Chapitre II

Les structures algorithmiques de contrôle

Durée : 16 Heures
Type : Théorique / Pratique

I. Les structures conditionnelles

I.1 La structure conditionnelle simple

a. Syntaxe (voir livre page 52)

b. Définition (voir livre page 53)

Remarque : La structure conditionnelle simple peut être de forme **réduite** si on n'utilise pas le bloc Sinon.

```
Si Condition Alors Traitement
Fin si
```

Activité 1 :

- Ecrire un algorithme qui permet de saisir un entier n et de vérifier si il est divisible par son chiffre des dizaines.

a. Analyse :

Résultat : Affichage

Traitements :

Soit n le nombre à tester et d son chiffre des dizaines

Si n **Mod** d = 0 **Alors** écrire ("le nombre est divisible par son chiffre des dizaines")
Sinon écrire ("le nombre n'est pas divisible par son chiffre des dizaines")
Fin si
d ← (n **Mod** 100) **Div** 10
n = Donnée ("Donner un entier : ")

b. Algorithme

0) Début dizaine
1) Ecrire ("Donner un entier : "), Lire (n)
2) d ← (n **Mod** 100) **Div** 10
3) **Si** n **Mod** D = 0 **Alors** écrire ("le nombre est divisible par son chiffre des dizaines")
Sinon écrire ("le nombre n'est pas divisible par son chiffre des dizaines")
Fin si
5) Fin dizaine

T.D.O

Objet	Type	Rôle
n	entier	Nombre entier
d	octet	Le chiffre des dizaines

c. Pascal (voir fichier : dizaine.pas)

I.2 La structure conditionnelle Forme généralisée (Imbriquée)

a. Syntaxe (voir livre page 60)

b. Définition (voir livre page 62)

Activité 2:

Le gérant d'un magasin désir automatiser l'opération de calcul des remises accordées à ses clients. Afficher le montant net à payer sachant que :

- Net à payer = montant total de la facture – Remise

La remise se calcul comme suit

- Pour un montant ≥ à 200DT et une ancienneté ≥ 1an → Taux = 20%
- Pour un montant ≥ à 100DT → Taux = 10%
- Pour un montant < à 100DT → Taux = 5%

L'ancienneté du client et le montant total de la facture sont des données.

a. Analyse :

Résultat : Affichage

Traitements :

```
NAP ← MTF – remise
Si (MTF >=  200) et (a >= 1) Alors remise = MTF * 0,2
                        Sinon Si MTF >=  100 Alors remise = MTF * 0,1
                                             Sinon remise = MTF * 0,05
Fin si
MTF = donnée ("Précisez le montant total de la facture : ")
a = donnée ("Précisez l'ancienneté du client : ")
```

b. Algorithme

0) Début remise
1) Ecrire ("Précisez le montant total de la facture : "), Lire (MTF)
2) Ecrire ("Précisez l'ancienneté du client : "), Lire (a)
3) Si (MTF >= 200) et (a >= 1) Alors remise ← MTF * 0,2
   Sinon Si MTF >= 100 Alors remise ← MTF * 0,1
   Sinon remise ← MTF * 0,05
   Fin si
4) NAP ← MTF – remise
5) écrire ("Le net à payer est : ", NAP)
6) Fin remise

T.D.O

Objet	**Type**	**Rôle**
MTF	Réel	montant total de la facture
NAP	Réel	net à payer
Remise	Réel	
a	Octet	Ancienneté

c. Pascal (voir fichier : remise.pas)

I.3 Structure à choix multiples

Discussion :

Voir Application 2 page 64 et algorithme page 65.

Que peut-on remarquer sur la structure conditionnelle utilisée ?

Y'a-t-il d'autres structures plus adaptées ?

a. Syntaxe (voir livre page 69)

b. Définition (voir livre page 71)

Activité 3:

- On veut réaliser un programme qui effectue des calculs d'aires à la demande. Les formes que nous allons considérer sont assez élémentaires (disque, carré, ellipse, rectangle).
- En entrée, le programme devra lire une lettre qui lui indique le type de figure
  - "D" pour disque
  - "C" pour carré
  - "E" ellipse
  - "R" pour rectangle
- Suivie d'un ou deux nombres réels représentant les dimensions. S'il y a doute sur la nature de la figure, le programme affiche la surface = 0.
  - Surface d'un disque = pi × rayon²
  - Surface d'un rectangle = longueur × largeur
  - Surface d'un disque = pi × d1 × d2 / 4
  - Surface d'un carré = côté²

a. Analyse :

Résultat : écrire ("La surface est : ", aire)

Traitements :
Selon **choix** faire

"D" : p1 = donnée ("Donnez le rayon : ")
aire ← pi × SQR (p1)

"R" : d1 = donnée ("Donnez la largeur : ")
d2 = donnée ("Donnez la longueur : ")
aire ← d1 × d2

"E" : d1 = donnée ("Donnez la diagonale 1: ")
d2 = donnée ("Donnez la diagonale 2: ")
aire ← pi × d1 × d2 / 4

"C" : p1 = donnée ("Donnez le côté : ")
aire ← SQR (p1)

Sinon
aire ← 0

Fin selon
choix = donnée ("Précisez l'ancienneté du client : ")

b. Algorithme

0) Début surface
1) Ecrire ("Entrez votre choix : "), Lire (choix)
2) Selon **choix** faire
"D" : écrire ("Donnez le rayon : ", p1), lire (p1)
aire ← pi × SQR (p1)

"R" : écrire ("Donnez la largeur : ", d1), lire (d1)
écrire ("Donnez la longueur : ", d2), lire (d2)
aire ← d1 × d2

"E" : écrire ("Donnez diagonale 1 : ", d1), lire (d1)
écrire ("Donnez diagonale 2 : ", d2), lire (d2)
aire ← pi × d1 × d2 / 4

"C" : écrire ("Donnez le côté : ", p1), lire (p1)
aire ← SQR (p1)
Sinon
aire ← 0
Fin selon
3) Ecrire ("La surface est : ", aire)
4) Fin surface

T.D.O

Objet	Type	Rôle
choix	caractère	le sélecteur
Aire	Réel	la valeur à calculée
p1	Réel	Rayon / côté
d1	Réel	Longueur / diagonale 1
d2	Réel	Largeur / diagonale 1

c. Pascal (voir fichier : choixsur.pas)

II. Les structures de contrôle itératives

II.1 la structure itérative complète

a. Syntaxe (voir livre page 78)

Activité 4 :

- Application 5 page 83.

a. Analyse :

Résultat : Affichage

Traitements :

Si T [1] = T [2] – 1 alors écrire (T [1])
Fin si

Pour i de 2 à n-1 faire
    Si (T [i] = T [i-1] + 1) et (T [i] = T [i +1] - 1) alors écrire (T [i])
    Fin si
Fin pour

Si T [n] = T [n - 1] + 1 alors écrire (T [n])
Fin si

Données : Saisir n le nombre de case par l'utilisateur,
Prévoir une boucle pour afin de saisir le tableau T.

b. Algorithme

0) Début encadrement
1) Ecrire ("Donnez n : "), lire (n)
2) Pour i de 1 à n faire
    Ecrire ("Donnez la case : "), lire (T[i])
   Fin pour
3) Si T [1] = T [2] – 1 alors écrire (T [1])
   Fin si

   Pour i de 2 à n-1 faire
    Si (T [i] = T [i-1] + 1) et (T [i] = T [i +1] - 1) alors écrire (T [i])
    Fin si
   Fin pour

   Si T [n] = T [n - 1] + 1 alors écrire (T [n])
   Fin si
4) Fin encadrement

Type
Tab = tableau de 50 octet

T.D.O

Objet	Type	Rôle
n	Octet	Taille du tableau
i	Octet	Compteur
T	Tab	Tableau d'entiers

c. Pascal (voir fichier : encad.pas)

II.2 Les structures de contrôle itérative à condition d'arrêt

a. La structure répéter … jusqu'à

- *Définition* (voir livre page 85)
- *Syntaxe* (voir livre page 86)

Activité 5 :

- *Ecrire un programme intitulé voyelles permettant de déterminer et d'afficher la position de la 2ème voyelle dans un tableau T de n caractères alphabétiques données. S'il y'a moins que deux voyelles dans T le programme affiche* -1.

Style de l'exercice : Algorithme de recherche

- Analyse

Résultat : Affichage

Traitements :

- *On suppose que nv est une variable qui contient le nombre d'occurrence des voyelles dans T et i la position de la 2ème voyelle si elle existe.*

  Si nv = 2 alors écrire (i)

  Sinon écrire ("-1")

  Fin si
- Prévoir une boucle répéter qui parcours le tableau T réalisant le test suivant :

  Si T[i] ∈ {l'ensemble des voyelles} alors incrémenter nv

  La boucle s'arrête lorsque (nv =2) ou (i=n)

Données :

- Faire une saisie contrôlée sur n
- Prévoir une boucle pour qui saisit T (Contrôler les éléments de T)

- Algorithme

0) Début voyelle

1) écrire ("Donner la taille du tableau : "), lire (n)

2) Pour i de 1 à n faire
   Répéter
      Lire (T[i])
   Jusqu'à Majus (T[i]) dans ["A".."Z"]
   Fin pour

3) nv ← 0 {on suppose qu'il n'y a aucune voyelle}
   i ← 0
   Répéter
      i ← i + 1
      Si Majus(T[i]) dans ["A","E","I","O","U","Y"] alors nv ← nv +1
      Fin si
   Jusqu'à (nv = 2) OU (i = n)

4) Si nv = 2 alors écrire (i)
      Sinon écrire ("-1")
   Fin si

5) Fin voyelle

Type
Tab = tableau de 50 octet

T.D.O

Objet	Type	Rôle
n	Octet	Taille du tableau
i	Octet	Compteur
T	Tab	Tableau d'entiers
nv	Octet	Nombre de voyelle

- Pascal (voir fichier : voyelle2.pas)

b. La structure Tant que … Faire

- *Définition* (voir livre page 95)
- *Syntaxe* (voir livre page 95)

Activité 6 :

- *Ecrire un algorithme qui permet de saisir une suite d'entiers qui se termine par -1 puis de calculer et d'afficher la somme de ces valeurs sauf la dernière.*

- Analyse

Résultat : écrire ("la somme est :", cumul)

Traitements :

Cumul est une variable qui va contenir la somme des valeurs saisies. Cette saisie s'arrête lorsque l'utilisateur entre -1. *Il serait intéressant d'utiliser une boucle tant que car si l'utilisateur saisit -1 la première fois il n'y a plus de traitement.*

nombre = **Donnée** ("Donner un entier : ")

- Algorithme

0) Début cumul
1) écrire ("Donner un entier : "), lire (nombre)
2) cumul ← 0

Tant que (nombre ≠ -1) faire
Cumul ← cumul + nombre
Lire (nombre)
Fin tant que

3) écrire (la somme est :", cumul)
4) Fin cumul

T.D.O

Objet	Type	Rôle
nombre	entier	Nombre à cumuler
cumul	entier	la somme

- Pascal (voir fichier : cumul2.pas)