
Les structures de contrôle

Algorithmique et Programmation en C

Chapitre 4 Karim Bouzoubaa

Chapitre 4 : Les structures de contrôle

- Objectifs
 - Être capable d'écrire des instructions conditionnelles
 - Être capable d'écrire des instructions de répétition

- Sommaire
 - Introduction
 - La structure de sélection si
 - La structure de sélection si ... sinon
 - La structure de répétition Tant que
 - La structure de répétition Pour
 - La structure de répétition Faire ... Tant que
 - Programme récapitulatif

Introduction

- Les algo jusqu'ici
 - instructions d'affectation ($a = 5$)
 - entrées/sorties (`afficher`, `lire`)
- Programmes informatiques sont plus complets et contiennent des instructions plus élaborées
- Exemple (rappel/chapitre 2):

Algorithme de plantation et d'arrosage de plusieurs arbres:

- 1- Creuser un trou
- 2- Placer un arbre dans le trou.
- 3- Reboucher le trou
- 4- **S**'il existe encore des arbres
Exécuter **les actions 1, 2, 3 et 4**
Sinon Exécuter les actions suivantes
- 5- Arroser les arbres

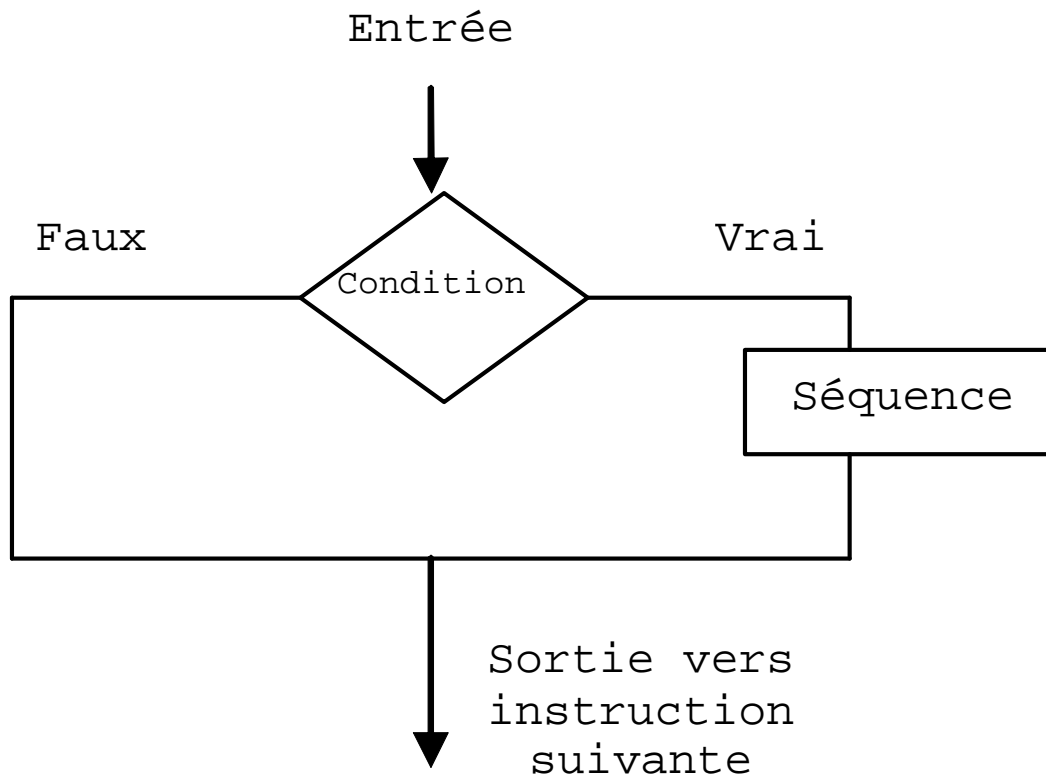
- *L'action 4 vérifie si une condition est vraie ou non*
⇒ sélection
- *Les actions 1,2,3 et 4 sont répétées un certain nombre de fois*
⇒ Itération

Introduction

- L'informatique fournit d'autres types d'instructions appelés **structures de contrôle**
- Ces structures permettent de contrôler l'ordre d'exécution des instructions
- Les structures conditionnelles (de sélection):
 - **si**
 - **si ... sinon**
- Les structures répétitives (Itérations):
 - **Tant que**
 - **Faire ... Tant que**
 - **Pour**

La structure de sélection si

- La sélection `si` consiste à exécuter une séquence d'instructions seulement lorsqu'une certaine condition est réalisée



```
instruction0 ;  
si (condition) {  
    instruction1 ;  
    instruction2 ;  
    ...  
    instructionn ;  
}  
instructionn+1 ;
```

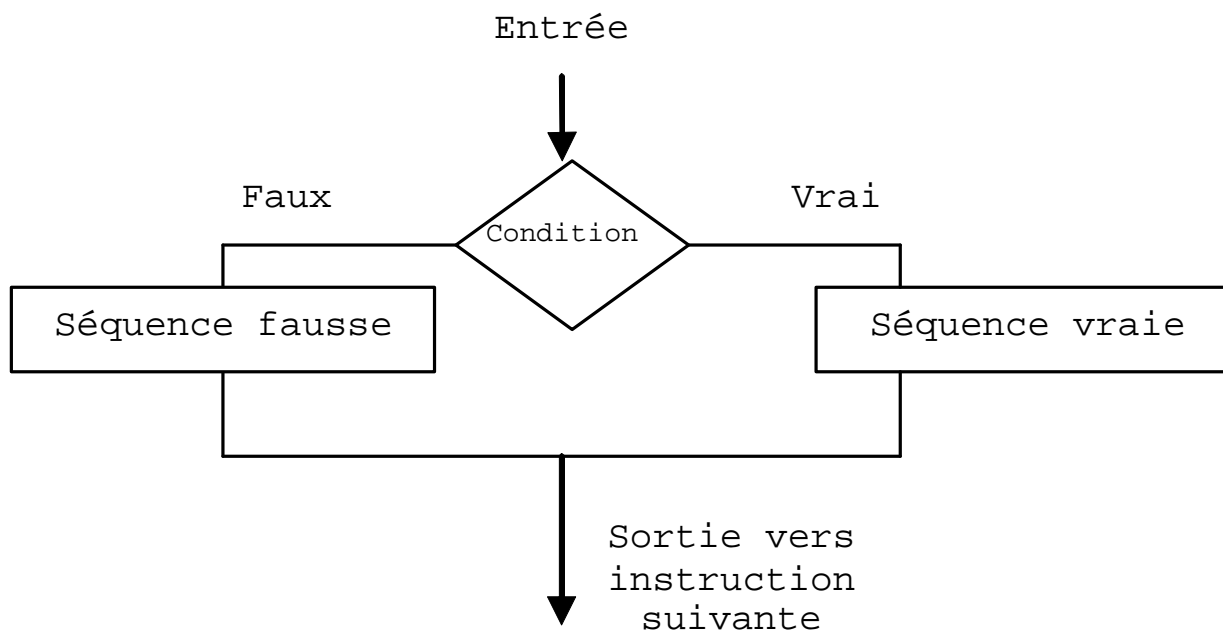
La structure de sélection si

- Soient a et b deux valeurs. Écrire un algo qui détermine et affiche le plus grand de ces deux valeurs

```
• Algorithme "maximum de 2 variables"  
• Entrées : x, y type entier  
• Sorties : max type entier  
• Traitement:  
  Début  
  Lire x, y  
  si (x > y) max = x  
  si (y ≥ x) max = y  
  Afficher(" le maximum est : ", max)  
  Fin
```

La structure de sélection si .. sinon

- La sélection `si ... sinon` consiste à choisir entre la réalisation de l'une ou l'autre de deux séquences selon la valeur de la condition.



```
instruction ;  
  
si (condition) {  
    instruction1 ;  
    instruction2 ;  
    ...  
    instructionn ;  
}  
  
sinon {  
    instruction1 ;  
    instruction2 ;  
    ...  
    instructionm ;  
}  
  
instruction ;
```

La structure de sélection si .. sinon

- Soient a et b deux valeurs. Écrire un algo qui détermine et affiche le plus grand de ces deux valeurs

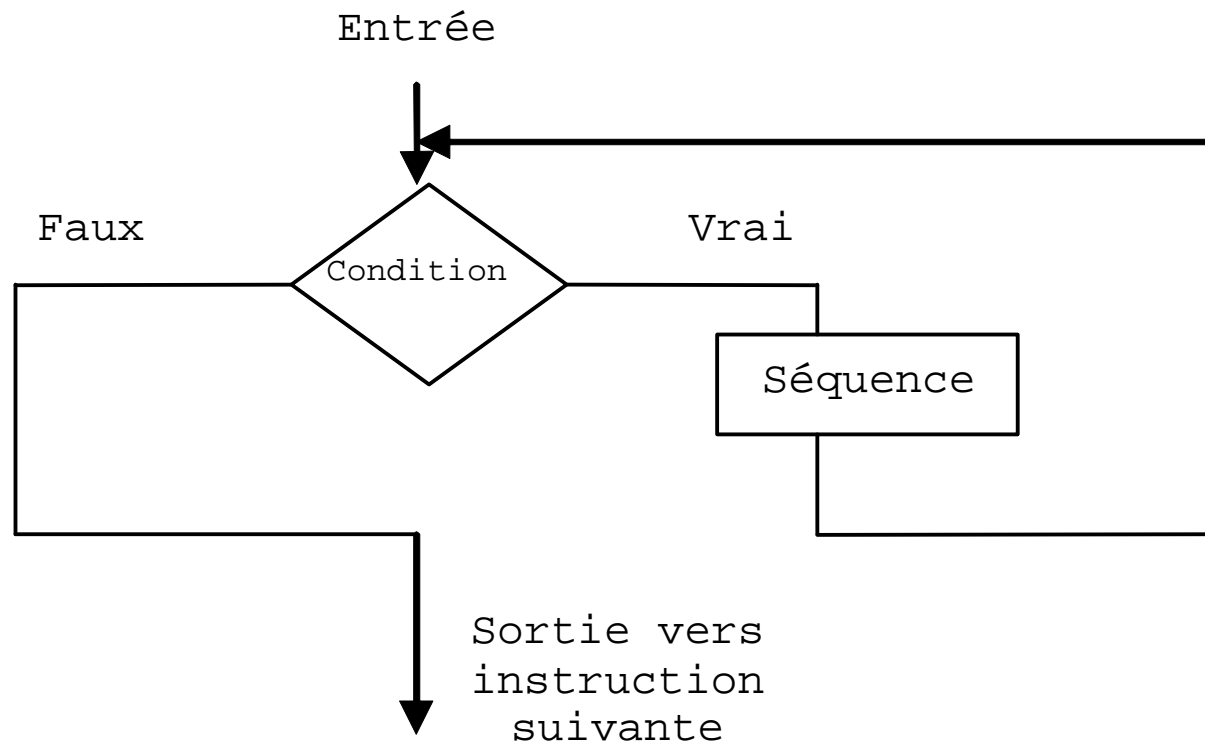
```
• Algorithme "maximum de 2 variables"  
• Entrées : x, y type entier  
• Sorties : max type entier  
• Traitement:  
  Début  
  Lire x, y  
  si (x > y) max = x  
  sinon max = y  
  Afficher(" le maximum est : ", max)  
  Fin
```

Exercices

1. Trouver un algo qui détermine et affiche le minimum de trois valeurs (+ieurs algo possibles)
2. Écrire un algo qui résout l'équation $ax + b = 0$
3. Écrire un algo qui résout l'équation $ax^2 + bx + c = 0$

La structure de répétition Tant que

- La séquence est exécutée autant de fois que nécessaire tant qu'une certaine condition est vérifiée



```
TQ (condition) {  
    instruction1 ;  
    instruction2 ;  
    ...  
    instructionn ;  
}
```

La structure de répétition Tant que

- Algo qui affiche la suite 0 2 4 6 8 ... 50

- Algorithmme "affichage suite 0 2 ... 50"
- Entrées :
- Sorties :
- Auxiliaires: val

- Traitement:
Début
val = 0
TQ (val < 52) {
 afficher val
 val = val + 2
}
Afficher("Fin")
Fin

La structure de répétition Tant que

- Algo qui calcule la moyenne de n valeurs ($n > 0$)

- Algorithme "moyenne de n valeurs"
- Entrées : n, val type entier
- Sorties : moy type reel
- Auxiliaires: compt, som

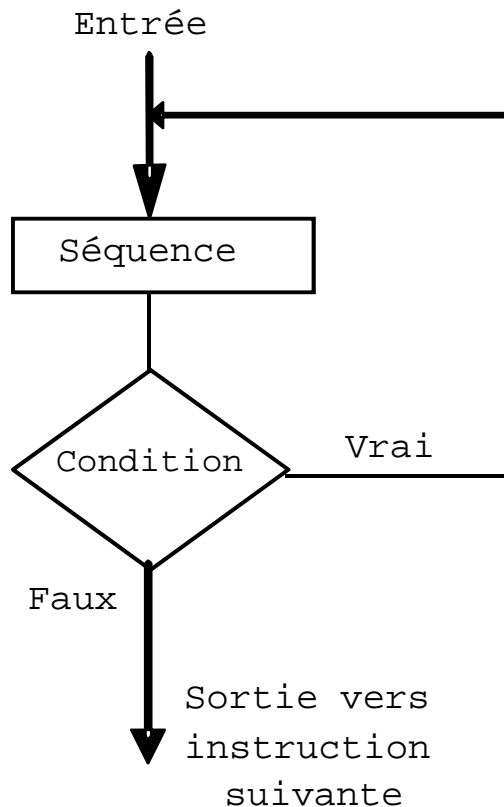
- Traitement:
Début
Lire n
compt = 0
som = 0
TQ (compt < n) {
 lire val
 som = som + val
 compt = compt + 1
}
moy = som / n
Afficher(" la moyenne est : ", moy)
Fin

Exercices

1. Algo qui affiche la suite 0 5 10 ... 100
2. Algo qui affiche la suite 100 90 80 ... 10
3. Algo qui calcule le factoriel d'un nombre n
4. Algo qui lit une série de valeurs, s'arrête à -1 et affiche la somme, produit et moyenne des valeurs saisies
5. Algo qui calcule le produit de deux nombres m et n (en utilisant seulement l'opération d'addition)

La structure de répétition Faire ..TQ

- La structure Faire-TQ utile principalement lorsque l'on désire exécuter les instructions d'une répétition au minimum une fois.
- La condition est vérifiée à la fin de la boucle au lieu du début
 - Même si la condition est fausse depuis le début, la séquence est exécutée au minimum une fois



```
Faire {  
    instruction1 ;  
    instruction2 ;  
    ...  
    instructionn ;  
} TQ (condition) ;
```

La structure de répétition Faire ..TQ

- Algo qui calcule le produit $P = 10 \cdot 9 \cdot 8 \cdot \dots \cdot 1$

```
• Algorithme "produit de 10 à 1"  
• Sorties : p type entier  
• Auxiliares: compt  
• Traitement:  
  
  Début  
  
  compt = 10  
  
  p = 1  
  
  Faire {  
    p = p * compt  
    compt = compt - 1  
  } TQ (compt > 0)  
  
  Afficher("le produit est:", p)  
  
  Fin
```

La structure de répétition Faire ..TQ

- L'utilisation la plus courante: Lecture d'une variable avec conditions

Val dans $[0, +\infty]$

- ...
- Entrée: val entier
- ...
- Traitement:
 - ...
 - Faire {
 lire val
} TQ (val < 0)
 - ...

Val dans $[5, 10]$

- ...
- Entrée: val entier
- ...
- Traitement:
 - ...
 - Faire {
 lire val
} TQ (val < 5 OU val > 10)
 - ...

La structure de répétition Pour

- La boucle Pour est utilisée pour exprimer un nombre fini de répétitions

```
Pour var = initial à final PAS pas {  
  instruction1 ;  
  instruction2 ;  
  ...  
  instructionn ;  
}
```

- Lorsque pas = 1, il peut être ignoré

La structure de répétition Pour

- Algo qui affiche la suite 0 2 4 6 8 ... 50

- Algorithme "suite pairs < 50"

- Auxiliaires: i entier

- Traitement:

Début

Pour i=0 à 50 pas 2 {

 afficher (i)

}

Fin

La structure de répétition Pour

- Algo qui affiche les n premières puissances de 2

```
• Algorithme "n puissances de 2"  
• Entrées : n entier  
• Auxiliaires: i, p entier  
• Traitement:  
  Début  
  Lire n  
  p = 1  
  Pour i=1 à n {  
    p = p * 2  
    afficher (p)  
  }  
  Fin
```

La structure de répétition Pour

- Algo qui calcule
- $f(x,y) = x / (x^2 + y^2)$
 - $x = 0, 2, \dots, 10$
 - $y = 1, 3, \dots, 21$
- Nous avons un emboîtement de boucles, c'est-à-dire que la deuxième boucle fait partie intégrante de la première boucle. La deuxième boucle est exécutée complètement pour chaque itération de la première

```
• Algorithmme "fonction f(x,y)"
• Auxilieres:
  - i, j entier
  - f réel
• Traitement:
  Début
  Pour x = 0 à 10 pas 2 {
    Pour y = 1 à 21 pas 2 {
      f = x / (x*x + y*y)
      afficher (f)
    }
  }
  Fin
```

Exercices

1. Les exercices sur TQ
2. Algo qui calcule la somme des n premiers nombres premiers qui est supérieur à 100
3. Algo qui calcule la somme S des 100 premiers termes de la suite

$$S = 1 - \frac{1}{2} + \frac{1}{4} - \dots + (-1)^p / 2^p$$

4. Algo qui calcule la somme
 - En additionnant les termes de gauche à droite
 - En additionnant les termes de droite à gauche
 - En additionnant séparément les termes positifs et négatifs chacun de gauche à droite

$$S = 1 - \frac{1}{2} + \frac{1}{3} \dots + \frac{1}{999} - \frac{1}{1000}$$