



INTERNATIONAL STANDARD ISO/IEC 13211-1:1995 TECHNICAL CORRIGENDUM 3

Published 2017-07

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Programming languages — Prolog — Part 1: **General core**

TECHNICAL CORRIGENDUM 3

Technologies de l'information — Langages de programmation — Prolog —

Partie 1: Noyau général

RECTIFICATIF TECHNIQUE 3

Technical Corrigendum 3 to ISO/IEC 13211-1:1995 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

Allow explicitly extensions to options in 5.5 Extensions. Add new subclause

5.5.12 Options

A processor may support one or more additional options such as stream-options (7.10.2.11), close-options (7.10.2.12), read-options (7.10.3), and write-options (7.10.4) as an implementation specific feature. An invalid option E shall be associated with only two error conditions: an instantiation error when there is an instance (3.95) of E that is a valid option, and a domain error for the domain *optname_option* when there is no instance of E that is a valid option. Further, an instantiation error may occur in place of the domain error if a component of E is a variable, and an instantiated component is required.

NOTE — A valid option may be associated with other error conditions like 8.11.5.3 l and m.

In 6.2.1 Prolog text, add an optional layout text sequence to the fourth and last production

p text = [layout text sequence (* 6.4.1 *)] ;

In 7.1.6.3 a, Iterated-goal term, replace unifies with by has the form

a) If T has the form $\wedge(_, G)$ then ...

Add term-to-body conversion in 7.8.3.4 Examples. Replace first paragraph by:

Table 21 and 22 show the execution stack before and after executing the control construct *call(G)* with *goal* obtained from G in step 7.8.3.1 f via 7.6.2.

Replace in Table 22 G by goal .

$N + 1 \quad ((goal, N - 1), \quad \Sigma \quad nil$

In 7.10.3 Read-options list, clarify unification for variables/1, specify left-to-right traversal order in read-options variable_names/1 and singletons/1. Replace the three paragraphs by:

variables(Vars) — After inputting a term, Vars shall be unified with a list of the variables in the term input, in left-to-right traversal order.

variable_names(VN_list) — After inputting a term, VN_list shall be unified with a list of elements where: (1) each element is a term $A = V$, and (2) V is a named variable of the term, and (3) A is an atom whose name is the characters of V, and (4) there is exactly one element for each named variable, and (5) the elements appear in the order of the first occurrence of their variables V in the term input, in left-to-right traversal order.

singletons(VN_list) — After inputting a term, VN_list shall be unified with a list of elements where: (1) each element is a term $A = V$, and (2) V is a named variable which occurs only once in the term, and (3) A is an atom whose name is the characters of V , and (4) there is exactly one element for each named variable occurring only once, and (5) the elements appear in the order of the first occurrence of their variables V in the term input, in left-to-right traversal order.

Remove "ground" in 3.206, add write-option variable_names(VN_list) before numbervars(Bool) in analogy to the read-option. Replace "an non-negative" by "a non-negative" in paragraph numbervars(Bool). Add Note 2

7.10.4 Write-options list

variable_names(VN_list) — Each variable V is output as the sequence of characters defined by the syntax for the atom A iff a term $A = V$ is an element of the list VN_list. If more than one element applies, the leftmost is used. VN_list is a list of terms $A = T$ with A an atom and T any term, possibly a variable.

NOTE 2 — Many Prolog processors modified write option numbervars/1 to print arbitrary variable names. The write option variable_names/1 serves this purpose and avoids vulnerabilities.

In 7.10.5, add writing with variable_names/1, correct terminology and writing of {}, lists, extra round brackets. Add as first subclause a1, rename and replace subclause a by a2, e by e1, g by e2, add new subclause e3, replace subclauses f and g:

7.10.5 Writing a term

a1) If Term is a variable and there is an effective write-option variable_names(VN_list) and there is an element $A = \text{Term}$ of the list VN_list with A an atom, then A is output with effective write-option quoted(false).

a2) Else if Term is a variable, a character sequence representing that variable is output. The sequence begins with _ (underscore) and the remaining characters are implementation dependent. During the execution of write_term/3, the same character sequence is used for each occurrence of a particular variable and a different character sequence is used for each distinct variable.

e1) If Term has the form '\$VAR'(N) for some non-negative integer N , and there is an effective write-option numbervars(true), a variable name as defined in subclause 7.10.4 is output,