Université Ahmed Zabana de Relizane

Spécialité : LMD MIAS

Année : 1ère (Semestre 1)

Module : Algorithmique et structure de données 1

Les structures de contrôle d'un algorithme :

Les structures conditionnelles

I. Structure séquentielle

Une structure séquentielle est une suite d'instructions. Les instructions sont exécutées dans l'ordre l'une après l'autre.

Syntaxe :

```
Début
        Instruction 1 ;
        Instruction 2 ;
        …..
        Instruction n ;
Fin.
```

Exemple : l'algorithme qui affiche l'inverse d'un nombre réel.

```
Algorithme  inverse ;
Var  X,Y : Réel ;
Début
Ecrire ("Entrez une valeur svp ") ;
Lire (X) ;
Y ← 1 /X ;
Ecrire ("L'inverse de" , X, "est", Y) ;
Fin.
```

L'algorithme ci-dessus calcule l'inverse d'un nombre réel. Si l'utilisateur tape la valeur 0 il y a une erreur d'exécution (division par 0), ce problème ne peut pas être résolu que par l'utilisation des structures conditionnelles.

II. Structure conditionnelle

Une instruction conditionnelle permet à un programme de modifier son traitement en fonction d'une condition. Les structures conditionnelles permettent de déterminer l'ordre dans lequel les instructions sont exécutées.

III. Types d'une structure conditionnelle

III.1 Structure conditionnelle simple : L'instruction « if..... »

Elle est subdivisée en deux parties : la condition et l'action.

L'instruction if est la structure de test la plus basique, on la retrouve dans tous les langages (avec une syntaxe différente...). Elle permet d'exécuter une série d'instructions si jamais une condition est vérifiée.

Syntaxe :

```
SI <Condition> ALORS
        < Bloc d'actions >
FINSI
```

Où : <Condition> est une expression de condition dont l'évaluation donne une valeur logique, et le < Bloc d'actions > est un groupe d'actions élémentaires.

- Si la condition est vérifiée (condition = vrai), les instructions du < Bloc d'actions > sont exécutées et on continue l'exécution des actions (instructions) situées après le Finsi.

- Si la condition n'est pas vérifiée (condition = faux), la partie < Bloc d'actions > à l'intérieur du Si n'est pas exécuté et on poursuit l'exécution de l'algorithme directement à partir de l'instruction qui suit le FinSi.

Syntaxe en en langage C :

```
if ( condition )
{
        INSTRUCTION1 ;
        NSTRUCTION2 ;
        ……
        INSTRUCTION N ;
}
```

Exemple :

SI ((A-B)*C) = 12 **ALORS**
B ← 3;
C ← (X+2+B);
FINSI;

	<condition> = VRAI			<condition> = FAUX	
	Avant l'action	Après l'action		Avant l'action	Après l'action
A	5	5		2	2
B	2	3		2	2
C	4	8		4	4
X	3	3		5	5

Exemple en langage C:

```
if ( note >= 10)
{
        printf (" vous êtes admis  ") ;
}
```

III.2 Structure alternative : L'instruction « if ... else »

L'expression **if ... else** permet d'exécuter une autre série d'instructions en cas de non-réalisation de la condition.

Syntaxe :

```
SI <Condition> ALORS
        < Bloc d'actions 1 >
SINON
        < Bloc d'actions 2 >
FINSI
```

Elle s'exécute comme suit :

- Si la condition est vérifiée, les instructions du < Bloc d'actions1 > sont exécutées puis poursuite de l'exécution de l'algorithme à partir de l'instruction qui suit le Finsi.
-
- Si la condition n'est pas vérifiée, les instructions du < Bloc d'actions2 > sont exécutées puis poursuite de l'exécution de l'algorithme à partir de l'instruction qui suit le Finsi.

Syntaxe en langage C:

```
if ( condition )
{
    //  liste d' instructions
}
else
{
   // autre bloc d' instructions
}
```

Exemple :

SI (A+B = 0) **ALORS**

A ⟵ (C-2);

SINON

B ⟵ (X*3);
A ⟵ (B-4);

FINSI;

	<condition> = VRAI (exécuté action(s)1)			<condition> = FAUX (exécuté Action (s)2)	
	Avant l'action	**Après l'action**		**Avant l'action**	**Après l'action**
A	3	0		5	2
B	-3	-3		2	6
C	2	2		4	4
X	1	1		2	2

Exemple en langage C:

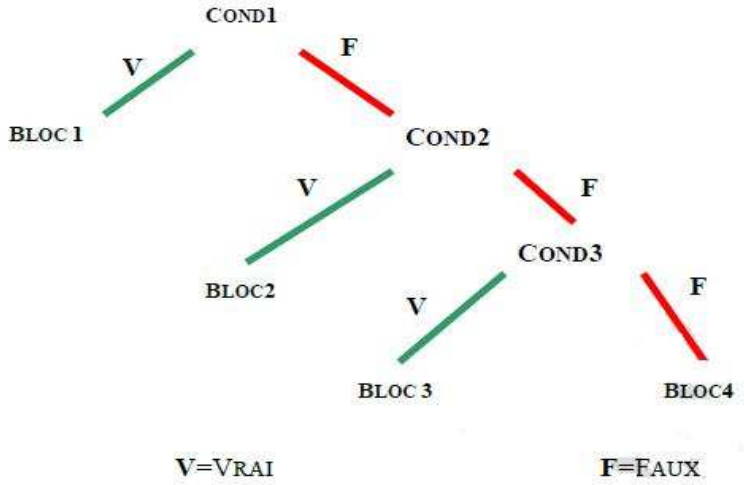
```
int main ()
        {
        int valeur1 ;
        int valeur2 ;
        printf (" Entrez une 1ere valeur : ");
        scanf ("%d" ,& valeur1 );
        printf (" Entrez 2eme valeur : ");
        scanf ("%d" ,& valeur2 );
        if ( valeur1 < valeur2 )
                printf ("%d > %d\n",valeur2 , valeur1 );
        else
                printf ("%d >= %d\n",valeur1 , valeur2 );
        return 0;
        }
```

III.3 Structure imbriquée :

Syntaxe :

SI <Cond 1 > **ALORS**
< Bloc d'actions 1 >
SINON SI <Cond 2 > **ALORS**
< Bloc d'actions 2 >
SINON SI <Cond 3 > **ALORS**
< Bloc d'actions 3 >
SINON
< Bloc d'actions 4 >
FINSI
FINSI
FINSI

Fonctionnement :



Exemple en langage C:

```
int main () {
        int a = 0;
        printf (" Saisie de a : ");
        scanf ("%d" ,&a);
        if (a < 0) /* Strictement negative ? */
                printf ("la variable a est negative .\n");
        else
        {
        if (a > 0) /* Strictement positive ? */
                printf ("la variable a est positive \n");
        else /* Sinon a est nulle */
                printf ("la variable a est nulle \n");
        }
        return 0;     }
```

III.4 Structure de branchement sélective : L'instruction « switch »

C'est une extension du si … alors … sinon … Elle permet une programmation plus claire en évitant une trop grande imbrication de Si successifs.

Le branchement multiple sélectif permet de faire plusieurs conditions sur une même variable. La structure globale est la suivante :

Syntaxe :

```
CASE E OF
      Cas 1 : debut ……Fin
      Cas 2 : début ……Fin

      Cas n : début .... ..Fin
Sinon     traitement par défaut
;
Fin
```

E est une expression ordinale (dont le type est un entier, un caractère, un booléen, mais pas un réel ni une chaîne de caractères). Les **CAS X** sont des constantes ordinales du même type que **E** (sont des valeurs qui peuvent être prise par la variable sélecteur E).

CAS X peut être :

- une valeur
- une suite de valeurs : 1, 3, 5, 7, 9
- une fourchette : 0 à 9
- une plage : >= 10

- Supprimer le début et fin si vous avez une seule instruction dans un bloc.
- Une seule Expression (ou une simple variable) est testée au début puis est comparée avec les listes de valeurs.
- A la première concordance les instructions correspondantes sont exécutées puis le programme sort de la structure.
- Si aucune concordance n'est trouvée les instructions placées après le Sinon sont exécutées.

Fonctionnement :
1- **E** est évalue
2- **E** est Recherchée parmi les valeurs possibles **CAS X**
3- Le Bloc correspondant **BLOC X** est alors exécuté puis on sort
4- Sinon, aucun bloc n'est exécuté

Syntaxe en langage C

```
switch ( Variable ) {
case Valeur1 :
        // Liste d' instructions ;
break ;
case Valeur2 :
        // Liste d' instructions ;
break ;
case Valeurs ... :
        // Liste d' instructions;
break ;
default :
        // Liste d'instructions ;        }
```

Exemple en langage C:

```
int main ()
{
int choixMenu ;
printf (" ===== Menu =====\ n\n");
printf ("1. Royal Cheese \n");
printf ("2. Big Burger \n");
printf ("3. Complet Poulet \n");
printf ("4. Panini Thon \n");
printf ("\ nVotre choix ? ");
scanf ("%d", & choixMenu ); /* Saisie du choix de l' utilisateur */
printf ("\n");
switch ( choixMenu ) /* Tester le choix de l' utilisateur */
{
case 1: printf (" Vous avez choisi un Royal Cheese !");
break ;
case 2: printf (" Vous avez choisi un Big Burger !");
break ;
case 3: printf (" Vous avez choisi un Complet Poulet !");
break ;
case 4: printf (" Vous avez choisi un Panini Thon !");
break ;
default : printf (" Choix incorrect . Vous ne mangerez rien !");
break ;
}
return 0;
}
```

- N'oubliez pas d'insérer des instructions break entre chaque test, ce genre d'oubli est difficile à détecter car aucune erreur n'est signalée...
- Ceci peut être utilisé judicieusement afin de faire exécuter les mêmes instructions pour différentes valeurs consécutives

```
switch ( variable ) {
case 1:
case 2:
// instructions executees pour les valeur 1 et 2
break ;
case 3:
// instructions executees pour la valeur 3
break ;
default :
// instructions executees pour toute autre
valeur
}
```

III.5 Une façon plus courte de faire un test :

Il est possible de faire un test avec une structure beaucoup moins lourde en langage C grâce à la structure suivante :

(condition) ? instruction si vrai : instruction si faux

Remarques :

- la condition doit être entre des parenthèses
- Lorsque la condition est vraie, l'instruction de gauche est exécutée
- Lorsque la condition est fausse, l'instruction de droite est exécutée
- En plus d'être exécutée, la structure ? : renvoie la valeur résultant de

Exemple :

```
( moyenne >=10) ? printf (" Admis ") : printf (" Ajourne ");

admis = (( moyenne >=10) ? 1 : 0);
```