UEF 1 : INITIATION À L'ALGORITHMIQUE

Dr. Djilali IDOUGHI
Maître de Conférences de Classe A

http://www.idoughi.com
djilali@idoughi.com
http://works.bepress.com/djilali_idoughi/

PLAN DU COURS

Chapitre 1 : Introduction

1. Description d'un ordinateur
2. Instructions de base d'un ordinateur
3. Différentes phases de résolution d'un problème par ordinateur

Chapitre 2 : algorithme

1. Définition
2. Caractéristiques d'un algorithme
3. Définition d'une variable et ses caractéristiques
4. Primitives de base
5. Action d'affectation
6. Action conditionnelle
7. Action alternative
8. Actions de répétition
9. Boucle **tantque**
10. Boucle **repeter**
11. Boucle **pour**

Chapitre 3 : procédure et fonction

1. Définitions
2. Mode de passages de paramètres
3. Exemples

Chapitre 4 : structures de données de base

1. Tableau
2. Matrice
3. Type énuméré
4. Ensemble

PLAN – CHAPITRE 2

CHAPITRE 2 : ALGORITHMES

1. Définition
2. Caractéristiques d'un algorithme
3. Définition d'une variable et ses caractéristiques
4. Primitives de base
5. Action d'affectation
6. Action conditionnelle
7. Action alternative
8. Actions de répétition
9. Boucle **tantque**
10. Boucle **repeter**
11. Boucle **pour**

I. ALGORITHME

Définition : Encyclopédie Universalis

Un algorithme est une suite finie de règles à appliquer dans un ordre déterminé à un nombre fini de données pour arriver, en un nombre fini d'étapes, à un certain résultat, et cela indépendamment des données

I. ALGORITHME

Définition :

- *C'est un pseudo-langage qui est conçu pour résoudre les problèmes et applications sans aucune contrainte due aux langages de programmation et aux spécificités de la machine.*
- *Ce pseudo-langage sera ensuite traduit et codé dans le langage de programmation désiré.*

I. LANGAGE ASSEMBLEUR

Définition :

Le langage Assembleur est un langage qui utilise des instructions sous forme symbolique (ADD, MOVE).

L'assembleur est lié au microprocesseur, puisque c'est le seul langage que le microprocesseur comprend.

I. LANGAGE DE PROGRAMMATION

Définition :

On appelle langage de programmation tout ensemble fini de mots réservés qui permettent de traduire les instructions de l'algorithme afin de l'exécuter par l'ordinateur.

Exemple :

Turbo Pascal, Cobol, Fortran, C, Delphi, Visual Basic (VB), C++, Java etc...

I. PROGRAMME SOURCE

Définition :

- Le programme source est le premier résultat de la traduction d'un algorithme en un langage évolué :
- *Un nouvel ensemble d'instructions non exécutables directement par la machine*

I. COMPILATEUR

Définition :

- On appelle compilateur tout programme spécial qui permet d'avoir un programme exécutable à partir d'un programme source :
- *Le programme ainsi obtenu est appelé programme Objet*

I. STRUCTURE GENERALE D'UN ALGORITHME

1. Titre du Problème

2. Déclaration des Objets
   - ✓ Déclaration des Constantes
   - ✓ Déclaration des Variables
   - ✓ Déclaration des Tableaux
   - ✓ Déclaration des Procédures et Fonctions

3. Traitement & Manipulation des objets
   - Début
   - *Actions*
   - Fin

I. STRUCTURE GENERALE D'UN ALGORITHME

Démarche à suivre pour résoudre un problème donné:

1. Identifier les données du départ (entrées) et celle(s) qu'il faut obtenir (sorties);
2. Structurer les données (variables ou constantes, type...);
3. Réfléchir pour déterminer les action nécessaires à l'obtention des résultats ;
4. Présenter les résultats.

II. DECLARATION DES OBJETS

II. DECLARATION DES OBJETS

Définition :

- Un objet est toute partie identifiable de l'information au cours d'un traitement.
- Il est caractérisé par son **nom**, son **type** et sa **valeur**.
- L'ensemble des objets manipulés par un algorithme est appelé: *environnement de cet algorithme*.

Les objets manipulés par un ordinateur sont :

*Les **Constantes** et*

*Les **Variables***

II. 1- LES CONSTANTES

Définition :

Les Constantes désignent des références à des valeurs invariantes dans le programme

Syntaxe de la déclaration :

```
Constante      Nom_Constante = Valeur
```

Exemple :

```
Constante    Pi = 3.14
```

II. 2- LES VARIABLES

Définition :

Ce sont des références (adresses mémoires ou EMPLACEMENTS MEMOIRES) où vont être stockées des valeurs variables.

Les différentes valeurs d'une référence vont appartenir au type de données auquel appartient la référence.

Important

1°- Le nom d'une variable est constituée d'une suite de caractères ALPHANUMERIQUE qui permet d'identifier la variable d'une manière unique dans un algorithme.

2- Il existe différents types de variables.

II. 2.1- LES VARIABLES : Type Entier

Définition :

- *Une variable de type Entier est un emplacement mémoire pouvant contenir des valeurs ou données de type entier;*
- *C'est l'ensemble des nombres entiers positifs ou négatifs.*

Syntaxe de la déclaration :

```
Variable      variable1,variable2,...  : Entier
```

Exemple :

```
Variable      a,b  : Entier
```

a et b sont, par exemple, les coefficients de l'équation : $ax + b = 0$

II. 2.2- LES VARIABLES : Type Réel

Définition :

C'est l'ensemble des nombres réels, c'est à dire les nombres décimaux sans limitation.

Syntaxe de la déclaration :

```
Variable      variable1,variable2,...  : Réel
```

Exemple :

```
Variable      x,y  : Réel
```

II. 2.3- LES VARIABLES : Type Chaîne de caractères

Définition :

C'est une suite de caractères, c'est à dire des combinaisons de caractères (lettres, chiffres, symboles..).

Syntaxe de la déclaration :

```
Variable      variable1,variable2,...  : Caractère
```

Exemple :

```
Variable      Nom, Catégorie  : Caractère
```

II. 2.4- LES VARIABLES : Type Booléen

Définition :

Il s'agit des objets qui ne peuvent prendre que deux valeurs vrai ou faux.

Syntaxe de la déclaration :

```
Variable      variable1,variable2,...  : Booléen
```

Exemple :

```
Variable      Décision  : Booléen
```

II. 2.5- LES VARIABLES : Type Tableau

Définition

Un tableau permet de représenter un ensemble de valeurs ayant des propriétés communes et appartenant toutes au même type.

Ces variables sont identifiées par un même nom mais un numéro de repère (indice) pour chacun.

II. 2.6- LES FONCTIONS ET LES PROCEDURES

Définition

Ce sont des sous-programmes auxquels on peut faire référence à l'intérieur d'un programme . Ils sont conçus pour éviter les répétitions et pour découper des programmes jugés trop longs; ce qui facilite la lisibilité du programme principal.

III. MANIPULATION DES OBJETS

III. 1. INSTRUCTION ET ACTION

Définitions :

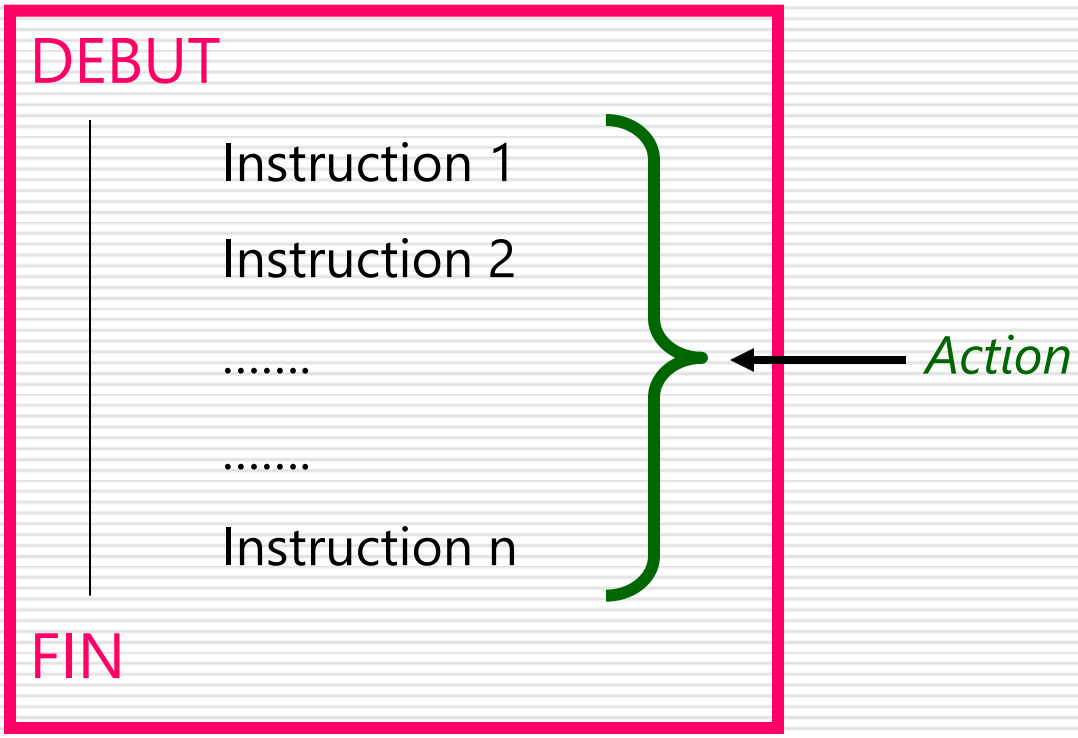
- *On appelle instruction toute commande élémentaire que l'on doit appliquer sur des objets pour avoir des sorties bien définies.*

- *Une action est un événement qui change l'état d'un objet d'un état initial donné à un état final désiré.*
- *Une action a une durée d'exécution finie et un effet propre et bien défini.*
- *Chaque action porte sur des objets sur lesquels elle s'exécute :*

L'Action peut être une seule instruction ou un groupe d'instructions

III. 2. LA STRUCTURE DE LA PARTIE MANIPULATION

La partie manipulation doit commencer par le mot **DEBUT** et se termine par le mot **FIN** :



III. 3. LES INSTRUCTIONS D'UN ALGOROTHME

La partie manipulation utilise les différents objets déclarés dans la partie déclaration et leur applique des opérations afin de retourner le(s) résultat(s) attendu(s) par le programmeur.

Pour ce fait, il y a différentes actions, dites instructions, à savoir :

1. Instructions de dialogue Homme-Machine ;
2. Instructions d'affectation ;
3. Instructions à structure alternative ;
4. Instructions à structure répétitive.
5. Etc…

III. 3.1- LES INSTRUCTIONS DE DIALOGUE HOMME -MACHINE

L'affichage des informations:

Pour faire comprendre qu'il faut afficher des informations à l'écran, on utilise l'instruction **écrire** qui obéit à la syntaxe suivante :

Écrire (Variable ou ' Message')

Exemples :

Écrire (' Saisissez la valeur de a ')

Écrire (' Saisissez la valeur de b ')

Écrire (' Saisissez les valeurs de a et b ')

Écrire ('Le résultat trouvé est :', r)

Écrire (r)

III. 3.1- LES INSTRUCTIONS DE DIALOGUE HOMME -MACHINE

La Saisie des informations:

Pour indiquer dans un algorithme que telle donnée doit être lue par le système, on utilise l'instruction **lire** qui obéit à la syntaxe suivante :

Lire (Variables)

Exemple :

```
Écrire (' Saisissez la valeur de a ')
Lire(a)
```

III. 3.2- INSTRUCTIONS D'AFFECTATION

Définition:

- C'est le stockage d'une valeur à un endroit spécifique(variable).
- Pour affecter une valeur à une variable, on écrit :

Exemple :

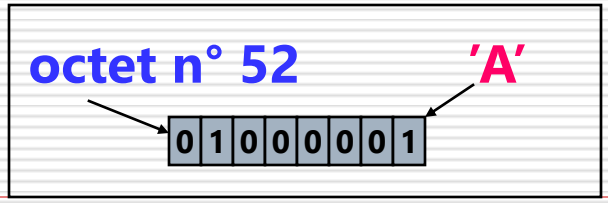
Variable ← valeur 1 + valeur 2

Variable ← valeur 1 * valeur 2

Variable ← valeur + Variable1

III. REMARQUES SUR LES CONSTANTES ET LES VARIABLES

- Les variables sont des références (adresses mémoires) où vont être stockées des valeurs qui peuvent changer au cours de l'exécution du programme.
- Les mémoires sont repérées par des numéros (pour l'ordinateur) ou des noms (pour le programmeur, qui a intérêt à choisir des noms significatifs).
- Chaque fois qu'on procède à une nouvelle affectation, l'ancien contenu de la mémoire est perdu et un nouveau contenu est placé dans la mémoire.
- Les constantes correspondent à des zones mémoires dont le contenu ne peut pas varier.



III. 3.3- EXERCICE1

Quels résultats produit l'algorithme suivant ? Les types de variables sont-ils corrects

Titre : Calcul

Déclaration

Variable A: Entier
C,B : Réel
D : caractère
E : Booléen

Manipulation

Début

A ← 30
B ← A * 2
Écrire('B=' , B)
C ← (B + A)/4
B ← C / 5
D ← 'Amine'
E ← (A > 40) Ou (C < B)
Écrire('les valeurs obtenues sont : A = ' , A ,
'B = ' ,B , ' C = ', C , ' D = ', D, ' E = ', E)

Fin

III. ETATS DE LA MÉMOIRE LORS DE L'EXECUTION DU PROGRAMME

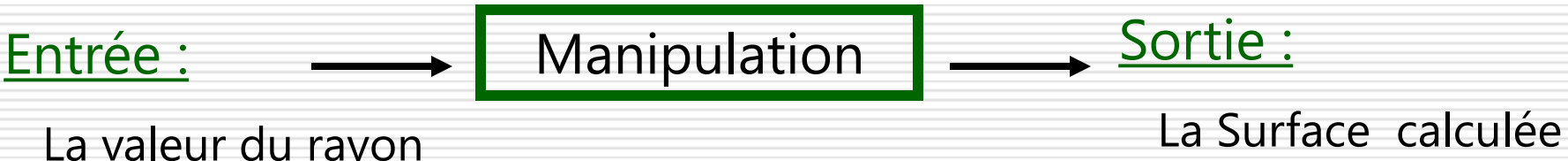
Instruction	*A*	*B*	*C*	*D*	*E*
Après l'instruction *A ← 30*	*30*	—	—	—	—
Après l'instruction *B ← A * 2*	*30*	*60*	—	—	—
Après l'instruction *C ← (B + A)/4*	*30*	*60*	*22.5*	—	—
Après l'instruction *B ← C / 5*	*30*	*4.5*	*22.5*	—	—
Après l'instruction *D ← 'Amine'*	*30*	*4.5*	*22.5*	*Amine*	—
Après l'instruction *E ← (A > 40) Ou (C< B)*	*30*	*4.5*	*22.5*	*Amine*	*Faux*

III. 3.3- EXERCICE 2

Énoncé:

Écrire l'algorithme permettant le calcul de la surface d'un cercle connaissant son rayon :

Surface = π * Rayon^2



III. 3.3- SOLUTION

Titre : Surface d'un cercle
Déclaration :
    Constante Pi=3.14
    Variable Rayon : Entier /* Donnée d'entrée*/
    Variable Surface : Réel /* Donnée de sortie*/

Manipulation :
DEBUT
    Écrire ('Saisir la valeur du rayon')
    Lire (Rayon)
    Surface ⟵ Rayon * Rayon * Pi
    Écrire (' La Surface du cercle est : ', Surface)
FIN

III. 3.3- EXERCICE 3

Exemple :

Écrire l'algorithme qui permet de déterminer le salaire mensuel d'un commercial sachant que ce salaire comporte un montant fixe de 4000 DA et une commission qui représente 10% du chiffre d'affaire réalisé par mois.

III. 3.3- CE QU'IL FAUT FAIRE

- Analyse du problème

- Recenser les données dont on dispose, celles qu'on cherche à produire

- Choisir les actions à utiliser pour résoudre le problème

- Présentation de l'algorithme

* Déclarer toutes les données utilisées (variables, constantes, types)

* Organiser les actions

* Présenter les résultats

III. 3.3- ALGORITHME

Titre : Commission

Déclaration :

```
Constante  M = 4000
Variable   CA : Entier     /* Chiffre d'Affaires,  Donnée d'entrée*/
           Com : Réel      / * Commission, Donnée intermédiaire*/
           Sal : Réel      / * Salaire, Donnée de sortie*/
```

Manipulation :

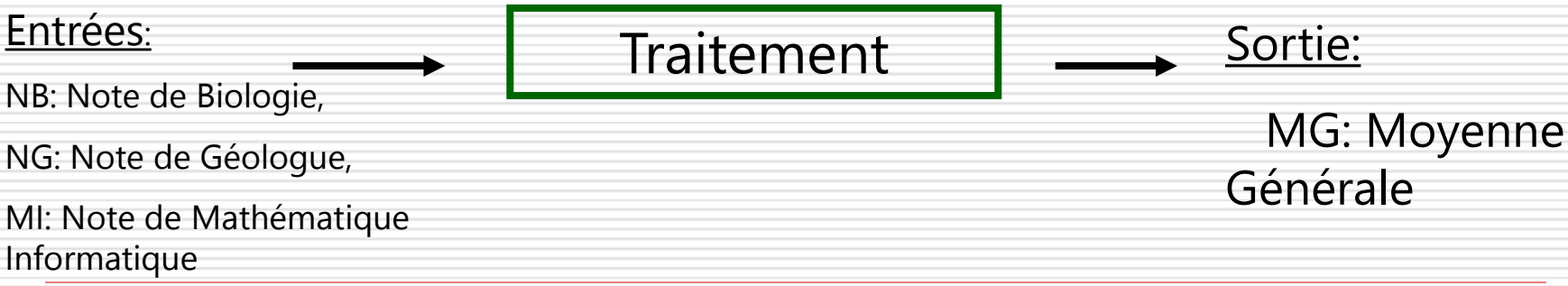
```
DEBUT
    Écrire ('Donner le CA mensuel en DA')
    Lire (CA)
    Com ⟵ CA * 10/100
    Sal ⟵ Com + M
    Écrire ('Le salaire mensuel est de : ', Sal, ' en DA ' )
FIN
```

III. 3.3- PROBLEME

Écrire un algorithme qui calcule la moyenne générale d'un étudiant sachant que celle-ci se calcule de la manière suivante :

Moyenne = [3 * Note(Biologie) + 2* Note(Géologie) + Note(NMI)] / 6

Indication :



III. 3.4- INSTRUCTIONS À STRUCTURE ALTERNATIVE

Les conditions :

*On appelle **condition simple** toute expression de la forme :*

Variable1 OPÉRATEUR Variable2

III. 3.5- LES OPERATEURS ARITHMETIQUES

Opérateur	Signification
+	Addition
-	Soustraction
*	Multiplication
/	Division
% ou mod	Modulo : le reste de la division de 2 valeurs entières
div	Division entière

III. 3.6- LES OPERATEURS DE COMPARAISON

Pour exprimer les conditions, on utilise les opérateurs conditionnels suivants :

Opérateur	Signification
=	Égal
<	Inférieur
>	Supérieur
<=	Inférieur ou égal
>=	Supérieur ou égal
<>	différent

III. 3.6- LES OPERATEURS LOGIQUES DE RELATION

On peut combiner des conditions à l'aide des opérateurs logiques :

Opérateurs	Signification
ET	Et logique
OU	Ou logique
NON	Négation logique
OU = XOR	Ou exclusif

III. 3.7- LA TABLE DE VERITE

A	B	A et B	A ou B	Non A
VRAI	VRAI	VRAI	VRAI	FAUX
VRAI	FAUX	FAUX	VRAI	FAUX
FAUX	VRAI	FAUX	VRAI	VRAI
FAUX	FAUX	FAUX	FAUX	VRAI

III. 3.8- EXEMPLE

Expression	Résultat
(4 <7) ET (9>0)	*Vrai*
(1 < 0) OU (1<>1)	*Faux*
Non(13.4 < 15)	*Faux*

III. 3.9- Priorités des opérateurs

1. Priorité de *, / div et % par rapport à + et -

   5 + 9 * 3 = 32 et non 42

   5*9+3 = 48 et non 60

2. Pour les opérateurs de même priorité, associativité à partir de la gauche

   15 / 5 * 3 = 9 et non 1

   5 – 2 + 4 = 7 et non –1

3. On peut utiliser des parenthèses pour changer l'ordre des opérations :

   15 / (5 * 3) = 1

   (5 + 9) * 3 = 42

III. 3.10- Opérateurs et variables

1. Opérateurs sur des variables de type entier

   *, / , % , div , + et -

2. Opérateurs sur des variables de type Réel

   *, / , + et -

3. Opérateurs sur des variables de type caractère ou chaîne de caractères

   +, CONCAT(énation) , Suc(cesseur), Pred(ésseur)

4. Opérateurs sur des variables de type Booléen

   Les opérateurs logiques de relation : ET, OU, NON

III. 3.10- L'Instruction Si

L'instruction Si :

Pour exprimer le fait que des instructions vont être exécutées dans un cas alors que d'autres instructions peuvent être exécutées dans l'autre cas, on utilise une ***structure alternative***.

Syntaxe :

```
Si    <condition>    alors
        [ Actions 1]
[ Sinon
        [Actions 2]        ← Option Facultative
Finsi
```

III. 3.10- L'Instruction Si - Exemples

```
Titre : Test 1
Variable x : entier
Début
   Écrire ('Saisir un entier x ')
   Lire (x)
   Si (x > 0) alors
    Écrire('x est un nombre positif ')
   Finsi
Fin
```

```
Titre : Test 2
Variable x : entier
Début
   Écrire ('Saisir un entier x ')
   Lire (x)
   Si (x > 0) alors
      Écrire (' x est un nombre positif ')
    Sinon
      Écrire (' x est un nombre négatif ou nul')
   Finsi
Fin
```

III. 3.10- L'Instruction Si - Exemples

Dans l'exercice où on a calculé la moyenne générale, Afficher '**Admis**' si un étudiant a une moyenne générale >=10 et Afficher '**Ajourné**' dans le cas contraire (Moyenne générale < 10).

Solution :

```
……(Reste de l'algorithme)
    Si (MG >= 10)  alors
        Écrire ('Admis')
    Sinon
        Écrire ('Ajourné')
    Finsi
FIN
```

III. 3.10- L'Instruction Si - Exercices

Écrire l'algorithme qui permet de calculer le maximum de deux entiers quelconques.

III. 3.10- L'Instruction Si - Solution

```
Algorithme : Maximum
Variable a ,b, max : entier
Début
   Écrire ('Saisir deux entiers a et b ')
   Lire(a, b)
   Si (a > b) alors
            max ⟵ a
   Sinon
             max ⟵ b
   Finsi
   Écrire ('le maximum de ' , a , ' et  de ' , b, ' est : '  , max)
 Fin
```

III. 3.10- L'Instruction Si - Exercices

Écrire l'algorithme qui permet de

1. déterminer le **Salaire Mensuel** d'un commercial sachant que ce salaire comporte
2. un **Montant Fixe** de **4000,00 DA** et
3. une **Commission** qui représente
   - **5%** du chiffre d'affaire réalisé par mois si ce chiffre est < 30.000,00 DA et de
   - **10 %** dans le cas contraire .

III. 3.10- L'Instruction Si - Solution

… Suite de l'algorithme

```
        Si  (CA < 30.000,00) alors
                Com ←—— CA * 0.05
        Sinon
                Com ←—— CA * 0.1
        Finsi
                Sal ←—— Com + M
        Écrire ('Le salaire mensuel est de : ', Sal , 'DA')
FIN
```

III. 3.10- L'Instruction Si - Exercices

Écrire l'algorithme qui permet de déterminer le salaire mensuel.

Sachant que la commission est calculée de la manière suivante :

- *Commission = 15% du CA quand CA > 100.000,00 DA*
- *Commission = 10% du CA quand 30.000,00 DA < CA < = 100.000,00 DA*
- *Dans le cas contraire pas de commission*

III. 3.10- L'Instruction Si - Solution

```
... Suite de l'algorithme
        Si  (CA > 100.000,00) alors
                Com ←—— CA * 0.15
        Sinon /* CA <= 100.000,00 */
            Si (CA > 30.000,00) alors
                Com ←—— CA * 0.1
            Sinon /* CA <= 30.000,00*/
                Com ←—— 0
            Finsi
        Finsi
                Sal ←—— Com + M
        Écrire ('Le salaire mensuel est de : ', Sal , 'DHS')
FIN
```

III. 3.11- L'Instruction Suivant Cas:

L'instruction ***Suivant cas*** constitue une structure alternative à la forme en bloc [Si … Alors …Sinon…] et permet de formuler de manière plus simple le choix du groupe d'instructions.

III. 3.11- L'Instruction Suivant Cas:

Syntaxe :

```
Suivant Cas  variable  Faire
  Cas  Valeur 1
          Actions 1
  Cas  Valeur 2, Valeur3, Valeur 4
          Actions 2
  Cas  Valeur 5 à Valeur 7
          Actions 3
    ..
    ..
  Sinon Cas
          Actions N
 Fin Suivant
```

III. 3.11- L'Instruction Suivant Cas: Exercices

Écrire l'algorithme qui permet de déterminer le nombre de jours d'un mois d'une année donnée

III. 3.11- L'Instruction Suivant Cas: Solution

```
....Reste de l'Algorithme
   Suivant Cas      Mois    Faire
         Cas  2
                  Action 1
         Cas  1, 3, 5, 7, 8, 10 , 12
                  Action 2
         Cas  4, 6, 9, 11
                  Action 3
         Sinon Cas
                  Écrire ( 'Attention : Mois Incorrect ')
          Fin Suivant
FIN
```

III. 3.11- L'Instruction Suivant Cas: Exercices

Un club de sport désire automatiser sa gestion. Les tarifs annuels d'inscription sont définis ainsi :

- De 0 à 3 ans ne sont pas autorisés à s'inscrire
- De 3 à 6 ans : gratuit
- De 6 à 12 ans: 1000 DA
- De 12 à 26 ans : 1500 DA
- Plus de 26 ans : 2000 DA

III. 3.11- L'Instruction Suivant Cas: Solution

....Reste de l'Algorithme

```
Suivant Cas      AGE    Faire
        Cas  1, 2
                    Écrire ('Vous n'êtes pas autorisé à vous inscrire ')
        Cas  3..5
                     TARIF   ←——    0
        Cas  6..11
                    TARIF   ←——    1000
         Cas  12..25
                    TARIF   ←——    1500
        Sinon Cas
                    TARIF   ←——    2000
    Fin Suivant
        Écrire (' le montant à payer en DA est : '  , TARIF )
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Problème :

Écrire un algorithme permettant d'afficher 300 fois le message : *" bonjour tout le monde"*.

Solution Classique :

```
DEBUT
    Écrire (' Bonjour tout le monde ')          1
    Écrire (' Bonjour tout le monde ')          2
        .
        .
    Écrire (' Bonjour tout le monde ')          300
FIN
```

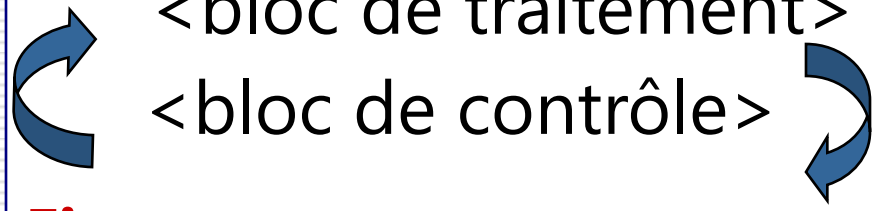
III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

L'instruction **Tant que** :

On utilise cette instruction pour exécuter des actions tant qu'une certaine condition reste vraie.

Syntaxe :

<initialisation>
Tant que <Condition> **Faire**



Action

Fin tant que

Solution de l'exercice précédent :

```
Variable      i : Entier
DEBUT
        i ⟵ 0                    * Initialisation *
      Tant que  (i < 300)  Faire
            Écrire (' Bonjour tout le monde ')
            i ← i + 1
      Fin tant que
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice : Afficher tous les multiples de 9 inférieurs à 485

```
Algorithme Multiples_de_9
Variable   M, i : Entier        *i: variable intermédiaire: compteur*
DEBUT
        i  ←——   0              * initialisation de la boucle*
        M ←——   0
        Tant que   (M < 485)   Faire
           Écrire (M , ' est un multiple de 9 ' )
           i ←  i + 1           * incrémentation du compteur *
           M ←—— i * 9
        Fin tant que
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exemple 1: Dans cet algorithme combien de fois la boucle est- elle exécutée ?

```
Algorithme Boucle1
Variable  i , y : Entier
Début
    i ← 2
    y ← 0
    Tant Que (i<7)  faire
            i ← i+1
            y ← y+i
          Écrire (' y = ' , y)
    Fin Tant que
Fin
```

i = variable intermédiaire
= compteur
i ← i+1 = incrémentation du compteur

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exemple 2: Dans cet algorithme combien de fois la boucle est- elle exécutée ?

```
Algorithme Boucle2
Variable  N: Entier
Début
    N← 15
    Tant Que (N<>0)  faire
          Écrire (N)
           N← N-2
    Fin Tant que
Fin
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

L'instruction **Répéter ,,,, jusqu'à :**

On utilise cette instruction pour exécuter des actions jusqu'à ce qu'une certaine condition devienne fausse.

Syntaxe :

<Initialisation>

Répéter

<bloc de traitement>
<bloc de progression >

} Action

Jusqu'à Condition

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exemple

```
Algorithme Boucle2
Variable  i , y : Entier
Début
    i ← 2
    y ← 0
    Répéter
          i ← i+1
          y ← y+i
          Écrire (' y = ' , y)
    Jusqu'à (i >=7)
Fin
```

Valeurs de y ?

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exemple

```
Titre : Boucle3
Variable  i , y : Entier
Début
    i ← 2
    y ← 0
    Répéter
          i ← i+1
          y ← y+i
        Écrire (' y = ' , y)
    Jusqu'à (i = 7)
Fin
```

Donner les valeurs de y

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice :

Écrire un algorithme permettant de calculer, pour un entier N> 0, la somme :

$$S_N = 1 + 2 + 3 + \ldots\ldots + N$$

Solution :

```
    Variable      N,S, i : Entier
DEBUT
        Écrire (' Saisir une valeur entière positive :')
        Lire (N)
            S  ⟵  0     * initialisation de la boucle*
            i  ⟵  0
        Répéter
                  i ⟵  i + 1
                  S ⟵  S + i
        Jusqu'à  (i > =N)
          Écrire (' La somme :  S = ', S)
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice :

Écrire un algorithme permettant de calculer la somme :

$$S_n = 1 + 1/2 + 1/3 + \ldots\ldots + 1/n$$

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

L'instruction **Pour** :

La spécification de cette instruction c'est qu'elle limite le nombre de fois où doit se répéter le bloc Action

Syntaxe :

<Initialisation>

Pour variable **allant de** valeur initiale **à** valeur finale **pas** valeur_pas **faire**

<Bloc de traitement> } Action

Fin Pour

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice :

Écrire un algorithme permettant le calcul du factoriel d'un entier $N > 0$ donné : $N !$

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Solution :

```
Algorithme Factoriel
Variable      N : Entier
              i  : Entier                    * i variable intermédiaire
              F : Entier                       i = compteur *
DEBUT
        Écrire (' Saisir une valeur entière N > 0: ')
        Lire (N )
           F ← 1                             * initialisation de la boucle*
           i ← 0
        Pour  i  allant de 1  à  N  Faire

                    F ← F*i
        Fin Pour

        Écrire ('  Le factoriel de ',  N , ' est : ' , F)
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice :

Écrire l'algorithme permettant de calculer la moyenne des notes de N étudiants

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Solution :

```
Algorithme Moyenne
Variable      N, i : Entier
              note, S, Moy : Réel
DEBUT
    Écrire (' Saisir le nombre d'étudiants: ')
    Lire (N)
    S  ←———  0          * initialisation de la boucle*
    i  ←———  0
       Pour   i ← 1  à  N  faire
         Écrire (' Saisir la note  de l'Etudiant ' , i , '  : ')
           Lire (note)
           S  ←———  S  +  note
        Fin Pour
           Moy  ←———  S/N
         Écrire (' La moyenne  est : ', Moy)
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Exercice

Écrire un algorithme permettant d'afficher les lettres de l'alphabet.

Solution :

```
Algorithme  alphabet
Variable   lettre: Caractère
                              * parcourir les lettres de l'alphabet*
Début
Pour lettre ← ' a '  à  lettre = ' z '  Faire
Écrire(lettre)
FinPour
Fin
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Quelle répétition ou Boucle choisir ?

Si nombre d'itérations connu Alors

Boucle Pour

Sinon

Si itération exécutée au moins une fois Alors

Boucle Répéter …. jusqu'à

Sinon

Boucle Tant que faire

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Boucles « Tant que faire » et « Répéter ….. jusqu'à »

Remarques: Soient T une condition et R l'action. Alors il y a équivalence entre les boucles Tant que faire et Répéter jusqu'à. La syntaxe est la suivante:

- Tant Que T faire
  R
  FinTantQue

~

- Si T alors
  Répéter
  R
  Jusqu 'à non(T)
  FinSi

Et

- Répéter
  R
  jusqu 'à T

~

- R
  Tant Que non(T) faire
  R
  FinTantQue

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Boucle « Pour »

Exercice :

1- Écrire un algorithme permettant de déterminer le $N^{ème}$ terme d'une suite numérique connaissant son premier terme et ses coefficients a et b et tels que:

$$U_n = a * U_{n-1} + b \qquad \forall \quad 1 \leq n \leq N$$

2- Écrire un algorithme permettant de définir le rang N et le terme correspondant de la suite tels que $U_N > 1000$

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Solution :

1) Le nombre d'itérations est connu : Boucle Pour

```
Variable      N, i : Entier
Variable      a, b, S : Réel

DEBUT
    Écrire (' Saisir la valeur de N: ')
    Lire  (N)
    Écrire ('Saisir la valeur du premier terme et les coefs a et b:')
    Lire (S , a, b )
    Pour   i ← 1   à   N   faire
           S  ←———   a * S  +  b
     Fin Pour
      Écrire ( ' La somme de la série est : ', S )
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Solution :

2) Le nombre d'itérations inconnu : Boucle Répéter jusqu'à

```
Variable     N : Entier
Variable     a, b, S : Réel

DEBUT
    Écrire ('Saisir la valeur du premier terme et les coefs a et b:')
    Lire (S , a, b)
    N ←—— 0
      Répéter
            S  ←———   a * S  +  b
            N ←—— N + 1
       Jusqu'à  S > 1000
       Écrire  (' La somme de la série est : ', S)
      Écrire (' Le rang est  : ', N )
FIN
```

III. 3.4- INSTRUCTIONS À STRUCTURE RÉPÉTÉTIVE

Boucle « Pour »

Exercice :

Écrire un algorithme permettant de calculer la somme :

$$S_n(x) = 1/x + 2/x^2 + 3/x^3 + \ldots\ldots + n/x^n$$

LIENS

http://troumad.developpez.com/C/algorigrammes/
http://www.pise.info/algo/variables.htm
http://ptrau.free.fr/program/pascal.htm
http://cours.univ-msila.dz/saoudi/ens/algo/chap2.htm