

4 - CONSTRUCTIONS DE BASE

Programmation Concurrente - LI330
Université P. & M. Curie - année scolaire 2013/2014

PrC

 **Texte libre n'étant pas traité par un compilateur ou un interpréteur**

-  **Fait partie du programme**
-  **Ne fait pas partie de l'algorithme**
-  **Indispensable à la lecture**

 **Ada**

- Ceci est un commentaire**
- sur deux lignes**

Caractère(s) ne pouvant être intégré(s)
dans un **identificateur** ou un **mot clef** du langage

Caractères de formatage

- Espace, tabulation, passage à la ligne

Délimiteurs

- Fin d'instruction (typiquement ";")
- Éléments de reconnaissance de structures (typiquement "(", ")"...)
- Caractères constituant des opérateurs (typiquement "=", "<", ...)

On les désignera au fur et à mesure...

Pas
d'utilisation de [

Constantes booléennes

False True

Constantes numériques entières ou réelles

10, 10.0

Constantes caractères ou chaînes de caractères

'c', "c", " voici une chaîne "

Constantes d'un type simple énuméré

 **Un type contient rouge, vert et bleu**

rouge

- **Permet de référencer une entité du programme**
 - Variable, type, constante, procédure, fonction, etc.
 - Respecte des règles de construction
- **Commence par une lettre, séparation des mots avec `_`, taille quelconque**
`Ceci_Est_Un_Identificateur`
- **Les identificateurs ne sont pas casse-sensitifs, par convention, la première lettre d'un mot est en majuscule**

Des opérations élémentaires

Affectation

- Recopier une zone mémoire (de taille identique)

ECRITURE

variable := expression

Comparaisons (fonctions de type booléen)

- Égalité, comparer deux zones mémoire de taille identique

ECRITURE

expression = expression

- Autres (<, >, ≤, ≥) la sémantique dépend du type

ECRITURE

expression op_cmp expression

Opérateurs numériques

- Addition, soustraction, division, multiplication, puissance, ...

Opérateurs booléens (fonctions de type booléen)

- and, or, not

ECRITURE

expression op_bool expression

Opérateurs numériques

`V * 10` -- pas de ; car dans une expression

`X + 5.0 * Y` -- fort typage => X et Y de type réel

`+, -, /, *, **, mod`

Comparaisons

`Variable_Entière = 10` -- pas de ; car dans une expression

`Variable_Flottante = 10.0`

`Variable_Caractère <= 'c'` -- `<=, <, >=, > /=`

Ada

`not Variable -- pas de ; car dans une expression`

`V1 or V2`

`(V1 and V2) and (V3 or V4) -- priorité du and sur le or`

`(V1 or else V2) and then V3 -- évaluation partielle possible`

Affectations

`Variable_Entière := 10 * Z;`

`Variable_Flottante := 10.0 + Y; -- noter ici le typage`

`Variable_Caractère := 'c';`