Information technology – Programming languages – Fortran

TECHNICAL CORRIGENDUM 2

Technical corrigendum 2 to international Standard ISO/IEC 1539-1:1997 (E) was prepared by Joint Technical Committee ISO/IEC JTC1, Information technology.

Subclause 1.5.1

Add these extra items at the end on 1.5.1:

(3) Earlier standards specified that if the second argument to MOD or MODULO was zero, the result was processor dependent. This standard specifies that the second argument shall not be zero.

(4) The PAD= specifier in the INQUIRE statement in this standard returns the value UNDEFINED if there is no connection or the connection is for unformatted input/output. The previous standard specified YES.

Subclause 4.4.1

Replace the second line of R429 by

\[
\text{R429a} \quad \text{pointer-initialization-expr} \quad \text{is} \quad \text{function-reference}
\]

Constraint: function-reference shall be a reference to the intrinsic function NULL with no arguments.
Subclause 4.4.4

Replace the paragraph that starts “Where a component” by

Where a component in the derived type is a pointer, the corresponding constructor expression shall be an allowable target for such a pointer in a pointer assignment statement (7.5.2).

Pages 45 and 46

Subclause 4.5

Add to end of the paragraph that follows the constraints: “The character length of an ac-value in an ac-implied-do whose iteration count is zero shall not depend on the value of the implied DO variable and shall not depend on the value of an expression that is not an initialization expression.”.

In the first line of page 46, replace “an ac-value sequence” with “a sequence of elements”.

Page 47

Subclause 5.1

Replace the second line of R505 by

```
or => pointer-initialization-expr
```

Page 48

Subclause 5.1

After line 26 of page 48, add:

Constraint: The object-name shall be the name of a data object.

Page 49

Subclause 5.1

In the paragraph that begins “If a length-selector”, replace “length-selector” by “char-len-param-value”.

In the paragraph that begins “If initialization”, replace “NULL( )” by “pointer-initialization-expr” twice.

Page 55

Subclause 5.1.2.4.3

In the fifth line from the bottom of page 55, delete “statement” and replace “in a” with “by”.

Page 62

Subclause 5.2.10

Replace the fifth line of R540 by

```
or pointer-initialization-expr
```

In the penultimate line of page 62, replace “NULL( )” by “pointer association status”.
Subclause 5.2.10
In lines 1, 7 and 10 of page 63, replace “NULL( )” by “pointer-initialization-expr” thrice.

Subclause 5.4
In the paragraph that begins “Any namelist-group-name”, replace “in more than one NAMELIST statement” by “more than once in the NAMELIST statements”.

Subclause 6.3.1.2
In item (1) in the list in clause 6.3.1.2, change “; it” to “. It shall not be supplied as an actual argument except to certain intrinsic inquiry functions. It”.

Subclause 7.1.4.1
In the last line of page 91, add “The optional argument shall also be present if the reference appears as an actual argument corresponding to a dummy argument with assumed character length.”.

Subclause 7.1.6.1
Add to end of list item (6) “where the argument is not of type character with a length that is assumed or defined by an expression that is not an initialization expression.”.

In the first line of the last paragraph of page 94, replace “for a type parameter” by “that depends on a type parameter”.

Replace the last sentence of page 94 by “The prior specification may be to the left of the inquiry function in the same statement, but shall not be within the same entity-decl.”.

Subclause 7.1.6.2
In the first line of the last paragraph of the subclause, replace “for a type parameter” by “that depends on a type parameter”.

Replace the second sentence of the last paragraph of the subclause by “The prior specification may be to the left of the inquiry function in the same statement, but shall not be within the same entity-decl.”.

Subclause 7.1.7
In the first line of page 97, replace “in an expression” with “of an intrinsic operation”.
In the fifth line of page 97, replace “all of its components” with “it”.
Subclause 7.5.3.2
In the first line of the paragraph following NOTE 7.48, delete “a WHERE statement or”; after the paragraph, add the new paragraph:

Upon execution of a WHERE statement that is part of a where-body-construct, the control mask is established to have the value \( m \text{.AND. mask-expr} \). The pending mask is not altered.

Subclause 9.2.1.3.1
At the end of the last paragraph of subclause 9.2.1.3.1 add “If a nonadvancing output statement leaves a file positioned within the current record and no further output statement is executed for the file before it is closed or a BACKSPACE, ENDFILE, or REWIND statement is executed for it, the file is positioned after the current record before the specified action is performed.”.

Subclause 9.4.4
In list item 6, delete “an error condition,” and change “, or” to “or”.
Add the following after list item (8):

If an error condition occurs during any of the above operations, execution of the data transfer statement terminates, any variable specified in an IOSTAT= specifier becomes defined, and the error condition is processed as described in 9.4.3.

Subclause 9.6.1.22
Replace the second sentence of the paragraph in section 9.6.1.22 with the following. “The scalar-default-char-variable in the PAD= specifier is assigned the value YES if the connection of the file to the unit included the PAD= specifier and its value was YES or if there was no PAD= specifier. If there is no connection or if the connection is not for formatted input/output, the scalar-default-char-variable is assigned the value UNDEFINED.”.

Subclause 9.7
After “statement” in the second line of the subclause, insert “or a STOP statement”.

Subclause 10.5.4.1.2
In the last line of the table in 10.5.4.1.2, change “.1” to “.0”.

Subclause 10.8
In the fifth line of page 175, replace “constant or” by “constant, optionally signed if integer or real, or”.
Subclause 10.8.1
Add the following as a new paragraph, just before NOTE 10.26

For the \( r*c \) form of an input value, the constant \( c \) is interpreted as a nondelimited character constant if the first list item corresponding to this value is of type default character, there is a nonblank character immediately after \( r^* \), and that character is not an apostrophe or a quotation mark; otherwise, \( c \) is interpreted as a literal constant.

Subclause 10.9
In the fifth line of the paragraph that starts “Each value is”, replace “constant and” by “constant, optionally signed if integer or real, and”.

Subclause 10.9.1.1
Replace the last sentence of subclause 10.9.1.1 by “In the input record, each object name or subobject designator may be preceded and followed by one or more optional blanks but shall not contain embedded blanks.”.

Subclause 12.3.2.1
In the fourth constraint following R1207 delete “and, if included, ... interface-stmt” and add: “If the end-interface-stmt includes generic-name, the interface-stmt shall specify the same generic-name. If the end-interface-stmt includes ASSIGNMENT(=), the interface-stmt shall specify ASSIGNMENT(=). If the end-interface-stmt includes OPERATOR(defined-operator), the interface-stmt shall specify the same defined-operator. If one defined-operator is .LT., .LE., .GT., .GE., .EQ., or .NE., the other is permitted to be the corresponding operator <, <=, >, >=, ==, or /=.”.

Replace the fifth constraint following R1207 with

Constraint: A procedure-name in a module-procedure-stmt shall not specify a procedure that is specified previously in any module-procedure-stmt in any accessible interface block with the same generic identifier.

Subclause 12.3.2.2
Add at the end of the first paragraph after R1208 “In an external subprogram, an EXTERNAL statement shall not specify the name of a procedure defined by the subprogram.”.

Subclause 12.3.2.3
Replace lines 7-9 of page 198 by

If a specific intrinsic function (13.13) is used as an actual argument, it shall have been explicitly declared to have the INTRINSIC attribute.
Subclause 12.4.1.5
Replace the first sentence of 12.4.1.5 by

A dummy argument or an entity that is host associated with a dummy argument is not present if the dummy argument

(1) is not associated with an actual argument, or
(2) is associated with an actual argument that is not present.
Otherwise, it is present.

Subclause 12.4.3
Replace the final sentence of subclause 12.4.3 by “A reference to an elemental subroutine (12.7) is an elemental reference if there is at least one actual argument corresponding to an INTENT(OUT) or INTENT(INOUT) dummy argument, all such actual arguments are arrays, and all actual arguments are conformable.”.

Subclause 12.5.2.2
Before “.” in the first constraint after R1220, insert “and shall not be the same as the entry-name in any ENTRY statement in the subprogram”.

In the third line of the final paragraph of page 207, change “are recursive function references” to “refer to the function itself”.

Subclause 12.5.2.5
In the last constraint after R1225, change “entry-name” to “the function-name in the FUNCTION statement and shall not be the same as the entry-name in any ENTRY statement in the subprogram.”

Subclause 12.6
On the penultimate line of page 212, change “assignment-stmt” to “intrinsic assignment statement”.

Subclause 12.7.2
In line 2 of subclause 12.7.2, after “If” insert “there are no actual arguments or”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”.

Subclause 12.7.3
In the second line of the final paragraph of subclause 12.7.3, after “may be the same variable”, add “and may be associated scalar variables or associated array variables all of whose corresponding elements are associated”. 
Subclause 13.6

Add new sentence to the end of the paragraph: “If the MOLD argument to this function is a variable, it need not be defined.”

Subclause 13.14.10

In subclause 13.14.10, replace the Result Value paragraph by

Result Value. The result is the integer nearest A, or if there are two integers equally near A, the result is whichever such integer has the greater magnitude.

Subclause 13.14.73

Append to line 5 of 13.14.73 “P shall not be zero.”.
In line 7 of 13.14.73, change “If P ≠ 0, the” to “The”.
In lines 7-8 of 13.14.73, delete “If P = 0, the result is processor dependent.”.

Subclause 13.14.74

Append to line 5 of 13.14.74 “P shall not be zero.”.

Subclause 13.14.74

In line 2 of page 258, delete “If P ≠ 0,.”.
In line 4 of page 258, delete “If P = 0, the result is processor dependent.”.
In line 5 of page 258, change “If P ≠ 0, the” to “The”.
In lines 5-6 of page 258, delete “If P = 0, the result is processor dependent.”.

Subclause 13.14.75

In the second line of the paragraph that defines the effect of TO, change “and may be the same variable as FROM” to “and may be associated with FROM (12.7.3)”.

Subclause 13.14.77

In subclause 13.14.77, replace the Result Value paragraph by

Result Value. The result is the integer nearest A, or if there are two integers equally near A, the result is whichever such integer has the greater magnitude.

Subclause 13.14.110

On line 4 of page 272, change “undefined” to “processor dependent”.
On line 17 of page 272, change “undefined” to “processor dependent”.
Subclause 14.1.2.5
In lines 2-3 of 14.1.2.5, replace “the type is accessible ... 14.6.1.3)” by “an entity of the type is accessible in another scoping unit”.

Subclause 14.7.1
Replace item (1) with

(1) An array is defined if and only if all of its elements are defined.
(2) A derived-type scalar object is defined if and only if all of its nonpointer components are defined.
(3) A complex or character scalar object is defined if and only if all of its subobjects are defined.

Renumber item (2) as item (4).

Annex A
Delete the glossary entry for present.

Subclause C.6.2
In the third line of the example code that starts with the line

    CHARACTER (LEN = 20) CH1

replace “SCRATCH” with “OLD”.

Subclause C.6.5
Change the entry for PAD in the table to:

<table>
<thead>
<tr>
<th>PAD=</th>
<th>UNDEFINED</th>
<th>YES, NO, or UNDEFINED</th>
<th>UNDEFINED</th>
</tr>
</thead>
</table>

Page 280

Page 288

Page 299

Page 322

Page 325