# Chapitre I : Les structures de données

Un programme informatique est une suite d'instructions qui manipulent des données qui peuvent être des constantes, des variables de plusieurs types.

## I- Les constantes :

Une constante est toute donnée dont on décide de garder sa valeur inchangée tout au long d'un algorithme ou un programme. Une constante est caractérisée par : * Son nom * Sa valeur

### Déclaration d'une constante:

<table>
<tr><td>Analyse et algorithme</td><td>pascal</td></tr>
<tr><td><table><tr><td>Objet</td><td>Type/nature</td></tr><tr><td>Nom de la constante</td><td>Constante=valeur</td></tr></table></td><td>Const<br>Nom constante=valeur ;</td></tr>
</table>

Remarque : En Pascal, la déclaration des constantes vient toujours avant la déclaration des variables.

## II- Les variables :

Une variable est toute donnée pouvant prendre différentes valeurs tout au long d'un algorithme ou un programme. Une variable est caractérisée par : * Son nom * Son type * Son contenu

### Déclaration d'une variable:

<table>
<tr><td>Analyse et algorithme</td><td>pascal</td></tr>
<tr><td><table><tr><td>Objet</td><td>Type/nature</td></tr><tr><td>Nom de la variable</td><td>Type de la variable</td></tr></table></td><td>VAR<br>Nom_variable : Type_variable ;</td></tr>
</table>

Remarque : Une variable peut être une donnée initiale, une variable intermédiaire ou un résultat ou les deuxensemble.

## III- Les types de données :

Chaque variable est caractérisée par son type qui détermine l'ensemble des valeurs qui peuvent y être affectés ainsi que l'ensemble des opérations appliquées sur cette variable.

Il existe plusieurs types de variables : Entier, Réel, Booléen, caractère, Chaîne de caractères.

### 1- Le type Entier:

C'est un sous ensemble de Z. Pascal fournit 5 types prédéfinis :

| Type | Plage de valeurs | Nombre d'octets |
|---|---|---|
| **Shortint** | -128..127 | Avec signe, sur 1 octet |
| **Integer** | -32768..32767 | Avec signe, sur 2 octets |
| **Longint** | -2147483648..2147483647 | Avec signe, sur 4 octets |
| **Byte** | 0..255 | Sans signe, sur 1 octet |
| **Word** | 0..65535 | Sans signe, sur 2 octets |

Remarque : Attention au problème de débordement qui engendre des calculs erronés, il faut choisir le type convenable lors de la traduction de l'algorithme en Pascal.
Les opérateurs applicables : +, -, *, Mod, Div, les opérateurs relationnels.

### 2- Le type Réel:

C'est un sous ensemble de R. Il peut être présenté sur 6 octets.
Les opérateurs applicables : +, -, *, /, les opérateurs relationnels.
Les fonctions applicables sur le type réel : Voir Annexe.

### 3- Le type Booléen:

C'est une expression logique : soit Vrai, soit Faux.
Les opérateurs applicables : Non, Et, Ou, Ou ex

| X | Y | Non(X) NOT(X) | X ET Y X AND Y | X OU Y X OR Y | X Ou ex Y X XOR Y |
|---|---|---|---|---|---|
| FAUX | FAUX | VRAI | FAUX | FAUX | FAUX |
| FAUX | VRAI | VRAI | FAUX | VRAI | VRAI |
| VRAI | VRAI | FAUX | VRAI | VRAI | FAUX |
| VRAI | FAUX | FAUX | FAUX | VRAI | VRAI |

NB: A Ou ex B= (Non A ET B) OU (A ET Non B)

### 4- Le type caractère:

Un caractère est représenté par lui-même entre guillemets en algorithmique et entre apostrophes en Pascal.
Remarque : Tous les caractères sont ordonnés par leurs codes ASCII.
Les opérateurs applicables : opérateurs relationnels (c'est comparer leurs codes ASCII)

| Nom algorithmique | Code en Pascal | Rôle | Exemples |
|---|---|---|---|
| ORD(c) | ORD(c) | Renvoie le code ASCII du caractère c. Le résultat est un entier positif | ORD("B" ) vaut 66 |
| CHR(n) | CHR(n) | Renvoie le caractère dont le code ASCII est n | CHR(97) vaut "a" |
| SUCC(c) | SUCC(c) | Renvoie le caractère successeur de c s'il existe | SUCC ("A") vaut "B "<br>SUCC ("8") vaut "9" |
| PRED(c) | PRED(c) | Renvoie le caractère prédécesseur de c s'il existe | PRED("B") vaut "A";<br>PRED("8") vaut "7" |
| MAJUS(c) | UPCASE(c) | Convertit le caractère c en majuscule s'il est possible | MAJUS ("c") vaut "C" |

Les fonctions applicables sur le type caractère

### 5- Le type chaîne de caractères:

Une chaîne de caractères est une suite de n caractères avec 0**≤ n ≤255.**
Une chaîne de caractères est définie entre guillemets en algorithmique et entre apostrophes en Pascal.

| Nom algorithmique | Code en Pascal | Rôle | Exemples |
|---|---|---|---|
| Concat(ch1, Ch2,…,Chn) | CONCAT (ch1, Ch2,…,Chn) | Retourne la concaténation des chaînes Ch1, Ch2,…, chn | Concat ("Ma","dame") retourne la chaîne "Madame" |
| Long (ch) | Length(ch) | Retourne un entier représentant la longueur de la chaîne | Long("Madame") retourne 6 |
| Sous_chaîne(ch, p,nbc) | COPY (ch,p,nbc) | Retourne une sous chaîne d'une longueur nbc à partir de la position p dans ch | Sous_chaîne("Madame", 3,4) retourne "dame" |
| Pos (ch1, ch2) | POS (ch1, ch2) | Retourne la première position de ch1 dans ch2 | POS ("Ma", "Madame") retourne 1 |
| Efface (ch, p, n) | DELETE (ch, p, n) | Enlève n caractères de ch à partir de la position p | Efface ("Informatique", 5,8) retourne "Info" |
| Insère (ch1, ch2, p) | INSERT (ch1, ch2, p) | Insère ch1 dans ch2 à partir de la position p | Insère ("rnava", "Cal",3) retourne "Carnaval" |
| Convch (d, ch1) | STR (d, ch1) | Convertit une valeur numérique en une chaîne | Convch (123, ch) retourne "123" |
| Valeur (ch, d, erreur) | VAL (ch, d, erreur) | convertit ch en une valeur numérique et l'affecte à d. erreur contiendra 0 si la conversion s'est bien déroulée sinon elle contiendra la position du caractère qui a déclenché l'erreur | Valeur ("19,75", d, erreur) retourne d avec 19,75 et erreur avec 0 ; valeur ("1c3", d, erreur) retourne d avec 0 et erreur contient 2 |

Les fonctions et procédures applicables sur le type chaîne de caractères :

## IV- Le type tableau:

C'est une structure de données permettant de ranger un nombre fini d'éléments de même type.

Un tableau est caractérisé par : * Un nom * Une taille * type d'éléments qu'il va contenir

Les éléments d'un tableau ont des indices qui sont de type scalaire. Un tableau unidimensionnel est dit Vecteur.

| Analyse & Algorithme | Pascal |
|---|---|
| Objet: Nom du tableau — Type/Nature: Tableau de **taille type d'éléments** | VAR<br>Nom_ Tableau: Array [Binf..Bsup] of type_elts; |

**Déclaration d'un type tableau:** On peut déclarer un nouveau type tableau :

| Analyse & Algorithme | Pascal |
|---|---|
| TYPE<br>Nom_Type= Tableau de **taille type d'éléments**<br><br>Objet: Nom du tableau — Type/Nature: Nom_Type | TYPE<br>Nom_Type= Array [Binf.. Bsup] of type_element;<br><br>VAR<br>Nom_Tableau: Nom-Type; |

### Déclaration d'un tableau:

Le nombre d'éléments d'un tableau : Bsup-Binf+1

Si l'indice est de type caractère : Ord(Bsup)-Ord(Binf) +1

L'accès à un élément du tableau est direct : nom_tab [indice] avec Binf **≤ Indice ≤** Bsup

Sur un élément du tableau, on peut appliquer toutes les opérations définies sur une variable de même type que
l'élément.

## V- Le type scalaire énuméré:

Le type scalaire est une donnée à valeur unique.
Le type scalaire par énumération définit un ensemble ordonné et fini de valeurs désignées par des
identificateurs.

| Analyse & Algorithme | Pascal |
|---|---|
| TYPE<br>Nom_Type= (cste1, cste2, cste3, ....., csten)<br><br>Objet \| Type/Nature<br>Nom de la variable \| Nom_Type | TYPE<br>Nom_Type= Array (cste1, cste2, cste3, ....., csten);<br><br>VAR<br>Nom_variable : Nom-Type; |

**Exemples :** les 4 saisons, les jours de la semaine, l'année administrative, l'année scolaire…

| Analyse & Algorithme | Pascal |
|---|---|
| TYPE<br>Jours= (Lundi, Mardi, Mercredi, Jeudi, vendredi, samedi, Dimanche)<br><br>Objet \| Type/Nature<br>J \| Jours | TYPE<br>Jours= (Lundi, Mardi, Mercredi, Jeudi, vendredi, samedi, Dimanche);<br><br>VAR<br>J : Jours; |

Déclaration d'un type scalaire par énumération:

Remarques:
- Ord(Lundi)=0; Ord(Jeudi)=3; Succ(Mercredi)=Jeudi; Pred(Mardi)=Lundi
- Un identificateur ne peut pas être un mot réservé.
- Un même identificateur ne peut pas désigner plusieurs choses différentes.
- On ne peut pas déclarer un type scalaire énuméré de type nombre ou caractère.
- On ne peut ni lire ni afficher un objet de type scalaire énuméré.

## VI- Le type intervalle:

Le type intervalle a les propriétés d'un type scalaire ordonné. (Entier, caractère, type scalaire énuméré)
Un intervalle a deux bornes : Binf et Bsup.

Déclaration d'un type intervalle:

| Analyse & Algorithme | Pascal |
|---|---|
| TYPE<br>Nom_Type= Binf .. Bsup<br><br>Objet \| Type/Nature<br>Nom de la variable \| Nom_Type | TYPE<br>Nom_Type= Binf .. Bsup ;<br><br>VAR<br>Nom_variable : Nom-Type; |

Exemples : Week-end, Jours de travail, Jours du mois, Semaines de l'année…

```
TYPE
Week_End= Vendredi .. Dimanche ;
VAR
WE : Week_End;
```

## VII- Les expressions:

### 1- Définition:

On appelle expression toute composition d'opérandes et d'opérateurs réalisant un calcul déterminé.

Nous distinguons 2 types d'expression :

- Expressions arithmétiques donnant une valeur numérique
- Expressions logiques donnant une valeur booléenne.

Une expression est constituée d'opérandes reliés par des opérateurs.

### 2- Les opérandes:

C'est l'élément sur lequel on applique l'opération.

Dans une expression, les opérandes utilisés doivent être de même type ou de types compatibles.

Les opérandes peuvent être des constantes, des variables ou des résultats envoyés par des fonctions.

### 3- Les opérateurs:

C'est l'opération appliquée sur les opérandes.

- Les opérateurs arithmétiques :

- Opérateurs unaires : appliqués sur un seul opérande. (-). On l'appelle aussi monodique.
- Opérateurs binaires : appliqués sur 2 opérandes. On l'appelle aussi dyadique. (+, _, *, /, Div, Mod).

- Les opérateurs logiques :

- Opérateurs unaires : Non.
- Opérateurs binaires : Et, Ou, Ou ex.

- Les opérateurs relationnels : Ces opérateurs peuvent être appliqués sur tous les types de données déjà vus. La comparaison est faite entre 2 éléments de même type ou de types compatibles.

| Algorithmique | Pascal |
|---|---|
| <, >, ≥, ≤, ≠, = | <, >, >=, <=, <>, = |

## 4- Évaluation d'une expression:

| Opérateurs | Priorité | Catégorie |
|---|---|---|
| Parenthèses | 1 | |
| NOT, - (-7) | 2 | Opérateurs unaires |
| *, /, DIV, MOD, AND | 3 | Opérateurs multiplicatifs |
| OR, XOR,+, - | 4 | Opérateurs additifs |
| <>, =, <, >, <=, >= | 5 | Opérateurs relationnels |

les fonctions arithmétiques standard appliquées sur les réels

| om algorithmique | Code en Pascal | Type du paramètre | Type du résultat | Rôle | Exemples |
|---|---|---|---|---|---|
| Tronc(x) | TRUNC(x) | Entier ou Réel | Entier | Supprime la partie décimale pour ne laisser que la composante entière de x | Tronc (-10,434) vaut -10 |
| Arrondi(x) | ROUND(x) | Entier ou Réel | Entier | Donne un entier qui est la valeur du réel x arrondie à la plus proche valeur | Arrondi(34,6) vaut 35; Arrondi(9,45) vaut 9 |
| Abs(x) | ABS(x) | Entier ou Réel | Entier ou Réel (même type que x) | Donne la valeur absolue de x | Abs(-1765) vaut 1765; Abs(6) vaut 6 |
| Carré(x) | SQR(x) | Entier ou Réel | Entier ou Réel (même type que x) | Donne le carré de x | Carré(2) vaut 4 |
| Racine Carré(x) | SQRT(x) | Entier ou Réel | Réel | Donne la racine carrée de x si x >=0 sinon provoque une erreur | Racine Carré(12,25) vaut 3,5 |
| Sin(x) | SIN(x) | Entier ou Réel | Réel | Donne le sinus de x (x en radians) | Sin(1,5705) vaut 1 |
| Cos(x) | COS(x) | Entier ou Réel | Réel | Donne le cosinus de x (x en radians) | Cos(1,5705) vaut 0 |
| Tang(x) | TAN(x) | Entier ou Réel | Réel | Donne la tangente de x (x en radians) | Tang(3,141) vaut 0 |
| Cotang(x) | COTAN(x) | Entier ou Réel | Réel | Donne la cotangente de x (x en radians) | Tang(1,5705) vaut 0 |
| Ent(x) | INT(x) | Entier ou Réel | Entier | Donne la partie entière d'un réel | Ent (5,6) vaut 5,0 Ent (-8,7) vaut -8,0 |
| Aléa | RANDOM | | Réel | Donne un réel compris entre 0 et 1 exclus | Aléa peut donner 0,56 |
| Aléa(x) | RANDOM(x) | Entier | Entier | Donne un entier entre 0 et x-1 | Aléa(12) peut donner 7 |
| Ln(x) | Ln(x) | Entier ou Réel | Réel | Renvoie le logarithme népérien d'un réel x | Ln(1) vaut 0 |
| Exp(x) | EXP(x) | Entier ou Réel | Réel | Renvoie l'exponentiel de x | Exp(0) vaut 1 |