Introduction au Langage Pascal

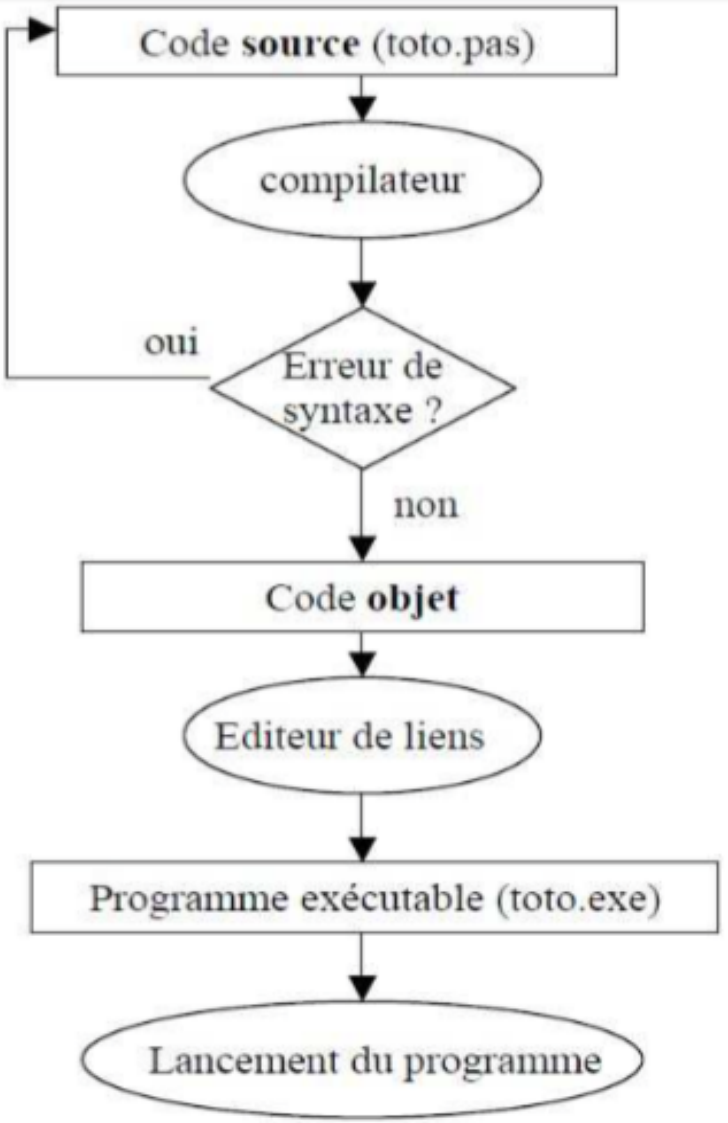
L1 MPI-PC

Dr Ousmane DIALLO

Chap 4: Introduction au Pascal

4.1. Programmer en Pascal

- Un programme pascal (*prog* par exemple) peut être écrit avec un simple éditeur de texte. Le programme ainsi réalisé est stocké sous forme de fichier avec l'extension **.pas** (*prog.pas*). Dans ce cours, les programmes sont écrits dans un IDE (Integrated Development Environment) appelé **Turbo Pascal** qui va permettre de les compiler et de les exécuter.
- ***Tout un processus du fichier source écrit en Pascal (langage de programmation proche de notre langage) au programme compréhensible par la machine (langage binaire, suite de 0 et 1)***
- La version en langage machine d'un programme s'appelle aussi le code objet. L'éditeur de liens est un programme qui intègre, après la compilation du fichier source, le code machine de toutes les fonctions utilisées dans le programme et non définies à l'intérieur. A partir du code objet du programme l'éditeur de liens génère un fichier exécutable d'extension **.exe**(*prog.exe*). Le fichier exécutable renferme alors le programme qui peut être chargé dans la mémoire pour être exécuté.



Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Alphabet

- Il est constitué des éléments suivants :
  - ✓ Les majuscules : **A, B,..., Z** (26 caractères)
  - ✓ Les minuscules : **a, b,..., z** (26 caractères)
  - ✓ Le caractère **« blanc »**
  - ✓ Les chiffres : **0, 1,..., 9**
  - ✓ Les symboles spéciaux :
  - ✓ Les opérateurs :
    - Aarithmétiques : + - * /
    - relationnels : <; <; = ; < =; >=; <>
  - ✓ Les séparateurs: (); {}; []; (* *)
  - ✓ Le signe « pointeur » : ^
  - ✓ Les signes de ponctuation: **. , ; : ' !?**

Remarque :
Le Pascal n'est pas sensible à la casse. Il ne fait pas de distinction entre majuscule et minuscule. L'écriture **begin** est correcte ainsi que les écritures suivantes **BEGIN** ou **Begin**

Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Les mots du langage

- Il est constitué des éléments suivants : Un mot est une suite de caractères encadrés par des espaces ou des caractères spéciaux.

Les mots réservés

- Ne peuvent être redéfinis par l'utilisateur et ont une signification précise
- Ils aident à la construction syntaxique des instructions.

Exemples de mots réservés :

AND BEGIN REPEAT WHILE PROCEDURE
END CASE VAR TYPE FUNCTION
IF GOTO DO UNTIL ARRAY
FOR WITH CONST ELSE …

Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Les mots du langage

Les identificateurs

- Un identificateur est un nom donné à un élément du programme introduit par l'utilisateur (constante, variable, fonction, procédure, programme).
- En Pascal, Un identificateur est une suite alphanumérique commençant nécessairement par une lettre de l'alphabet et ne comportant pas d'espaces.
- Il est possible possible de lier plusieurs mots à l'aide de "_".
- Existence d'une limite au nombre de caractères. (dépend du compilateur)

Exemples d'identificateurs légaux :

TERME terme TRES_LONG_TERME X23 Z1Z2

Exemples d'identificateurs non légaux :

3MOT U.T.C. MOT-BIS A!8 $PROG AUTRE MOT

Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Les mots du langage

Les identificateurs standards

- Identificateurs prédéfinis ayant une signification standard. Peuvent être redéfinis par l'utilisateur (action peu conseillée).

Exemples d'identificateurs standards :

Fonctions :

COS SIN EXP SQR SQRT SUCC PRED

Constantes :

MAXINT TRUE FALSE

Types :

INTEGER REAL BOOLEAN CHAR

Procédures :

READ WRITE NEW RESET REWRITE

Fichiers d'entrée-sortie :

INPUT OUTPUT

Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Les identificateurs standards

Fonctions de base:

ABS (X) valeur absolue de X
ARCTAN (X) arc-tangente de X
CHR (X) caractère dont le numéro d'ordre est X
COS (X) cosinus de X
EOF (X) vrai si fin du fichier X
EOLN (X) vrai si fin de ligne du fichier X
EXP (X) exponentielle de X
LN (X) logarithme népérien de X
ODD (X) vrai si X est impair, faux sinon
ORD (X) numéro d'ordre dans l'ensemble de X
PRED (X) prédécesseur de X dans son ensemble
ROUND (X) arrondi de X
SIN (X) sinus de X
SQR (X) carré de X
SQRT (X) racine carrée de X
SUCC (X) successeur de X dans son ensemble
TRUNC (X) partie entière de X

Chap 4: Introduction au Pascal

4.2. Constituants élémentaires du Pascal

Le langage Pascal

Caractéristiques globales

C'est un Langage Typé

- ✓ Toutes les variables doivent être pré-déclarées, ce qui évite les erreurs
- ✓ Leur type doit être explicitement défini, ce qui enlève toute ambiguïté
- ✓ Il s'agit d'un langage très *pédagogique* et bien adapté pour les débutants

C'est un Langage Structuré

- ✓ Organisation du programme en "blocs d'instructions" emboîtés,
- ✓ Utilisation d'identificateurs pour spécifier les blocs,
- ✓ Utilisation d'indentations pour visualiser l'architecture du programme.

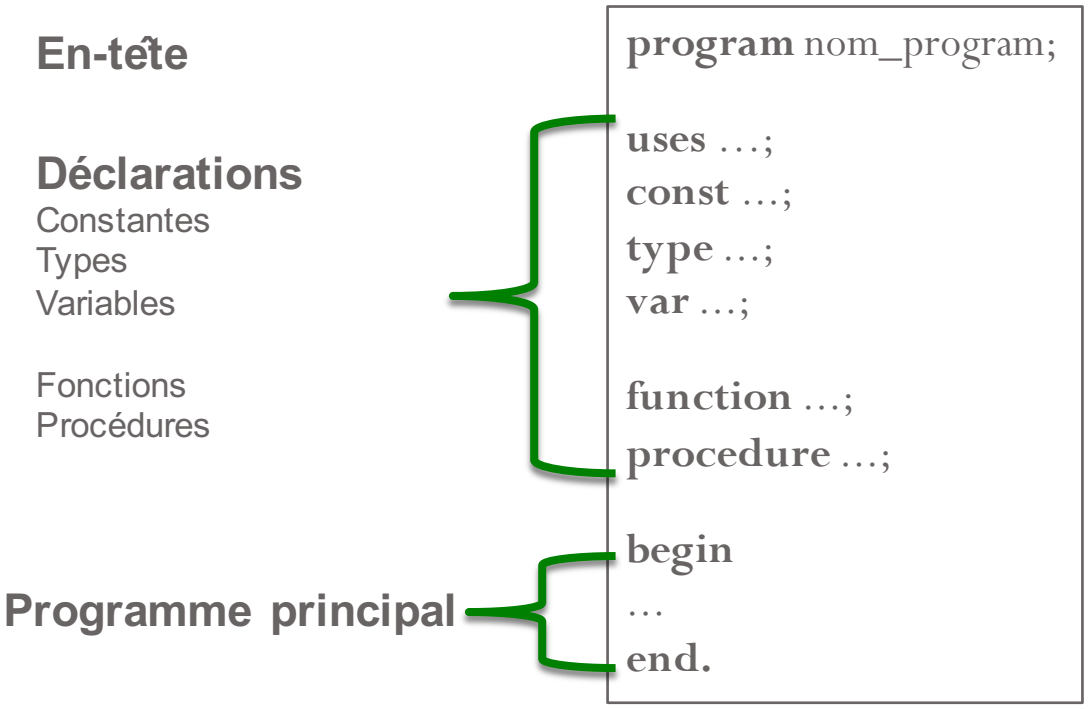
C'est un Langage Récursif

- ✓ Programmation concise et efficace

Chap 4: Introduction au Pascal

4.3. Structure globale d'un programme en Pascal

- Semblable à la structure d'un algorithme, un programme en Pascal respectera la structure suivante:



Remarque:
Dans la partie des déclarations, ***l'ordre indiqué est obligatoire***, mais les ***clauses sont optionnelles*** et dépendent des besoins du programmeur
Seuls l'entête et le corps du programme sont obligatoire.

Uses: précise les informations dont le compilateur a besoin pour utiliser les primitives prédéfinies (clrscr, …) se trouvant dans des fichiers spéciaux.

Chap 4: Introduction au Pascal

4.3. Structure globale d'un programme en Pascal

Exemples: les premiers programmes qui affichent « Bonsoir»

```
program bonsoir;

begin
writeln ('bonsoir !');
end.
```

En-tête

Déclarations

Corps

```
program bonsoir_bis;
uses crt;
begin
clrscr;
writeln ('bonsoir !');
end.
```

Chap 4: Introduction au Pascal

4.4. Déclarations de constantes

Syntaxe

Const
identificateur = valeur_constante;

Exemples

- ✓ Déclarations de type numérique

```
const
    DEUX = 2;
    PI = 3.14;
```

- ✓ Déclarations de type booléen

```
VRAI = true;
FAUX = false;
```

- ✓ Déclarations de type caractère

```
CARA = 'A';
```

- ✓ Déclarations de type chaîne de caractères

```
PHRASE = 'Vaut mieux une fin effroyable qu"un effroi sans fin';
```

Chap 4: Introduction au Pascal

4.5. Déclaration des types

- Un type est un ensemble de valeurs que peut prendre une donnée.

Les types standards

- Un type standard est un type qui est normalement connu de tout langage Pascal et qui n'a donc pas été déclaré par l'utilisateur.

```
byte :          0..255                      (8 bits non signé)
shortint:       -128..127                   (8 bits signé)
word:           0..65535                    (16 bits non signé)
integer:        -32768..32767               (16 bits signé)
longint:        -2147483648..2147486647     (32 bits signé)
real:           2.9e-39..1.7e38             (64 bits signé) 6 chiffres signif
double          5.0e-324..1.7e308                           15 chiffres signif
boolean :       true ou False
file...of :     fichier de...
string :        chaîne de caractère (maximum 255)
string[num] :   chaîne de caractère de longueur num
char :          caractère (les 256 caractères ASCII)
pointer :       pointeur ;
```

Chap 4: Introduction au Pascal

4.5. Déclaration des types

Les types scalaires et non standards

Principe: on fabrique les types dont on a besoin par l'intermédiaire du mot réservé ***TYPE***.

- Les types ***scalaires standards*** sont les ***entiers*** et les ***caractères***.

Type énuméré

Un ***type énuméré*** est une séquence ordonnée d'identificateurs.

Syntaxe : ***TYPE identificateur = (id1, id2, … , idn) ;***

Exemple

Type COULEUR = (jaune, vert, rouge, bleu, marron);
SEMAINE=(lundi, mardi, mercredi, jeudi, vendredi, samedi, dimanche);
SEXE =(masculin, féminin);
VOYELLE = (A, E, I, O, U);

NB: Le mot réservé ***TYPE*** ne doit être écrit qu'une seule fois.

Chap 4: Introduction au Pascal

4.5. Déclaration des types

Les types scalaires et non standards

Type énuméré

Restriction: Deux types énumérés différents ne peuvent contenir le même identificateur

Ensembles ordonnés: Le système reconnaît automatiquement les successeurs et les prédécesseurs d'un élément. *succ(mardi)= mercredi, pred(samedi)=vendredi*.

Type intervalle

Un ***type intervalle*** est un sous-type d'un type scalaire déjà défini.

Syntaxe : ***TYPE identificateur = [borne inf].. [borne sup] ;***

Points importants :

- ✓ Valeurs autorisées : toutes celles de l'intervalle
- ✓ Deux sortes de type intervalle :
  - les types issus d'un type énuméré standard
  - les types issus d'un type énuméré déclaré.

Chap 4: Introduction au Pascal

4.5. Déclaration des types

Les types scalaires et non standards

Type intervalle

Exemples issus d'un type standard:

✓ Intervalle d'entiers :

```
Type DECIMAL = 0 .. 9 ;
     OCTAL = 0 .. 7 ;
     AGE = 0..150;
```

✓ Intervalle de caractères :

```
Type ABC = 'A' .. 'C' ;
     MAJ = 'A' .. 'Z' ;
```

Exemples issus d'un type non-standard

```
Type OUVRABLE = lundi .. vendredi ;
     WEEK-END = samedi .. dimanche ;
     LETTRES = 'A' .. 'Z' ;
```

4.5. Déclaration des types

Les types scalaires et non standards

Type intervalle

Ordre ascendant requis : borne-inf doit être placé avant borne-sup dans le type énuméré source.

Déclarations de type intervalle incorrectes :

Type OCTAL = 7 .. 0 ;

OUVRABLE = vendredi .. lundi ;

Pas de type intervalle issu du type réel (non scalaire) !

Chap 4: Introduction au Pascal

4.6. Déclaration des variables

- Une variable est une donnée manipulée par un programme et pouvant être modifiée. Elle peut être :
  - ✓ une donnée d'entrée ;
  - ✓ le résultat final d'un calcul ;
  - ✓ un résultat intermédiaire de calcul.
- ***Déclarer une variable***, c'est définir l'ensemble des valeurs qu'elle peut prendre.
- Toutes les variables utilisées dans un programme doivent être déclarées.

Chap 4: Introduction au Pascal

4.6. Déclaration des variables

Deux façons pour déclarer une variable :

- l'aide d'un type standard ou d'un type pré-déclaré
- par une déclaration explicite et spécifique à cette variable de l'ensemble des valeurs qu'elle peut prendre.

Syntaxe : *VAR identificateur : type ;*

- ✓ *VAR* ne peut apparaître qu'une seule fois
- ✓ Possibilité de grouper plusieurs variables pour le même type
- ✓ Séparation des variables par une virgule

Exemple: déclarations de variables

avec référence à un type existant

```
var JOUR : semaine ;
    A,B,C : real ;
    I,J,K : integer;
    CONGE  : week-end ;
    VIVANT  : boolean ;
```

avec déclaration locale explicite :

```
var
LETTRE: 'A' .. 'Z';
   FEUX : (vert, orange, rouge) ;
```

Chap 4: Introduction au Pascal

4.6. Déclaration des variables

Exemple: déclarations de constante, type et variable

```
const JOUR_MAX = 31 ;
      AN_MIN = 1901 ;
      AN_MAX = 2000 ;
type    SIECLE = AN_MIN . . AN_MAX ;
      SEMAINE = (LUNDI, MARDI, MERCREDI, JEUDI, VENDREDI, SAMEDI, DIMANCHE) ;
       ANNEE = (JANVIER, FEVRIER, MARS, AVRIL, MAI, JUIN, JUILLET, AOUT,
                SEPTEMBLE, OCTOBRE , NOVEMBRE, DECEMBRE);
var   MOIS      : ANNEE ;
      JOUR      : SEMAINE ;
  NB_JOUR       : 1 .. JOUR_MAX ;
        AN      : SIECLE ;
OUVRABLE        : lundi .. vendredi ;
       I,J      : integer ;
```

Chap 4: Introduction au Pascal

4.7. Instructions composées

Définition

Les ***instructions composées*** permettent de regrouper, dans un même bloc, un ensemble d'instructions qui seront exécutées au même niveau.

Syntaxe : ***Séquence de deux ou plusieurs instructions comprises entre BEGIN et END et séparées par des points virgules***

Instruction d'affectation

<VARIABLE> := <expression>;

- ✓ Evaluation de l'expression (calcul)
- ✓ Puis affectation (rangement) dans la variable (identificateur)

NB: Nécessité d'avoir des types compatibles (les mélanges de types sont interdits)

Ne pas confondre ":=", l'opérateur d'affectation et "=", l'opérateur de test.

Chap 4: Introduction au Pascal

4.7. Instructions composées

Instruction d'affectation

<VARIABLE> := <expression>;

Exemple: *Soit X une variable de type integer et on lui donne comme valeur 20*

X := 20 signifie que l'on affecte la valeur 20 à la variable X donc X vaut 20.

On peut tester si X est égal à une certaine valeur avant d'effectuer un calcul :

Si X = 3 alors X := X / 2

Ici X vaut toujours 20 car le test X = 3 n'est pas vérifié (puisque la valeur 20 a été placée dans X)

Chap 4: Introduction au Pascal

4.8. Opérateurs et fonctions arithmétiques

Opérateurs disponibles

Opérateur	Description
+	somme
-	soustraction
*	multiplication
/	division
DIV	division entière (5 div 3 = 1)
MOD	modulo (5 mod 3 = 2)

Exemple:

```
var A, I, J, K : integer;
begin
  A := 7.4 ; B := 8.3 ;
  C := A + B ;
  D := A / B + C ;
  I := 42; J:=9;
  K:=I mod J;          {K vaut 6}
end.
```

Chap 4: Introduction au Pascal

4.8. Opérateurs et fonctions arithmétiques

Expressions

- Une ***expression*** désigne une valeur, exprimée par composition d'opérateurs appliquées à des opérandes, qui sont : des valeurs, des constantes, des variables, des appels de fonction ou des sous-expressions.

Exemple: *Etant donnée une variable* ***x****, une constante* ***max*** *et une fonction* ***cos()****, chaque ligne contient une expression :*

5

x + 3.14

*2 * cos(x)*

*(x < max) or (cos(x-1) > 2 * (x+1))*

*2.08E3 * x*

(x>2) OR (x<8)

Chap 4: Introduction au Pascal

4.8. Opérateurs et fonctions arithmétiques

Fonctions arithmétiques

ABS (X) valeur absolue de X

ARCTAN (X) arctangente de X

CHR (X) caractère dont le numéro d'ordre est X

COS (X) cosinus de X

EXP (X) exponentielle de X

LN (X) logarithme népérien de X

ORD (X) numéro d'ordre dans l'ensemble de X

PRED (X) prédécesseur de X dans son ensemble

ROUND (X) arrondi de X

SIN (X) sinus de X

SQR (X) carré de X

SQRT (X) racine carrée de X

SUCC (X) successeur de X dans son ensemble

TRUNC (X) partie entière de X

4.8. Opérateurs et fonctions arithmétiques
Fonctions logiques

EOF (X) vrai si la fin du fichier X est atteinte

EOLN (X) vrai si fin de ligne du fichier X

ODD (X) vrai si X est impair, faux sinon

Chap 4: Introduction au Pascal

4.9. Entrées et Sorties

- Ce sont des échanges d'informations entre la mémoire (variables et constantes) et les périphériques (clavier, écran …).
- Types autorisés : réel, booléen, caractères et chaînes de caractères.

Entrée (Lecture)

On utilise la fonction ***readln*** pour la saisie des données depuis le clavier. La fonction admet la syntaxe suivante :***readln(argument1, argument2,…, argumentn)***

Le programme va lire ce que l'utilisateur a tapé et va stocker les valeurs dans les variables argument1,…, argumentn

Exemple

Write('Entrez un nombre entier : ') ; Readln(a) ;

Writeln('vous avez entré la nombre ', a) ;

Write('Entrez 3 nombre réels : ') ; Readln(b, c, d) ;

Readln lit des valeurs sur le périphérique d'entrée standard (clavier) les interprète dans le format de la variable et les range dans les arguments spécifiés. A chaque valeur saisie il faut valider par la touche ***entrée*** pour que la saisie soit prise en compte.

Readln(…); ---> passage à la ligne suivante en ignorant ce qui reste sur la ligne

Readln; peut être employé sans paramètre

4.9. Entrées et Sorties

Sortie (Ecriture)

On utilise la fonction ***write*** ou ***writeln*** pour l'affichage formaté des données. La fonction admet la syntaxe suivante : ***Write(argument1,argument2,..., argumentn)*** ou ***Writeln(argument1,argument2,...,argumentn)*** avec argument1,…, argumentn : les arguments à afficher

Exemple

```
write('bonjour ');
writeln('monsieur');
a :=2+3 ;
writeln('la somme de 2 + 3 donne :',a) ;
```

La fonction ***write*** écrit ici à l'écran les arguments (chaîne de caractères, constante, variable)
La fonction ***writeln*** écrit la même chose. La seule différence est que à la fin de l'écriture du dernier argument, il y a un passage à la ligne suivante.

Chap 4: Introduction au Pascal

4.9. Entrées et Sorties

Largeur d'affichage

Cette largeur correspond au nombre de caractères ou de chiffres qui doivent être affichés pour une variable. Pour effectuer un tel formatage à l'affichage, il faut placer un nombre entier suivant la variable à afficher et séparé par ***deux points***. Si la représentation d'une valeur nécessite moins de position que spécifiée par la valeur, le reste est complété par des espaces à gauche.

Exemple : pi: = 3.14159; writeln(pi); (* affiche 3.141590000E+00*)
writeln(pi:20); (*affiche ␣␣␣␣3.141590000E+00*); writeln(pi:0); (* affiche 3.1E+00*);

Précision

On peut spécifier la précision des valeurs à afficher. La spécification de précision est un nombre entier qui suit la largeur minimale et est séparée de cette dernière par ***deux points***. Pour les réels, il indique le nombre de chiffres après la virgule

Exemple : pi: = 3.14159; writeln(pi); (* affiche 3.141590000E+00*)
writeln(pi:20:5); (* affiche␣␣␣␣␣␣␣␣␣␣3.14159*); writeln(pi:0:4); (* affiche 3.1415*);

4.9. Entrées et Sorties

Lecture directe au clavier

Il existe une fonction avec laquelle on peut entrer une valeur sans valider avec la touche ***entrée***. Cette entrée manipule uniquement des caractères. Il faut donc déclarer des variables de type caractère

Exemple :

```
C:=readkey;        {lit une touche au clavier}
C:=upcase(readkey); {lit une touche au clavier et la convertit en minuscule}
```