Subject: Comments on Section 4
From: Van Snyder

1 Edits

Edits refer to 02-007. Page and line numbers are displayed in the margin. Absent other instructions, a page and line number or line number range implies all of the indicated text is to be replaced by associated text, while a page and line number followed by + (-) indicates that associated text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

[What good does Note 4.1 do? Editor: Delete Note 4.1.] 31:8+ff

---

[What's the point of "in this standard?" Why so wordy?] 31:20-21
The syntax for literal constants of each intrinsic data type is specified in 4.4.

C402½ (R401) An asterisk may be used as a *type-param-value* only in the declaration or allocation of a dummy argument. 33:2+

[LaTeX's pretty ‘, “, ’ and ” are nice for ordinary text, but don't seem quite right for BNF. Editor: Set them `\small` size in math mode as superscripts using {`\small $^\prime$`} and {`\small $^{\prime\prime}$`}, *viz.* ′ and ″.] 35:3-10

[It isn't the *digits* but the *constants* that are represented in "their respective number systems" (and it isn't specified what that means).] 35:18-19
The *hex-digits* A through F represent the numbers ten through fifteen, respectively; they may be represented by their lower-case equivalents. The binary, octal and hexadecimal constants are interpreted according to base two, base eight and base sixteen number systems, respectively.

[The paragraph at [36:1-2] is wrong (one can specify double precision with the REAL keyword), it duplicates [36:3-6], and it uses the term "double precision," which hasn't been defined yet. Editor: Delete [36:1-2]; at [36:3-4], "keyword" ⇒ "specifier" twice.] 36:1-4

[Same comments as for [35:3-10] above.] 38:20-21

[Sounds like there's a special character set for fixed form. Editor: Move "in fixed source form" to the beginning of the sentence, put a comma and "it is" after it, and adjust capitalization.] 38:28

[Sounds like there's a special character set for free form. Editor: Move "in free source form" to the beginning of the sentence, put a comma and "it is" after it, and adjust capitalization.] 38:28

[Editor: Insert a comma before "which".] 40:18

[Doesn't account for type parameters. Editor: "the name of the type" ⇒ "a *derived-type-spec* (R447)" at [41:4]; At [41:5] add a new sentence "A *derived-type-spec* is a type name optionally followed by type parameters in parentheses."] 41:4-5

[Unnecessarily split infinitive. Editor: "to explicitly" ⇒ "explicitly to".] 41:8

[Editor: Insert "(4.5.3.1)" after "component".] 47:4

[Editor: Insert "(5.1.2.5.1)" after "shape".] 48:4

[Eighth line of Note 4.29: We don't discuss "pointer to" anywhere else. Editor: "to ... holding" ⇒ "array of type CHARACTER named".] 49:7+9

[Line 9 of Note 4.30: Components of a type aren't accessible – it's components of *objects* of the type that are accessible. Editor: Insert "of objects of the type" before "are".] 50:19+10

[Doesn't cover the case of accessing two different types that have the same name. Editor: "A ... unit" ⇒ "Within a scoping unit, two different derived type definitions shall not be accessible by the same name".] 53:2

[Editor: "which" ⇒ "that".] 53:12

[Line 1 of Note 4.47 duplicates the normative text two lines above. Editor: Delete "is the name that ... This".] 54:5+2-3

[Note 4.49: Leaves out bindings and type parameters. Editor: Insert ", bindings, or parameters" after "components" twice.] 55:0+6-7

[Editor: "nongeneric" ⇒ "specific" for consistency with [44:24].] 55:6

[Unnecessarily wordy compared to other ones. Editor: Delete "procedure of the" and "that of".] 55:11

[Editor: Insert a comma after "overriding" in the first line of Note 4.52.] 56:0+2

[Editor: Insert "are said to" before "correspond".] 56:9

[Editor: "acording" ⇒ "according".] 57:22

[Awkward to say "the type parameters of the type parameters." It's also a bit coy, given that the only type parameter is the kind type parameter. Editor: "agrees ... of" ⇒ "of the same kind as".] 57:23

[Editor: "actual-arg-spec-list" ⇒ "`\si{actual-arg-spec}\st{-list}`", *viz.* "*actual-arg-spec-list*".] 58:2

[Editor: "nonintrinsic assignment" ⇒ "defined assignment (7.5.1.2)".] 59:12

[Too wordy. Editor: "If ... but" ⇒ "Otherwise if".] 59:18

[Too wordy. Editor: "If ... requirements" ⇒ "Otherwise".] 59:20-21

[Reads at first as though array components are finalized differently from array objects that aren't components. I hope this helps. Editor: Insert "being finalized" after "entity"; "element" ⇒ "each of its elements".] 59:22-23

[Doesn't account for type aliases. Editor: "or" ⇒ ","; Insert ", or a previously defined type alias" at the end.] 61:2-3

[Editor: In the first line of Note 4.63, "C" ⇒ "The C standard".] 62:0+7

[Doesn't account for asterisk. Now covered by new constraint at [33:2+] above (which does account for asterisk). Editor: Delete.] 63:25

2 Derived type constructors aren't finished

[Need to allow for procedure pointers.] 57:29+

or [*keyword* =] *proc-target*

[Editor: Insert "or *proc-target*" after '*expr*".] 58:4

[Editor: Insert "nonallocatable" after "nonpointer".] 58:8

[Rank remapping can't work during construction, so [59:1-2] isn't adequate. Editor: Add a new sentence "For nonprocedure pointer components, the rank of *expr* shall be the same as the rank of the component."] 58:9

[Editor: Insert "or *proc-target*" after "*expr*".] 58:10

[Editor: "constructor expression" ⇒ "*expr* or *proc-target*".] 58:17

[Editor: "object" ⇒ "entity".] 59:1

3 Proposed spec change

Revival of pseudo-elementalism. Sometimes you are perfectly happy to have the same finalization done on every element of an array of derived type, but you can't use an elemental procedure because you want to do something forbidden therein, such as I/O.

[Editor: After "argument." insert a new sentence: "Otherwise, if there is a final subroutine whose dummy argument is a scalar that has the same type and kind type parameters as the entity, it is invoked once for each element of the entity with that element as an actual argument; the elements are finalized in array element order."] 59:20

4 Questions for J3 to ponder

Is there a need to specify that all of the integers between the smallest and largest representable ones are representable? It would be undesirable if every integer from, say, −32767 to 32767 EXCEPT 42 were representable. 34:5

Do we want to specify any accuracy requirements? How about the same as for NEAREST rounding for I/O at [218:29-31]? 36:25-26

Does C507 apply? Probably at least somewhat, since a derived type definition is a scoping unit. But does it apply to prevent a component or binding to have the same name as a type parameter, or to prevent a binding to have the same name as a component? Does it cover the relation between inherited components or bindings and type parameter names? 42-44