

# — Informatique — Licence de Mathématiques — Cours ——

# Mémento Pascal

Les définitions suivantes donnent une idée des commandes et déclarations les plus importantes (et non toutes les commandes et déclarations) en Pascal standard, c'est-à-dire celles communes aux pascals Sun, GNU et Free.

Les éléments entre < > appartiennent à la grammaire définissant le langage, et sont (généralement) définis par ::= suivi d'une liste de possibilités séparées par des |. Les éléments en gras appartiennent au langage lui-même. Un élément entre [] est optionnel,  $(e)^+$  désigne l'élément e répété une ou plusieurs fois, alors que  $(e)^*$  est équivalent à  $[(e)^+]$  (e est répété zéro, une ou plusieurs fois).

Les schémas syntaxiques qui suivent offrent une autre façon de présenter ces commandes et déclarations. Ils sont plus complets, mais me semblent aussi avoir quelques erreurs.

# Structure d'un programme

# Programme

# Corps du programme

#### Commentaires

Des commentaires peuvent être ajoutés où que ce soit.

```
< Commentaire > ::= (* < Texte > *) | { < Texte > }
```

#### **Identificateurs**

- < Identificateur > ::= < Caractère alphabétique > (< Caractère alpha-numérique > )\* < Caractère alphabétique > ::=  $\mathbf{a} \mid \dots \mid \mathbf{z} \mid \mathbf{A} \mid \dots \mid \mathbf{Z}$
- < Caractère alpha-numérique > ::= < Caractère alphabétique > | 0 | ... | 9 | \_\_

### Définitions et déclarations

#### Définition de constantes

```
< Définition de constantes > ::= < Identificateur > = < Expression constante >
```

## Définition de types

```
< Définition de type >
                           ::= < Identificateur >
                                                       = < Type >
< Type >
                           := < Type simple >
                                                           < Type construit >
                                < Identificateur >
< Type simple >
                           ::= < Type scalaire >
                                                            < Type intervalle >
                                < Type référence >
                                                            < Type chaîne >
< Type scalaire >
                           ::= < Type prédéfini >
                                                            < Type énuméré >
                                                           boolean char
< Type prédéfini >
                               integer | real
                           ::=
< Type énuméré >
                                                       (, < Identificateur >)^*)
                           ::= ( < Identificateur >
< Type intervalle >
                                < valeur inférieure >
                                                           < valeur supérieure >
                           ::=
< Type référence >
                               ^{\circ} < Type >
                           ::=
< Type construit >
                           ::=
                                < Type tableau >
                                                            < Type enregistrement >
                                array[ < Type intervalle > ( , < Type intervalle > )^* ]
< Type tableau >
                           ::=
                                \mathbf{of} < \mathrm{Type} >
< Type enregistrement >
                                record < Déclaration de variables >
                           ::=
                                        (; < Déclaration de variables > )* end
```

## Remarques:

- un identificateur ne peut être utilisé comme type qu'après avoir été définit comme tel ;
- bien que non standard, le type chaîne de caractères est disponible dans la plupart des Pascals;
- les valeurs inférieures et supérieures d'un intervalle sont des constantes correspondant à un type scalaire différent du type réel.

#### Définition de procédure ou de fonction

#### Déclaration de variables

```
< Déclaration de variables > ::= < Identificateur > (, < Identificateur > )*: < Type >
```

## Instructions

```
< Instruction > ::= | < Instruction composée > | < Instruction d'affectation > |
                    if < Expression > then < Instruction > [else < Instruction > ]
                    case < Expression > of < Instruction pour un cas >
                      (; < Instruction pour un cas >)^* end
                    while < Expression > do < Instruction > |
                    repeat < Instruction > (; < Instruction > )*
                      until < Expression >
                    for < Instruction d'affectation > [down]to < Expression > do
                      < Instruction >
                    < Appel de procédure >
                             ::= begin < Instruction > (; < Instruction > )* end
< Instruction composée >
                             ::= < Identificateur > := < Expression >
< Instruction d'affectation >
                             ::= else: < Instruction >
< Instruction pour un cas >
              < valeur constante > (, < valeur constante > )*: < Instruction >
< Appel de procédure >
              < Identificateur > [ ( < Expression > ( , < Expression > )* ) ]
```

## Remarques:

- l'expression utilisée dans un case doit être d'un type scalaire non réel;
- l'identificateur d'une instruction d'affectation doit correspondre à un identificateur utilisé dans une déclaration de variable ou, dans le corps d'une fonction, à l'identificateur de cette fonction;
- l'identificateur d'un appel de procédure doit correspondre à un identificateur utilisé dans une définition de procédure, et les expressions donnant les paramètres de cet appel doivent correspondent aux types indiqués lors de la définition de cette procédure.

# Expressions

#### Remarques:

- tout identificateur utilisé dans une expression doit avoir été utilisé soit dans une déclaration de variable, soit dans la définition d'une fonction et être alors suivi d'expressions correspondant aux types indiqués lors de la définition de cette fonction;
- chaque expression utilisée en paramètre d'un opérateur (unaire ou binaire) doit être d'un type correspondant à cet opérateur (en particulier, toute comparaison doit porter sur des types scalaires ou intervalle).