Notion de variable

1. Présentation des notions de variable et de type

Un algorithme manipule des **objets** sur lesquels il peut effectuer des **actions**. Les objets simples sont :

- des **nombres** (3.14159, 1980, 9...),
- des **caractères** ("A", "9"...) et des **chaînes de caractères** ("PAULINA"...),
- des valeurs **booléennes** ou **logiques** (vrai ou faux).

Pour pouvoir manipuler ces objets, des opérations sont disponibles. Ces objets ainsi que les opérations qui leur sont associées doivent être parfaitement définis. Analysons les trois exemples suivants qui sont des situations d'échange entre un client et un vendeur.

Exemple 1 :

- "Bonjour Monsieur l'épicier, je voudrais 1 kg et 1 kg."
- "??"

Exemple 2 :

- "Bonjour Monsieur l'épicier, je voudrais 1 kg de riz et 1 kg de vin."
- "Voici votre kilo de riz et que voulez-vous par ailleurs ?"

Exemple 3 :

- "Bonjour Monsieur l'épicier, je voudrais 1 kg de riz et 1 litre de vin."
- "Voilà je vous ai tout rangé dans un petit sac"

On constate sur ces exemples la nécessité :

- de **citer la nature des objets** que l'on veut manipuler (riz, vin...),
- de **ne pas utiliser ces objets n'importe comment** (on pèse du riz, on boit du vin...), c'est-à-dire qu'à chaque nature d'objet sont liées des opérations particulières.

On appelle **type** l'association :

- d'une **nature d'objets** (riz, vin, ou encore entiers, réels...),
- et des **opérations associées** (peser, laver, cuire le riz, additionner, multiplier des entiers...).

2. Types de base et opérations associées

On distingue quatre types de base : entier, réel, booléen, caractère.

Le type entier :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

- Les valeurs sont des nombres entiers.
- Notation : la notation décimale est utilisée (Exemples : 365, -15).
- Opérations : addition, soustraction, multiplication, division entière, modulo (reste de la division entière)...

Le type réel :

- Les valeurs sont des nombres réels.
- Notation : le point anglo-saxon remplace la virgule (Ex : 124.89, 0.136, 1986).
- Opérations : celles utilisées usuellement sur les réels en arithmétique.

Le type booléen :

- Il n'existe que deux valeurs booléennes signifiant vrai ou faux.
- Notation : Vrai, Faux.
- Opérations : on utilise les opérateurs logiques usuels (Non, Ou, Et).

Rappelons le fonctionnement des opérateurs logiques Non, Et et Ou au travers de tableaux de synthèse :

A	Non A
Vrai	Faux
Faux	Vrai

A	B	A Ou B
Vrai	Faux	Vrai
Faux	Vrai	Vrai
Faux	Faux	Faux
Vrai	Vrai	Vrai

A	B	A Et B
Vrai	Faux	Faux
Faux	Vrai	Faux
Faux	Faux	Faux
Vrai	Vrai	Vrai

En résumé :

- Le contraire de Faux est Vrai et inversement (cf. tableau du Non).
- Avec le Ou logique, il suffit que l'un des opérandes (A, B) soit Vrai pour que le résultat A Ou B soit Vrai (cf. tableau du Ou).
- Avec le Et logique, il faut que les deux opérandes (A, B) soient simultanément Vrai pour que le résultat A Et B soit Vrai (cf. tableau du Et).

3. Intérêt des types

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

Faisons ici un petit effort d'abstraction en imaginant un langage (de communication) basé exclusivement sur deux types.

La définition de ces types (nature des objets et opérations associées) se veut volontairement proche d'exemple de la "vraie vie". Ce langage permettra de manipuler des objets de type "Solide" et de type "Liquide".

Caractérisons ces deux types comme suit :

Solide :

- Nature : matière ayant une forme propre
- Opérations : fondre, manger

Liquide :

- Nature : matière tendant à s'écouler
- Opérations : bouillir, boire, vider

et écrivons un premier algorithme :

Début

Solide : beurre, gruyère, caoutchouc

Liquide : huile, eau, vin, coca

boire huile, ***fondre*** beurre, ***fondre*** vin, ***manger*** caoutchouc, ***boire*** cidre, ***cuire*** caoutchouc

Fin

Deux avantages peuvent être perçus à l'utilisation de types dans notre "langage" :

- Pour décrire des objets de même nature admettant les mêmes opérations, il suffit d'avoir décrit le type une fois pour toutes et d'annoncer que tel ou tel objet est de tel ou tel type. **Les répétitions sont ainsi évitées**.
- Le typage (utilisation des types) fournit un moyen pour détecter un certain nombre d'erreurs sans exécuter l'algorithme par simple examen des opérations par rapport au type annoncé.

Sans effectuer les actions décrites dans cet algorithme, on s'aperçoit que l'opération *fondre vin* est erronée car le vin est un objet déclaré de type *Liquide* et l'opération *fondre* n'est pas associée à ce type. L'opération *manger caoutchouc* est correcte bien que le caoutchouc soit indigeste. Enfin *boire cidre* n'est pas possible dans la mesure où *cidre* n'a pas été cité comme étant un *Liquide*. Enfin, il est inutile de déclarer l'*eau* et le *coca* dans la mesure où ils ne sont pas employés par la suite.

4. Utilisation des variables dans des expressions

La manière la plus courante de manipuler des objets est de les faire intervenir dans des calculs. On utilise à cet effet des opérateurs élémentaires à un opérande (opérateurs unaires) ou à deux opérandes (opérateurs binaires).

Une **expression est l'association d'opérateurs et d'opérandes**. Des règles régissent l'ordre d'évaluation des différents termes d'une expression :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

- Une opération est à évaluer d'abord si sa priorité est plus forte que celle des opérations adjacentes (exemple : le calcul de 3+5/2 commence par 5/2).
- En cas d'égalité de priorité, l'évaluation a lieu de gauche à droite (exemple : 3+2-5 s'évalue en commençant par 3+2).
- On peut toujours utiliser des parenthèses pour forcer l'ordre d'évaluation (exemple : 3*(2-5) s'évalue en commençant par 2-5).

5. Tableau récapitulatif des opérateurs

Le tableau ci-après liste les opérateurs utilisables en algorithmique ainsi que la priorité existant entre ces opérateurs lors de l'évaluation d'expressions.

Bien évidemment ces opérateurs seront pratiquement toujours retrouvés (avec les mêmes notations) dans les langages de programmation. Nous reviendrons plus tard dans ce livre sur ceux spécifiques à JavaScript.

Opérateur	Notation	Type des opérandes	Type du résultat
+ et - unaires Négation logique	+ - Non	Entier ou Réel Booléen	Celui de l'opérande Booléen
Élévation à la puissance	**	Entier ou Réel	Entier ou Réel
Multiplication Division de réels Quotient d'entiers Reste (modulo)	* / Div Mod	Entier ou Réel Réel Entier Entier	Entier ou Réel Réel Entier Entier
Addition Soustraction	+ -	Entier ou Réel Entier ou Réel	Entier ou Réel Entier ou Réel
Comparaisons	>, <, >=, <=, =, <>	Tout type	Booléen
Et logique	Et	Booléen	Booléen
Ou logique	Ou	Booléen	Booléen

Les **opérateurs sont classés par ordre de priorité décroissant** dans ce tableau (le Non est par exemple prioritaire sur le Ou).

L'ordre des caractères est régi par la table ASCII. Dans cette table chaque caractère occupe une position numérique, le rang 65 pour la lettre "A", le rang 97 pour la lettre "a", ce qui revient à considérer que "a" est plus grand que "A" !

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare