

---

---

**Information technology — Programming  
languages — Fortran —**

**Part 1:  
Base language**

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

*Partie 1: Langage de base*

ISO/IEC 1539-1:1997

<https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997>



## Contents

	Foreword .....	xiii
	Introduction .....	xiv
1	Overview .....	1
1.1	Scope .....	1
1.2	Processor .....	1
1.3	Inclusions .....	1
1.4	Exclusions .....	1
1.5	Conformance .....	2
1.5.1	Fortran 90 compatibility .....	3
1.5.2	FORTRAN 77 compatibility .....	3
1.6	Notation used in this standard .....	4
1.6.1	Informative notes .....	4
1.6.2	Syntax rules .....	4
1.6.3	Assumed syntax rules .....	5
1.6.4	Syntax conventions and characteristics .....	5
1.6.5	Text conventions .....	6
1.7	Deleted and obsolescent features .....	6
1.7.1	Nature of deleted features .....	6
1.7.2	Nature of obsolescent features .....	6
1.8	Modules .....	6
1.9	Normative references .....	7
2	Fortran terms and concepts .....	9
2.1	High level syntax .....	9
2.2	Program unit concepts .....	11
2.2.1	Program .....	12
2.2.2	Main program .....	12
2.2.3	Procedure .....	12
2.2.4	Module .....	13
2.3	Execution concepts .....	13
2.3.1	Executable/nonexecutable statements .....	13
2.3.2	Statement order .....	13
2.3.3	The END statement .....	14
2.3.4	Execution sequence .....	14

© ISO/IEC 1997

All rights reserved. Unless otherwise specified, 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

2.4	Data concepts . . . . .	15
2.4.1	Data type . . . . .	15
2.4.2	Data value . . . . .	15
2.4.3	Data entity . . . . .	15
2.4.4	Scalar . . . . .	17
2.4.5	Array . . . . .	17
2.4.6	Pointer . . . . .	17
2.4.7	Storage . . . . .	17
2.5	Fundamental terms . . . . .	17
2.5.1	Name and designator . . . . .	18
2.5.2	Keyword . . . . .	18
2.5.3	Declaration . . . . .	18
2.5.4	Definition . . . . .	18
2.5.5	Reference . . . . .	18
2.5.6	Association . . . . .	18
2.5.7	Intrinsic . . . . .	19
2.5.8	Operator . . . . .	19
2.5.9	Sequence . . . . .	19
3	Characters, lexical tokens, and source form . . . . .	21
3.1	Processor character set . . . . .	21
3.1.1	Letters . . . . .	21
3.1.2	Digits . . . . .	21
3.1.3	Underscore . . . . .	21
3.1.4	Special characters . . . . .	22
3.1.5	Other characters . . . . .	22
3.2	Low-level syntax . . . . .	22
3.2.1	Names . . . . .	22
3.2.2	Constants . . . . .	23
3.2.3	Operators . . . . .	23
3.2.4	Statement labels . . . . .	24
3.2.5	Delimiters . . . . .	24
3.3	Source form . . . . .	24
3.3.1	Free source form . . . . .	25
3.3.2	Fixed source form . . . . .	27
3.4	Including source text . . . . .	27
4	Intrinsic and derived data types . . . . .	29
4.1	The concept of data type . . . . .	29
4.1.1	Set of values . . . . .	29
4.1.2	Constants . . . . .	30
4.1.3	Operations . . . . .	30
4.2	Relationship of types and values to objects . . . . .	30
4.3	Intrinsic data types . . . . .	31
4.3.1	Numeric types . . . . .	31
4.3.2	Nonnumeric types . . . . .	35
4.4	Derived types . . . . .	37
4.4.1	Derived-type definition . . . . .	38
4.4.2	Determination of derived types . . . . .	43
4.4.3	Derived-type values . . . . .	44
4.4.4	Construction of derived-type values . . . . .	44
4.4.5	Derived-type operations and assignment . . . . .	45

4.5	Construction of array values .....	45
5	Data object declarations and specifications .....	47
5.1	Type declaration statements .....	47
5.1.1	Type specifiers .....	50
5.1.2	Attributes .....	52
5.2	Attribute specification statements .....	57
5.2.1	INTENT statement .....	58
5.2.2	OPTIONAL statement .....	58
5.2.3	Accessibility statements .....	58
5.2.4	SAVE statement .....	59
5.2.5	DIMENSION statement .....	59
5.2.6	ALLOCATABLE statement .....	60
5.2.7	POINTER statement .....	60
5.2.8	TARGET statement .....	60
5.2.9	PARAMETER statement .....	60
5.2.10	DATA statement .....	61
5.3	IMPLICIT statement .....	63
5.4	NAMELIST statement .....	65
5.5	Storage association of data objects .....	66
5.5.1	EQUIVALENCE statement .....	66
5.5.2	COMMON statement .....	68
6	Use of data objects .....	73
6.1	Scalars .....	74
6.1.1	Substrings .....	74
6.1.2	Structure components .....	75
6.2	Arrays .....	75
6.2.1	Whole arrays .....	76
6.2.2	Array elements and array sections .....	76
6.3	Dynamic association .....	79
6.3.1	ALLOCATE statement .....	79
6.3.2	NULLIFY statement .....	82
6.3.3	DEALLOCATE statement .....	82
7	Expressions and assignment .....	85
7.1	Expressions .....	85
7.1.1	Form of an expression .....	85
7.1.2	Intrinsic operations .....	89
7.1.3	Defined operations .....	90
7.1.4	Data type, type parameters, and shape of an expression .....	90
7.1.5	Conformability rules for elemental operations .....	92
7.1.6	Scalar and array expressions .....	93
7.1.7	Evaluation of operations .....	96
7.2	Interpretation of intrinsic operations .....	101
7.2.1	Numeric intrinsic operations .....	101
7.2.2	Character intrinsic operation .....	102
7.2.3	Relational intrinsic operations .....	102
7.2.4	Logical intrinsic operations .....	104
7.3	Interpretation of defined operations .....	104
7.3.1	Unary defined operation .....	104

	7.3.2	Binary defined operation .....	105
7.4		Precedence of operators .....	105
7.5		Assignment .....	107
	7.5.1	Assignment statement .....	107
	7.5.2	Pointer assignment .....	110
	7.5.3	Masked array assignment - WHERE .....	111
	7.5.4	FORALL .....	114
8		Execution control .....	121
8.1		Executable constructs containing blocks .....	121
	8.1.1	Rules governing blocks .....	121
	8.1.2	IF construct .....	122
	8.1.3	CASE construct .....	123
	8.1.4	DO construct .....	126
8.2		Branching .....	130
	8.2.1	Statement labels .....	130
	8.2.2	GO TO statement .....	131
	8.2.3	Computed GO TO statement .....	131
	8.2.4	Arithmetic IF statement .....	131
8.3		CONTINUE statement .....	131
8.4		STOP statement .....	131
9		Input/output statements .....	133
9.1		Records .....	133
	9.1.1	Formatted record .....	133
	9.1.2	Unformatted record .....	133
	9.1.3	Endfile record .....	134
9.2		Files .....	134
	9.2.1	External files .....	134
	9.2.2	Internal files .....	137
9.3		File connection .....	138
	9.3.1	Unit existence .....	138
	9.3.2	Connection of a file to a unit .....	138
	9.3.3	Preconnection .....	139
	9.3.4	The OPEN statement .....	139
	9.3.5	The CLOSE statement .....	143
9.4		Data transfer statements .....	144
	9.4.1	Control information list .....	144
	9.4.2	Data transfer input/output list .....	148
	9.4.3	Error, end-of-record, and end-of-file conditions .....	149
	9.4.4	Execution of a data transfer input/output statement .....	150
	9.4.5	Printing of formatted records .....	153
	9.4.6	Termination of data transfer statements .....	154
9.5		File positioning statements .....	154
	9.5.1	BACKSPACE statement .....	154
	9.5.2	ENDFILE statement .....	155
	9.5.3	REWIND statement .....	155
9.6		File inquiry .....	155
	9.6.1	Inquiry specifiers .....	156
	9.6.2	Restrictions on inquiry specifiers .....	160
	9.6.3	Inquire by output list .....	160
9.7		Restrictions on function references and list items .....	160

9.8	Restriction on input/output statements . . . . .	160
10	Input/output editing . . . . .	161
10.1	Explicit format specification methods . . . . .	161
10.1.1	FORMAT statement . . . . .	161
10.1.2	Character format specification . . . . .	161
10.2	Form of a format item list. . . . .	162
10.2.1	Edit descriptors . . . . .	162
10.2.2	Fields . . . . .	164
10.3	Interaction between input/output list and format . . . . .	164
10.4	Positioning by format control . . . . .	165
10.5	Data edit descriptors. . . . .	165
10.5.1	Numeric editing . . . . .	165
10.5.2	Logical editing . . . . .	170
10.5.3	Character editing . . . . .	170
10.5.4	Generalized editing . . . . .	170
10.6	Control edit descriptors . . . . .	171
10.6.1	Position editing . . . . .	172
10.6.2	Slash editing . . . . .	173
10.6.3	Colon editing . . . . .	173
10.6.4	S, SP, and SS editing. . . . .	173
10.6.5	P editing . . . . .	173
10.6.6	BN and BZ editing . . . . .	174
10.7	Character string edit descriptors. . . . .	174
10.8	List-directed formatting . . . . .	174
10.8.1	List-directed input . . . . .	175
10.8.2	List-directed output. . . . .	177
10.9	Namelist formatting . . . . .	178
10.9.1	Namelist input . . . . .	179
10.9.2	Namelist output . . . . .	182
11	Program units . . . . .	185
11.1	Main program . . . . .	185
11.1.1	Main program specifications . . . . .	185
11.1.2	Main program executable part. . . . .	186
11.1.3	Main program internal subprograms . . . . .	186
11.2	External subprograms. . . . .	186
11.3	Modules . . . . .	186
11.3.1	Module reference . . . . .	187
11.3.2	The USE statement and use association . . . . .	187
11.4	Block data program units. . . . .	189
12	Procedures . . . . .	191
12.1	Procedure classifications . . . . .	191
12.1.1	Procedure classification by reference. . . . .	191
12.1.2	Procedure classification by means of definition. . . . .	191
12.2	Characteristics of procedures. . . . .	192
12.2.1	Characteristics of dummy arguments . . . . .	192
12.2.2	Characteristics of function results . . . . .	192
12.3	Procedure interface . . . . .	192
12.3.1	Implicit and explicit interfaces . . . . .	192

12.3.2	Specification of the procedure interface . . . . .	193
12.4	Procedure reference . . . . .	198
12.4.1	Actual arguments, dummy arguments, and argument association . . . . .	199
12.4.2	Function reference . . . . .	205
12.4.3	Subroutine reference . . . . .	206
12.5	Procedure definition . . . . .	206
12.5.1	Intrinsic procedure definition . . . . .	206
12.5.2	Procedures defined by subprograms . . . . .	206
12.5.3	Definition of procedures by means other than Fortran . . . . .	211
12.5.4	Statement function . . . . .	211
12.6	Pure procedures . . . . .	212
12.7	Elemental procedures . . . . .	213
12.7.1	Elemental procedure declaration and interface . . . . .	213
12.7.2	Elemental function actual arguments and results . . . . .	214
12.7.3	Elemental subroutine actual arguments . . . . .	214
13	Intrinsic procedures . . . . .	217
13.1	Intrinsic functions . . . . .	217
13.2	Elemental intrinsic procedures . . . . .	217
13.3	Arguments to intrinsic procedures . . . . .	217
13.4	Argument presence inquiry function . . . . .	218
13.5	Numeric, mathematical, character, kind, logical, and bit procedures . . . . .	218
13.5.1	Numeric functions . . . . .	218
13.5.2	Mathematical functions . . . . .	218
13.5.3	Character functions . . . . .	218
13.5.4	Character inquiry function . . . . .	218
13.5.5	Kind functions . . . . .	218
13.5.6	Logical function . . . . .	218
13.5.7	Bit manipulation and inquiry procedures . . . . .	219
13.6	Transfer function . . . . .	219
13.7	Numeric manipulation and inquiry functions . . . . .	219
13.7.1	Models for integer and real data . . . . .	219
13.7.2	Numeric inquiry functions . . . . .	220
13.7.3	Floating point manipulation functions . . . . .	220
13.8	Array intrinsic functions . . . . .	220
13.8.1	The shape of array arguments . . . . .	220
13.8.2	Mask arguments . . . . .	221
13.8.3	Vector and matrix multiplication functions . . . . .	221
13.8.4	Array reduction functions . . . . .	221
13.8.5	Array inquiry functions . . . . .	221
13.8.6	Array construction functions . . . . .	221
13.8.7	Array reshape function . . . . .	221
13.8.8	Array manipulation functions . . . . .	222
13.8.9	Array location functions . . . . .	222
13.9	Pointer association status functions . . . . .	222
13.10	Intrinsic subroutines . . . . .	222
13.10.1	Date and time subroutines . . . . .	222
13.10.2	Pseudorandom numbers . . . . .	222
13.10.3	Bit copy subroutine . . . . .	222
13.11	Generic intrinsic functions . . . . .	223
13.11.1	Argument presence inquiry function . . . . .	223
13.11.2	Numeric functions . . . . .	223
13.11.3	Mathematical functions . . . . .	223

13.11.4	Character functions . . . . .	224
13.11.5	Character inquiry function . . . . .	224
13.11.6	Kind functions . . . . .	224
13.11.7	Logical function . . . . .	224
13.11.8	Numeric inquiry functions . . . . .	224
13.11.9	Bit inquiry function . . . . .	225
13.11.10	Bit manipulation functions . . . . .	225
13.11.11	Transfer function . . . . .	225
13.11.12	Floating-point manipulation functions . . . . .	225
13.11.13	Vector and matrix multiply functions . . . . .	225
13.11.14	Array reduction functions . . . . .	225
13.11.15	Array inquiry functions . . . . .	226
13.11.16	Array construction functions . . . . .	226
13.11.17	Array reshape function . . . . .	226
13.11.18	Array manipulation functions . . . . .	226
13.11.19	Array location functions . . . . .	226
13.11.20	Pointer association status functions . . . . .	226
13.12	Intrinsic subroutines . . . . .	226
13.13	Specific names for intrinsic functions . . . . .	227
13.14	Specifications of the intrinsic procedures . . . . .	228
14	Scope, association, and definition . . . . .	275
14.1	Scope of names . . . . .	275
14.1.1	Global entities . . . . .	275
14.1.2	Local entities . . . . .	275
14.1.3	Statement and construct entities . . . . .	280
14.2	Scope of labels . . . . .	281
14.3	Scope of external input/output units . . . . .	281
14.4	Scope of operators . . . . .	281
14.5	Scope of the assignment symbol . . . . .	281
14.6	Association . . . . .	281
14.6.1	Name association . . . . .	282
14.6.2	Pointer association . . . . .	284
14.6.3	Storage association . . . . .	285
14.7	Definition and undefinition of variables . . . . .	288
14.7.1	Definition of objects and subobjects . . . . .	288
14.7.2	Variables that are always defined . . . . .	288
14.7.3	Variables that are initially defined . . . . .	288
14.7.4	Variables that are initially undefined . . . . .	288
14.7.5	Events that cause variables to become defined . . . . .	288
14.7.6	Events that cause variables to become undefined . . . . .	290
A.	Glossary of technical terms . . . . .	293
B.	Decremental features . . . . .	303
B.1	Deleted features . . . . .	303
B.1.1	Real and double precision DO variables . . . . .	303
B.1.2	Branching to an END IF statement from outside its IF block . . . . .	304
B.1.3	PAUSE statement . . . . .	304
B.1.4	ASSIGN, assigned GO TO, and assigned FORMAT . . . . .	304
B.1.5	H edit descriptor . . . . .	306
B.2	Obsolescent features . . . . .	306



B.2.1	Alternate return .....	306
B.2.2	Computed GO TO statement .....	307
B.2.3	Statement functions .....	307
B.2.4	DATA statements among executables .....	307
B.2.5	Assumed character length functions .....	307
B.2.6	Fixed form source .....	307
B.2.7	CHARACTER* form of CHARACTER declaration .....	307
C.	Extended notes .....	309
C.1	Section 4 notes .....	309
C.1.1	Intrinsic and derived data types (4.3, 4.4) .....	309
C.1.2	Selection of the approximation methods (4.3.1.2) .....	310
C.1.3	Pointers (4.4.1) .....	311
C.2	Section 5 notes .....	312
C.2.1	The POINTER attribute (5.1.2.7) .....	312
C.2.2	The TARGET attribute (5.1.2.8) .....	312
C.3	Section 6 notes .....	313
C.3.1	Structure components (6.1.2) .....	313
C.3.2	Pointer allocation and association .....	314
C.4	Section 7 notes .....	314
C.4.1	Character assignment .....	314
C.4.2	Evaluation of function references .....	315
C.4.3	Pointers in expressions .....	315
C.4.4	Pointers on the left side of an assignment .....	315
C.4.5	An example of a FORALL construct containing a WHERE construct .....	316
C.4.6	Examples of FORALL statements .....	316
C.5	Section 8 notes .....	317
C.5.1	Loop control .....	317
C.5.2	The CASE construct .....	317
C.5.3	Additional examples of DO constructs .....	317
C.5.4	Examples of invalid DO constructs .....	319
C.6	Section 9 notes .....	319
C.6.1	Files (9.2) .....	319
C.6.2	OPEN statement (9.3.4) .....	322
C.6.3	Connection properties (9.3.2) .....	323
C.6.4	CLOSE statement (9.3.5) .....	324
C.6.5	INQUIRE statement (9.6) .....	325
C.7	Section 10 notes .....	325
C.7.1	Number of records (10.3, 10.4, 10.6.2) .....	325
C.7.2	List-directed input (10.8.1) .....	326
C.8	Section 11 notes .....	327
C.8.1	Main program and block data program unit (11.1, 11.4) .....	327
C.8.2	Dependent compilation (11.3) .....	327
C.8.3	Examples of the use of modules .....	329
C.9	Section 12 notes .....	334
C.9.1	Portability problems with external procedures (12.3.2.2) .....	334
C.9.2	Procedures defined by means other than Fortran (12.5.3) .....	334
C.9.3	Procedure interfaces (12.3) .....	335
C.9.4	Argument association and evaluation (12.4.1.1) .....	335
C.9.5	Pointers and targets as arguments (12.4.1.1) .....	336
C.10	Section 14 notes .....	337
C.10.1	Examples of host association (14.6.1.3) .....	337
C.11	Array feature notes .....	338

C.11.1	Summary of features .....	338
C.11.2	Examples .....	339
C.11.3	FORmula TRANslation and array processing .....	343
C.11.4	Sum of squared residuals .....	344
C.11.5	Vector norms: infinity-norm and one-norm .....	344
C.11.6	Matrix norms: infinity-norm and one-norm .....	344
C.11.7	Logical queries .....	344
C.11.8	Parallel computations .....	345
C.11.9	Example of element-by-element computation .....	345
C.11.10	Bit manipulation and inquiry procedures .....	346
D.	Index .....	347

iTeh STANDARD PREVIEW  
(standards.iteh.ai)

[ISO/IEC 1539-1:1997](https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997)

<https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997>

## Tables

Table 2.1	Requirements on statement ordering .....	13
Table 2.2	Statements allowed in scoping units .....	14
Table 3.1	Special characters .....	22
Table 6.1	Subscript order value .....	77
Table 7.1	Type of operands and results for intrinsic operators .....	89
Table 7.2	Type, type parameters, and rank of the result of NULL (). .....	91
Table 7.3	Interpretation of the numeric intrinsic operators .....	101
Table 7.4	Interpretation of the character intrinsic operator // .....	102
Table 7.5	Interpretation of the relational intrinsic operators .....	103
Table 7.6	Interpretation of the logical intrinsic operators .....	104
Table 7.7	The values of operations involving logical intrinsic operators .....	104
Table 7.8	Categories of operations and relative precedence .....	105
Table 7.9	Type conformance for the intrinsic assignment statement .....	108
Table 7.10	Numeric conversion and the assignment statement .....	108
Table C.1	Values assigned to INQUIRE specifier variables .....	325

iTeh STANDARD PREVIEW  
(standards.iteh.ai)

ISO/IEC 1539-1:1997

<https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997>

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

ISO/IEC 1539-1:1997

<https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997>

## 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 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 edition cancels and replaces ISO/IEC 1539:1991, 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*

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

[ISO/IEC 1539-1:1997](https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997)

<https://standards.iteh.ai/catalog/standards/sist/291492a3-4273-4d02-ae66-3081c5d2d453/iso-iec-1539-1-1997>

## Introduction

### *Standard programming language Fortran*

This part of the international standard comprises the specification of the base Fortran language. With the limitations noted in 1.5.1, and the deletions described in Annex B, the syntax and semantics of Fortran 90 are contained entirely within Fortran 95. Therefore, any standard-conforming Fortran 90 program not containing deleted features or affected by such limitations is a standard conforming Fortran 95 program. New features of Fortran 95 can be compatibly incorporated into such Fortran 90 programs, with any exceptions indicated in the text of this part of the standard.

Fortran 95 continues the evolutionary model introduced in Fortran 90 by deleting several of the features marked as obsolescent in Fortran 90 and identifying a few newly-obsolescent features (Annex B).

Fortran 95 is a relatively minor evolution of standard Fortran, with the emphasis in this revision being upon correcting defects in the Fortran 90 standard, including providing interpretation to a number of questions that have arisen concerning Fortran 90 semantics and syntax (e.g., whether blanks are permitted within edit descriptors in free source form). In addition to such corrections and clarifications, Fortran 95 contains several extensions to Fortran 90; there are three major extensions:

- (1) The FORALL statement and construct
- (2) PURE and ELEMENTAL procedures
- (3) Pointer initialization and structure default initialization

### *FORALL*

The Fortran 90 array constructor and SPREAD and RESHAPE intrinsic functions are powerful tools for element-by-element construction of an array value. Their use in combination, which is required for many array values, can be awkward. Fortran 95 therefore provides a simple and efficient alternative: the FORALL statement allows array elements, array sections, character substrings, or pointer targets to be explicitly specified as a function of the element subscripts. The form of the FORALL statement is very much like a functionally equivalent set of nested DO loops for computing and assigning the elements of an array, except that conceptually all elements are computed simultaneously and then assigned simultaneously. An added benefit of FORALL is that it simplifies conversion from sequential DO loops to parallel array operations. A FORALL construct allows several such array assignments to share the same element subscript control. This control includes masking in a manner similar to the masking facilities of WHERE, the main difference between WHERE and FORALL being that FORALL makes use of element subscripts whereas WHERE is whole array oriented.

### *PURE*

As has always been the case in Fortran, Fortran 95 functions may have side effects (e.g., change the value of an argument or a global variable). Side effects cause problems in parallel processing, however, and because parallel processing has become an important high performance technology, Fortran 95 makes it possible to specify a function to be side effect free. Such a function is called "pure" and is declared with the keyword PURE in the function statement. A restricted form of PURE functions may be called elementally; such ELEMENTAL functions are especially important to high performance parallel processing. An added advantage of pure functions is that it is reasonable to allow them in specification expressions; this provides a significant amount of functionality, with very little cost, and therefore this capability has also been included in Fortran 95.

*Initialization*

In Fortran 90 there was no way to define the initial pointer association status — a pointer has to be explicitly nullified, allocated, or associated with a target during execution before it can be tested by the ASSOCIATED intrinsic function. This limits the usefulness of pointers, especially the use of pointers as derived-type components. Fortran 95 therefore solves this problem by providing (a) a NULL intrinsic function that may be used to nullify a pointer and (b) a means to specify default initial values for derived-type components. In the latter case the specification of initial values is part of the derived-type definition, and objects declared of this type automatically have all their components so initialized.

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

This part of ISO/IEC 1539 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 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 (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; 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.