Structures Alternatives

Mehdi ROUAN-SERIK

Institut de maintenance et de sécurité industrielle, IMSI
Université d'Oran 2 Mohamed Ben Ahmed
mehdi.rouan@gmail+L1HSI.com

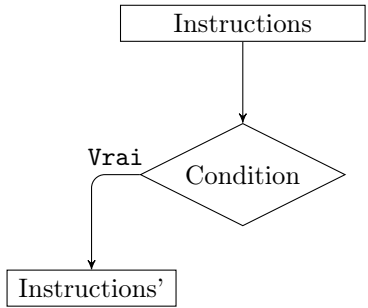
11 novembre 2022

Introduction

- Les structures alternatives ont deux syntaxes (ou formes) différentes.
- Il s'agit dans la première forme de faire un traitement si une condition est vérifiée et rien dans le cas contraire.
- Quant à la deuxième forme, on a à exécuter un traitement si une condition est satisfaite et un autre traitement si elle ne l'est pas.

Première forme

- Voici un organigramme pour représenter la 1ère forme du **si**.
- Un losange pour exprimer les conditions.



Première forme

Voici la première forme d'une condition dans un algorithme :

si Condition **alors**
    instructions' ;
finsi;

Exemple

Division

Divison

Ecrivez un algorithme qui calcul D comme quotient de A sur B (deux réels).

Exemple

Division

Divison

Ecrivez un algorithme qui calcul D comme quotient de A sur B (deux réels).

Algorithme : Division2 ;

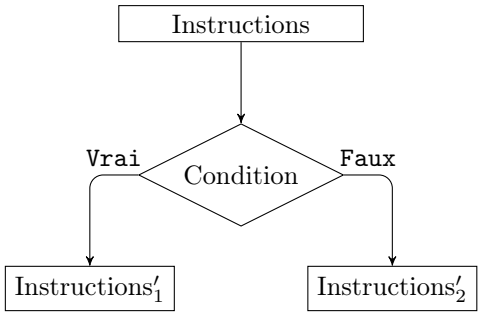
var A, B, D : **réel** ;

Début

```
    ecrire("Donnez la valeur de A");
    lire(A);
    ecrire("Donnez la valeur de B");
    lire(B);
    si B ≠ 0 alors
        D ← A / B;
        ecrire("D = ",D);
    finsi;
Fin.
```

Seconde forme

- Voici un organigramme pour représenter la 2ème forme du **si**.
- Un losange pour exprimer les conditions.



Seconde forme

Voici la seconde forme d'une condition dans un algorithme :

si Condition **alors**
  instructions'$_1$;
sinon
  instructions'$_2$;
finsi;

Exemple

Division

Divison

Écrivez un algorithme qui détermine si $n \in \mathbb{N}$ donné par l'utilisateur est pair ou impaire :

$$n : \begin{cases} \text{si } \exists\, k \in \mathbb{N},\ n = 2k, & \text{pair,} \\ \text{sinon,} & \text{impair.} \end{cases}$$

Exemple

Division

Divison

Écrivez un algorithme qui détermine si $n \in \mathbb{N}$ donné par l'utilisateur est pair ou impaire :

$$n : \begin{cases} \text{si } \exists k \in \mathbb{N},\ n = 2k, & \text{pair,} \\ \text{sinon,} & \text{impair.} \end{cases}$$

Algorithme : Parity ;

var n : **entier** ;

Début

```
ecrire("Donnez la valeur de n");
lire(n);
si n mod 2 = 0 alors
    ecrire(n, "est un nombre paire");
sinon
    ecrire(n, "est un nombre
     impaire");
finsi;
```

Fin.

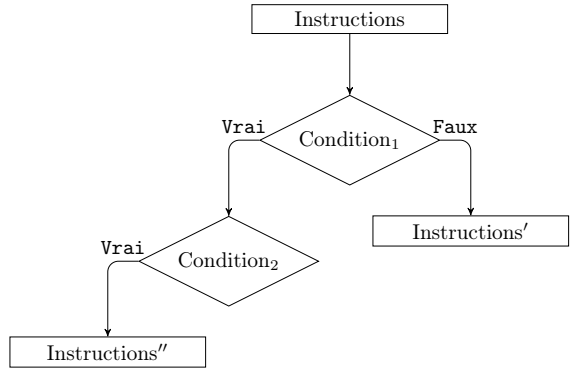
Conditions composées

- Construite à partir d'une combinaison de deux ou plusieurs conditions simples.
- Vérifier si $x \in \mathbb{R}$ donnée $x \in [-2, 0]$?
- Composées à partir de (Et, Ou, Non) et ($<$,$>$,$\neq$).

si x ≥ -2 **Et** x ≤ 0 **alors**
  `ecrire("message 1");`
sinon
  `ecrire("message 2");`
finsi;

Conditions imbriquées

- Une condition à l'intérieur d'une autre.
- Nécessaire pour certains problèmes.
- Organigramme montrant une imbrication :



Exemple

$ax^2 + bx + c = 0$

Second degrée

Écrivez un algorithme qui résout :

$$ax^2 + bx + c = 0, \quad a, b, c \in \mathbb{R}$$

Exemple

$ax^2 + bx + c = 0$

Second degrée

Écrivez un algorithme qui résout :

$$ax^2 + bx + c = 0, \quad a, b, c \in \mathbb{R}$$

1. $S = \varnothing$ si $\Delta < 0$.
2. $S = \left\{ \dfrac{-b}{2a} \right\}$, si $\Delta = 0$.
3. $S = \left\{ x_{1,2} = \dfrac{-b \mp \sqrt{\Delta}}{2a} \right\}$.

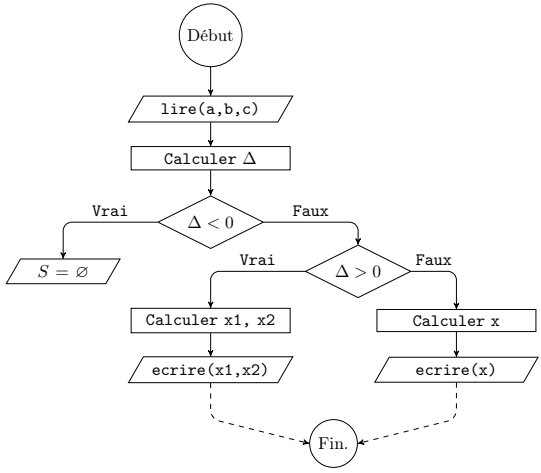
Exemple

$ax^2 + bx + c = 0$

Second degrée

Écrivez un algorithme qui résout :

$$ax^2 + bx + c = 0, \quad a, b, c \in \mathbb{R}$$



Déroulement

Algorithme : Deroulement ;

var x : **réel** ;
Début
    ecrire(*"Donnez x"*);
    lire(x);
    si $x \geq -2$ **Et** $x \leq 0$ **alors**
        ecrire(*"message 1"*);
    sinon
        ecrire(*"message 2"*);
    finsi;
Fin.

1. Dérouler pour $x = 3$.
2. Dérouler pour $x = -1$
3. Que fait cet algorithme ?

Déroulement

Algorithme : Deroulement ;

var x : **réel** ;
Début
    ecrire(*"Donnez x"*);
    lire(x);
    si $x \geq -2$ **Et** $x \leq 0$ **alors**
        ecrire(*"message 1"*);
    sinon
        ecrire(*"message 2"*);
    finsi;
Fin.

1. Dérouler pour $x = 3$.
2. Dérouler pour $x = -1$
3. Que fait cet algorithme ?

x	$x \geq -2$ **Et** $x \leq 0$	Écran
		Donnez x
		3
3		
	Faux	
		message 2

Déroulement

Algorithme : Deroulement ;

var x : **réel** ;
Début

```
ecrire("Donnez x");
lire(x);
si x ≥ −2 Et x ≤ 0 alors
    ecrire("message 1");
sinon
    ecrire("message 2");
finsi;
```

Fin.

1. Dérouler pour $x = 3$.
2. Dérouler pour $x = -1$
3. Que fait cet algorithme ?

x	$x \geq -2$ **Et** $x \leq 0$	Écran
		Donnez x
		-1
-1		
	Vrai	
		message 1

Déroulement

```
Algorithme : Deroulement ;
var x : réel ;
Début
    ecrire("Donnez x");
    lire(x);
    si x ≥ −2 Et x ≤ 0 alors
        ecrire("message 1");
    sinon
        ecrire("message 2");
    finsi;
Fin.
```

1. Dérouler pour $x = 3$.
2. Dérouler pour $x = -1$
3. Que fait cet algorithme ?

message 1 = x appartient à [-2,0]
message 2 = x n'appartient pas à [-2,0]

Conclusion

- La conception des algorithmes c'est une étape importante dans tout processus de résolution de problèmes.

- L'algorithme final repose sur cette conception et des erreurs dans l'analyse des besoins ou de cette même conception faussera ensuite l'algorithme.

- Les conditions sont des instructions nécessaires à l'écriture de nombre d'algorithmes.