Langage Pascal

Langage Pascal Objet

Programmation avec des Unités

Langage Pascal

Langage Pascal

Structure d ’un programme :

```
program nom_programme;
        <Déclaration globale des unités, constantes, types et variables>
begin
        <Instructions>
end.
```

Langage Pascal

Les déclarations globales :

```
uses graph;                    {une unité nommée graph doit exister}

const
        valeur_max=100; (* la valeur d ’une constante n ’est pas modifiable*)
        valeur_min=50;

type
        T1_valeur= array[1..10] of integer;
        T2_valeur= array[valeur_min..valeur_max] of char;

var
        tab1, tab2 : T1_valeur;
        tab3 : T2_valeur;
        tab4 : array [-5..5] of real;
```

Langage Pascal

Les instructions :

```
program test;

var entier: integer;
      caractere: char;
      chaine: string;
      reel: real;
     bool: boolean;

begin
  entier:=6;                    {les affectations}
  caractere:=’c’;
  chaine:='ceci est une chaîne de caractères';
  reel:=5.06;
  bool:=true;
end.
```

Langage Pascal

Les instructions :

```
program test;

var entier1, entier2: integer;
      chaine1, chaine2: string;
      reel: real;

begin
  entier1:=6;                   {les opérations}
  entier2:=entier1-10;
  chaine1:='ceci est une chaîne de caractères';
  chaine2:=chaine1 + ' sans intérêt';
  reel:=(entier1+56.98)/59;
end.
```

Langage Pascal

Les entrées / sorties standards :

```
program test;

var entier1, entier2: integer;

begin
  entier1:=6;                    {les Entrées / Sorties}
  readln (entier2);
  writeln ('ceci est une chaîne de caractères');
  writeln ('vous avez ', entier2, ' euros');
  writeln ('vous avez ', entier2*0.9, ' euros');
end.
```

Langage Pascal

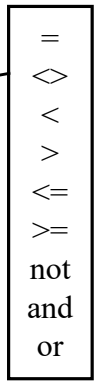
Les boucles :

```
program test;

var somme, produit, entier: integer;

begin
  somme:=0;
  for entier:=1 to 10 do                {boucle FOR}
          somme:=somme +entier;

  somme:=0;
  produit:=1;
  for entier:=1 to 5 do
  begin
    somme:=somme +entier;
    produit:=produit*entier;
  end;
end.
```

Langage Pascal

Les boucles :

```
program test;

var somme, produit, entier: integer;

begin
 entier:=1;
 while (entier<=10) do                {boucle WHILE}
   entier:=entier+1;

 somme:=0; produit:=1; entier:=1;
 while (entier<=5) do
 begin
   somme:=somme +entier;
   produit:=produit*entier;
   entier:=entier+1;
 end;
end.
```



Langage Pascal

Les boucles :

```
program test;

var somme, produit, entier: integer;

begin
 somme:=0;
 produit:=1;
 entier:=1;
 repeat                          {boucle REPEAT}
   somme:=somme +entier;
   produit:=produit*entier;
   entier:=entier+1;
 until (entier>10);

end.
```

Langage Pascal

L'instruction conditionnelle IF :

```
program test;

var entier: integer;
    resultat: string;

begin
  entier:=1;

  if (entier>0) then resultat:='positif';

  if (entier>0) then begin
                       resultat:='positif';
                       entier:=entier-1;
                     end;
end.
```

Langage Pascal

L'instruction conditionnelle IF :

*Avant un **"else"** pas de point virgule*

```
program test;

var entier: integer;
    resultat: string;

begin
  entier:=1;

  if (entier>0) then resultat:='positif'
                else resultat:='négatif';

  if (entier>0) then begin
                       resultat:='positif';
                       entier:=entier-1;
                     end
                else begin
                       resultat:='négatif';
                       entier:=entier+1;
                     end;
end.
```

Langage Pascal

Les tableaux :

```
program test;

var i : integer;
    tab : array [1..10] of real;        {Tableau à une dimension}

begin

  tab[5]:=12.5;
  tab[6]:=tab[5]+5.96;

  for i:=1 to 10 do
      tab[i]:=0;

end.
```

Langage Pascal

Les tableaux :

```
program test;       {Tableau à deux dimensions}

var i, j: integer;
    tab2 : array [1..10, -5..5] of real;
    tab3 : array [1..10] of array [-5..5] of real ;

begin
  tab2[5,0]:=12.23;
  tab3[1][1]:=tab2[5,0]-59.56;

  for i:=1 to 10 do
     for j:=-5 to 5 do
     begin
          tab2[i,j]:=0;
          tab3[i][j]:=0;
     end;
end.
```

Langage Pascal

Les enregistrements :

```
program test;

type
  Temploye = record
                nom: string(40);
                prenom : string(60);
                age : integer;
             end;

var paul, jacques: Temploye;

begin
  paul.nom:='Durant';
  jacques.age:=18;
  paul.age:=jacques.age + 1;
  paul:=jacques;   {les comparaisons globales sont interdites}
end.
```

Langage Pascal

Les fonctions :

```
program test;       {Exemple de fonction simple}

var i, j, k: integer;

function max (a, b : integer): integer;
begin
  if (a<b) then max:=b else max:=a;
end;

begin
  i:=5;
  j:=6;
  k:=max (i, j);
  k:=max (i, 10);
  k:=max (max (i, j), 10);
end.
```

Langage Pascal

Les procedures :

```
program test;        {Exemple de procédure simple}

var i, j: integer;

procedure max (a, b : integer);
begin
  if (a<b) then writeln('le max est ', b)
           else writeln('le max est ', a);
end;

begin
  i:=5;
  j:=6;
  max (i, j);
  max (i, 10);
end.
```

Langage Pascal

Les fichiers :

```
program test;        {Ouverture d'un fichier en Lecture}

var pf: file of Integer;
    i: integer;
    T: array[1..1000] of integer;

begin
  assignfile (pf, 'doc.txt'); {Association d'un pointeur et d'un fichier }
  reset (pf);               {Ouverture du fichier 'doc.txt'}
  i:=1;
  while not (eof (pf)) do   {Tant que le pf ne pointe pas sur la fin de fichier}
  begin
    read(pf, T[i]);         {Lecture d'un entier et stockage dans T}
    i:= i+1;
  end;
  closefile(pf);            {Fermeture du fichier}
end.
```

Langage Pascal

Les fichiers :

```
program test;       {Ouverture d'un fichier en Ecriture}

var pf: file of Integer;
    i: integer;
    T: array[1..1000] of integer;

begin
  assignfile (pf, 'doc.txt'); {Association d'un pointeur et d'un fichier }
  rewrite (pf);             {Ouverture du fichier 'doc.txt'}

  for i:=1 to 1000 do
     write(pf, T[i]);       {Écriture d'un entier T[i] dans le fichier }

  closefile(pf);            {Fermeture du fichier}
end.
```

Langage Pascal

Les notions non abordées :

- Les pointeurs
- L'allocation dynamique
- La récursivité

Programmation avec des Unités

Langage Pascal - Les unités

Principes :

- Fournir des bibliothèques de fonctions / procédures sans fournir le code source
- Pratique à utiliser dans plusieurs programmes
- Permettre le développement en parallèle des différentes parties d'un projet

Langage Pascal - Les unités

Syntaxe :

```
Unit Nom;

Interface

   Uses …
   Const ...
   Type …
   Var …
   Déclaration des entêtes de fonctions - procédures

Implementation

   Uses …
   Const ...
   Type …
   Var …
   Description des fonctions - procédures

end.
```

} Partie visible

} Partie interne à l'unité

Langage Pascal - Les unités

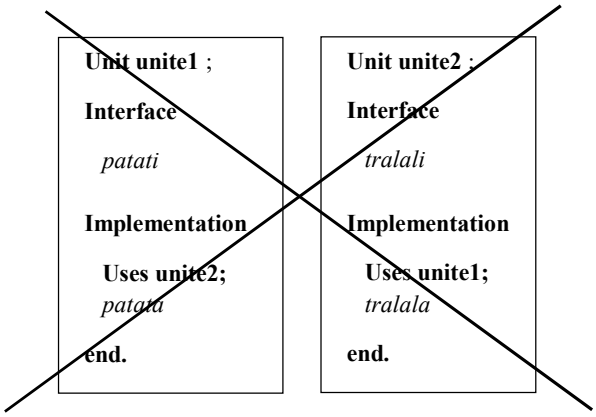
Exemple :

```
Unit tableau ;

Interface

  Const Max=100;
  Type tableau = array [1..Max] of integer;
  Var T1 : tableau ;

  procedure Initialisation (var T: tableau);

Implémentation

  procedure Initialisation (var T: tableau)
  var i: integer;
  begin
    for i:=1 to Max do
       T[i]:=0;
  end;

end.
```

```
Program test;

Uses tableau ;

begin
  initialisation(T1);
  T1[Max]:=5;
End.
```

Langage Pascal - Les unités

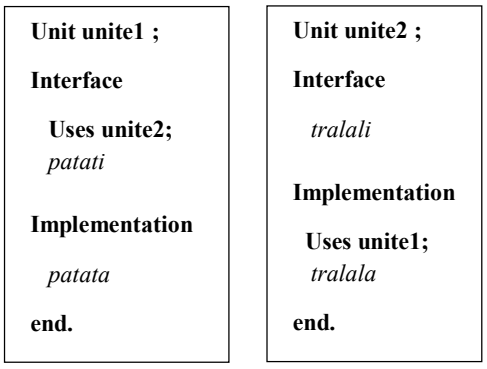
Les références circulaires à deux unités :



Application : deux fenêtres qui s'appellent

Langage Pascal - Les unités

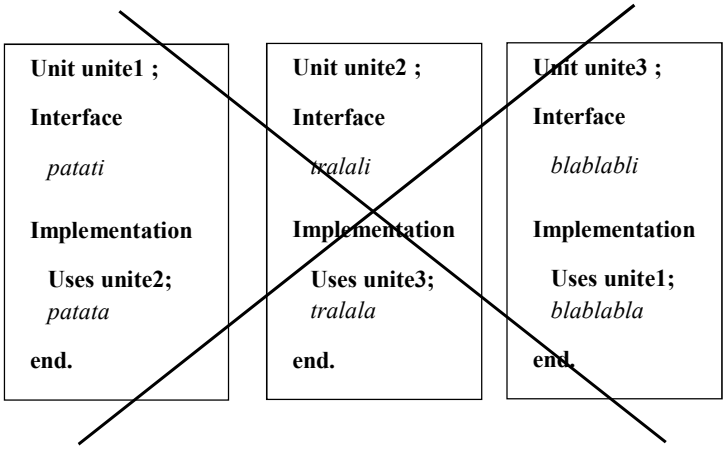
Les références circulaires à deux unités :



Application : deux fenêtres qui s'appellent

Langage Pascal - Les unités

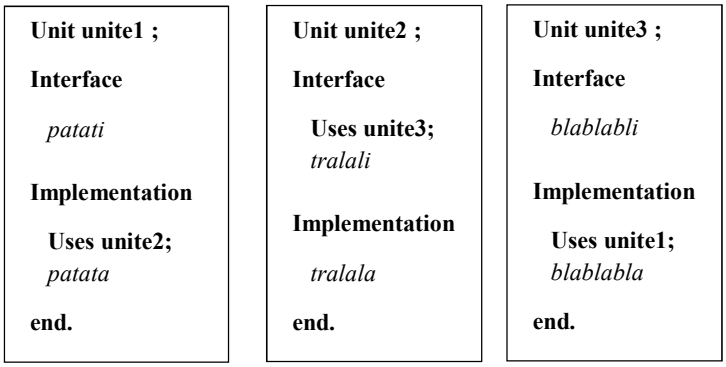
Les références circulaires à plus de deux unités :



Application : trois fenêtres qui s'appellent

Langage Pascal - Les unités

Les références circulaires à plus de deux unités :

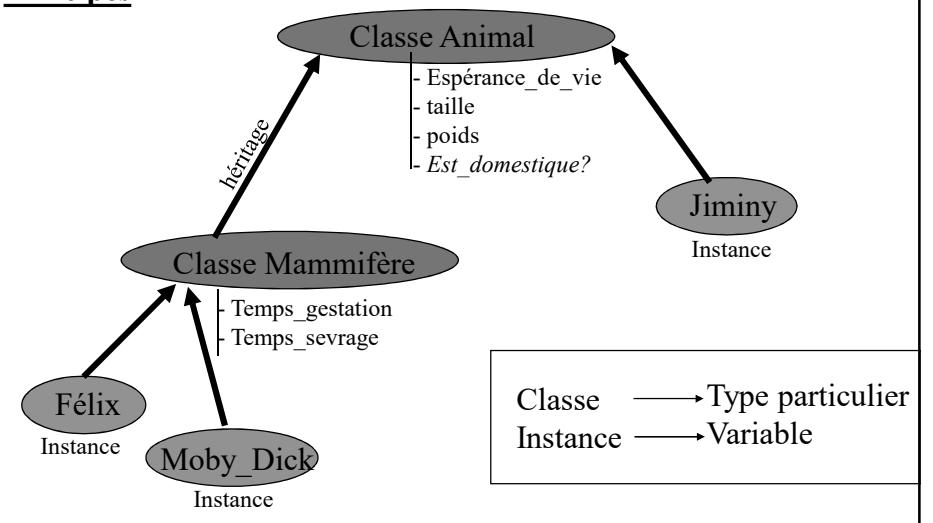


Application : trois fenêtres qui s'appellent

Langage Pascal Objet

Langage Pascal Objet

Principes



Langage Pascal Objet

Une Classe

Type
UneClasse **= class (***ClasseMere***)**
Private
< Accessible uniquement aux méthodes de la classe >
Protected
< Accessible aux méthodes de la classe et de ses descendants >
Public
< Accessible à tout le code >
Published
< Accessible à tout le code et depuis l'inspecteur d'objets >
Automated
< Accessible à tout le code. Les informations d'automatisation sont générées >
end;

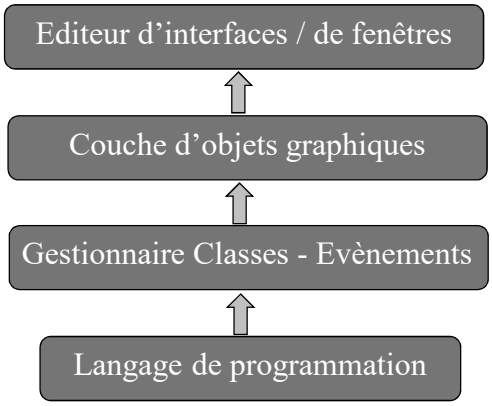
Langage Pascal Objet

Exemple

```
Var Felix, Moby_dick : Mammifere ;
    Jiminy, autre : Animal ;
    b : boolean ;
…
autre:=Animal.create;

Felix.Temps_gestation := 12 ;
b:=Felix.Est_domestique? ;

Moby_dick.taille := 40;
Moby_dick.poids := 1000;
Moby_dick.Temps_gestation := 10;
Moby_dick.Temps_sevrage := 4;

Jiminy.Esperance_de_vie := 2;
Jiminy.Temps_gestation := 0.1;
...
```

```
...
Type
 Animal = class (TObject)
   Protected
        Esperance_de_vie : integer;
        taille : integer;
        poids : integer;
   Public
        function Est_domestique? : boolean;
 end;

Mammifere = class (Animal)
   Public
        Temps_gestation : integer;
        Temps_sevrage : integer;
    end;

….
Function Animal.Est_domestique? : boolean;
begin … end;
```

IDE (Editeurs d'Interface)



IDE (Editeurs d'Interface)

Démonstrations