

INTERNATIONAL  
STANDARD

**ISO/IEC**  
**1539**

Second edition  
1991-07-01

---

---

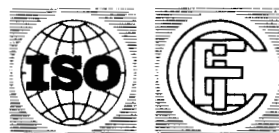
**Information technology — Programming  
languages — Fortran**

*Technologies de l'information — Langages de programmation — Fortran*

**iTeh STANDARD PREVIEW**  
**(standards.iteh.ai)**

ISO/IEC 1539:1991

<https://standards.iteh.ai/catalog/standards/sist/58d391bf-e571-4a5e-91de-4ad20949d95b/iso-iec-1539-1991>



Reference number  
ISO/IEC 1539 : 1991 (E)

# Contents

Foreword.....		xii
1. Overview.....		1
1.1 Scope.....		1
1.2 Processor.....		1
1.3 Inclusions and exclusions.....		1
1.3.1 Inclusions.....		1
1.3.2 Exclusions.....		1
1.4 Conformance.....		2
1.4.1 FORTRAN 77 compatibility.....		3
1.5 Notation used in this International Standard.....		3
1.5.1 Syntax rules.....		3
1.5.2 Assumed syntax rules.....		4
1.5.3 Syntax conventions and characteristics.....		5
1.5.4 Text conventions.....		5
1.6 Deleted and obsolescent features.....		5
1.6.1 Nature of deleted features.....		5
1.6.2 Nature of obsolescent features.....		5
1.7 Modules.....		6
1.8 Normative references.....		6
2. Fortran terms and concepts.....		7
2.1 High level syntax.....		7
2.2 Program unit concepts.....		9
2.2.1 Executable program.....		10
2.2.2 Main program.....		10
2.2.3 Procedure.....		10
2.2.4 Module.....		10
2.3 Execution concepts.....		11
2.3.1 Executable/nonexecutable statements.....		11
2.3.2 Statement order.....		11
2.3.3 The END statement.....		12
2.3.4 Execution sequence.....		12
2.4 Data concepts.....		13
2.4.1 Data type.....		13
2.4.2 Data value.....		13
2.4.3 Data entity.....		13
2.4.4 Scalar.....		14
2.4.5 Array.....		15
2.4.6 Pointer.....		15
2.4.7 Storage.....		15
2.5 Fundamental terms.....		15
2.5.1 Name and designator.....		15
2.5.2 Keyword.....		16

© ISO/IEC 1991

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

2.5.3	Declaration .....	16
2.5.4	Definition .....	16
2.5.5	Reference .....	16
2.5.6	Association .....	16
2.5.7	Intrinsic .....	16
2.5.8	Operator .....	16
2.5.9	Sequence .....	17
3.	Characters, lexical tokens, and source form .....	18
3.1	Processor character set .....	18
3.1.1	Letters .....	18
3.1.2	Digits .....	18
3.1.3	Underscore .....	18
3.1.4	Special characters .....	19
3.1.5	Other characters .....	19
3.2	Low-level syntax .....	19
3.2.1	Keywords .....	19
3.2.2	Names .....	19
3.2.3	Constants .....	20
3.2.4	Operators .....	20
3.2.5	Statement labels .....	21
3.2.6	Delimiters .....	21
3.3	Source form .....	21
3.3.1	Free source form .....	22
3.3.2	Fixed source form .....	23
3.4	Including source text .....	24
4.	Intrinsic and derived data types .....	25
4.1	The concept of data type .....	25
4.1.1	Set of values .....	25
4.1.2	Constants .....	25
4.1.3	Operations .....	26
4.2	Relationship of types and values to objects .....	26
4.3	Intrinsic data types .....	26
4.3.1	Numeric types .....	26
4.3.2	Nonnumeric types .....	30
4.4	Derived types .....	32
4.4.1	Derived-type definition .....	32
4.4.2	Determination of derived types .....	35
4.4.3	Derived-type values .....	36
4.4.4	Construction of derived-type values .....	37
4.4.5	Derived-type operations and assignment .....	37
4.5	Construction of array values .....	37
5.	Data object declarations and specifications .....	39
5.1	Type declaration statements .....	39
5.1.1	Type specifiers .....	41
5.1.2	Attributes .....	43
5.2	Attribute specification statements .....	48
5.2.1	INTENT statement .....	48

5.2.2	OPTIONAL statement .....	49
5.2.3	Accessibility statements .....	49
5.2.4	SAVE statement .....	50
5.2.5	DIMENSION statement .....	50
5.2.6	ALLOCATABLE statement .....	50
5.2.7	POINTER statement .....	51
5.2.8	TARGET statement .....	51
5.2.9	DATA statement .....	51
5.2.10	PARAMETER statement .....	53
5.3	IMPLICIT statement .....	54
5.4	NAMELIST statement .....	56
5.5	Storage association of data objects .....	56
5.5.1	EQUIVALENCE statement .....	56
5.5.2	COMMON statement .....	58
6.	Use of data objects .....	61
6.1	Scalars .....	62
6.1.1	Substrings .....	62
6.1.2	Structure components .....	62
6.2	Arrays .....	63
6.2.1	Whole arrays .....	63
6.2.2	Array elements and array sections .....	63
6.3	Dynamic association .....	67
6.3.1	ALLOCATE statement .....	67
6.3.2	NULLIFY statement .....	68
6.3.3	DEALLOCATE statement .....	68
7.	Expressions and assignment .....	70
7.1	Expressions .....	70
7.1.1	Form of an expression .....	70
7.1.2	Intrinsic operations .....	74
7.1.3	Defined operations .....	75
7.1.4	Data type, type parameters, and shape of an expression .....	75
7.1.5	Conformability rules for intrinsic operations .....	77
7.1.6	Scalar and array expressions .....	77
7.1.7	Evaluation of operations .....	79
7.2	Interpretation of intrinsic operations .....	83
7.2.1	Numeric intrinsic operations .....	83
7.2.2	Character intrinsic operation .....	84
7.2.3	Relational intrinsic operations .....	85
7.2.4	Logical intrinsic operations .....	86
7.3	Interpretation of defined operations .....	86
7.3.1	Unary defined operation .....	86
7.3.2	Binary defined operation .....	87
7.4	Precedence of operators .....	87
7.5	Assignment .....	89
7.5.1	Assignment statement .....	89
7.5.2	Pointer assignment .....	92
7.5.3	Masked array assignment—WHERE .....	92

8.	Execution control.....	95
8.1	Executable constructs containing blocks.....	95
	8.1.1 Rules governing blocks.....	95
	8.1.2 IF construct.....	96
	8.1.3 CASE construct.....	97
	8.1.4 DO construct.....	100
8.2	Branching.....	107
	8.2.1 Statement labels.....	107
	8.2.2 GO TO statement.....	107
	8.2.3 Computed GO TO statement.....	107
	8.2.4 ASSIGN and assigned GO TO statement.....	107
	8.2.5 Arithmetic IF statement.....	108
8.3	CONTINUE statement.....	108
8.4	STOP statement.....	108
8.5	PAUSE statement.....	108
9.	Input/output statements.....	109
9.1	Records.....	109
	9.1.1 Formatted record.....	109
	9.1.2 Unformatted record.....	109
	9.1.3 Endfile record.....	110
9.2	Files.....	110
	9.2.1 External files.....	110
	9.2.2 Internal files.....	112
9.3	File connection.....	113
	9.3.1 Unit existence.....	114
	9.3.2 Connection of a file to a unit.....	114
	9.3.3 Preconnection.....	115
	9.3.4 The OPEN statement.....	115
	9.3.5 The CLOSE statement.....	118
9.4	Data transfer statements.....	119
	9.4.1 Control information list.....	119
	9.4.2 Data transfer input/output list.....	123
	9.4.3 Error, end-of-record, and end-of-file conditions.....	124
	9.4.4 Execution of a data transfer input/output statement.....	125
	9.4.5 Printing of formatted records.....	128
	9.4.6 Termination of data transfer statements.....	128
9.5	File positioning statements.....	128
	9.5.1 BACKSPACE statement.....	129
	9.5.2 ENDFILE statement.....	129
	9.5.3 REWIND statement.....	129
9.6	File inquiry.....	130
	9.6.1 Inquiry specifiers.....	130
	9.6.2 Restrictions on inquiry specifiers.....	134
	9.6.3 Inquire by output list.....	134
9.7	Restrictions on function references and list items.....	134
9.8	Restriction on input/output statements.....	134

10.	Input/output editing.....	135
10.1	Explicit format specification methods.....	135
10.1.1	FORMAT statement .....	135
10.1.2	Character format specification .....	135
10.2	Form of a format item list .....	136
10.2.1	Edit descriptors.....	136
10.2.2	Fields.....	137
10.3	Interaction between input/output list and format .....	137
10.4	Positioning by format control .....	138
10.5	Data edit descriptors .....	139
10.5.1	Numeric editing .....	139
10.5.2	Logical editing .....	143
10.5.3	Character editing .....	144
10.5.4	Generalized editing .....	144
10.6	Control edit descriptors .....	145
10.6.1	Position editing.....	145
10.6.2	Slash editing .....	146
10.6.3	Colon editing .....	146
10.6.4	S, SP, and SS editing .....	146
10.6.5	P editing .....	147
10.6.6	BN and BZ editing .....	147
10.7	Character string edit descriptors .....	147
10.7.1	Character constant edit descriptor .....	147
10.7.2	H editing .....	148
10.8	List-directed formatting.....	148
10.8.1	List-directed input .....	148
10.8.2	List-directed output .....	150
10.9	Namelist formatting .....	151
10.9.1	Namelist input .....	151
10.9.2	Namelist output .....	154
11.	Program units.....	156
11.1	Main program.....	156
11.1.1	Main program specifications.....	156
11.1.2	Main program executable part .....	156
11.1.3	Main program internal procedures .....	157
11.2	External subprograms .....	157
11.3	Modules .....	157
11.3.1	Module reference .....	157
11.3.2	The USE statement and use association.....	158
11.3.3	Examples of the use of modules.....	159
11.4	Block data program units.....	161
12.	Procedures .....	163
12.1	Procedure classifications.....	163
12.1.1	Procedure classification by reference .....	163
12.1.2	Procedure classification by means of definition.....	163
12.2	Characteristics of procedures.....	165
12.2.1	Characteristics of dummy arguments.....	165
12.2.2	Characteristics of function results .....	166

12.3	Procedure interface .....	166
12.3.1	Implicit and explicit interfaces.....	166
12.3.2	Specification of the procedure interface.....	167
12.4	Procedure reference.....	171
12.4.1	Actual argument list .....	171
12.4.2	Function reference .....	174
12.4.3	Elemental intrinsic function reference .....	174
12.4.4	Subroutine reference .....	174
12.4.5	Elemental intrinsic subroutine reference .....	175
12.5	Procedure definition .....	175
12.5.1	Intrinsic procedure definition .....	175
12.5.2	Procedures defined by subprograms .....	175
12.5.3	Definition of procedures by means other than Fortran .....	181
12.5.4	Statement function .....	182
13.	Intrinsic procedures.....	183
13.1	Intrinsic functions.....	183
13.2	Elemental intrinsic procedures .....	183
13.2.1	Elemental intrinsic function arguments and results .....	183
13.2.2	Elemental intrinsic subroutine arguments.....	183
13.3	Positional arguments or argument keywords .....	183
13.4	Argument presence inquiry function .....	184
13.5	Numeric, mathematical, character, kind, logical, and bit procedures .....	184
13.5.1	Numeric functions .....	184
13.5.2	Mathematical functions .....	184
13.5.3	Character functions .....	184
13.5.4	Character inquiry function .....	184
13.5.5	Kind functions .....	184
13.5.6	Logical function .....	184
13.5.7	Bit manipulation and inquiry procedures .....	184
13.6	Transfer function .....	185
13.7	Numeric manipulation and inquiry functions.....	185
13.7.1	Models for integer and real data.....	185
13.7.2	Numeric inquiry functions .....	186
13.7.3	Floating point manipulation functions .....	186
13.8	Array intrinsic functions .....	186
13.8.1	The shape of array arguments.....	186
13.8.2	Mask arguments .....	186
13.8.3	Vector and matrix multiplication functions.....	186
13.8.4	Array reduction functions .....	187
13.8.5	Array inquiry functions.....	187
13.8.6	Array construction functions .....	187
13.8.7	Array reshape function.....	187
13.8.8	Array manipulation functions .....	187
13.8.9	Array location functions.....	187
13.8.10	Pointer association status inquiry functions .....	187
13.9	Intrinsic subroutines.....	187
13.9.1	Date and time subroutines .....	188
13.9.2	Pseudorandom numbers .....	188
13.9.3	Bit copy subroutine .....	188

13.10	Generic intrinsic functions .....	188
13.10.1	Argument presence inquiry function.....	188
13.10.2	Numeric functions .....	188
13.10.3	Mathematical functions .....	189
13.10.4	Character functions .....	189
13.10.5	Character inquiry function .....	189
13.10.6	Kind functions.....	190
13.10.7	Logical function.....	190
13.10.8	Numeric inquiry functions .....	190
13.10.9	Bit inquiry function .....	190
13.10.10	Bit manipulation functions .....	190
13.10.11	Transfer function .....	190
13.10.12	Floating-point manipulation functions .....	190
13.10.13	Vector and matrix multiply functions .....	191
13.10.14	Array reduction functions .....	191
13.10.15	Array inquiry functions.....	191
13.10.16	Array construction functions .....	191
13.10.17	Array reshape function.....	191
13.10.18	Array manipulation functions.....	192
13.10.19	Array location functions.....	192
13.10.20	Pointer association status inquiry function .....	192
13.11	Intrinsic subroutines.....	192
13.12	Specific names for intrinsic functions.....	192
13.13	Specifications of the intrinsic procedures .....	194
14.	Scope, association, and definition .....	241
14.1	Scope of names .....	241
14.1.1	Global entities .....	241
14.1.2	Local entities .....	241
14.1.3	Statement entities .....	245
14.2	Scope of labels.....	245
14.3	Scope of external input/output units.....	245
14.4	Scope of operators.....	245
14.5	Scope of the assignment symbol .....	245
14.6	Association .....	245
14.6.1	Name association .....	246
14.6.2	Pointer association .....	246
14.6.3	Storage association .....	247
14.7	Definition and undefinition of variables .....	249
14.7.1	Definition of objects and subobjects .....	249
14.7.2	Variables that are always defined .....	249
14.7.3	Variables that are initially defined .....	249
14.7.4	Variables that are initially undefined .....	250
14.7.5	Events that cause variables to become defined .....	250
14.7.6	Events that cause variables to become undefined .....	251
14.8	Allocation status .....	252

ITeH STANDARD PREVIEW  
 (standards.iteh.ai)



## Annexes

A.	Glossary of technical terms.....	254
B.	Decremental features .....	262
B.1	Deleted features.....	262
B.2	Obsolescent features.....	262
B.2.1	Alternate return .....	262
B.2.2	PAUSE statement .....	263
B.2.3	ASSIGN and assigned GO TO statements .....	263
B.2.4	Assigned FORMAT specifiers .....	263
B.2.5	H editing .....	263
C.	Section notes .....	264
C.1	Section 1 notes.....	264
C.1.1	Conformance (1.4) .....	264
C.2	Section 2 notes.....	264
C.2.1	Keywords .....	264
C.3	Section 3 notes.....	264
C.3.1	Representable characters (3.1.5) .....	264
C.3.2	Comment lines (3.3.1.1, 3.3.2.1) .....	264
C.3.3	Statement labels (3.2.5).....	265
C.3.4	Source form (3.3) .....	265
C.4	Section 4 notes.....	265
C.4.1	Zero (4.3.1) .....	265
C.4.2	Characters (4.2) .....	265
C.4.3	Intrinsic and derived data types (4.3, 4.4) .....	266
C.4.4	Selection of the approximation methods.....	266
C.4.5	Storage of derived types (4.4.1).....	267
C.4.6	Pointers .....	267
C.5	Section 5 notes.....	268
C.5.1	Type declaration statements (5.1) .....	268
C.5.2	The POINTER attribute (5.1.2.7).....	268
C.5.3	The TARGET attribute (5.1.2.8).....	269
C.5.4	PARAMETER statements and IMPLICIT NONE (5.2.10, 5.3).....	270
C.5.5	EQUIVALENCE statement extensions (5.5.1) .....	270
C.5.6	COMMON statement extensions (5.5.2).....	270
C.6	Section 6 notes.....	270
C.6.1	Substrings (6.1.1) .....	270
C.6.2	Array element references (6.2.2) .....	270
C.6.3	Structure components (6.1.2) .....	270
C.6.4	Pointer allocation and association.....	271
C.7	Section 7 notes.....	272
C.7.1	Character assignment .....	272
C.7.2	Evaluation of function references .....	272
C.7.3	Pointers in expressions .....	272
C.7.4	Pointers on the left side of an assignment.....	273
C.8	Section 8 notes.....	273
C.8.1	Loop control .....	273

	C.8.2	The CASE construct .....	274
	C.8.3	Examples of invalid DO constructs .....	274
C.9	Section 9	notes.....	274
	C.9.1	Input/output records (9.1) .....	274
	C.9.2	Files (9.2) .....	275
	C.9.3	OPEN statement (9.3.4) .....	276
	C.9.4	Connection properties (9.3.2).....	278
	C.9.5	CLOSE statement (9.3.5) .....	279
	C.9.6	INQUIRE statement (9.6) .....	280
	C.9.7	Keyword specifiers .....	280
	C.9.8	Format specifications (9.4.1.1) .....	280
	C.9.9	Unformatted input/output (9.4.4.4.1) .....	280
	C.9.10	Input/output restrictions.....	280
	C.9.11	Pointers in an input/output list .....	280
	C.9.12	Derived type objects in an input/output list (9.4.2) .....	280
C.10	Section 10	notes .....	281
	C.10.1	Character constant format specification (10.1.2, 10.7.1) .....	281
	C.10.2	T edit descriptor (10.6.1.1).....	281
	C.10.3	Length of formatted records .....	281
	C.10.4	Number of records (10.3, 10.4, 10.6.2) .....	281
	C.10.5	List-directed input/output (10.8).....	282
	C.10.6	List-directed input (10.8.1) .....	282
	C.10.7	Namelist list items for character input (10.9.1.3).....	282
	C.10.8	Namelist output records (10.9.2.2) .....	282
C.11	Section 11	notes .....	283
	C.11.1	Main program and block data program unit (11.1, 11.4) .....	283
	C.11.2	Dependent compilation (11.3) .....	283
	C.11.3	Pointers in modules .....	285
	C.11.4	Example of a module (11.3) .....	285
C.12	Section 12	notes .....	288
	C.12.1	Examples of host association (12.1.2.2.1) .....	288
	C.12.2	External procedures (12.3.2.2).....	289
	C.12.3	Procedures defined by means other than Fortran (12.5.3) .....	289
	C.12.4	Procedure interfaces (12.3) .....	290
	C.12.5	Argument association and evaluation (12.4.1) .....	290
	C.12.6	Argument intent specification (12.4.1.1).....	291
	C.12.7	Dummy argument restrictions (12.5.2.9) .....	291
	C.12.8	Pointers and targets as arguments.....	291
	C.12.9	The ASSOCIATED function (13.13.13).....	292
	C.12.10	Internal procedure restrictions.....	292
	C.12.11	The result variable (12.5.2.2) .....	293
C.13	Section 13	notes .....	293
	C.13.1	Summary of features.....	293
	C.13.2	Examples .....	295
	C.13.3	FORMula TRANslation and array processing .....	299
	C.13.4	Sum of squared residuals .....	300
	C.13.5	Vector norms: infinity-norm and one-norm .....	300
	C.13.6	Matrix norms: infinity-norm and one-norm .....	300
	C.13.7	Logical queries.....	300
	C.13.8	Parallel computations.....	301
	C.13.9	Example of element-by-element computation.....	301

	C.13.10	Bit manipulation and inquiry procedures.....	302
C.14		Section 14 notes .....	302
	C.14.1	Storage association of zero-sized objects .....	302
D.		Syntax rules .....	303
	D.1	Syntax rules and constraints .....	303
		D.1.1 Overview .....	303
		D.1.2 Fortran terms and concepts .....	303
		D.1.3 Characters, lexical tokens, and source form .....	306
		D.1.4 Intrinsic and derived data types .....	307
		D.1.5 Data object declarations and specifications .....	310
		D.1.6 Use of data objects .....	315
		D.1.7 Expressions and assignment .....	317
		D.1.8 Execution control .....	320
		D.1.9 Input/output statements .....	323
		D.1.10 Input/output editing .....	326
		D.1.11 Program units .....	328
		D.1.12 Procedures .....	329
		D.1.13 Intrinsic procedures .....	331
		D.1.14 Scope, association, and definition .....	332
	D.2	Cross references .....	332
		D.2.1 Nonterminal symbols that are defined .....	332
		D.2.2 Nonterminal symbols that are not defined .....	338
		D.2.3 Terminal symbols .....	339
E.		Permuted index for headings.....	344
F.		Index .....	363

## Figures

Figure 2.1	Requirements on statement ordering.....	11
------------	---	----

## Tables

Table 2.1	Statements allowed in scoping units.....	12
Table 3.1	Special characters.....	19
Table 6.1	Subscript order value.....	65
Table 7.1	Type of operands and result for the intrinsic operation $\{x_1\} \text{ op } x_2$ .....	74
Table 7.2	Interpretation of the numeric intrinsic operators.....	84
Table 7.3	Interpretation of the character intrinsic operator <code>//</code> .....	84
Table 7.4	Interpretation of the relational intrinsic operators.....	85
Table 7.5	Interpretation of the logical intrinsic operators.....	86
Table 7.6	The values of operations involving logical intrinsic operators.....	86
Table 7.7	Categories of operations and relative precedences.....	87
Table 7.8	Type conformance for the intrinsic assignment statement $\text{variable} = \text{expr}$ .....	89
Table 7.9	Numeric conversion and assignment statement $\text{variable} = \text{expr}$ .....	90
Table C.1	Values assigned to INQUIRE specifier variables.....	279

## 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 non-governmental, 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 as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 1539 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

This second edition cancels and replaces the first edition (ISO 1539 : 1980), which has been technically revised.

Annexes A, B, C, D, E and F are for information only.

**ITeH STANDARD PREVIEW**  
**(standards.iteh.ai)**

ISO/IEC 1539:1991

<https://standards.iteh.ai/catalog/standards/sist/58d391bf-e571-4a5e-91de-4ad20949d95b/iso-iec-1539-1991>

## Introduction

### *Standard programming language Fortran*

This International Standard specifies the form and establishes the interpretation of programs expressed in the Fortran language (known informally as "Fortran 90"). It consists of the specification of the language Fortran. No subsets are specified in this International Standard. With limitations noted in 1.4.1, the syntax and semantics of the International Standard commonly known as "FORTRAN 77" are contained entirely within this International Standard. Therefore, any standard-conforming FORTRAN 77 program is standard conforming under this International Standard. New features can be compatibly incorporated into such programs, with any exceptions indicated in the text of this International Standard.

A standard-conforming Fortran processor is also a standard-conforming FORTRAN 77 processor.

Note that the name of this language, Fortran, differs from that in FORTRAN 77 in that only the first letter is capitalized. Both FORTRAN 77 and FORTRAN 66 used only capital letters in the official name of the language, but Fortran 90 does not continue this tradition.

## Overview

Among the additions to FORTRAN 77 in this International Standard, seven stand out as the major ones:

- (1) Array operations
- (2) Improved facilities for numerical computation
- (3) Parameterized intrinsic data types
- (4) User-defined data types
- (5) Facilities for modular data and procedure definitions
- (6) Pointers
- (7) The concept of language evolution

A number of other additions are also included in this International Standard, such as improved source form facilities, more control constructs, recursion, additional input/output facilities, and dynamically allocatable arrays.

### *Array operations*

Computation involving large arrays is an important part of engineering and scientific computing. Arrays may be used as entities in Fortran. Operations for processing whole arrays and subarrays (array sections) are included in the language for two principal reasons: (1) these features provide a more concise and higher level language that will allow programmers more quickly and reliably to develop and maintain scientific/engineering applications, and (2) these features can significantly facilitate optimization of array operations on many computer architectures.

The FORTRAN 77 arithmetic, logical, and character operations and intrinsic (predefined) functions are extended to operate on array-valued operands. The array extensions include whole, partial, and masked array assignment, array-valued constants and expressions, and facilities to define user-supplied array-valued functions. New intrinsic procedures are provided to manipulate and construct arrays, to perform gather/scatter operations, and to support extended computational capabilities involving arrays. For example, an intrinsic function is provided to sum the elements of an array.

### *Numerical computation*

Scientific computation is one of the principal application domains of Fortran, and a guiding objective for all of the technical work is to strengthen Fortran as a vehicle for implementing scientific software. Though nonnumeric computations are increasing dramatically in scientific applications, numeric computation remains dominant. Accordingly, the additions include portable control over numeric precision specification, inquiry as to the characteristics of numeric representation, and improved control of the performance of numerical programs (for example, improved argument range reduction and scaling).

### *Parameterized character data type*

Optional facilities for multibyte character data for languages with large character sets, such as those in China and Japan, are added by using a kind parameter for the character data type. This facility allows additional character sets for special purposes as well, such as characters for mathematics, chemistry, or music.

### *Derived types*

“Derived type” is the term given to that set of features in this International Standard that allows the programmer to define arbitrary data structures and operations on them. Data structures are user-defined aggregations of intrinsic and derived data types. Intrinsic uses of structured objects include assignment, input/output, and as procedure arguments. With no additional derived-type operations defined by the user, the derived data type facility is a simple data structuring mechanism. With additional operation definitions, derived types provide an effective implementation mechanism for data abstractions.

Procedure definitions may be used to define operations on intrinsic or derived types and nonintrinsic assignments for intrinsic and derived types.

### *Modular definitions*

In FORTRAN 77, there was no way to define a global data area in only one place and have all the program units in an application use that definition. In addition, the ENTRY statement is awkward and restrictive for implementing a related set of procedures, possibly involving common data objects. Finally, there was no means in FORTRAN 77 by which procedure definitions, especially interface information, could be made known locally to a program unit. These and other deficiencies are remedied by a new type of program unit that may contain any combination of data object declarations, derived-type definitions, procedure definitions, and procedure interface information. This program unit, called a module, may be considered to be a generalization and replacement for the block data program unit. A module may be accessed by any program unit, thereby making the module contents available to that program unit. Thus, modules provide improved facilities for defining global data areas, procedure packages, and encapsulated data abstractions.

### *Pointers*

Pointers allow arrays to be sized dynamically and ranged, and structures to be linked to create lists, trees, and graphs. An object of any intrinsic or derived type may be declared to have the pointer attribute. Once such an object becomes associated with a target, it may appear almost anywhere a nonpointer object with the same type, type parameters, and shape may appear.

### *Language evolution*

With the addition of new facilities, certain old features become redundant and may eventually be phased out of the language as their usage declines. For example, the numeric facilities alluded to above provide the functionality of double precision; with the new array facilities, nonconformable argument association (such as associating an array element with a dummy array) is unnecessary (and in fact is not useful as an array operation); and block data program units are redundant and inferior to modules.

As part of the evolution of the language, categories of language features (deleted and obsolescent) are provided which allow unused features of the language to be removed from future standards.

## Organization of this International Standard

This document is organized in 14 sections, dealing with 7 conceptual areas. These 7 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
Execution Control	Section 8
Input/Output	Sections 9, 10
Program Units	Sections 11, 12
Scoping and Association Rules	Section 14

### *High/low level concepts*

Section 2 (Fortran Terms and Concepts) contains many of the high level concepts of Fortran. This includes the concept of an executable 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 the document.

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 (arrays as data objects) 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 (Intrinsic and Derived 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 as intrinsic types are predefined 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.

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 object's 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. Arrays are considered to be variables, as are array sections. Among the array facilities described here are array sections (subarrays), and array