# Algorithmique et programmation

## II - Langages de programmation

### 1- Notion de programme

***Définition***

Un programme est une suite d'instructions à effectuer sur des données avec un ordre précis dans un langage donné.
L'exécution d'un programme correspond à effectuer la suite des instructions qui le composent.
L'activité d'établir des programmes est appelée : programmation.

### 2- Langages de programmation

Pour faire exécuter un programme par l'ordinateur, il doit être écrit dans un langage compréhensible par l'ordinateur (langage machine).
Le langage utilisé pour programmer un ordinateur est appelé : langage de programmation.

***Définition :***

Un langage de programmation est un code permettant de programmer l'ordinateur afin de lui faire réaliser des tâches.
Un langage de programmation est l'intermédiaire entre l'être humain et la machine.

Un langage de programmation est différent du langage machine. Il faut donc le traduire pour le rendre compréhensible par le processeur.
Les langages de programmation peuvent se classer en deux catégories :

#### a - Langages interprétés

Un programme écrit dans un langage interprété a besoins d'un programme auxiliaire (l'interpréteur) pour traduire au fur et à mesure les instructions du programme. [Ex : Visual Basic, MATLAB, Prolog,…etc.]

#### b - Langages compilés

Un programme écrit dans un langage dit "compilé" va être traduit une fois pour toutes par un programme annexe (le compilateur) afin de générer un nouveau fichier qui sera autonome. [Ex : Pascal, C, C++, Cobol…etc.]

### 3- Raisons du choix du langage de programmation PASCAL

- Le langage Pascal est un langage compilé, c'est-à-dire qu'il faut :
  - Entrer un texte dans l'ordinateur (à l'aide d'un programme appelé éditeur);
  - Le traduire en langage machine ;
  - L'exécuter.
- C'est un langage utile à tous ceux qui débutent dans la programmation.
- Le langage Pascal est très structuré et constitue en lui-même une très bonne approche de la programmation.

### 4- Format simple d'un programme en langage PASCAL

Un programme Pascal est constitué de trois parties principales : **l'en-tête**, **la partie de déclaration** (variables, constantes,…) et **le corps du programme.**
Le format d'un programme en langage Pascal peut être présenté comme suit :

```
Program TITRE ;                                  {en-tête}
Déclarations des variables, des constantes,… ;   {partie déclarative}
Begin
  Instructions du programme ;
  …                                              } corps du programme
  Instruction du programme
End.
```

* **Program** : C'est le premier mot de chaque programme Pascal.

* **TITRE** : C'est le nom que le programmeur donne au programme qu'il écrit. C'est un identificateur. Il doit être suivi par ' ;'

* **Déclarations des variables** : Dans cette partie, le programmeur doit déclarer toutes les variables utilisées par son programme.
  L'instruction **VAR** permet de déclarer les variables utilisées.
  *Syntaxe :*

  **VAR** nom_de_la_variable : type;

* **Begin** : Précise le point de départ du programme et fournit un moyen de grouper les instructions d'un même programme. Le mot-clé est toujours fermé par un **end** suivi d'un point.

* **End** : Ce mot-clé doit être suivi d'un point, s'il est la dernière instruction d'un programme Pascal. Il est aussi utilisé pour limiter une instruction.

Toutes les instructions du programme doivent être séparées par un point-virgule. Les instructions du programme précédant une instruction **end** ne nécessitent pas le point-virgule.

### 5- Types de variables standards et opérateurs associés

Parmi les types de variables que le langage Pascal supporte, on peut citer :

* **integer** : sert à utiliser les entiers simples sans virgule.
* **Char** : sert à manipuler les caractères.
* **boolean** : ce type de variable peut prendre soit la valeur TRUE (vrai), soit la valeur FALSE (faux).
* **Real** : nombres réels.

Le langage Pascal réserve pour chaque type de variable un ensemble d'opérations à utiliser.
Le tableau ci-dessous résume ces types de variables ainsi que les opérations utilisées avec chaque type :

| Déclaration | Quelques opérations | Exemples |
|---|---|---|
| VAR A, B : INTEGER ; | + - * **div** (division entière) **mod** (reste de la division entière).<br>Elles sont toutes à résultat entier, et nécessitent deux arguments entiers. | Soit A = 6, B = 4<br>• A – B = 2<br>• A * B = 24<br>• A **div** B = 1<br>• A **mod** B = 2 |
| VAR A , B, C : REAL ; | + - * / (division)<br>Quand une opération comprend un argument réel et un entier, le résultat est réel.<br>L'opération / donne toujours un résultat réel, même si les deux arguments sont entiers. | Soit A = 8, B = 6, C = 2<br>• (A+B) / C = 7<br>• A+B/C = 11 |
| VAR A ,B : BOOLEAN ; | **AND, OR, NOT** | Soit A = TRUE, B = FALSE<br>• A **AND** B = FALSE<br>• A **OR** B=TRUE |
| VAR a : CHAR ; | **Ord** (retourne le code ASCII d'un caractère) | Ord ('a') = 97 |
| VAR A, B : STRING ; | + (concaténation)<br>**Upcase** (conversion en majuscule) | A = 'Tronc'<br>B = 'Commun'<br>A+B = TroncCommun |
| CONST Nom = valeur; | | taux_TVA = 20<br>Pi = 3.14 |

## 6- Utilisation des Entrées / Sorties

### 6 - 1 Write et writeln

***Définition***

L'instruction **write** affiche à l'écran le contenu d'une variable, d'une constante ou encore du texte.

L'instruction **writeln** a le même effet que **write**, mais le curseur passe à la ligne suivante après l'affichage.

*Syntaxes :*

| Instruction | Résultat |
|---|---|
| write (nom) ; | Affichage du contenu de la variable (ou de la constante) **nom**. |
| write ('Ce message sera affiché') ; | Affichage du texte **Ce message sera affiché.** |
| write (nom : 5) ; | Affichage du contenu de la variable (ou de la constante) **nom** sur une largeur d'au moins 5 colonnes. |
| Write (nom : 7 : 3) ; | Affichage du contenu de la variable (ou de la constante) **nom** sur une largeur d'au moins 7 colonnes avec 3 chiffres après la virgule. |

### 6 - 2 Read et Readln

***Définition***

L'instruction **read** permet à l'utilisateur de rentrer une valeur qui sera utilisée par le programme.

L'instruction **readln** a le même effet que **read**, mais elle provoque un retour à la ligne et attend la saisie de la part de l'utilisateur.

*Syntaxes :*

**read (variable) ;**
**readln (variable) ;**

## 7- Affectation

***Définition***

L'affectation est l'opération permettant la mise d'une valeur dans une variable.

En langage Pascal, une affectation est représentée par le signe : =

*Syntaxes :*

**Variable := Valeur1 ;** Ce qui signifie mettre la valeur **Valeur1** dans la variable.

**Variable_1 := Variable_2 ;** Ce qui signifie mettre la valeur de la **variable_2** dans la variable_1.

**8- Utilisation des instructions conditionnelles**

En langage Pascal, la structure conditionnelle peut se présenter sous l'une des formes suivantes :

| **Pour exécuter une seule instruction** | **Pour exécuter plusieurs instructions** |
|---|---|
| **IF** condition **THEN**<br>Instruction_1 { ; est interdit}<br>**ELSE**<br>Instruction_2 ; | **IF** condition **THEN**<br>Begin<br>Instruction_1 ;<br>…<br>Instruction_n ; { ; est facultatif}<br>End { ; est interdit}<br>**ELSE**<br>Begin<br>Instruction_1 ;<br>…<br>Instruction_n ; { ; est facultatif}<br>End ; |

**9- Utilisation des instructions sélectives (CASE…OF…END)**

***Description***

Cette instruction compare la valeur d'une variable de type entier ou caractère à un ensemble de valeurs constantes.

| *Syntaxes :* | |
|---|---|
| **Case** variable **of**<br><br>Liste_valeur_1_1 : instruction_1;<br>Liste_valeur_1_2 : instruction_2;<br>{…}<br>Liste_valeur_1_n : instruction_n;<br><br>**End;** | **Case** variable **of**<br><br>Liste_valeur_1_1 : instruction_1;<br>Liste_valeur_1_2 : instruction_2;<br>{…}<br>Liste_valeur_1_n : instruction_n<br>**Else** instruction ;<br><br>**End;** |