Le présent fichier PDF **Norme iso APL Sommaire** reproduit des extraits du document :

Programming Language APL, Extended

International Standards Organisation

Working Draft 2

Daté July 1990

Ce document est disponible sous le numéro **AC_23427-09** dans l'inventaire de la collection de l'ACONIT, Association pour un conservatoire de l'informatique et de la télématique, Grenoble.

Le numéro d'inventaire AC_23427 regroupe des documents en provenance de la bibliothèque personnelle de Gérard Langlet, un informaticien du CEA.

Ce PDF reproduit :

-la page de couverture du document normatif, en cours de rédaction,

-les deux pages de l'introduction qui indique que « APL stands for A Programming Language. It is a notation invented by K. E. Iverson in the late 1950s for the description of algorithmes, and expanded on and made into the programming system *APL\360* by Iverson and his colleagues Adin Falkoff, Larry Breed, Dick Lathwell, and Roger Moore in the mid-1960s. This document « Programming Language APL, Extended, is a sequel to Programming Language APL, ISO 8485 (1989) », et y sont listés les ajouts prévus pour « l'extension ».

-les 9 pages (i à ix) de table des matières (Contents)

-la page-16 listant les caractères utilisés dans le langage APL

-les 4 pages (73 à 76) listant les formules utilisables dans le langage APL et les pages où leur fonctionnement est décrit

Pour utiliser ce document, mentionnez, l'ISO, ainsi que la Base de Données ACONIT n° 23427-09.

Fiche rédigée le 21/12/2019 par Jean Ricodeau.

Programming Language APL, Extended
International Standards Organisation

Working Draft 2

Eugene McDonnell, Editor

9 July 1990

0

Introduction

APL stands for **A Programming Language**. It is a notation invented by K. E. Iverson in the late 1950s for the description of algorithms, and expanded on and made into the programming system *APL\360* by Iverson and his colleagues Adin Falkoff, Larry Breed, Dick Lathwell, and Roger Moore in the mid-1960s. This document, **Programming Language APL, Extended**, is a sequel to **Programming Language APL**, ISO 8485 (1989).

The principal differences that the reader will find here have to do with new features that have been added. These topics are:

without
gcd
lcm
duplicate
commute
table
first-axis-join
mixed arrays
overbar in names
underbar in names
replicate
character grades
unique
alpha as a name
omega as a name
ambivalent defined functions
event handling
transfer form
n-wise reduction
complex arithmetic
left
right

0. INTRODUCTION

function rank operator
defined operators
transfer form
component file system

An entry for each of these topics will be found in the index. Some new system commands have been added. Shared variable extensions have been added. Workspace Interchange Standard 2 is given, in which canonical representation vectors of type "E" are used to represent generalised arrays.

Contents

CONTENTS

List of Figures

List of Tables

Alphabetic Characters

A B C D E F G H I J K L M N O P Q R S T U V W
X Y Z
A B C D E F G H I J K L M N O P Q R S T U V W
X Y Z

Numeric Characters

0 1 2 3 4 5 6 7 8 9

Special Characters

⍺ alpha	⍫ del tilde	⍞ quote quad
↓ down arrow	Δ delta	⍴ rho
← left arrow	⍙ delta stile	; semicolon
→ right arrow	⍚ delta underbar	∪ down shoe
↑ up arrow	¨ diaeresis	⊂ left shoe
− bar	⋄ diamond	⊃ right shoe
blank	÷ divide	∩ up shoe
{ left brace	$ dollar sign	⍈ up shoe jot
} right brace	. dot	/ slash
[left bracket	∈ epsilon	\ back slash
] right bracket	= equal	⌿ slash bar
∨ down caret	≥ greater-than or equal	⍀ back slash bar
⍱ down caret tilde	⍳ iota	⋆ star
< left caret	∘ jot	| stile
> right caret	≤ less-than or equal	⌊ down stile
∧ up caret	× multiply	⌈ up stile
⍲ up caret tilde	≠ not equal	⊥ down tack
○ circle	¯ overbar	⍊ down tack jot
⍉ circle backslash	(left parenthesis	⊢ left tack
⊖ circle bar	) right parenthesis	⊣ right tack
⍟ circle star	+ plus	⊤ up tack
⌽ circle stile	⍞ quad	⍎ up tack jot
: colon	⌹ quad divide	~ tilde
, comma	? query	_ underbar
⍪ comma bar	⍰ diaeresis jot	⍨ diaeresis tilde
∇ del	' quote	⍵ omega
⍫ del stile	! quote dot	

Table 1: The Required Character Set

Quelques uns des « current-statement » et leur aspect (extrait de la page 49)

Identification	Content	Class
T01	'ABC'	simple-identifier
T02	'←'	primitive
T03	'FN'	simple-identifier
T04	'□'	distinguished-identifier
T05	'⍵'	primitive
T06	'['	primitive
T07	'1'	numeric-literal
T08	'+'	primitive
T09	'0'	numeric-literal
T10	']'	primitive
T11	'DEF'	simple-identifier
T12	'['	primitive
T13	'1'	numeric-literal
T14	';'	primitive
T15	'5 6'	numeric-literal
T16	']'	primitive
T17	'×'	primitive
T18	'3.45E4'	numeric-literal
T19	','	primitive
T20	'ρ'	primitive
T21	''ABC''	character-literal

Les 4 pages listant les formules existant dans le langage APL, leur syntaxe et les pages dans le document où l'on trouve décrit leur fonctionnement

Form	Operation Name	Page
Z ← + B	Conjugate	78
Z ← − B	Negative	78
Z ← × B	Direction	79
Z ← ÷ B	Reciprocal	79
Z ← ⌊ B	Floor	80
Z ← ⌈ B	Ceiling	80
Z ← ⋆ B	Exponential	81
Z ← ⍟ B	Natural Logarithm	81
Z ← | B	Magnitude	82
Z ← ! B	Factorial	82
Z ← ○ B	Pi times	83
Z ← ~ B	Not	84
Z ← A + B	Plus	85
Z ← A − B	Minus	85
Z ← A × B	Times	86
Z ← A ÷ B	Divide	86
Z ← A ⌈ B	Maximum	87
Z ← A ⌊ B	Minimum	87
Z ← A ⋆ B	Power	88
Z ← A ⍟ B	Logarithm	89
Z ← A | B	Residue	90
Z ← A ! B	Binomial	91
Z ← A ○ B	Circular Functions	92
Z ← A ∧ B	And/LCM	94
Z ← A ∨ B	Or/GCD	95
Z ← A ⍲ B	Nand	96
Z ← A ⍱ B	Nor	96
Z ← A = B	Equal	97
Z ← A < B	Less than	98
Z ← A ≤ B	Less than or equal to	99
Z ← A ≠ B	Not equal	100
Z ← A ≥ B	Greater than or equal to	101
Z ← A > B	Greater than	102
Z ← , B	Ravel	103
Z ← ⍴ B	Shape	104
Z ← ⍳ B	Index Generator	105
Z ← ⍪ B	Table	106
Z ← A ⍴ B	Reshape	106
Z ← A , B	Join	108
Z ← A ⍪ B	Join	108

Table 4: The Form Table

Form	Operation Name	Page
Z ← f/ B	Reduction	109
Z ← f/[K] B	Reduction	109
Z ← f⌿B	Reduction	109
Z ← f⌿[K] B	Reduction	109
Z ← f⍀B	Scan	112
Z ← f⍀[K] B	Scan	112
Z ← f⍁B	Scan	112
Z ← f⍁[K] B	Scan	112
Z ← N f / B	N-wise Reduction	114
Z ← N f /[K] B	N-wise Reduction	114
Z ← N f ⌿ B	N-wise Reduction	114
Z ← N f ⌿[K] B	N-wise Reduction	114
Z ← f⍨ B	Duplicate	117
Z ← A f⍨ B	Commute	118
Z ← A ∘ . f B	Outer Product	118
Z ← A f . g B	Inner Product	120
Z ← f⍤ y B	Rank	122
Z ← A f⍤ y B	Rank	123
Z ← ? B	Roll	125
Z ← ⍋ B	Grade Up	127
Z ← ⍒ B	Grade Down	129
Z ← ⌽ B	Reverse	130
Z ← ⊖ B	Reverse	130
Z ← ⌽[K] B	Reverse	130
Z ← ⊖[K] B	Reverse	130
Z ← ⍉ B	Monadic Transpose	131
Z ← ⌹ B	Matrix Inverse	132
Z ← ⍎ B	Execute	133
Z ← ∪ B	Unique	134
Z ← A ,[K] B	Join Along an Axis	134
Z ← A ⍳ B	Index of	137
Z ← A ∈ B	Member of	138
Z ← A ? B	Deal	139
Z ← A / B	Replicate	140
Z ← A ⌿ B	Replicate	140
Z ← A /[K] B	Replicate	140
Z ← A ⌿[K] B	Replicate	140
Z ← A \ B	Expand	142
Z ← A ⍀ B	Expand	142
Z ← A \[K] B	Expand	142
Z ← A ⍀[K] B	Expand	142

Table 4: (Continued)

Form	Operation Name	Page
Z ← A ⌽ B	Rotate	144
Z ← A ⊖ B	Rotate	144
Z ← A ⌽[K] B	Rotate	144
Z ← A ⊖[K] B	Rotate	144
Z ← A ⊥ B	Base Value	146
Z ← A ⊤ B	Representation	148
Z ← A ⍉ B	Dyadic Transpose	150
Z ← A ↑ B	Take	152
Z ← A ↓ B	Drop	153
Z ← A ⌹ B	Matrix Divide	154
Z ← A[I]	Indexed Reference	155
Z ← V[I] ←B	Indexed Assignment	156
Z ← A ~ B	Without	158
Z ← A ⊣ B	Left	158
Z ← A ⊢ B	Right	159
Z ← A ⍒ B	Character Grade Down	160
Z ← A ⍋ B	Character Grade Up	161
Z ← ⎕TS	Time Stamp	164
Z ← ⎕AV	Atomic Vector	165
Z ← ⎕LC	Line Counter	165
Z ← ⎕EM	Event Message	166
Z ← ⎕ET	Event Type	166
Z ← ⎕DL B	Delay	167
Z ← ⎕NC B	Name Class	168
Z ← ⎕EX B	Expunge	169
Z ← ⎕NL B	Name List	170
Z ← ⎕STOP B	Query Stop	171
Z ← ⎕TRACE B	Query Trace	172
⎕ES B	Monadic Event Simulation	173
Z ← A ⎕NL B	Name List	173
Z ← A ⎕STOP B	Set Stop	174
Z ← A ⎕TRACE B	Set Trace	175
Z ← A ⎕EA B	Execute Alternate	176
A ⎕ES B	Dyadic Event Simulation	177
Z ← A ⍁ ⎕TF B	Transfer Form	178
Z ← ⎕CT ← B	Comparison Tolerance	179
Z ← ⎕CT	Comparison Tolerance	179
Z ← ⎕RL ← B	Random Link	181
Z ← ⎕RL	Random Link	181
Z ← ⎕PP ← B	Print Precision	182
Z ← ⎕PP	Print Precision	182

Table 4: (Continued)

Form	Operation Name	Page
Z ← ⎕*IO* ← *B*	**Index Origin**	183
Z ← ⎕*IO*	**Index Origin**	183
Z ← ⎕*LX* ← *B*	**Latent Expression**	184
Z ← ⎕*LX*	**Latent Expression**	184
Z ← ⎕*LX*[*I*] ← *B*	**Latent Expression**	184
Z ← *DFN*	**Call-Defined-Function**	193
Z ← *DFN B*	**Call-Defined-Function**	193
Z ← *A DFN B*	**Call-Defined-Function**	193
Z ← **f** *DFN B*	**Call-Defined-Function**	193
Z ← *A* **f** *DFN B*	**Call-Defined-Function**	193
Z ← **f** *DFN* ⍺ *B*	**Call-Defined-Function**	193
Z ← *A* **f** *DFN* ⍺ *B*	**Call-Defined-Function**	193
Z ← ⎕*FX B*	**Function Fix**	196
Z ← ⎕*CR B*	**Character Representation**	197
Z ← *SHV*	**Shared Variable Reference**	210
Z ← *SHV* ← *B*	**Shared Variable Assignment**	211
Z ← *SHV*[*I*] ← *B*	**Shared Variable Indexed Assignment**	212
Z ← ⎕*SVC B*	**Shared Variable Access Control Inquiry**	212
Z ← ⎕*SVQ B*	**Shared Variable Query**	214
Z ← ⎕*SVO B*	**Shared Variable Degree of Coupling**	215
Z ← *A* ⎕*SVO B*	**Shared Variable Offer**	216
Z ← ⎕*SVR B*	**Shared Variable Retraction**	217
Z ← *A* ⎕*SVC B*	**Shared Variable Access Control Set**	218
Z ← ⎕*SVS B*	**Shared Variable State Inquiry**	219
⎕*SVE* ← *B*	**Shared Variable Event**	220
Z ← ⎕*SVE*	**Shared Variable Event**	220
Z ← ⍎ *B*	**Monadic Format**	226
Z ← *A* ⍎ *B*	**Dyadic Format**	230
Z ← ⎕	**Quad Input**	238
Z ← ⍞	**Quote Quad Input**	239
Z ← ⎕ ← *B*	**Quad Output**	239
Z ← ⍞ ← *B*	**Quote Quad Output**	240

Table 4: (Concluded)