Spécialité : L.M.D. Economie & Commerce.

Année : 2ième année(3ième semestre).

Module : INFORMATIQUE 2(Algorithmique 1).

CHAPITRE 2 : LE Langage de programmation PASCAL

I- Structure d'un programme écrit en langage pascal

La structure d'un programme PASCAL est définie selon la figure suivante:

ENTETE
رأس البرنامج

```
PROGRAM  NomDuProgramme;
```

Déclarations
جزء التصريحات

```
Uses wincrt ; { bibliothèque nécessaire pour read et write }
CONST
  identificateur = valeur ; { déclaration de(s) constante(s) }
TYPE
  nom_de_type = description ; { déclaration de(s) type(s) }

VAR
  identificateur1 : type1 ;
  identificateur2 : type2 ; { déclaration des variables }
```

Instructions
جزء التعليمات

```
BEGIN { début des instructions }

  instruction_1 ;
  instruction_2 ;
  _
  instruction_n ;

END. { fin du programme }
```

En première analyse, on trouve:

- le mot réservé **program** suivi d'un identificateur le nommant, suivi d'un point-virgule,
- les différentes déclarations dont il a besoin (types, constantes, variables, fonctions et procédures),
- le bloc d'instructions encadré par les mots réservés **begin** et **end**,
- et finalement un point. Tout ce qui suivra sera ignoré par le compilateur.
- Les instructions sont séparées par ' ;' ,

en général, le point-virgule (;) est utilisé pour séparer les instructions, tandis que la virgule (,) est utilisée pour séparer les listes d'identificateurs.

تستعمل النقطة الفاصلة للفصل بين التعليمات، أما الفاصلة فتستعمل لفصل عناصر قائمة معرفات.

- Le retour à la ligne et les tabulations sont facultatifs mais fortement recommandés.
- Ce qui est entre accolades { } constitue un commentaire.

Les commentaires :(تعليقات) sont des éléments du texte d'un programme qui sont ignorés par le compilateur. Ils servent à donner des informations sur le programme.

En Pascal, on peut écrire des commentaires de deux façons:
- commentaires encadrés par {et}
- commentaires encadrés par (* et *)

– تستعمل التعليقات من أجل توضيح البرنامج فقط، حيث يتم تجاهلها من قبل المترجم.

Premier petit programme

Prenons comme exemple un petit programme PASCAL qui va afficher sur l'écran la somme de deux nombres obtenus par une lecture au clavier:

ENTETE	**PROGRAM** Addition;
SECTION DECLARATIVE	**Uses wincrt ;** **VAR** Somme : INTEGER; Nombre1, Nombre2 : INTEGER;
BLOC D'INSTRUCTIONS	**BEGIN** Write ('Premier nombre ? : '); ReadLn (Nombre1); *{ Lecture 1er nombre }* Write ('Deuxième nombre ? : '); ReadLn (Nombre2); *{ Lecture 2ème nombre }* Somme := Nombre1 + Nombre2; WriteLn ('La somme vaut: ', Somme); **END.**

II- Expressions (العبارات) :

Il existe deux types d'expressions, les expressions de déclaration (عبارات التصريحات) et celles de traitement(عبارات الأفعال).

On trouve les expressions de déclaration dans la partie déclaration, elles permettent de préciser les données nécessaires pour résoudre le problème, ainsi que leur type.

Les expressions de traitement sont utilisées pour décrire les traitements exécutés sur les données pour obtenir les résultats du problème.

نجد عبارات التصريح في جزء التصريحات، و هي تسمح بتحديد المعطيات و كذالك أنواعها.

أما عبارات الأفعال فتوجد في جزء التعليمات، تستعمل لوصف المعالجات التي نجريها على المعطيات لإيجاد حل للمسألة المطروحة.

III- Les identificateurs(المعرفات)

Les identificateurs sont des noms symbolique qui sont utiliser pour nommer (identifier) des objets dans un programme.ces objets peut être des variables, des constantes, des procédures, et des fonctions (sous programmes).

En Pascal, les identificateurs doivent commencer par une lettre et peuvent être poursuivi par n'importe quelle combinaison de lettres (A-Z, a-z), chiffres (0-9) et blancs soulignés (_).

المعرفات هي الأسماء التي نطلقها على الأشياء في برنامج (اسم البرنامج، اسم الثابت، اسم متغير أو اسم نوع).هذه الأسماء عبارة عن مزيج من الحروف و الأرقام، كذلك يمكن استعمال الرمز _ شرط أن يبدأ الإسم بحرف من أحرف اللغة الفرنسية

Remarque :

- Vous pouvez utiliser n'importe quel mot (**non réservé غير محجوز**)qui ne contient ni espace, ni accents, ni apostrophes, et un chiffre ne peut pas être placé en début.
- Le langage Pascal ne fait aucune distinction entre lettre minuscule et lettre majuscule.
- Certains identificateurs sont des mots réservés du langage, et ne peuvent donc pas être utilisés pour désigner des variables ou autres entités.

Exemple d'identificateurs corrects et incorrect:

- **Identificateurs corrects:**

Amine BlackBird RS232 Au_Debut Count

- **Identificateurs incorrects**

Identificateurs incorrects	Pourquoi?
Au-Dela	Le signe (-) est un opérateur arithmétique, il désigne, dans Pascal, l'opération de soustraction
7eme	Un identificateur doit commencer par une lettre ou un blanc souligné (_)
SYS**$**READ	Les caractère spéciaux ne sont pas permis
Pas a pas	Un identificateur ne doit pas contenir un espace
Program	Le mot Program peut apparaitre uniquement dans l'entête du programme

Les mots réservés du langage PASCAL الكلمات المحجوزة

and	**array**	**begin**	**case**	**const**	**div**
do	**downto**	**else**	**end**	**exit**	**false**
file	**for**	**function**	**if of**	**implementation**	**in**
interface	**mod**	**new**	**not to**	**or**	**procedure**
program	**record**	**repeat**	**then**	**true**	**type**
until	**uses**	**var**	**while**	**With**	

IV- Constantes et variables

a) Constantes : الثوابت

Les constantes sont déclarées dans la partie des déclarations d'un programme, après le mot réservé **const.**

Syntaxe :(قاعدة التصريح)

```
Const
      NomConstante = ValeurConstante;
```

On utilise le symbole **=** dans les déclarations de constantes.

Il est possible de déclarer plusieurs constantes,
Exemple:

```
Const
      Pi=3.14 ;
      LARG = 640;
      HAUT = 480;
      NB_PIX = LARG*HAUT;
```

b) Variables :(المتغيرات)

Une variable peut être représentée par une case mémoire, qui contient la valeur d'une donnée. Chaque variable (case mémoire) possède un nom unique (identificateur) par lequel on peut accéder à son contenu.
Par exemple, on peut avoir en mémoire une variable X et une variable Y qui contiennent les valeurs 5 et 12 :

X | 5 | Y | 12 |

On dit : la variable X contient la valeur 5, et la variable Y contient la valeur 12.

Donc une variable sert à stocker l'information traitée par un programme.

Une variable est identifiée par :
- Un identificateur (اسم المتغير): nom qui sert à repérer la variable
- Un type (نوع المتغير): format des données qu'elle contient

- Une valeur(قيمة المتغير)

- **Il ne faut pas confondre la variable et son contenu. Une variable est une sorte d'un récipient, alors que le contenu d'une variable est une valeur numérique, alphanumérique ou booléenne, ou autre type.**
- **la valeur d'une variable peut varier au cours du programme. L'ancienne valeur est tout simplement écrasée et remplacée par la nouvelle.**

Une variable peut être :

- **Donnée** : variable nécessaire pour faire fonctionner un programme.
- **Résultat** : résultat d'un calcul.

- **Auxiliaire** : qui sert à stocker des informations temporaires, à faire des calculs ou à faire fonctionner des structures de contrôle

Déclaration des variables
La déclaration d'une variable indique deux choses à préciser:
- son identificateur (son nom) ;
- son type (pour connaître sa taille).

Les variables sont déclarées dans la partie de déclarations d'un programme, après le mot réservé **var.**

Syntaxe : (قاعدة التصريح)

```
var
     NomVariable : TypeVariable;
```

Il est possible de déclarer plusieurs variables

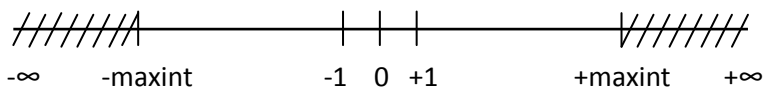
```
Var
     x : REAL;
     s , t : STRING;
```

Les types de variables

Les différents types de variables sont :

1. Le type entier :

   Ce type est désigné par l'identificateur prédéfini **integer.**
   Les valeurs de ce type forment un sous-ensemble de l'ensemble de nombres entiers, les bornes de ce sous ensemble dépendent de l'implémentation du compilateur.
   La valeur prédéfinie maxint définit la valeur absolue du plus grand nombre entier représentable.
   Le sous ensemble désigné par le type integer est donné par le schéma qui suit:



2. Le type réel :

Ce type est désigné par l'identificateur prédéfini **real.**
Même que le type integer, les valeurs de ce type forment un sous-ensemble de l'ensemble de nombres réels.

3. Le type booléen:

Désignation	Description	Bornes
boolean	Booléen (logique)	Deux valeurs possible: true (vrai) ou false (faux)

4. Caractères et chaines de caractères

Désignation	Description	Bornes
Char	nombre correspondant à un caractère ASCII codé	0 et 255
String	chaîne de caractères	peut contenir des chaînes d'au plus 255 caractères; le premier octet représente en effet la longueur de la chaîne.
String[n]	chaîne de caractère ne devant pas excéder n caractères	peut contenir des chaînes d'au plus n caractères.

Les variables de type caractère (Char): contiennent un caractère. Celui-ci doit être écrit entre deux quotes.

exemples

- Les caractères lettres: 'A', 'B',...., 'a','b',...
- Les caractères chiffres: '1','2',.....,'9'
- Les caractères spéciaux; '.', '=', ';', '%',...
- Le caractère quote:'''', la 1ère quote pour dire qu'il va y avoir un caractère, les 2 suivantes qui symbolisent la quote (car une seule quote voudrait dire fin du caractère), la dernière signifie fin du caractère.

Les valeurs de ce type sont en général ordonnées suivant l'ordre des codes internes des caractères sur l'installation considérée (code interne: ASCII,..).

Les variables de type chaine de caractère (string): contiennent une séquence de caractères(un mot ou une phrase) Celle-ci doit être écrit entre deux quotes.

Exemples: 'algorithme', 'Langage de programmation','and', 'software design',...

Une chaine de caractère doit être tapée entièrement sur une seule ligne.

Exemple:

```
'Cette chaîne de caractère est invalide parce qu'elle

     est tapée en deux ligne'
```

Nouveaux types

Il existe d'autres types prédéfinis et le programmeur peut aussi définir des types.les types définis par l'utilisateur sont appelés **types manufacturés.**

Nous pouvons classer ces types en deux classes :

1. Les types énumérés (الأنواع المعددة) : dans ces types, il faut citer toutes ses valeurs, dans l'ordre, et il ne faut pas déclarer une valeur dans deux types différents.

Syntaxe : (قاعدة التصريح)

Type NomType =(Val1,Val2,…,Valn) ;

Exemple :

```
TYPE
     couleurs_feux_t = (Rouge, Orange, Vert, Clignotant);

VAR
     feux : couleurs_feux_t;
```

Ecriture incorrecte : Type X=(samedi, **dimanche**, lundi) ;

Y=(**dimanche**, lundi, mardi) ;

2. Le type intervalle: (النوع مجال)

Les valeurs de ce type sont comprises entre une valeur inférieure et une valeur supérieure.

Syntaxe : (قاعدة التصريح)

Type NomType= Val_inf . . Val_sup ;

Exemple :

```
Type   xx = 20..45 ;
```

- le type real n'est pas ordinal, et donc on ne pas créer un type intervalle avec des réels, il n'y a pas de notion de réels consécutifs.
- Lorsqu'on crée un type T2 à partir d'un type T1, ce type T1 doit déjà exister ; donc T1 doit être déclaré avant T2.

Opérations sur les variables et constantes

Opérateurs arithmétiques:

Opèrent sur les variables numériques.

Multiplication	*
Division entière	div

Division réelle	/
Modulo (reste de la division entière)	mod
Addition	+
Soustraction	-

Niveaux de priorité (مستويات الأولوية): c'est l'ordre dans lequel sont appliquées les différentes opérations d'une expression.
Exemple : soit l'expression : x+y*z div d*3-x, il faut connaitre dans quel ordre les différentes opérations sont évaluées.

Il existe quatre niveaux de priorité :

Niveau	opération
1	Le signe du nombre(le - unitaire)
2	() les parenthèses
3	* / div
4	+ -

Remarque : les opérations du même niveau sont appliquées de gauche à droite

Exemples:

a) 10+(4*3) div 2 =

10 + 12 div 2

10 + 6 = 16

b) 14 mod (4 div 3)

14 mod 1 = 14

c) 11 mod 3 + 5.2 / 2 =

2 + 2.6 = 4.6

Autres exemples d'expressions numériques (soit les variables: A=3, B=4, C=2):

- A+B/C = A+(B/C) = 5
- (A+B)/C = 3.5
- A/B*C = (A/B)*C (1.5)
- A/BC = valeur de A sur valeur de la variable de nom BC et non A sur B*C
- B*A-5*C = (B*A)-(5*C) = 2

Opérateurs logiques

Opèrent sur les variables booléennes.

and	Et logique (الوصل)
or	Ou logique (الفصل)
not	Négation (النفي)

Niveaux de priorité (مستويات الأولوية) :

Niveau	opération
1	() les parenthèses
2	not
3	and
4	or

Table de vérité des opérateurs:

A	**B**	**Not A**	**A and B**	**A or B**
true	true	false	true	true
true	false	false	false	true
false	true	true	false	true
false	false	true	false	false

Exemple:

Not(12 < > (3*16.8/4))and True

Not(12 < > 12.6) and True

Not(True) and True

False and True

False

Opérateurs relationnels (عمليات المقارنة)

>	Supérieur
>=	Supérieur ou égal
<	Inférieur
<=	Inférieur ou égal
=	Egal
<>	Différent

- Le = et < > opèrent sur tous les types de variables précédemment définis, par contre les autres opérateurs ne peuvent pas s'appliquer sur des opérandes de type booléen.
- Le résultat d'une comparaison est un booléen.

Récapitulatif:

Voici la table des priorités classées par ordre décroissant, les opérateurs sur une même ligne ayant une priorité égale.

Niveau	opération
1	( ) fonction()

2	+ - not (unaire)
3	* / div mod and
4	+ - or
5	= <> < <= >= >

Exemple:

L'expression a < b or c <= d est mal formée, car écrire une telle expression booléenne sans parenthèses est une erreur.

En effet dans la table de priorités, l'opérateur or a une priorité plus élevée que les opérateurs < et <=, et donc l'expression sera évaluée:
a < (b or c) <= d , ce qui est faux.

L'expression bien formée est ici (a < b) or (c <= d) .

Les fonctions standards

Fonctions arithmétiques

Fonction	Signification	Type de l'argument	Type du résultat
ABS(x)	|X|	Entier ou Réel	De même type
SQR(x)	X^2	Entier ou Réel	De même type
SQRT(X)	$\sqrt{X}$	Entier ou Réel ≥0	Réel
SIN(X)	Sin(X)	Entier ou Réel	Réel
COS(X)	Cos(X)	Entier ou Réel	Réel
ARTAN(X)	Arctg(X)	Entier ou Réel	Réel
LN(X)	Log(X)	Entier ou Réel	Réel
Exp(X)	e^x	Entier ou Réel ≥0	Réel

Fonctions de conversion

Fonction	Signification	Type de l'argument	Type du résultat
Trunc(X)	Partie entière de X	Réel	Entier
Round(X)	Arrondi de X à l'entier le plus proche	Réel	Entier
ORD(X)	Rang de X	Entier, Caractère, ou Booléen	Entier
CHR(X)	Caractère ayant le rang X	Entier	Caractère

Fonctions d'ordre

Fonction	Signification	Type de l'argument	Type du résultat
PRED(X)	Valeur qui précède X	Entier, Caractère, ou Booléen	De même type

SUCC(X)	Valeur qui suit X	Entier, Caractère, ou Booléen	De même type
ODD(X)	Vrai si X est impaire Faux si X est pair	Entier	Booléen

V- Les instructions :

V-1) Instruction READ/READLN (lire)

Un programme a besoin de données sur lesquelles il opère. Si nous écrivons toutes les valeurs des données dans le programme lui-même, sous forme de constantes ou d'affectation, nous devons réécrire le programme chaque fois que nous voulions l'appliquer à un autre ensemble de valeurs différent. Pour éviter cela, les données doivent être séparé du programme jusqu'à ce qu'il soit exécuté. Ensuite les instructions du programme copient les valeurs des données dans les variable du programme. Après la mémorisation de ces valeurs, le programme peut effectuer des calculs avec eux.(voir la figure qui suit)

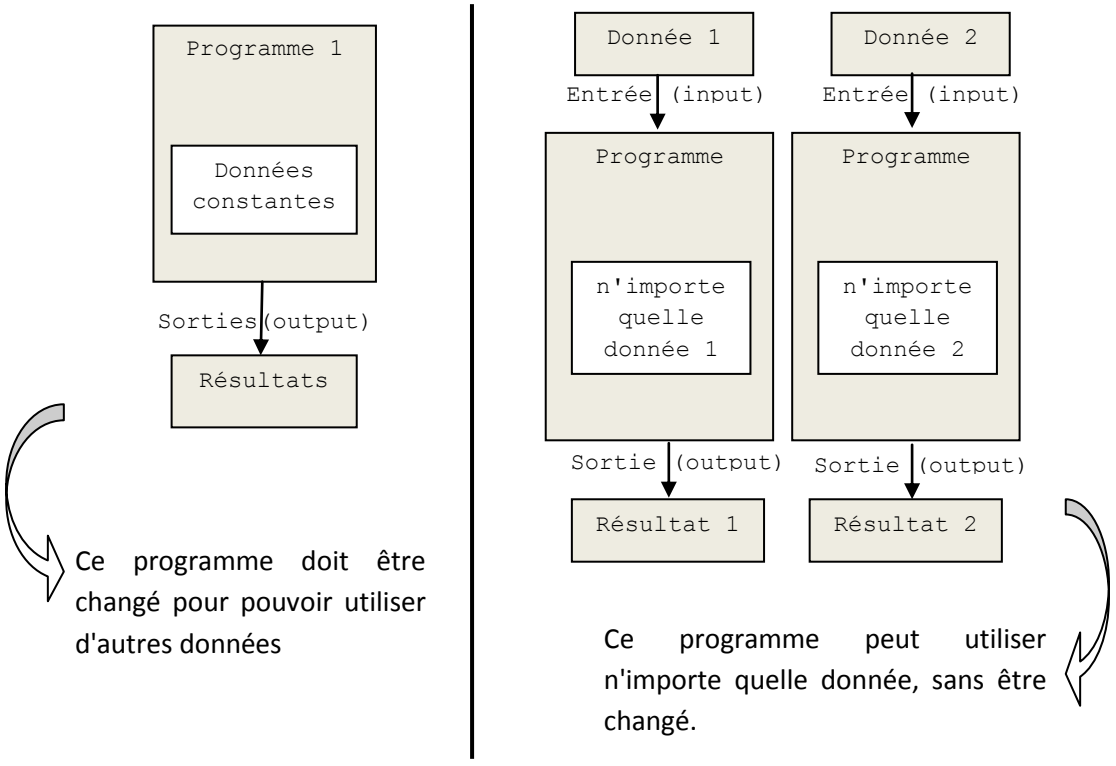


Figure1: Séparer les données du programme

Le processus de placement des valeurs d'un ensemble de données extérieures dans des variables d' un programme est appelée : les entrée (input).

Pour entrer des données dans un programme en cours d'exécution, le langage pascal nous offre les procédures READ et READLN.

1.READ

Syntaxe : (قاعدة الكتابة)

Algorithmique	Langage PASCAL
Lire(val1,val2,…,valn)	**Read(val1, val2,.., valn)**

Read : permet de lire n valeurs entrées par le clavier et les affectent aux variables val1, val2,..,valn.

- Les valeurs lues peuvent être du type : integer, real, char.
- **Il est impossible de lire une valeur booléenne.**
- **la liste des paramètres d'une instruction read(readln) ne peut contenir que des variables.**
- S'il ya plusieurs variables dans une liste de paramètres, vous devez les séparer par des virgules.

Exemple:

Instructions*	**Données**	**Contenus des variable après readln**
1.Read(x);	5	X=5
2.Read(a,b,c);	6 7 8	a=6, b=7, c=8
3.Read(d,e);	28	d=28, e=47
	47	
4.Read(y,z);	46 36.5 8	y=46, z=36.5

* a, b, c, x, y sont de type integer, et z de type real.

Si le nombre de valeurs entrées est supérieur à celui des paramètres dans l'instruction read, le système conserve le reste des valeurs pour un prochain read s'il existe, si non le système les néglige(voir 4 dans l'exemple).

S'il n'existe pas suffisamment de valeurs entrées sur une ligne pour remplir les paramètres de read, le système lit automatiquement à partir de ligne d'entrée suivante.(voir 3 dans l'exemple)

les deux écritures sont identiques:

```
READ(a,b);
  et
READ(a);
READ(b);
```

2.READLN

Le langage pascal nous offre une autre procédure pour entrer des données dans un programme, c'est l'instruction **READLN**

Exemple:

Supposons qu'il existe deux valeurs entières sur chaque ligne d'entrée:

10 20

15 16

22 21

Instructions*	**Contenus des variable après readln**
1.Readln(a); Readln(b); Readln(c);	A=10 B=15 C=22
2.Readln(a,b,c); Readln(d,e);	A=10, b=20, c=15 D=22, e=21
3.Readln(a,b); Readln(c,d); Readln(e);	A=10,b=20 C=15,d=16 E==22
4.Readln(a); Readln(b); Readln(c,d,e);	A=10 B=15 C=22, d=21, vous devez entrer une valeur pour la variable e.
5.Readln(a,b); Readln; Readln(c);	A=10, b=20, C=22, Readln sans paramètresprovoque un saut de ligne d'entrée.

* a,b,c,d,e sont des variable de type integer.

V-2) Instruction write (écrire)

Syntaxe : (قاعدة الكتابة)

Algorithmique	Langage PASCAL
ecrire(var1,var2,…,varn)	**write(var1, var2,.., varn)**

Write : permet d'afficher toutes les var1,var2,…,varn ,

La liste des paramètres d'une instruction write (writeln) peut contenir des constantes, des variables et des expressions,

L'instruction writeln(x) permet de d'afficher la valeur de x avec un retour à la ligne à la fin de l'affichage.

L'instruction Writeln sans paramètre permet de sauter une ligne.

V-3) Instruction d'affectation (تعليمة الإسناد)

Elle est représentée par le signe(←) dans le langage algorithmique et (:=) dans le langage PASCAL.
On appelle AFFECTATION la mise d'une valeur dans une variable. Celle-ci peut être sous forme directe (A := 2, ou A:=B) ou sous forme d'un calcul (A:=B*C).

1. Affectation par une valeur

L'affectation (variable := valeur) permet de changer la valeur d'une variable.
L'affectation modifie le contenu de la variable. La valeur de la variable à gauche du signe := est remplacée par la valeur à droite de :=.

Remarque
Après une affectation, l'ancienne valeur de la variable est écrasée par la nouvelle valeur.

2. Affectation par une expression

L'affectation (variable := expression)est effectuée par :
1. évaluation de l'expression
2. placement du résultat dans la variable à gauche.

Attention
A droite de :=, les variables sont utilisées pour désigner les valeurs qu'elles contiennent.

Exemple : x:= x + N a pour effet de mettre le résultat de la somme de la valeur de x avec la valeur de N dans le récipient x.

Le signe := signifie "mettre la VALEUR à droite du := dans la zone mémoire désignée à gauche" (mettre le contenu de B dans A ou mettre le résultat du calcul (contenu de B) fois (contenu de C) dans A).

Remarques :
- **Une affectation du type B*C:=A est IMPOSSIBLE.**
- **Ne pas confondre := avec le = réservé aux données constantes et aux symboles de comparaison.**

Une affectation ne peut se faire qu'entre une variable et une expression de même type (si A est réel, impossible de faire A:='bonjour').

Exemples

1)

```
X :=2 ;

Y :=3 ;

Z :=((x+y)*2) ;

X:=Z-2;
```

Après exécution : z prend la valeur : 10 et X prend la valeur: 8.

2)

Si x est un réel et y est un entier:

on peut écrire x:=y; mais on ne peut pas écrire y:=x;

V-4) Instruction conditionnelle (تعليمة الشرط)

On a deux types d'instructions conditionnelles :

- L'instruction conditionnelle simple : SI...alors...

  [Ne vouloir exécuter certaines instructions que dans des cas bien précis]

- L'instruction conditionnelles alternative (تعليمة الشرط المتناوب) : SI... alors...sinon...

  [vouloir exécuter différentes instructions selon le résultat du test]

1. L'instruction conditionnelle simple :

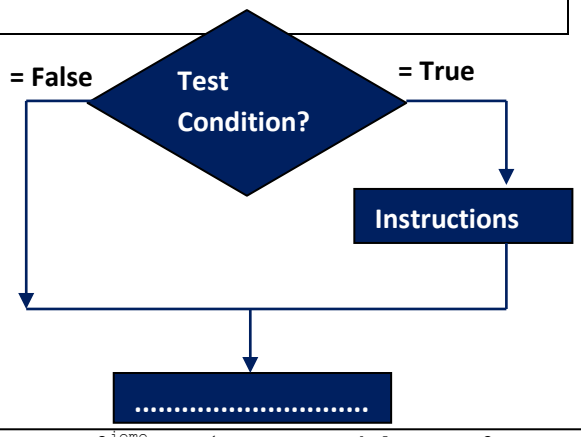
Syntaxe : (قاعدة الكتابة)

Algorithmique	Langage PASCAL

SI condition **ALORS**
Action(s)
FINSI
	IF condition **THEN** Instruction ;
	
	Ou bien
	
	IF condition **THEN**
	BEGIN
	Instructions ;
	END;

Exécution

- La condition est tout d'abord calculée,
- les instructions(ou les actions) ne sont exécutées que si la valeur de la condition est égale à True.



Exemple:

```
Program exemple;
Uses wincrt;
Const
  P=10;
Var
  A,b,c :integer ;
Begin
  Readln(a,b,c);
  If (a>b) and (a<c)
then
  Begin
    A:= b*2;
    B:= c-3*p;
    C:=(c+1)div 3;
  End;
End.
```

	1ier cas: Condition= false (2>14) and (2>-2)= false	
	Valeur avant exécution	Valeur après exécution
p	10	10
a	2	2
b	14	14
c	-2	-2

	2ième cas: Condition= true (12>4) and (12>8)=true	
	Valeur avant exécution	Valeur après exécution
p	10	10
a	12	8
b	4	-22
c	8	3

2. L'instruction conditionnelle alternative (تعليمة الشرط المتناوب) :

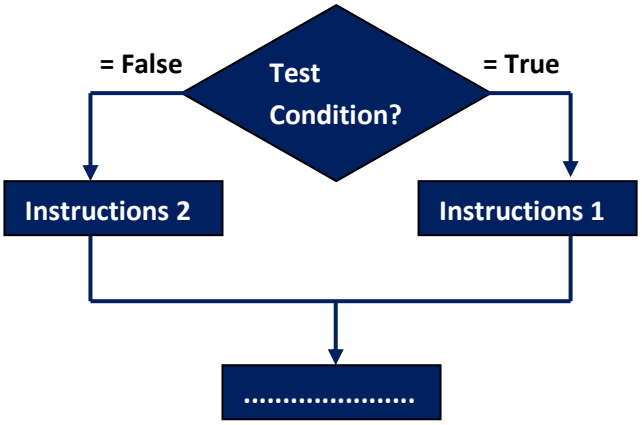
Syntaxe : (قاعدة الكتابة)

Algorithmiques	Langage PASCAL

SI condition **ALORS**
Action(s)1
SINON
Action(s)2 ;
FINSI
	IF condition **THEN**
	Instruction1
	Else
	Instruction2;
	
	Ou bien
	
	IF condition **THEN**
	BEGIN
	Instructions 1;
	END
	Else
	Begin
	Instructions2;
	End;

Exécution:

- La condition est tout d'abord calculée
- Si le résultat du calcul égale à true, le bloc d'instructions 1 est exécuté,
- Si le résultat du calcul égale à false, le bloc d'instructions 2 est exécuté.



Exemple:

```
Program exemple;
Uses wincrt;
Var
  A,b,c,d :integer ;
Begin
  Readln(a,b,c);
  If (a>b) or (c<d) then
  Begin
    A:= b*3;
    B:= (c-b)*2;
  End
  Else
  Begin
    B:=a*3 mod 2;
    C:=d div 2 - 2;
  End;
End.
```

	1ier cas: Condition= false (8>12) or (6<-2)= false	
	Valeur avant exécution	Valeur après exécution
A	8	8
B	12	0
C	6	-3
d	-2	-2

	2ième cas: Condition= vrai (10>8) or (9<4)=vrai	
	Valeur avant exécution	Valeur après exécution
A	10	24
B	8	2
C	9	9
d	4	4

Remarque:

On n'a le droit qu'à deux options selon le résultat du test (**un seul then et un seul else**); pour résoudre des problèmes plus complexes, on doit combiner(imbriquer) plusieurs instructions if- then- else.

Exemple:

```
program ifelse_ifelse;
var
a : integer;
begin
a := 100;
if (a = 10)
      then  writeln('Valeur de a= 10' )
      else if ( a = 20 ) then  writeln('Valeur de a= 20 )
                        else if( a = 30)then writeln('valeur de a= 30' )
                                        else
                                         begin
```

```
                                writeln('a est différente de10,de20,de30');
                                writeln('La valeur exacte de a=  ', a );
                                end;
```

end.

Après exécution du programme, le système affiche:

```
a est différente de10,de20,de30
La valeur exacte de a=  100
```

V-5) Instructions répétitives (تعليمات التكرار)

1. Instruction For - do (boucle For - do) :

La boucle for est la boucle la plus simple. Elle permet de répéter une séquence d'instructions un nombre fixé de fois.

Syntaxe : (قاعدة الكتابة)

Algorithmiques	Langage PASCAL

POUR(indice ← Val_inf à Val_sup)
FAIRE
 	
Action(s) ;
 	
FINPOUR
	FOR indice:= Val_inf **to** Val_sup **DO**
	Instruction;
	
	Ou bien
	
	FOR indice:= Val_inf **to** Val_sup **DO**
	Begin
	Bloc d'instructions;
	End;
	
	Ou bien
	
	FOR indice:= Val_sup **downto** Val_inf **DO**
	Instruction;
	
	Ou bien
	
	FOR indice:= Val_sup **downto** Val_inf **DO**
	Begin
	Bloc d'instructions;
	End;

Fonctionnement

L'instruction (ou le bloc d'instructions) est répétée pour toutes les valeurs de l'indice comprise (inclusivement) entre val_inf et val_sup. indice doit être d'un type ordonné, comme integer, et augmente d'une unité à chaque passage dans la boucle.

L'instruction sera donc exécutée (val_sup - val_inf + 1) fois. Cette forme de boucle est utilisée chaque fois que l'on connaît le nombre de fois à répéter les instructions.

Si on utilise un pas dégressif (allant de Val_sup jusqu'à Val_inf, on remplace TO par DOWNTO.

La variable indice peut être utilisée (mais pas modifiée) dans l'instruction ou le bloc d'instructions. Elle est souvent appelée "indice" de la boucle. Sa valeur est perdue dès que l'on sort de la boucle.

Exemple1:

```
Program exemple_1;
Uses wincrt;
Var
  puis,i, x :integer ;
Begin
     puis:=1; read(x);
     For i:=1 to 4 do
     puis:= puis*x;
     Writeln('x à la puissance 4 = ', puis);
End.
```

* Si x =3 par exemple, l'instruction puis := puis*x est exécutée (4-1+1 fois c.-à-d. 4 fois :

puis	i	Numéro d'itération
3	1	1
9	2	2
27	3	3
81	4	4

Le résultat est 3^4 =81.

Exemple2:

```
Program exemple_2;
Uses wincrt;
Var
  som,i, n :integer ;
Begin
     read(n); som:=0;
     For i:=1 to n do
     som:= som+i;
     Writeln('La somme des entier compris entre 1 et n est: ', som);
End.
```

* Si n =4 par exemple, l'instruction som := som+i est exécutée (4-1+1 fois c.-à-d. 4 fois :

som	i	Numéro d'itération
0+1=1	1	1
1+2=3	2	2
3+3=6	3	3

6+4=10	4	4

Le résultat est 1+2+3+4 =10.

Comme on peut le voir dans cet exemple, on peut utiliser la valeur de indice dans le bloc d'instructions.

2. Instruction While - do (boucle While - do) :

Syntaxe : (قاعدة الكتابة)

Algorithmiques	Langage PASCAL

TANTQUE condition **FAIRE**
 	
Action(s) ;
 	
FINTANTQUE
	WHILE condition **DO**
	Instruction;
	
	Ou bien
	
	WHILE condition **DO**
	BEGIN
	Instruction(s) ;
	END;

Fonctionnement

La condition est tout d'abord calculée, si le résultat du calcul égale à vrai, le bloc d'instructions est répété, tant que la condition est vraie.

Il faut noter que si la condition est fausse dès le début, l'instruction (les instructions) n'est jamais exécutée.

Exemple:

```
Program exemple;
Uses wincrt;
Var
  Somme, nombre :integer ;
Begin
Somme:=0;
While somme < 15 do
begin
  Writeln('Donnez un nombre');
  Readln(nombre);
  Somme :=somme+nombre ;
End ;
Writeln('somme obtenue=', somme) ;
End.
```

Condition Somme<10	Instructions : (1) Somme<15 (2)Writeln('Donnez un nombre'); (3)Readln(nombre); (4)Somme :=somme+nombre ;	nomdre	somme	Numéro d'itération

vrai	(1) – (2) – (3) –(4)	2	2	1
vrai	(1) – (2) – (3) –(4)	3	5	2
vrai	(1) – (2) – (3) –(4)	6	11	3
vrai	(1) – (2) – (3) –(4)	10	21	4
Faux	(1)		21	fin

ATTENTION : Vous êtes vous-même obligé d'ajouter une instruction pour modifier la valeur de la variable **Somme qui compose la condition** sinon votre programme tournera infiniment. Dans notre exemple : c'est l'instruction Somme:= somme+nombre.

3. Instruction Repeat - until (boucle Repeat... until) :

Syntaxe : (قاعدة الكتابة)

Algorithmiques	Langage PASCAL
REPETER FAIRE Action(s) ; **JUSQU'A Condition**	**REPEAT** Instruction(s); **UNTIL** Condition;

Fonctionnement

L'instruction ou le bloc d'instruction est répété jusqu'à ce que la valeur de la condition égale à vrai.

Même si la condition est vraie dès le début, L'instruction ou le bloc d'instruction sont au moins exécutées une fois.

Exemple:

```
PROGRAM multiples ;
USES wincrt ;
VAR n :integer ;
Begin
     n :=100 ;
     REPEAT
     n :=n+1 ;
     IF (n mod 9 =0) THEN Writeln(n) ;
     UNTIL n>200 ;
End.
```

Ce programme affiche à l'écran tous les multiples de 9 compris entre 100 et 200.
Pour chaque nombre n compris entre 100 et 200, on calcule le reste de sa division par 9, si ce reste est égale à 0, alors n est un multiple de 9.

Différence entre les deux boucles conditionnelles:

While-do	Repeat-until
▪ Le test est effectué avant d'entrer dans la boucle. ▪ on sort de la boucle si la condition est fausse ▪ Les instructions de la boucle peuvent ne jamais être exécutées, si dès la première fois la condition est fausse.	▪ Le test est effectué après les instructions de la boucle ▪ On sort de la boucle si la condition est vraie ▪ Les instructions de la boucle sont exécutées au moins une fois.

Démarche à suivre pour écrire un programme:

Pour construire un programme, on doit suivre les étapes suivantes:

1. Introduire les données: faire entrer les valeurs des données.

2. Résoudre le problème: manipuler les données pour obtenir la solution au problème donné, cette étape se diffère selon le problème posé.

3. Affichage du résultat obtenu.

On peut représenter les trois étapes précédentes par le schéma suivant: