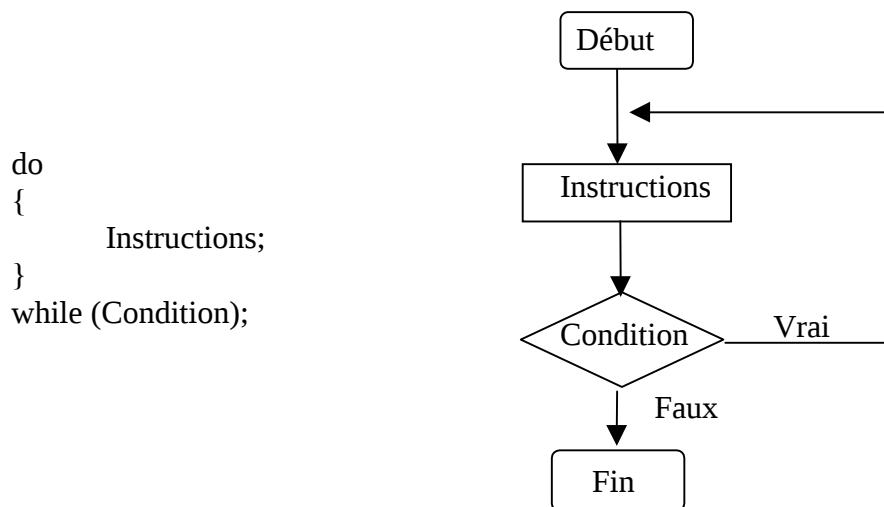


8.1.2 If (Condition) Alors Instructions 1 Else Instructions 2

```
if (Condition)
{
    Instructions 1;
}
else
{
    Instructions 2;
}
```



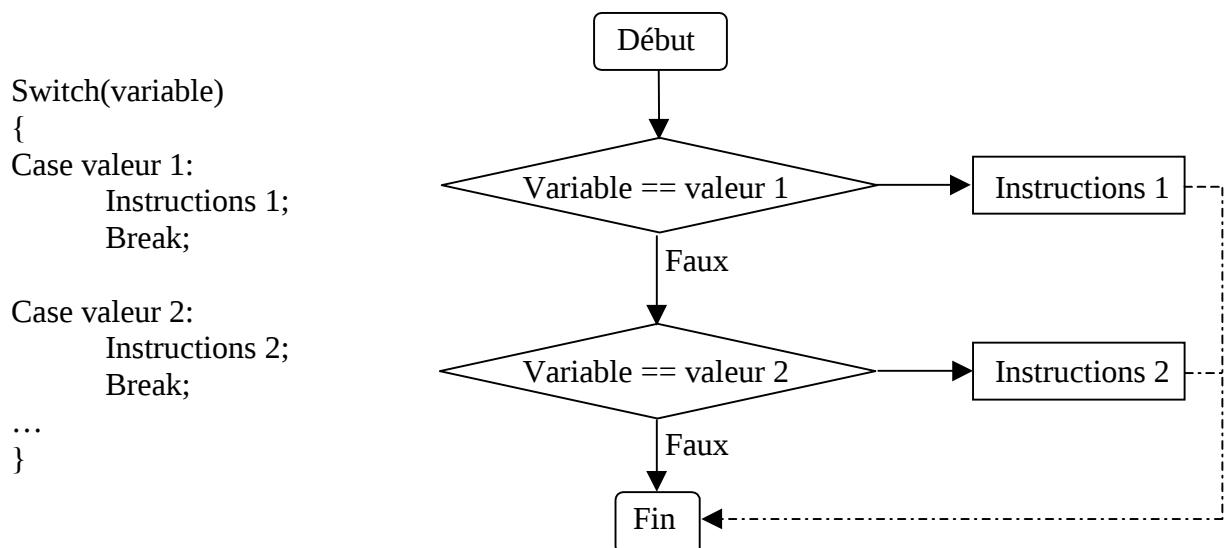
8.1.3 Do ... While (Condition)



8.1.4 While (Condition)

Je ne détaillerai pas cet algorithme au vue qu'il est très peu différent de celui-ci dessus et constituera un excellent exercice.

8.1.5 Switch (variable)... Case ...



8.1.6 Conclusion sur les organigrammes

Les organigrammes permettent de construire des programmes à l'aide de dessins, ceci simplifie énormément la tâche d'analyse du programmeur. Il a été maintenant remplacé par les algorithmes qui en deux mots peuvent être définis comme une transcription francisé de ce que l'on désire faire (pour i allant de 1 à 10 faire ...). Les algorithmes seront vus succinctement dans l'initiation au C niveau 2.

8.2 Exercice 1 : Et les shadoks crièrent : "Pourquoi tant que on a pas tout compris, on continue ?"

Traduire en langage C, compilez, exécutez, comprenez ...

- Déclarez une variable nb_rech de type entier
 - Déclarez une variable nb_user de type entier
 - Déclarez une variable essai de type entier et l'assigner à la valeur 0
 - Déclarez une variable gagne de type entier et l'assigner à la valeur 0
 - Assignez à la variable nb_rech la valeur de random (101)
 - Effacez l'écran
 - Faire
- ```
{
 • Demandez et saisissez un nombre entre 0 et 100. Soit nb_user la variable utilisée.
 • Si nb_user < nb_rech
 {
 Afficher « Le nombre à trouver est plus grand »
 essai = essai + 1
 }
 Sinon
 {
 Si (nb_user > nb_rech)
 {
 Afficher « Le nombre à trouver est plus petit »
 essai = essai + 1
 }
 Sinon
 gagne = 1;
 }
 Tant que (essai != 5) et (!gagne);

 • Si (gagne)
 Ecrire "Gagné"
 Sinon
 {
 Ecrire "Perdu"
 Faire afficher le nombre nb_rech
 }

 • Attendre l'appui d'une touche
```

## 8.3 Et les shadoks découvrirent la couleur

La fonction **textcolor (no\_couleur)** permet de changer la couleur du texte. Le no de couleur (no\_couleur) est un entier allant de 0 à 15. Pour pouvoir afficher en couleur, il est nécessaire d'utiliser la fonction **cprintf** à la place de printf. Cprintf possède la même syntaxe que printf.

| Constante           | Couleur       | Valeur | Couleur de Fond | Couleur de caractère |
|---------------------|---------------|--------|-----------------|----------------------|
| <b>BLACK</b>        | Noir          | 0      | Oui             | Oui                  |
| <b>BLUE</b>         | Bleu          | 1      | Oui             | Oui                  |
| <b>GREEN</b>        | Vert          | 2      | Oui             | Oui                  |
| <b>CYAN</b>         | Cyan          | 3      | Oui             | Oui                  |
| <b>RED</b>          | Rouge         | 4      | Oui             | Oui                  |
| <b>MAGENTA</b>      | Magenta       | 5      | Oui             | Oui                  |
| <b>BROWN</b>        | Brun          | 6      | Oui             | Oui                  |
| <b>LIGHTGRAY</b>    | Gris Clair    | 7      | Oui             | Oui                  |
| <b>DARKGRAY</b>     | Gris Foncé    | 8      | Non             | Oui                  |
| <b>LIGHTBLUE</b>    | Bleu Clair    | 9      | Non             | Oui                  |
| <b>LIGHTGREEN</b>   | Vert Clair    | 10     | Non             | Oui                  |
| <b>LIGHTCYAN</b>    | Cyan Clair    | 11     | Non             | Oui                  |
| <b>LIGHTRED</b>     | Rouge Clair   | 12     | Non             | Oui                  |
| <b>LIGHTMAGENTA</b> | Magenta Clair | 13     | Non             | Oui                  |
| <b>YELLOW</b>       | Jaune         | 14     | Non             | Oui                  |
| <b>WHITE</b>        | Blanc         | 15     | Non             | Oui                  |
|                     |               |        |                 |                      |
| <b>BLINK</b>        | Clignotant    | 128    | Non             | ***                  |

#### 8.4 Utilisation du clignotement et des codes couleur.

Les valeurs du tableau précédent s'utilisent soit par leur nom, soit par leur code couleur. Ceci signifie que :

textcolor (2) est équivalent à textcolor (GREEN);

En fait GREEN est défini comme valant 2. (cf. #define dans Initiation au C niveau 2).

Pour faire clignoter du texte, il faut ajouter la valeur BLINK à la valeur de la couleur ce qui nous donne :

textcolor (2+128);  
ou                   textcolor (GREEN+BLINK);  
ou encore       textcolor (130);

Exemple :

```
#include <conio.h>

int main ()
{
 clrscr ();
 textbackground (GREEN);
 textcolor (RED);
 cprintf ("Coucou");
 getch ();
}
```

affiche "Coucou" en couleur rouge sur fond vert.

Test :

Au lieu de  
    clrscr ();

```
textbackground (GREEN);
écrivez :
 textbackground (GREEN);
 clrscr ();
Aucun commentaire ...
```

### **8.5 Exercice 2 :**

Ecrivez un programme qui fait afficher "Je suis vert de peur" en bleu clignotant sur fond jaune poussin. Eh oui, pas shadok pour rien.

### **8.6 Exercice 3 : Et les shadoks créèrent un mini – yam ...**

Je dispose de 5 dés numérotés de 1 à 5.

Soient d1, d2, d3, d4, d5 ces 5 dés.

Utilisez la fonction Random pour afficher le tirage des 5 dés.

Affichez gagné si le résultat est un Yam (5 dés identiques).

## Corrigés des exercices du chapitre 8

! **Exercice 1** : *Et les shadoks crièrent : "Pourquoi tant que on a pas tout compris, on continue ?"*

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

int main ()
{
 int nb_rech=0, nb_user=0;
 int gagne=0;
 int essai=0;

 randomize ();
 nb_rech= random (101); /* Nombre entre 0 et 100 */

 clrscr ();

 do
 {
 printf("Saisir un nombre entre 0 et 100 :");
 scanf ("%d",&nb_user);

 if (nb_user<nb_rech)
 {
 printf("Le nombre à trouver est plus grand\n");
 essai=(essai+1);
 }
 else
 {
 if(nb_user>nb_rech)
 {
 printf("Le nombre à trouver est plus petit\n");
 essai=(essai+1);
 }
 else
 {
 gagne=1;
 }
 }
 }while((essai!=5)&&(!gagne));

 if (gagne)
 printf("Gagné !");
 else
 printf("Perdu !");

 printf("\nNombre recherché : %d",nb_rech);

 getch ();
}
```

! **Exercice 2**

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
```

```
int main()
{
 clrscr ();
 gotoxy(20,20);
 textbackground (YELLOW);
 textcolor (BLUE + BLINK);
 cprintf ("Je suis vert de peur");
 getch ();
}
```

**! Exercice 3 : : Et les shadoks créèrent un mini – yam ...**

```
#include <stdio.h>
#include<stdlib.h>

int main ()
{
 int d1,d2,d3,d4,d5;

 randomize ();

 /* Tirage des 5 dés */
 d1 = random (6) + 1;
 d2 = random (6) + 1;
 d3 = random (6) + 1;
 d4 = random (6) + 1;
 d5 = random (6) + 1;

 clrscr ();

 /* Affichage du tirage */
 printf ("\nD1 : %d, D2 : %d, D3 : %d, D4 : %d, D5 : %d"
 ,d1,d2,d3,d4,d5);

 /* Résultat */
 if ((d1 == d2) && (d2 == d3) && (d3 == d4) && (d4 == d5))
 printf ("\nGagné !");
 else
 printf ("\nPerdu !");

 getch ();
}
```