Avec du « J » (comme Joie !)
par Robert Coquidé

Il y a ... quelques années ..., j'étais enseignant à Corbeil-Essonnes, chez IBM. Monsieur Leborgne, de retour des Etats Unis (c'était , je crois, en 1966 ou 67), fit une « démo. » d'APL sur une « 72CMC » (sorte de grosse machine à écrire reliée par téléphone à un ordinateur situé à Orléans pour laquelle l'expression « Retour Chariot » gardait un sens précis, bruyant ... et pesant !). Ce fut une révélation. J'ai ensuite pratiqué et enseigné l'APL sur 1130, 360, 370, 5100, PC ; les versions d'APL se nommaient APL, APL\SV, VS-APL, APL2, APL-PC, APL2-PC. Plus par obligation que par goût j'ai pratiqué et enseigné le FORTRAN, le BASIC, le PL1, le PASCAL, le REXX, le C, le C++, le FORTH, le LISP.... Leur étude m'a permis d'apprécier de plus en plus APL....

On a dit que l'APL (interprété) est moins « rapide » que, par exemple, le FORTRAN (compilé). Cette remarque n'est pas fausse si l'on ne tient compte que du « temps machine ». Elle est pour le moins discutable si on considère le « temps bonhomme » (si l'on place un programmeur FORTRAN et un programmeur APL « au pied d'un mur », le premier « arrivé en haut » est généralement le programmeur APL). De plus, depuis 1965, le coût du « temps machine » diminue constamment, celui du « temps bonhomme » augmente !

On a dit que les programmes APL sont « illisibles ». Mais les personnes qui affirmaient cela avaient pratiqué le FORTRAN ou le PL1 pendant 10 ans, et l'APL pendant ... 10 minutes ! Je pense même qu'un « produit matriciel » écrit en APL est beaucoup plus lisible qu'en FORTRAN, C, PASCAL, BASIC .... Qui n'a pas eu entre les mains des programmes PASCAL ou C parfaitement illisibles parce que mal commentés ?

On a dit que les programmes APL sont « non maintenables ». C'est vrai s'ils sont mal documentés. Mais c'est également vrai pour les programmes C, PASCAL ....

La «raison » de cette « bouderie » des Informaticiens (avec un I majuscule) est qu'il est aisé de passer du FORTRAN au BASIC, au PASCAL , au C (maintenant ++) ... ; l'esprit est le même : il faut perpétuellement réinventer la roue (utilisation de boucles imbriquées) ; des détails de syntaxe changent : ici il faut des parenthèses, là des crochets, ici des virgules, là des points virgules.... En APL, le mieux est d'éviter les boucles : l'expression « v ← v1 + v2 » programme 100 additions « en parallèle » si les vecteurs v1 et v2 ont 100 composantes ! Cela change trop les habitudes des Informaticiens (avec I majuscule)....

On a beaucoup dit que l'APL rebute les Informaticiens (avec un grand I) parce que des programmes pilotes de claviers, d'écrans, d'imprimantes, sont nécessaires pour utiliser les caractères spéciaux indispensables. Le langage J n'en a plus besoin.

Le langage J mérite totalement d'être qualifié de « langage » ! Les données numériques ou α-numériques entrantes (les arguments) ou sortantes (résultats), sont des « noms ». Les programmes, fonctions, sous-programmes ..., sont des « verbes ». Un opérateur qui modifie un verbe est un « adverbe ». Un opérateur malaxant 2 verbes pour en créer un 3^{e} est une « conjonction ». Ces verbes, adverbes, conjonctions peuvent être écrits « explicitement » (comme en APL traditionnel , en utilisant les arguments), ou « tacitement » (sans les utiliser). Ce résumé est beaucoup trop bref pour présenter toute la richesse du langage J. Voir à ce sujet les cours de J publiés par Monsieur M.Dumontier dans les « Nouvelles d'APL » et les articles de Monsieur S.Baron.

A la demande de Madame L.Lemagnen, voici 3 petites applications en J. Leur caractère « scolaire » sera peut être diversement apprécié ! Qui a dit que les enseignants sont des adultes parmi les enfants ... et des enfants parmi les adultes... ?

La première est une transposition d'un article de Monsieur G.Langlet, que j'ai le regret de n'avoir connu que beaucoup trop peu de temps, mais suffisamment pour pouvoir apprécier la profondeur de ses réflexions, la justesse de son jugement, l'étendue de ses connaissances, l'éclectisme de ses activités intellectuelles, ..., sa passion pour le « différent scan ».... Il est montré ici que cette opération, qui lui était chère, peut être utilisée avec le même bonheur en J qu'en APL traditionnel.

La deuxième est une application à la géométrie des possibilités du J dans le domaine des nombres complexes. Les propriétés mathématiques des barycentres sont exploitées systématiquement. L'écriture est proche de l'APL2 traditionnel, sans forme tacite.

La troisième traite des développement limités en séries de Mac-Laurin des fonctions mathématiques « usuelles » et des opérations classiques sur ceux-ci. Il est fait le plus souvent possible usage de l'écriture sous forme tacite propre au J.

Actuellement, je pressens toute la richesse potentielle cachée dans l'utilisation des adverbes et conjonctions ainsi que leur programmation tacite (« pro verbes » et « pro conjonctions »). Mais je ne les domine pas ... encore !

Je souhaite que le verbe « c. » (calcul des valeurs et vecteurs propres d'une matrice) ne reste pas une boîte vide, comme le « quad-slash » de APL2, pour pouvoir traiter aisément l'analyse des données multidimensionnelles....

Je voudrais terminer en exprimant tout le bien que je pense de Monsieur K.E.Iverson . En effet, je lui dois de nombreuses heures de bonheur (le mot n'est pas trop fort) à programmer, utiliser, enseigner,..., découvrir et admirer les

possibilités de l'APL. Sa vision de l'informatique et des mathématiques est unique. Je suis particulièrement sensible à l'« esthétique » qui se dégage d'une application APL bien pensée. Je retrouve actuellement, avec les découvertes des possibilités du J les « sensations » éprouvées il y a ... 30 ans en découvrant le langage APL. Grand « MERCI » à Monsieur Iverson pour son J (comme Joie)