

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and nongovernmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication of an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 1539-1 was prepared by Joint Technical Committee ISO/IEC/JTC1, *Information technology*, Subcommittee SC22, *Programming languages, their environments and system software interfaces*.

This fourth edition cancels and replaces the third edition (ISO/IEC 1539-1:1997), which has been technically revised.

ISO/IEC 1539 consists of the following parts, under the general title *Information technology — Programming languages — Fortran*:

- *Part 1: Base language*
- *Part 2: Varying length character strings*
- *Part 3: Conditional Compilation*

Annexes A to D of this part of ISO/IEC 1539 are for information only.

## Introduction

### *Standard programming language Fortran*

This part of the international standard comprises the specification of the base Fortran language, informally known as Fortran 2000. With the limitations noted in 1.5.2, the syntax and semantics of Fortran 95 are contained entirely within Fortran 2000. Therefore, any standard-conforming Fortran 95 program not affected by such limitations is a standard conforming Fortran 2000 program. New features of Fortran 2000 can be compatibly incorporated into such Fortran 95 programs, with any exceptions indicated in the text of this part of the standard.

Fortran 2000 contains several extensions to Fortran 95; the major ones are:

- (1) Asynchronous and stream I/O
- (2) Derived type enhancements (parameterized derived types, derived type I/O)
- (3) Object oriented technologies (including extensible types, inheritance, polymorphism, procedure pointers, and initializers/finalizers)
- (4) Support of exceptions and IEEE arithmetic
- (5) Interoperability with programming language C
- (6) Support for cultural adaptability

### Organization of this part of ISO/IEC 1539

This part of ISO/IEC 1539 is organized in 16 sections, dealing with 8 conceptual areas. These 8 areas, and the sections in which they are treated, are :

High/low level concepts	Sections 1, 2, 3
Data concepts	Sections 4, 5, 6
Computations	Sections 7, 13, 15
Execution control	Section 8
Input/output	Sections 9, 10
Program units	Sections 11, 12
Scoping and association rules	Section 14
Interoperability with C	Section 16

#### *High/low level concepts*

Section 2 (Fortran terms and concepts) contains many of the high level concepts of Fortran. This includes the concept of a program and the relationships among its major parts. Also included are the syntax of program units, the rules for statement ordering, and the definitions of many of the fundamental terms used throughout this part of ISO/IEC 1539.

Section 3 (Characters, lexical tokens, and source form) describes the low level elements of Fortran, such as the character set and the allowable forms for source programs. It also contains the rules for constructing literal constants and names for Fortran entities, and lists all of the Fortran operators.

#### *Data concepts*

The array operations and data structures provide a rich set of data concepts in Fortran. The main concepts are those of data type, data object, and the use of data objects, which are described in Sections 4, 5, and 6, respectively.

Section 4 (Data types) describes the distinction between a data type and a data object, and then focuses on data type. It defines a data type as a set of data values, corresponding forms (constants) for representing these values, and operations on these values. The concept of an intrinsic data type is introduced, and the properties of Fortran's intrinsic types (integer, real, complex, logical, and character) are described. Note that only type concepts are described here, and not the declaration and properties of data objects.

Section 4 also introduces the concept of derived (user-defined) data types, which are compound types whose components ultimately resolve into intrinsic types. The details of defining a derived type are given (note that this has no counterpart with intrinsic types; intrinsic types are defined by the language and therefore need not - indeed cannot - be redefined by the programmer). As with intrinsic types, this section deals only with type properties, and not with the declaration of data objects of derived type. New in Fortran 2000 are kind and non-kind type parameters, mixed public and private accessibility of components, facilities for object-oriented programming (type extension, type-bound procedures, and overloading), the ability to specify a type in an array constructor, and facilities to support interoperability with the C programming language (type aliases and enumerators)

### J3 internal note

Unresolved issue 208

Add stuff about destructor procedures.

Except that all the discussion about what's new in F2k doesn't belong mixed in the explanation of the organization at all, so it may get added here as a temporary measure, but it better end up getting moved.

Section 5 (Data object declarations and specifications) describes in detail how named data objects are declared and given the desired properties (attributes). An important attribute (the only one required for each data object) is the data type, so the type declaration statement is the main feature of this section. The various attributes are described in detail, as well as the two ways that attributes may be specified (type declaration statements and attribute specification statements). Implicit typing and storage association (COMMON and EQUIVALENCE) are also described in this section, as well as data object value initialization.

Section 6 (Use of data objects) deals mainly with the concept of a variable, and describes the various forms that variables may take. Scalar variables include character strings and substrings, structured (derived-type) objects, structure components, and array elements. Array variables include whole arrays and array sections. Among the array facilities described here are array operations, allocation and deallocation (user controlled dynamic arrays). New in Fortran 95 is automatic deallocation of allocatable arrays in situations that caused them in Fortran 90 to have undefined allocation status; this decreases potential problems due to allocated memory leaks. Note that this applies only to arrays declared with the ALLOCATABLE attribute - not to pointers. New in Fortran 2000 are the abilities to declare the ALLOCATABLE attribute for a scalar structure component or object, and to declare the type and non-kind parameters during allocation.

### *Computations*

Section 7 (Expressions and assignment) describes how computations are expressed in Fortran. This includes the forms that expression operands (primaries) may take and the role of operators in these expressions. Operator precedence is rigorously defined in syntax rules and summarized in tabular form. This description includes the relationship of defined operators (user-defined operators) to the intrinsic operators (+, \*, .AND., .OR., etc.). The rules for both expression evaluation and the interpretation (semantics) of intrinsic and defined operators are described in detail.

Section 7 also describes assignment of computational results to data objects, which has four principal forms: the conventional assignment statement, the pointer assignment statement, the

WHERE statement and construct, and the FORALL statement and construct. The WHERE and FORALL statements and constructs allow masked array assignment, the main difference between WHERE and FORALL being that FORALL makes use of element subscripts whereas WHERE is whole array oriented.

Section 13 (Intrinsic procedures and modules) describes more than one hundred intrinsic procedures that provide a rich set of computational capabilities. In addition to the Fortran 95 intrinsic procedures, this includes EXTENDS\_TYPE\_OF, SAME\_TYPE\_AS, SELECTED\_CHAR\_KIND, and GET\_ENVIRONMENT\_VARIABLE. Fortran 2000 extends the functionality of the intrinsic procedure SYSTEM\_CLOCK. New in Fortran 2000 is the ISO\_FORTRAN\_ENV module, which provides named constants for several processor-dependent values.

Section 15 (Exceptions and IEEE arithmetic) describes intrinsic modules to provide support for the five exceptions specified by the IEEE standard for floating-point arithmetic and for other features of that IEEE standard. A processor is permitted to provide partial support; there are facilities for inquiring about which features are supported or for requiring support of certain features.

### *Execution control*

Section 8 (Execution control) describes the control constructs (IF, CASE, and DO), and the control statements (IF, CONTINUE, GO TO, and STOP). New in Fortran 2000 are the SELECT TYPE and ASSOCIATE constructs.

### *Input/output*

Section 9 (Input/output statements) contains definitions for records, files, file connections (OPEN, CLOSE, and preconnected files), data transfer statements (READ, WRITE, and PRINT) that include processing of partial and variable length records, file positioning (REWIND and BACKSPACE), and file inquiry (INQUIRE).

#### **J3 internal note**

Unresolved issue 209

Need some comment on asynchronous, stream, and derived type I/O.

Section 10 (Input/output editing) describes input/output formatting. This includes the FORMAT statement and FMT= specifier, edit descriptors, list-directed formatting, and namelist formatting.

### *Program units*

Section 11 (Program units) describes main programs, external subprograms, modules, and block data program units. Modules, along with the USE statement, are described as a mechanism for encapsulating data and procedure definitions that are to be used by (accessible to) other program units. Modules are described as vehicles for defining global derived-type definitions, global data object declarations, procedure libraries, and combinations thereof. A module may be intrinsic (defined by the processor) or nonintrinsic.

Section 12 (Procedures) contains a comprehensive treatment of procedure definition and invocation, including that for user-defined functions and subroutines. The concepts of implicit and explicit procedure interfaces are explained, and situations requiring explicit procedure interfaces are identified. The rules governing actual and dummy arguments, and their association, are described. PURE procedures and ELEMENTAL procedures (which are PURE procedures that may be called elementally) are free of side effects, thereby facilitating parallel processing. New in Fortran 2000 are abstract interfaces, and procedure pointers and pointer declarations that use abstract interfaces.

Section 12 also describes the use of the OPERATOR option in interface blocks to allow function invocation in the form of infix and prefix operators as well as the traditional functional form. Similarly, the use of the ASSIGNMENT option in interface blocks is described as allowing an alternate syntax for certain subroutine calls. This section also contains descriptions of recursive procedures, the RETURN statement, the ENTRY statement, internal procedures and the CONTAINS statement, statement functions, generic procedure names, and the means of accessing non-Fortran procedures.

**J3 internal note**

Unresolved issue 221

Add stuff about controlling host association into interface bodies.

*Scoping and association rules*

Section 14 (Scope, association, and definition) explains the use of the term "scope" and describes the scope properties of various entities, including names and operators. Also described are the general rules governing procedure argument association, pointer association, and storage association. Finally, Section 14 describes the events that cause variables to become defined (have predictable values) and events that cause variables to become undefined.

*Interoperability with C*

Section 16 (Interoperability with C) describes mechanisms that allow Fortran data objects and procedures to interoperate with data objects and procedures defined by means other than Fortran.

**Annexes**

Annex A. A glossary of common and important terms used in this part of ISO/IEC 1539.

Annex B. A list of all obsolescent features and descriptions of all deleted features. Obsolescent features are still part of Fortran 2000 and are described in the normative portions of this part of ISO/IEC 1539. Deleted features are not part of standard Fortran 2000, but they are described completely in this annex for the benefit of those implementations that provide complete backward compatibility with Fortran 90.

Annex C. Long notes of explanation, examples, rationales and other informative material. Wherever feasible such material is integrated into the normative sections of this part of ISO/IEC 1539, but clearly identified as supporting informative material. In those cases in which such informative material is so extensive that it would unduly disrupt the flow of normative discourse, the material is placed in this annex.

Annex D. A comprehensive index to this part of ISO/IEC 1539, including the use of principal terms in the syntax rules.

