

Algorithmique et Structures de Données 1

Niveau MPI

Année universitaire
2019-2020

Dr. Aymen SELLAOUTI
Dr. Majdi JRIBI

Chapitre 2

Les structures conditionnelles

Plan

3

- Partie 1: Introduction
- Partie 2: Les formes conditionnelles
- Partie 3: De l'algorithmique au langage C

Plan

4

- Partie 1: Introduction
- Partie 2: Les formes conditionnelles
- Partie 3: De l'algorithmique au langage C

Introduction

5

- Les structures conditionnelles servent à n'exécuter une instruction ou une séquence d'instructions que si une **condition** est vérifiée.
- Une condition est une expression écrite entre parenthèses à **valeur booléenne**.

Plan

6

- Partie 1: Introduction
- Partie 2: Les formes conditionnelles
- Partie 3: De l'algorithmique au langage C

Structure d'un test

7

■ En Algorithmique, il y a **deux formes possibles** pour un test :

■ Forme conditionnelle simple

- ✓ Forme simple réduite
- ✓ Forme complète ou alternative

■ Forme conditionnelle généralisée

Forme conditionnelle simple

8

Forme Simple Réduite

[Instructions d'initialisation]

Si condition Alors

Instructions à exécuter dans le cas où le résultat de l'évaluation de la condition est **VRAI**.

Fin Si

Remarque :

Dans le cas où l'évaluation de la condition est FAUX, les instructions ne seront pas exécutées.

Forme Alternative ou Complète

[Instructions d'initialisation]

Si condition Alors

instructions 1 à exécuter si l'évaluation de la condition est **VRAI**.

Sinon

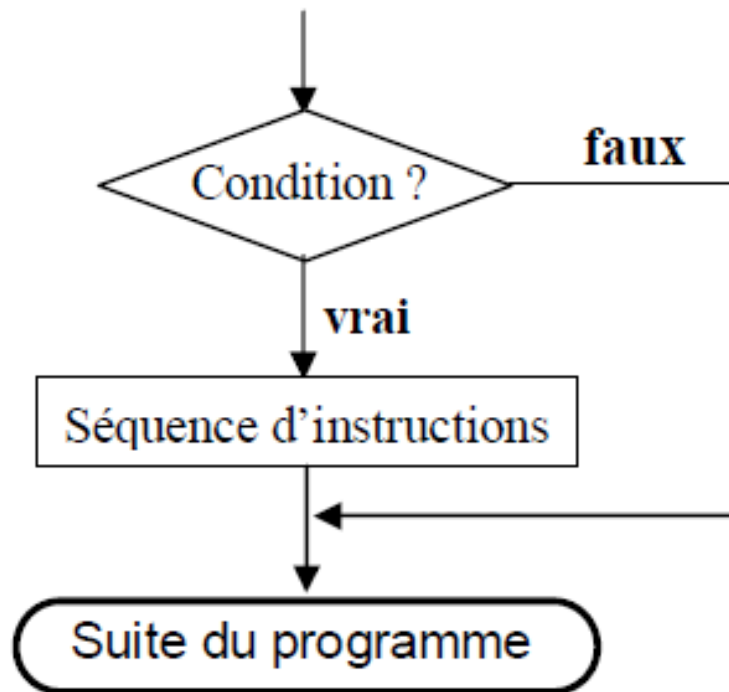
instructions 2 à exécuter si l'évaluation de la condition est **FAUX**.

Fin Si

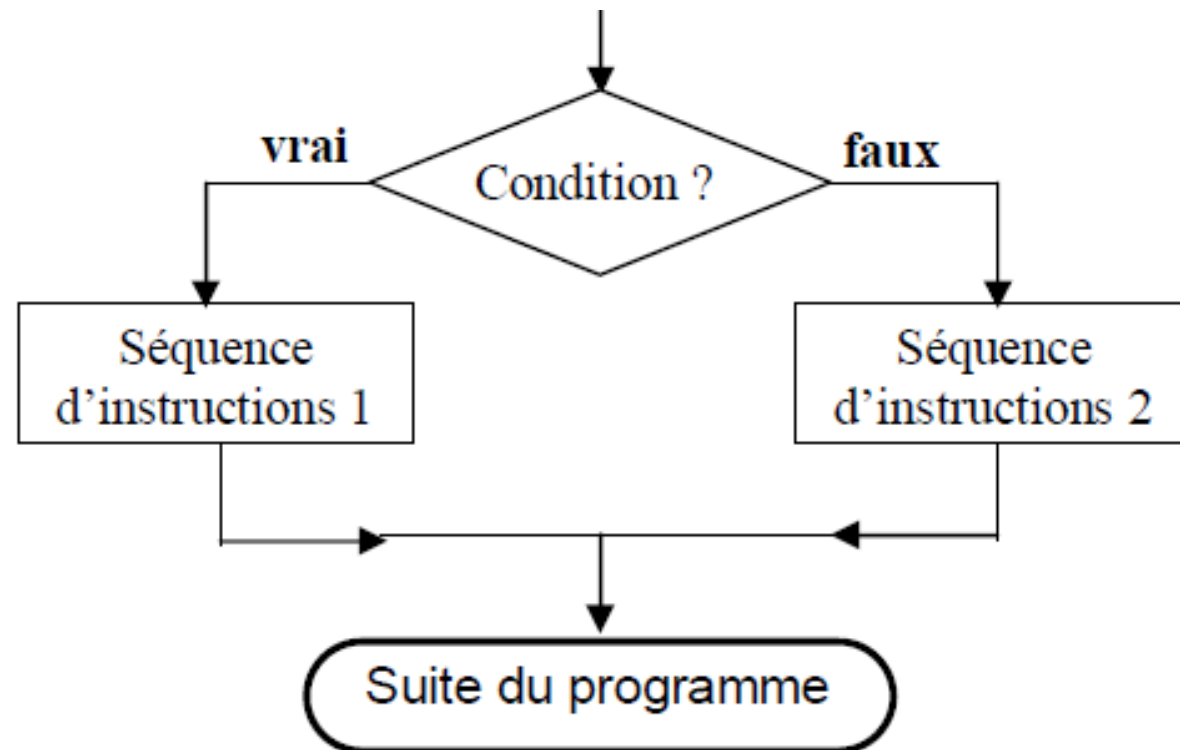
Forme conditionnelle simple

9

Forme Simple Réduite



Forme Alternative ou Complète



Qu'est ce qu'une condition?

10

- Une condition est une **comparaison** qui, à un moment donné, est vraie ou fausse.
- Les **opérateurs de comparaison** sont:
 - $=$: égal à ...
 - \neq : différent de ...
 - $<$: strictement plus petit que ..
 - $>$: strictement plus grand que .
 - \leq : plus petit ou égal à ...
 - \geq : plus grand ou égal à ...

Qu'est ce qu'une condition ?

11

- la condition peut être une condition simple ou une condition composée de plusieurs conditions.
- Une **condition composée** est une condition formée de plusieurs conditions simples reliées par des opérateurs logiques: **ET**, **OU**, **OU Exclusif** et **NON**

Exemples:

- x est compris entre 2 et 6: $(x > 2) \text{ ET } (x < 6)$
- n appartient à l'intervalle $]2,4] \cup]8,12]$: $((n > 2) \text{ ET } (n \leq 4)) \text{ OU } ((n > 8) \text{ ET } (n \leq 12))$

Exemple 1

12

- Ecrire un algorithme qui calcule et affiche la valeur absolue d'un entier quelconque lu au clavier.

Version 1 : avec une forme simple

Algorithme Valeur_Absolue2

Var : x, abs : entier

Début

Ecrire("Entrer un entier")

Lire(x)

abs \leftarrow x

si (x < 0) **alors**

abs \leftarrow -x

Finsi

Ecrire("la valeur absolue de ",x, "est",abs)

Fin

Exemple 1

13

- Ecrire un algorithme qui calcule et affiche la valeur absolue d'un entier quelconque lu au clavier.

Version 2 : avec une forme alternative

Algorithme Valeur_Absolue2

Var : x, abs : entier

Début

Ecrire("Entrer un entier")

Lire(x)

si ($x > 0$) **alors**

abs \leftarrow x

sinon

abs \leftarrow -x

Finsi

Ecrire("la valeur absolue de",x, "est",abs)

Fin

Exemple 2

14

- Ecrire un algorithme qui demande un nombre entier à l'utilisateur puis teste et affiche s'il est divisible par 3

Algorithme Divisible_par3

Var : n : entier

Début

Ecrire("Entrer un entier")

Lire(n)

si ((n % 3)=0) **alors**

Ecrire(n, "est divisible par 3")

sinon

Ecrire(n, "n'est divisible par 3")

Finsi

Fin

Exemple 3

15

- Ecrire un algorithme qui calcule le maximum de deux entiers x et y

Version 1 : avec une forme alternative

Algorithme Maximum1

Var : x,y,maxi : entier

Début

Ecrire("Entrer deux entiers x et y")

Lire(x,y)

si (x < y) **alors**

 maxi \leftarrow y

sinon

 maxi \leftarrow x

Finsi

Ecrire("Le maximum de",x, "et" ,y, "est", maxi)

Fin

Exemple 3

16

- Ecrire un algorithme qui calcule le maximum de deux entiers x et y

Version 2 : avec une forme simple

Algorithme Maximum2

Var : x,y,maxi : entier

Début

Ecrire("Entrer deux entiers x et y")

Lire(x,y)

maxi \leftarrow x

si (x < y) **alors**

maxi \leftarrow y

Finsi

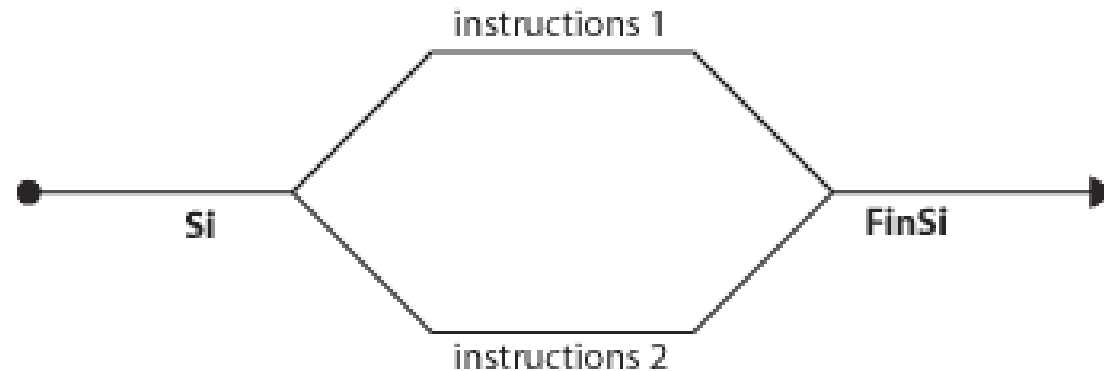
Ecrire("Le maximum de",x, "et" ,y, "est", maxi)

Fin

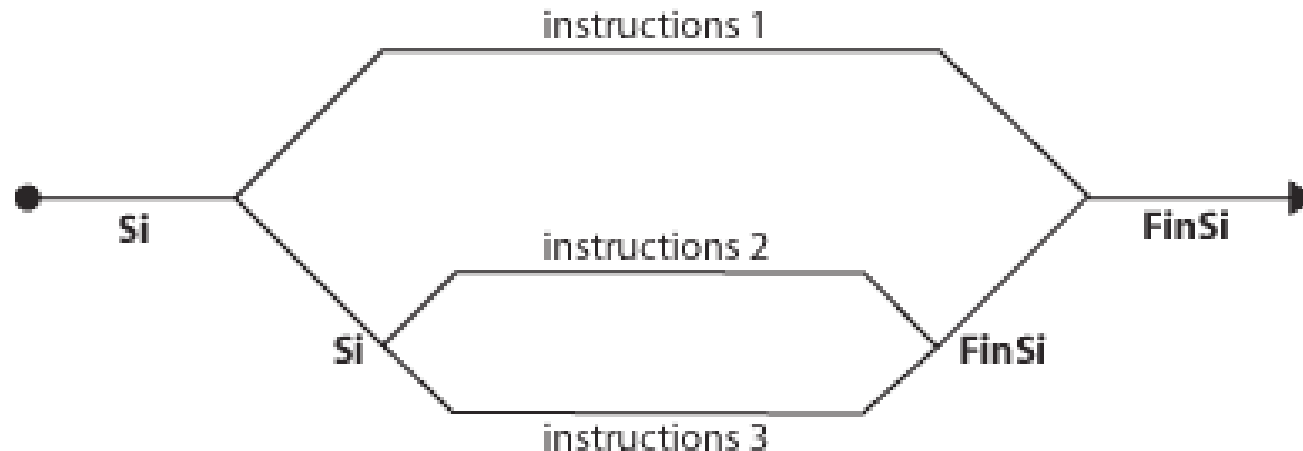
Forme conditionnelle généralisée

17

- Une condition permet de choisir **entre deux actions**



- Dans plusieurs cas, le choix doit se faire entre plus de deux actions



Exemple 4

18

- Un algorithme qui donne l'état de l'eau selon sa température doit pouvoir fournir trois réponses: (1) solide, (2) liquide, (3) gazeux.

Algorithme Température_Eau_1

Var : Temp: réel

Début

Ecrire("Entrer la température de l'eau")

Lire(Temp)

si (Temp \leq 0) **alors** **Ecrire**("C'est de la glace")

Finsi

si (Temp $>$ 0) **ET** (Temp $<$ 100) **alors** **Ecrire**("C'est du liquide")

Finsi

si (Temp \geq 100) **alors** **Ecrire**("C'est de la vapeur")

Finsi

Fin

Exemple 4

19

Algorithme Température_Eau_2

Var : Temp: réel

Début

Ecrire("Entrer la température de l'eau")

Lire(Temp)

si (Temp \leq 0) **alors** **Ecrire**("C'est de la glace")

Sinon

si (Temp < 100) **alors** **Ecrire**("C'est du liquide")

Sinon **Ecrire**("C'est de la vapeur")

Finsi

Finsi

Fin

- Les structures de **tests imbriqués** sont un outil indispensable à la **simplification** et à **l'optimisation** des algorithmes.

Forme conditionnelle généralisée

20

[Instructions d'initialisation]

Si condition 1 **Alors**

instructions 1

Sinon Si condition 2 **Alors**

instructions 2

Sinon Si condition 3 **Alors**

instructions 3

Sinon

instructions

Fin Si

Fin Si

Fin Si

[Instructions d'initialisation]

Si condition 1 **Alors**

instructions 1

SinonSi condition 2 **Alors**

instructions 2

SinonSi condition 3 **Alors**

instructions 3

Sinon

instructions

Fin Si

Exemple 5

21

- Ecrire un algorithme qui demande deux nombres à l'utilisateur et l'informe ensuite si leur produit est négatif ou positif (on inclut le cas où le produit peut être nul sans le calculer)

Algorithme Signe_Produit

Var : x, y : réels

Début

Ecrire('Entrer deux réels x et y')

Lire(x, y)

Si ($x = 0$) OU ($y = 0$) **Alors**

Ecrire ("Le produit est nul")

SinonSi ($x < 0$ ET $y < 0$) OU ($x > 0$ ET $y > 0$) **Alors**

Ecrire ("Le produit est positif")

Sinon

Ecrire ("Le produit est négatif")

Finsi

Fin

Choix multiple

22

Choix multiple

Selon *< expression >* **faire**

valeur1 : première séquence d'instructions;

valeur2 : deuxième séquence d'instructions;

...

valeurN : Nème séquence d'instructions;

Sinon (*N+1*) ème séquence d'instructions;

FinSelon

Exemple 6

23

moisA30Jours

Algorithme moisA30Jours

Var : mois : Entier, résultat : Booléen

début

selon mois **faire**

 4,6,9,11 : résultat \leftarrow Vrai

 sinon : résultat \leftarrow Faux

finselon

fin

Plan

24

- Partie 1: Introduction
- Partie 2: Les formes conditionnelles
- Partie 3: De l'algorithmique au langage C

Traduire Si...Sinon...

25

Forme Simple Réduite

```
if (condition réalisée)  
    {liste d'instructions;}
```

Forme généralisée

```
if ( condition1 )  
    Instruction à exécuter  
    si la condition1 est vraie.
```

```
else if ( condition2 )  
    Instruction à exécuter  
    si la condition2 est vraie.
```

```
else if ( condition3 )  
    Instruction à exécuter  
    si la condition3 est vraie.
```

...

```
else if (conditionN-1)
```

```
else
```

Instruction à exécuter
si les N-1 conditions sont toutes fausses

Forme Simple complète

```
if (condition réalisée)  
    {liste d'instructions}  
else  
    {autre série d'instructions}
```

Traduire Selon

26

switch (Variable)

```
{  
  case Valeur 1:  
    Liste d'instructions;  
    break;  
  case Valeur 2:  
    Liste d'instructions;  
    break;  
  .....  
  case Valeur n:  
    Liste d'instructions;  
    break;  
  default:  
    Liste d'instructions;  
}
```

Pour exécuter les mêmes instructions
pour différentes valeurs :

Exemple :

switch(variable)

```
{  
  case 1:  
  case 2:  
    {instructions exécutées pour variable = 1  
    ou pour variable = 2}  
    break;  
  case 3:  
    {instructions exécutées pour variable = 3}  
    break;  
  default:  
    {instructions exécutées pour toute  
    autre valeur de variable} }  
}
```

Switch Remarque

27

- Vous devez mettre une instruction `break`; obligatoirement à la fin de chaque cas. Si vous ne le faites pas, alors l'ordinateur ira lire les instructions en dessous censées être réservées aux autres cas ! L'instruction `break`; commande en fait à l'ordinateur de « sortir » des accolades.

Exemple

28

Exemple

```
switch(choix) {  
    case 't' : printf("vous voulez un triangle \n"); break;  
    case 'c' : printf("vous voulez un carre \n"); break;  
    case 'r' : printf("vous voulez un rectangle \n"); break;  
    default : printf("erreur . recommencez !\n");  
}
```

Exemple

29

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

    main(void)
{
    char reponse;

    printf("Que voulez-vous faire ?\n");
    printf("q pour quitter\n");
    printf("s pour sauvegarder\n");
    printf("o pour ouvrir un nouveau fichier\n");

    scanf("%c", &reponse);

    switch(reponse) {
    case 'q': printf("au revoir\n"); break;
    case 's': printf("sauvegarde \n"); break;
    case 'o': printf("ouverture\n");break;
    default: printf("ce n'est pas un choix valable\n");
    }
}
```