Subject: Allow any combination of assumed and explicit shape
From: Van Snyder
Reference: 03-258r1, section 2.9.2.2

1 Number

2 TBD

3 Title

4 Allow any combination of assumed and explicit shape.

5 Submitted By

6 J3

7 Status

8 For consideration.

9 Basic Functionality

10 Allow any combination of assumed and explicit shape.

11 Rationale

12 In many applications, one knows the values of some array bounds, but not all. In one application, I
13 have a $2{\times}2$ matrix at every point along a path of indeterminate length. If I could declare this using
14 `incoptdepth(2,2,:)`, I would have some confidence that the processor would optimize the MATMUL
15 operations along the path, without needing to write `incoptdepth(1:2,1:2,j)`. At another point in the
16 same application, I have an array that corresponds to the σ_-, π and σ_+ components of a Zeeman-split
17 spectral line. The first dimension here is naturally `-1:1`.

18 Estimated Impact

19 Minor.

20 Detailed Specification

21 Allow any dimension of a pointer, allocatable or dummy array to be declared with explicit, assumed or
22 deferred shape, independently of the others. Allow the last dimension of a pointer array to be specified
23 by an asterisk. If the bounds for any dimension are given explicitly in the declaration, the same values
24 shall be specified for those bounds in an ALLOCATE statement. If a pointer with such bounds is the
25 left-hand side in a pointer assignment statement, and any bounds are specified, any bounds explicitly
26 specified in its declaration shall have the same values in the pointer assignment statement.

27 Allow an explicit bound of a pointer or allocatable component of a derived type to be specified by
28 another component of the same type, or a subcomponent of a component of that type. Since there is no
29 object in whose context that component can be referenced, it is referenced with a prefix "%", meaning
30 "The component that specifies the bound is in the same object as the component for which it specifies
31 the bound."

32 Allow to define dynamic parameters within the type (see 04-200 and 04-162) in terms of components of
33 a type, again using prefix "%" notation to refer to them within the type definition.

34 History