s/bound/extent/

Document #: WG21 N3549
Date: 2013-03-12
Revises: None
Project: JTC1.22.32 Programming Language C++
Reply to: Walter E. Brown <webrown.cpp@gmail.com>

Contents

Abstract

The C++11 standard uses two distinct terms of art to denote an array's number of elements. For the sake of consistency, as well as improved technical accuracy, this paper proposes to adopt a single term of art throughout the standard.

1 Proposal

As we all know, a C++ array declaration typically takes the form **`T a[`**N**`]`**. In such a declaration, "The constant expression specifies the *bound* of (number of elements in) the array. If the value of the constant expression is N, the array has N elements numbered 0 to $N-1$ . . . " [dcl.array]/1.

Although this wording has been in place for a very long time, the nomenclature remains inconsistent with standard practice in very many (if not most) areas of computing. What C++ historically terms a *bound* is more commonly known as an *extent*. For example, Cray's Fortran Language Reference Manual, Volume 1, provides the following definitions under the heading "Array Terminology":[1]

> An array consists of elements that extend in one or more dimensions to represent columns, rows, planes, and so on. . . . The number of dimensions in an array is called the *rank* of the array. The number of elements in a dimension is called the *extent* of the array in that dimension. . . . The *size* of an array is the product of the extents; that is, it is the total number of elements in the array.

Even in our own C++11 standard library, the corresponding type property query trait is named **`extent`** [meta.unary.prop.query], and the array modification traits are named **`remove_extent`** and **`remove_all_extents`** [meta.query].

We propose to restore consistency within the standard by clarifying the relationship between an array's (lower and upper) bounds and its extent. The usual formula seems perfectly applicable to C++: $extent = ubound - lbound + 1$. As a bonus, such a definition makes meaningful the phrase

[1] http://docs.cray.com/books/S-3692-51/html-S-3692-51/ju1vlchri.html

"out of bounds"; it seems somewhat embarrassing to have a *bound* that exceeds the upper bound and so denotes an out of bounds value.

2 Proposed wording

All proposed wording is relative to WG21 draft [DuT12]. Green text is to be added; text in ~~red~~ is to be removed. Editorial instructions and notes are displayed against a gray background.

1. Edit [dcl.array]/1:
   ... The constant expression specifies the array's ~~*bound*~~ *extent* ~~of~~ (number of elements ~~in~~) ~~the array~~. If the value of the constant expression is N, the array has N elements numbered 0 (the *lower bound*) to $N-1$ (the *upper bound*) ... " [dcl.array]/1.

2. Edit [dcl.array]/2:
   ... only the first of the constant expressions that specify the ~~bounds~~ extents of the arrays may be omitted. In addition to declarations in which an incomplete object type is allowed, an array ~~bound~~ extent may be omitted in some cases in the declaration of a function parameter (8.3.5). An array ~~bound~~ extent may also be omitted when the declarator is followed by an *initializer* (8.5). In this case the ~~bound~~ extent is calculated from the number of initial elements (say, N) supplied (8.5.1), and the type of the identifier of `D` is "array of N `T`." Furthermore, if there is a preceding declaration of the entity in the same scope in which the ~~bound~~ extent was specified, an omitted array ~~bound~~ extent is taken to be the same as in that earlier declaration, and similarly for the definition of a static data member of a class.

3. Apply the change "`//OK: ~~bound~~ extent is 10`" 2× in [dcl.array]/4.

4. Apply the change "~~__bound~~ `__extent`" 2× in [stmt.ranged]/4.

5. Apply the change "different ~~bound~~ extent" in [temp.static]/2.

6. Apply the changes "the ~~bound~~ extent (8.3.4) of the ~~I'th~~ `I`$^{\text{th}}$ dimension of `T`" in [Table 50].

7. Apply the change "array ~~bounds~~ extents" 3× throughout the draft.

8. Apply the change "array ~~bound~~ extent" 8× more throughout the draft.

9. Apply the change "unknown ~~bound~~ extent" 38× throughout the draft.

3 Acknowledgments

Many thanks to the readers of early drafts of this paper for their helpful feedback.

4 Bibliography

[DuT12] Stefanus Du Toit: "Working Draft, Standard for Programming Language C++." ISO/IEC JTC1/SC22/WG21 document N3485 (post-Portland mailing), 2012-11-02. http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3485.pdf.

5 Revision history

Revision	Date	Changes
1.0	2013-03-12	• Published as N3549.