**Chapitre 4**

**Les structures de Contrôles : Conditionnelles et Itératives**

1. Introduction

Les algorithmes vus précédemment étaient très simples. En effet, pour l'instant, nous sommes seulement en mesure de décrire une suite d'opérations, chacune devant être exécutée une et une seule fois. Autrement dit on ne peut ni conditionner l'exécution de certaines instructions, ni répéter plusieurs fois la même action. Le but de ce cours est d’enrichir nos algorithmes par des structures conditionnelles et itératives.

1. Les Structures de contrôles conditionnelles

La structure de contrôle conditionnelle permet à un programme de modifier son traitement en fonction d'une condition.

Il existe trois formes d'instructions conditionnelles:

* Forme simple
* Forme généralisée.
* Forme à choix
  1. La structure de contrôle conditionnelle simple
     1. La forme réduite

Définition

Une structure de contrôle conditionnelle est dite à forme simple réduite lorsque le traitement dépend d'une condition. Si la condition est évaluée à « vrai », le traitement est exécuté.

Exemple

Pour que l’enfant puisse étudier en première année de base il doit avoir 6 ans.

Vocabulaire et syntaxe

|  |  |
| --- | --- |
| En Algorithme | En langage C |
| Si condition  Alors  Instruction 1 ;  Instruction 2 ;  ….  Instruction N ;  FinSi | If (condition)  {  Instruction 1 ;  Instruction 2 ;  ….  Instruction N ;  } |

* + 1. La forme Alternative

**Définition**

Une structure de contrôle conditionnelle est dite à forme alternative lorsque le traitement dépend d'une condition à deux états: Si la condition est évaluée à « vrai », le premier traitement est exécuté, si la condition est évaluée à « faux », le second traitement est exécuté.

**Vocabulaire et syntaxe**

|  |  |
| --- | --- |
| **En Algorithme** | **En langage C** |
| Si condition  Alors  Instruction 1 ;  ….  Instruction N ;  Sinon  Instruction 1 ;  ….  Instruction N ;  FinSi | If (condition)  {  Instruction 1 ;  ….  Instruction N ;  }  Else  {  Instruction 1 ;  ….  Instruction N ;  } |

**Exemple**

Chaque weekend, monsieur Ali se lève le matin, il prend sa douche ainsi que son petit déjeuner puis il s’habille. Monsieur Ali est un amateur de natation, s’il fait beau, il se déplace vers la plage pour se baigner. S’il pleut, il se dirige vers le super marché pour faire ses courses. A la fin de la journée il rentre à la maison. Décrire en utilisant un algorigramme et une séquence d’instructions algorithmiques le weekend de monsieur Ali.

* 1. La structure de contrôle conditionnelle généralisée

**Définition**

Une structure de contrôle conditionnelle est dite généralisée lorsqu'elle permet de résoudre des problèmes comportant plus de deux traitements en fonction des conditions. L'exécution d'un traitement entraîne automatiquement le non exécution des autres traitements.

**Vocabulaire et syntaxe**

|  |  |
| --- | --- |
| **En Algorithme** | **En langage C** |
| **Si** condition 1 **Alors**  Traitement 1 ;  **Sinon Si** condition 2 **Alors**  traitement 2 ;  **Sinon Si** condition 3 **Alors**  traitement 3 ;  **….**  **Sinon Si** condition N-1  **Alors** traitement N-1 ;  **Sinon** traitement N ;  **FinSi** | If (condition1)  {  Instruction 1 ;  }  Else If (condition2)  {  Instruction 2 ;  }  Else If (condition N-1)  {  Instruction N-1 ;  }  Else  Instruction N ; |

**Exemple**

Ecrire un algorithme et un programme en C qui compare deux entiers A et B.

* 1. La structure de contrôle conditionnelle à choix

**Définition**

La structure de choix permet de faire un choix parmi plusieurs possibilités. Le choix du traitement à effectuer dépend de la valeur que prendra un sélecteur. Ce sélecteur (soit une expression soit une variable) est comparé à une série de valeurs ou à un ou plusieurs intervalles. En cas d'égalité (ou fait partie d'un intervalle), seulement le traitement qui lui est associé est exécuté. Les autres seront ignorés (sautés).

**Vocabulaire et syntaxe**

|  |  |
| --- | --- |
| **En Algorithme** | **En langage C** |
| selon sélecteur Faire  Valeur 1 : Action 1 ;  Valeur 2 : Action 2 ;  …..  Valeur N : Action N ;  Sinon Action R ;  FinSelon | switch (sélecteur)  {  case Valeur 1 : Action 1; break ;  case Valeur 2 : Action 2 ; break ;  …..  case Valeur N : Action N ; break ;  default : Action R ;  } |

**Exemple**

Ecrivez un programme qui donne la saison, à partir d'un numéro de mois lu au clavier.

* 1. Evaluation

1. Ecrire un programme qui détermine si un entier saisi du clavier est paire ou impaire.
2. Sachant qu'un étudiant est admis lorsqu'il a 10 de moyenne, qu'il a la mention passable si sa note est inférieure à 12, la mention assez bien si sa note est inférieure à 14, la mention bien si sa note est inférieure à 16, la mention très bien si sa note est inférieure à 18 et les félicitations du jury à partir de 18, écrivez le programme qui demande la note de l'étudiant et affiche s'il est admis, ainsi que sa mention éventuelle.
3. Ecrire un programme qui calcule les solutions réelles d'une équation du second degré ax2+bx+c = 0 en discutant la formule.
4. Les Structures de contrôles itératives

Une structure itérative (une boucle) permet d’exécuter plusieurs fois une même séquence d’instructions. Cet ensemble d’instructions s’appelle le corps de la boucle. Chaque exécution du corps d’une boucle s’appelle une itération, ou encore un passage dans la boucle. On distingue deux types de structures itératives :

* La structure itérative complète où le nombre de répétition est connu à l’avance.
* Les structures itératives à condition d'arrêt où le nombre de répétition est inconnu à l’avance.
  1. Structure de contrôle itérative complète : la boucle pour

**Définition**

Cette structure exprime la répétition d'un traitement, un nombre de fois connu à l’avance.

**Syntaxe**

|  |  |
| --- | --- |
| **En algorithme** | **En langage C** |
| **POUR** Vc **DE** Vi **A** Vf [PAS Vp] **FAIRE**              <Traitement>  **FIN POUR** | **For (Vc = Vi; Vc<Vf+1; Vc++)**  **{**  <Traitement>  **}** |

**Remarques :**

**En algorithme**

* Vp la valeur du pas, c'est la valeur qu'on rajoute à Vc à chaque fin de traitement.
* La valeur du pas est égale à 1 par défaut.
* La valeur du pas peut être positive ou négative et par conséquent, il faut; au départ de la boucle; que Vi <= Vf ou Vi >= Vf selon la positivité ou la négativité de cette valeur.

**Exemple 1**

POUR i = 1 à 10 Faire

Ecrire (i) ;

FIN POUR

**Exemple 2**

POUR i = 10 à 1 pas -1 Faire

Ecrire (i) ;

FIN POUR

**Exemples**

**1-** Calculer la somme S des N premiers nombres entiers ; S=1+2+3+...+N.

**2-** Ecrire l'algorithme qui permet d'afficher tous les nombres pairs qui existent entre 1 et 10.

|  |  |  |
| --- | --- | --- |
| 1ière solution  POUR i de 2 à 10 pas 2 Faire              ECRIRE(i)  FINFAIRE | 2ième solution  POUR i de 2 à 10 Faire  SI (i mod 2 = 0) ALORS              ECRIRE(i)  FINSI  FINFAIRE | 3ième solution  POUR i de 1 à 5 Faire              ECRIRE(2\*i)  FINFAIRE |

* 1. Les structures itératives à condition d’arrêt

Les structures itératives à condition d'arrêt sont utilisées dans le cas où le nombre de répétition d’un traitement est inconnu à l'avance.

On distingue deux structures : Répéter.. jusqu’à et Tant que .. Faire

* + 1. La boucle Répéter … Jusqu'à

**Définition**

Dans ce cas la suite d’actions est exécutée au moins une fois, car le test de l’expression logique est effectué après exécution de l’ensemble d’actions.

**Syntaxe**

|  |  |
| --- | --- |
| **En Algo** | **En C** |
| Répéter              <Traitement>  Jusqu'à (condition d'arrêt) | Do  { <Traitement> }  While (condition d'arrêt) |

Phase d’exécution (orale)

-1.Le bloc d’actions est exécuté .

-2.L’expression logique est testée

-3.Dans le cas où elle est égale à faux, on recommence au point 1.

-4.Dans le cas où elle est égale à vrai. Le programme poursuit son exécution après l’instruction''jusqu’à''.

**Remarques**

1.      Dans cette boucle, le traitement est exécuté au moins une fois avant l'évaluation de la condition d'arrêt.

2.      Il doit y avoir une action dans le <Traitement> qui modifie la valeur de la condition.

**Exemple**

1- Calculer la somme S des N premiers nombres entiers ; S=1+2+3+...+N.

* + 1. La structure Tant que.. faire

Définition

L’ensemble d’actions doit être exécuté tant que l’expression logique est vraie. Lorsque l’expression logique est fausse, le processus itératif s’arrête.

**Syntaxe**

|  |  |
| --- | --- |
| **En Algo** | **En C** |
| Tant que <expression logique> faire    <Traitement>  Fin Tant Que | while (expression)  {  <instruction>  } |

Phases d’exécution : (orale)

-1.Evaluation de l’expression logique.

-2.Résultat vrai :Exécuter les actions.

Reprise de l’étape précédente 1.

-3.Résultat faux :Arrêt de l’itération et le programme poursuit son exécution après FTQ.

**Remarques**

1.      Dans cette boucle, le traitement peut ne pas être exécuté aucune fois, c'est lorsque la condition d'exécution est à faux dés le départ.

2.      Les paramètres de la condition doivent être initialisés par lecture ou par affectation avant la boucle.

3.      Il doit y avoir une action dans le <Traitement> qui modifie la valeur de la condition.

**Exemple**

1- Calculer la somme des N premiers nombres entiers.

Algo Somme

Var

S,I,N : Entier

La variable S est utilisée pour additionner les différentes valeurs. Elle est initialisée à 0.

Début

Afficher ''Entrer la valeur de N''

Saisir N

La variable I est appelée variable d’itération. Elle passe en revue les nombres de 1 à N

S 0

I 1

Tant que I<=N faire

S S+I

I I+1

Cette instruction permet de faire évoluer I à la valeur suivante.

Fin Tant Que

Afficher ''La somme des '',N,'' premiers entiers est : '', S

Fin

Récapitulation

Votre professeur d’Informatique vous demande de résoudre 5 exercices de la série. Comment y procéder ? Donc le nombre de répétitions est connu à l’avance qui est le nombre d’exercices.

Pour exercice de 1 à 5 faire

Lire (énoncé)

Résoudre (exercice)

Fin Pour

Suite à l’activité précédente, l’élève, avant de commencer la résolution des exercices, il doit tout d’abord lire le cours au moins une fois jusqu’à ce qu’il le comprenne. Donc le nombre de répétition est inconnu à l’avance et le cours va être lu au moins une fois.

Répéter

Lire (cours)

Jusqu’à comprendre

En classe l’élève peut poser des questions à son professeur en cas où il n’a pas compris la leçon. Donc le nombre de répétition est inconnu à l’avance et si l’élève a compris la leçon il peut ne pas poser des questions

Tant que leçon non claire faire

Poser (question)

Fin Tant que