

Public Review Comment #21

From: Nagle, Dan [dnagle@erols.com]
Sent: Monday, November 18, 2002 3:13 PM
To: Donovan, Deborah
Subject: public comment

<<File: My_PC.txt>>

Hi,

Here's my Public Comment on Fortran 2000.

TIA

--

Cheers!

Dan Nagle
Purple Sage Computing Solutions, Inc.

1. Allow the rule that "any keyword composed of two English words may be split into two English rules".
Add: "type alias" to the list at 38:2+
2. selected_char_kind()- How to exhaustively search for all kinds?
The only way at present is to try to guess all possible names of character sets.
3. 4.4.4, pg 38 The CD tries to say that graphic characters are guaranteed to have the expected value, but control characters are processor dependent. We should clearly say so if that's the intent.
Add (somewhere near here): "A program shall not use a control character unless specifically allowed by this standard." and then "A program may use any graphic character which has its usual meaning." or some such.
4. 24:18 (3.1.5) We say a character set must designate a character to be used as the blank character. There have been several efforts to add a new_line() character/function/etc to Fortran, but we can't be sure a character set even has control characters, let alone a control character specifically lending itself to duty as newline. We should specifically require that a character set designate a control character as newline. Near 24:18, add: "One character in each non-default character set shall be designated as the newline character to be used to separate lines in formatted stream input/output." or some such. Note that, in combination with 3. above, this requires a Fortran non-default character set to have control characters.
5. Not that we have control characters and graphic characters, let's give the poor programmer a clue as to which is which. Add intrinsics: is_control() and is_graphic() returning s-d-1 values depending on the argument, which is any kind of character.
6. While we're diagnosing characters, let's say what the largest and smallest character codes are for a given character set. To add intrinsics to return the largest and smallest character codes for a given character set, extend tiny() and huge() to take character arguments.
7. Now that we've got some reason for thinking that: 1- there are control characters and 2- that one of them has been designated newline, we can add a newline intrinsic to take a character of any kind and return the newline character in that character set. Add intrinsic: new_line(), its argument is a character of any kind.
8. For any sane floating point representation, the number one is exactly representable. But epsilon differs on either side of 1.0, (and for many, but not all, other numbers as well), so add an s= argument to epsilon(), spacing() and rrspaceing() to indicate in which direction the delta is sought. This s= argument should be defined as per nearest().
9. Now that we allow array rank remapping, the question may arise: which index i of a rank 1 array corresponds to j, k, l, ..., m of a rank n array, n > 1. Add intrinsics to convert from array indices to offset and from offset to indices given upper and lower bounds.

10. The name given a construct should have the scope of the construct.
11. While not as much fun as BOZs for reals and integers, BOZs as arguments for `logical()` and `char()` should be included for symmetry.