UNIVERSITÉ FRANÇOIS - RABELAIS TOURS

UFR Sciences et Techniques
Licence S&T 1ère année

Informatique — UE 102

Architecture des ordinateurs et Algorithmique de base

Jean-Yves Antoine

http://www.info.univ-tours.fr/~antoine/

UNIVERSITÉ FRANÇOIS - RABELAIS TOURS

UFR Sciences et Techniques

Licence S&T 1ère année

Informatique UE 102

Chap III – Bases du langage Pascal

Où l'on découvre la syntaxe générale du langage avant de faire de l'algorithmique

LANGAGE PASCAL

- ✓ **Créateur** : Niklaus Wirth (1969)
- ✓ Langage impératif de haut-niveau
- ✓ Langage compilé
- ✓ **Syntaxe normalisée** : ISO 7185 (1990) et 10206 (1991)
- ✓ La norme laisse cependant la place à des enrichissements : langage légèrement différent suivant le compilateur utilisé

Licence 1 (UE 102)

- ✓ Turbo Pascal Borland V7.0
- ✓ Compilateur libre de droit
- ✓ Pascal Objet

PROGRAMME PASCAL

OSSATURE GENERALE

```
program nom_prog;


begin


end.
```

Zone des déclarations

constantes
variables
types
fonctions
procédures

Zone des instructions

PASCAL : CONSTANTES ET VARIABLES

CONSTANTES

Identificateur (étiquette) servant à désigner une valeur immuable dans tout le programme

Intérêt (génie logiciel)

- ✓ nom explicite : programme plus lisible
- ✓ un seul changement dans tout le programme si on veut changer la valeur de la constante

Déclaration en Pascal

```
program nom_prog;

const (* les constantes se declarent en premier *)
        cours_euro = 6.559;
        taux_CSG = 9.50;
```

PASCAL : CONSTANTES ET VARIABLES

VARIABLES (Types Simples)

Identificateur qui désigne une zone mémoire qui pourra prendre des valeurs variables au cours de l'exécution du programme.

Types codage différent en mémoire suivant le type de données

✓ Entier relatif	`integer`	`+23, 23, -14`
✓ Réel	`real`	`5.0, -14.3` `0.123E2 = 123E-1 = 12.3`
✓ Booléen	`boolean`	`true, false`
✓ Caractère	`char`	`'a','A','1','+'`

Déclaration en Pascal

```
var (* les variables se declarent après les const *)
        prix, revenu : integer;
        remise : real;
        solde : boolean;
```

PASCAL : INSTRUCTIONS

INSTRUCTION se termine toujours par un point virgule

AFFECTATION symbole **:=**

```
prix := 3 ;    remise := -3.5 ;    solde := true;
```

OPERATIONS ARITHMETIQUES (entiers, réels)

opérateurs

- ✓ Addition — `y := x + 3;`
- ✓ Soustraction — `x := x - 3;`
- ✓ Multiplication — `x := y * x;`
- ✓ Division euclidienne — `x := y / 2;`
- ✓ Division entière — `x := y div 2;`
- ✓ Modulo — `x := y mod 5;`

fonctions

- ✓ Partie entière — `x := round(y);`
- ✓ Racine carrée — `x := sqrt(y);`

Parenthèsage idem arithmétique + associativité à gauche

PASCAL : INSTRUCTIONS

OPERATIONS LOGIQUES

- ✓ Disjonction (addition booléenne) `y := x OR y;`
- ✓ Conjonction (multiplication booléenne) `y := x AND true;`
- ✓ Négation (complémentation) `y := NOT x;`

EXPRESSIONS LOGIQUES

Test sur tout type de données pour donner un résultat booléen

```
x < y    x <= 3      y = 'B'      y >= 1      y <> 0
```

OPERATIONS SUR LES CARACTERES

- ✓ caractère → code ASCII `code := ORD('c');`
- ✓ Code ASCII → caractère `c := CHR(code);`
- ✓ `pred`, `succ` caractère précédent ou suivant, au sens ASCII

PASCAL : ENTREES / SORTIES

LECTURE CARACTERE SAISI AU CLAVIER

readln : lecture au clavier jusqu'au prochain retour chariot (Enter) et affectation dans une variable donnée en argument. Erreur si type saisi incorrect

affectation implicite : pas de symbole d'affectation

```
readln(car);                    readln(c1,c2,c3) ;
```

AFFICHAGE A L'ECRAN

write : affichage à l'écran de la valeur de la variable en argument

writeln : affichage avec retour à la ligne pour l'affichage suivant

```
writeln(prix);                  writeln(p1,p2,p3);
write('exemple d''affichage de message');
writeln('idem avec l''apostrophe');
```

PASCAL : UN EXEMPLE DE PROGRAMME

```
(* programme convertissant un prix en euro *)
program convertisseur_euro ;
    const taux = 6.559 ;
    var prix_init, prix_convert : real;
        a : char;
begin
    write(´entrez le prix à convertir :´);
    readln(prix_init);
    prix_convert := prix_init / taux;
    write(´le prix en euros est de : ´);
    writeln(prix_convert);
    writeln('entrez n''importe quelle touche pour terminer');
    readln(a);
    writeln('--------------------------');
end.
```

GENIE LOGICIEL rendre le code aussi lisible que possible

- Nom des variables, constantes aussi explicite que possible
- Jouer sur les décalages sur chaque ligne
- Partitionner le code à l'aide de commentaires
- Commenter si nécessaire les parties complexes

PASCAL : TYPES ENUMERES

TYPE INTERVALLE

- ✓ Type limité à un sous-domaine (plage de valeur) d'un type simple
- ✓ Fonctions du type simple toujours applicables au type intervalle

```
var intervalle_minuscules : 'a' .. 'z' ;
```

```
var intervalle_entiers : 1 .. 10 ;
```

TYPE SCALAIRE

- ✓ Sous-domaine défini par une énumérations de valeurs

```
var chiffres_pair : (0,2,4,6,8);
```

```
var jour : ('lu','ma','me','je','ve','sa','di');
```

- ✓ Restrictions d'usage : pas de read(ln) / write(ln) sur les scalaires

PASCAL : TYPES STRUCTURES

TYPE TABLEAU (*array*)

- ✓ Ensemble de valeurs d'un même type en nombre fixe connu a priori
- ✓ Les valeurs peuvent être d'un type simple ou structuré

```
var tab1 : array [0..10] of integer ;
var tab2 : array [1..25] of char ;
var tab3 : array ['a'..'z'] of real ;
```

- ✓ Travail sur l'ensemble du tableau ou sur un de ces éléments
- ✓ Accès à une valeur particulière du tableau par son indice

```
tab1[0] := 4 ;
tab2 := tab1 ; (* si type et dimension compatibles*)
```

- ✓ Exemple : somme des éléments d'un tableau de 10 entiers

PASCAL : TYPES STRUCTURES

TABLEAU A PLUSIEURS DIMENSIONS

- ✓ Les valeurs peuvent être de type structuré

```
var tab2D : array [0..10] of
              array[0..10] of integer ;
```

- ✓ Notation simplifiée : tableau à N dimension

```
var tab2D : array [0..10,0..10] of integer ;
```

```
var tab3D : array [0..9,1..10,'a'..'j'] of char;
```

- ✓ Accès à un élément particulier

```
tab2D[0,0] := 1;
tab3D[0,1,'a'] := 'r';
```

TYPE ENREGISTREMENT (*record*)

- ✓ Etudié au second semestre
- ✓ Cf. type *struct* du langage C

PASCAL : TYPES STRUCTURES

CHAÎNE DE CARACTERES (type STRING)

- ✓ Tableau de plusieurs caractères

```
var chainechar : string [200] ;
```

```
var chainechar : array [1..200] of char ;
```

- ✓ Travail sur l'ensemble de la chaîne ou un de ses élément

```
var ch1,ch2 : string [200] ;
begin
    ch2 := '';          (* chaîne vide *)
    read(ch1);
    ch1 := 'exemple d''affectation directe';
    ch2 := ch1;
    writeln(ch1[1]) ; writeln(ch2);
end.
```

- ✓ Longueur maximale ≠ longueur effective lgr := length(ch1) ;

PASCAL : TYPES STRUCTURES

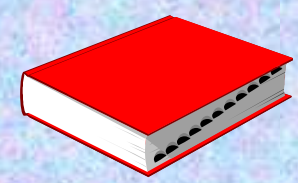
MANIPULATION DE CHAÎNES DE CARACTERES

- ✓ length(str) retourne la longueur effective de la chaîne *str*
- ✓ concat(s1,s2) retourne la chaîne obtenue par concaténation de *s1* et *s2*
- ✓ insert(s_insert,s_init,pos)

  modifie la chaîne *s_init* par insertion de *s_insert* dans la chaîne *s_init* à partir de la position *pos*
- ✓ delete(s_init,N,pos)

  modifie la chaîne *s_init* par suppression d'un nombre *N* de caractère à partir de la position *pos*
- ✓ copy(s_init,N,pos)

  retourne la sous-chaîne chaînede *s_init* de longueur N commençant à partir de la position *pos*
- ✓ pos(s1,s2) retourne la position initiale de la la sous-chaîne s2 dans s1 et 0 si s2 n'est pas une sous-chaîne de s1



Bibliographie

Ouvrages généraux

A faire…

Cours sur la Toile

Supports du cours : www.sir.blois.univ-tours.fr/~antoine/enseignement/pascal