Les caractères APL sont faciles à retenir

par Charles Hubert

Généralités

A la vue d'un algorithme écrit en APL, le profane le trouve incompréhensible à cause des caractères spéciaux dont il ignore la signification. En réalité on peut les mémoriser facilement en quelques jours de pratique ; on décrit ici des recettes mnémotechniques.

En APL toutes les fonctions ont la même syntaxe copiée sur la soustraction (-A ou B-A), il exécute toute instruction de droite à gauche et aucune fonction n'est prioritaire sur une autre. Les caractères spéciaux à APL désignent des fonctions ou des variables que d'autres langages représentent par des symboles réservés. Par exemple le minimum de a et b s'écrit

en langage C : min(a,b) en APL : a⌊b

et le plancher de a s'écrit

en langage C : floor(a) en APL : ⌊a

En APL les données sont des tableaux avec un nombre d'indices quelconque ; chaque case d'un tableau peut être un nombre, un caractère ou une boîte contenant un tableau.

Ces caractéristiques et d'autres permettent des algorithmes beaucoup plus courts en APL qu'en d'autres langages.

On notera une fonction

R ← fonction D ou R ← G fonction D

D = argument droit

G = argument gauche s'il existe

R = résultat s'il y en a un

Les caractères spéciaux sont ou des idéogrammes simples, ou dérivés de caractères ordinaires un peu suggestifs. Certains caractères sont des superpositions de deux caractères plus simples ; par exemple "⍜" est la superposition de "○" et "\", ce qu'on indiquera par

"○" puis +"\".

Sur les claviers modernes on les obtient directement au moyen de Alt ou Alt+Maj.

Caractères ne représentant pas des fonctions

⍝ à partir de ce caractère la fin de la ligne est un commentaire : une lampe "○" en haut d'un poteau +"∩" éclaire la route.

( ) les parenthèses enferment une évaluation prioritaire ; comme dans d'autres langages.

' deux apostrophes enferment une chaîne de

caractères ; comme dans d'autres langages.

Δ ⍙ _ aux 26 majuscules et 26 minuscules, APL ajoute le delta majuscule, et le delta majuscule souligné qu'il aurait dû remplacer par un delta minuscule, comme il l'a fait pour les 26 majuscules soulignées d'autrefois ; APL autorise ces deux deltas et le trait bas "_" dans les noms symboliques.

. dans une constante numérique le point sépare les parties entière et fractionnaire ; comme dans d'autres langages.

¯ le moins haut "¯" indique qu'une constante numérique est négative, tandis que le moins habituel "-" indique une soustraction.

⍬ vecteur vide de nombres : prenons un simple zéro "0" et rayons-le +"~", le reste est vide.

: séparateur indiquant que le symbole qui est à sa gauche est l'étiquette d'une instruction : comme dans d'autres langages.

⋄ ce losange sépare des instructions sur une même ligne ; d'autres langages utilisent ";".

∇ début et fin du texte d'une fonction.

⍢ début et fin du texte d'une fonction "∇" verrouillée +"~".

Les fonctions qui calculent

+ - addition, soustraction.

× ÷ multiplication, division ; comme sur les calculettes.

⋆ puissance, exponentielle : complication de "×".

⍟ logarithme : inverser (retourner) "○" l'exponentielle +"⋆".

! factorielle (ou coefficient du binôme) ; APL écrit !n et non n!

⌹ inversion ou division "÷" matricielle +"⌷".

○ fonctions circulaires (sin, cos, ...) et par prolongement fonctions hyperboliques : ce caractère est un cercle, ce n'est ni une lettre ni un chiffre.

| module (valeur absolue) si G est absent, ou modulo G ; APL écrit |x et non ⎪x⎪.

⌈ plafond ou maximum : l'ergot de "⌈" est en haut.

⌊ plancher ou minimum : l'ergot de "⌊" est en bas.

? tirage aléatoire : on ne peut pas prévoir la valeur de R "?".

/ réduire si G est une fonction : +/ pour la somme des valeurs de D, ×/ pour leur produit, etc... la dimension de R est réduite par rapport à celle de D "⌿/".

⌿ réduire sur le premier indice : les lignes "-"+"/" pour une matrice.

\ balayer : propager la fonction G sur D "⌿\".

⍀ balayer sur le premier indice : les lignes "-"+"\" pour une matrice.

+.× produit (interne) matriciel : un autre langage pourrait écrire G.D pour le produit des matrices G et D ; mais APL précise que chaque terme de R est une somme "+" de produits "×", ce qui lui permet de généraliser avec d'autres fonctions comme "×.*" pour un produit de puissances, ou "×.-" pour un produit de différences, etc...

∘.× produit (externe ou tensoriel) de tous les couples possibles : "×" peut être remplacé par n'importe quelle fonction par exemple "∘.+" pour toutes les sommes, "∘.-" pour toutes les différences, etc... le petit cercle "∘" ne représente aucune fonction.

⍴ traduit D en base de numération G.

⊥ valeur représentée par D en base de numération G : "⍴" renversé.

⍕ traduit D "⍴" en tableau de caractères +"∘".

⍎ calcule (exécute) "⊥" les nombres ou l'instruction décrits par la chaîne de caractères +"∘" D : c'est "⍕" renversé.

Les fonctions logiques

Les valeurs logiques possibles sont les deux nombres
1 = vrai = oui 0 = faux = non
comme en langage C par exemple.

< ≤ = ≥ > ≠ comparaisons usuelles ; le caractère "=" n'indique pas une affectation de valeur mais répond à la question G et D sont-ils égaux ?

~ non logique ; échange les valeurs logiques 0 et 1.

∧ et logique.

∨ ou logique.

⍲ non-et "~"+"∧".

⍱ non-ou "~"+"∨".

≡ D et G sont-ils identiques ?

∈ chaque valeur de G est-elle un élément de D ?

⍷ occurence : "∈" étendu à une chaîne +"_" de valeurs dans G.

∈ vecteur de tous les éléments enfouis dans D si G est absent.

Les fonctions qui ne calculent pas

← affecte : envoie D dans G ; c'est "=" ou ":=' dans d'autres langages.

→ brancher ou aller à D.

↑ prend : soulève "↑".

↓ laisse : repose "↓".

⌽ rotation ou miroir : par exemple pour une matrice fait tourner "∘" les colonnes +"|" ou autour de la verticale +"|" sur le dernier indice.

⊖ rotation ou miroir : par exemple pour une matrice fait tourner "∘" les lignes +"-" ou autour de l'horizontale +"-" sur le premier indice.

⍉ transposer : par exemple pour une matrice fait tourner "⍥" autour de la diagonale +"\".

⍋ tri croissant : en plaçant D[R] en colonne "|" les grandes valeurs sont dans le bas de la liste +"⍙".

⍒ tri décroissant : en plaçant D[R] en colonne "|" les grandes valeurs sont dans le haut de la liste +"⍫".

/ comprimer si G est logique : la dimension de R est plus petite que celle de D "⍀/".

⌿ comprimer sur le premier indice : les lignes "-"+"/" pour une matrice.

\ dilater si G est logique : la dimension de R est plus grande que celle de D "⍀\".

⍀ dilater sur le premier indice : les lignes "-"+"\" pour une matrice.

⊂ emballer D dans une boîte : en lisant de droite à gauche on place quelque chose dans une boîte.

⊃ déballer les boîtes de D : en lisant de droite à gauche on sort quelque chose d'une boîte.

¨ appliquer la fonction dans chaque boîte de D et de l'éventuel G : le tréma "¨" c'est deux points qui suggèrent deux boîtes.

≡ profondeur de D : trois traits horizontaux suggèrent trois niveaux.

⍴ redimensionner (restructurer) D à la dimension de G en reprenant au début s'il n'y a pas assez de valeurs : "⍴" = "r grec".

⍴ si G est absent alors la dimension (structure) de D.

⍳ l'indice dans G de chaque valeur de D : "⍳" = iota = "i grec".

⍳ si G est absent alors R = la liste des D premières valeurs de n'importe quel indice.

[;] caractères enfermant les indices pour adresser dans un tableau ; comme dans d'autres langages.

⌷ fonction d'adressage dans D par les indices G : ce caractère est un "I" gras évidé.

, caténation (chaînage) de G avec D.

, transformation de D en un vecteur si G est absent : caténation de toutes les composantes de D.

⍪ caténation suivant le premier indice : pour des matrices caténation "," des lignes +"-".

⍞ exécute ce qui est tapé au clavier "⍞' si c'est D ; met en forme à l'écran "⍞" si c'est G.

⎕ saisit les caractères "'" entrés au clavier +"⍞" si c'est D ; affiche à l'écran "⍞" sans mise en forme +"'" si c'est G.

⎕ est aussi l'initiale de tous les symboles réservés par exemple "⎕IO" "⎕CR".