

## Annex D

(informative)

## Index

In this index, entries in *italics* denote BNF terms, entries in **bold face** denote language keywords, and page numbers in **bold** denote primary or defining text.

### Symbols

! 26  
 - 109, 122  
 % 94  
 & 26  
 & **224**  
 \* 109, 122  
 \* (symbol) 64, 67, 72, 80, 168, 177, **220**, **224**, 248, 261  
 \*\* 108, 122  
 + 109, 122  
 . 108, 110  
 .AND. 110, 125  
 .EQ. 109, 124  
 .EQV. 110, 125  
 .FALSE. 38  
 .GE. 109, 124  
 .GT. 109, 124  
 .LE. 109, 124  
 .LT. 109, 124  
 .NE. 109, 124  
 .NEQV. 110, 125  
 .NOT. 110, 125  
 .OR. 110, 125  
 .TRUE. 38  
 / 109, 122, 207  
 / (symbol) 79, 82, 86, 89  
 // 36, 109, 123  
 /= 109, 124  
 : 71, 94, 97  
 :: 63  
 ; 26  
 < 109, 124  
 <= 109, 124  
 = 129, 240, 248, 265  
 == 109, 124  
 => (pointer assignment) 64, 133  
 => (rename) 233

> 109, 124  
 >= 110, 124

### A

abstract interface **241**  
 abstract interface block **241**  
**ACCESS=** specifier **172**, **197**  
 accessibility attribute **69**  
 accessibility statements **78**  
*access-spec* 40, **69**  
*access-stmt* **78**  
**ACTION=** specifier **172**, **197**  
*actual-arg* **248**  
**ADVANCE=** specifier **178**  
 affector **178**  
**ALLOCATABLE** 40, 63, 78  
 allocatable array **72**  
 ALLOCATABLE attribute **69**  
 ALLOCATABLE statement **78**  
*allocatable-stmt* **78**  
**ALLOCATE** 100  
 ALLOCATE statement **100**  
*allocate-stmt* **100**  
*alt-return-spec* **248**  
**APOSTROPHE** (DELIM value) **229**  
 argument association **249**, 374  
 argument keyword **249**  
 argument keywords 19, 271, 372  
 arithmetic IF statement **159**  
*arithmetic-if-stmt* **159**  
 array **17**, **70–73**, **96–99**  
     assumed-shape **71**  
     assumed-size **72**  
     automatic **71**  
     explicit-shape **71**  
 array constructor **60**

array element 17, 97  
 array element order 97  
 array pointer 72  
 array section 17, 98  
*array-constructor* 60  
*array-element* 96  
*array-section* 96  
*array-spec* 70  
 ASCII 322  
 ASCII collating sequence 37  
**ASSIGNMENT** 240  
 assignment 129–143  
     defined 130  
     elemental array (FORALL) 138  
     intrinsic 129  
     masked array (WHERE) 135  
     pointer 133  
 assignment statement 129  
*assignment-stmt* 129  
 associating entity 381  
 association 19  
     argument 249, 374  
     host 374  
     name 374  
     pointer 377  
     sequence 253  
     storage 379  
     use 374  
 association status  
     pointer 377  
 assumed type parameter 31  
 assumed-shape array 71  
 assumed-size array 72  
*asynchronous-stmt* 78  
 ASYNCHRONOUS attribute 69  
 ASYNCHRONOUS statement 78  
**ASYNCHRONOUS=** specifier 172, 197  
**ASYNCHRONOUS** 63  
 attribute specification statements 77–92  
 attributes 69–??  
     accessibility 69  
     ALLOCATABLE 69  
     ASYNCHRONOUS 69  
     DIMENSION 70  
     EXTERNAL 73  
     INTENT 73  
     INTRINSIC 75  
     OPTIONAL 75  
     PARAMETER 75  
     POINTER 76  
     PRIVATE 69  
     PUBLIC 69  
     SAVE 76, 79  
     TARGET 76

*attr-spec* 63  
 automatic array 71  
 automatic data object 65

## B

**BACKSPACE** 194  
 BACKSPACE statement 195  
*backspace-stmt* 194  
 base object 95  
 base type 51  
**BIND** 70  
 BIND statement 78  
**BIND(C)** 38, 58, 238, 239, 263, 358, 359, 361  
 binding 46  
 binding label 263  
**BINDNAME=** specifier 70, 264  
*bind-spec* 70  
*bind-stmt* 78  
 bit model 272  
 blank common 89  
**BLANK=** specifier 172, 198  
*blank-interp-edit-desc* 207  
 block 145  
*block* 145  
**BLOCK DATA** 235  
 block data program 235  
*block-data* 235  
*block-data-stmt* 235  
*block-do-construct* 154  
*bounds-spec* 133  
*boz-literal-constant* 32  
 branch target statement 158

## C

C\_(C type) 355–362  
 C\_LOC function 358  
 CALL statement 247  
*call-stmt* 248  
**CASE** 148  
 CASE construct 147  
*case-construct* 147  
*case-stmt* 148  
 CHAR intrinsic 37  
**CHARACTER** 36, 63  
*character* 21  
 character context 25  
 character intrinsic operation 123  
 character literal constant 36  
 character sequence type 49

character set 21  
 character string 35  
 character type 35–38  
 CHARACTER type specifier 67  
 characteristics of a procedure 238  
*char-constant* 23  
*char-expr* 112  
*char-literal-constant* 36  
*char-string-edit-desc* 208  
 child data transfer statement 187–192, 204  
**CLASS** 63  
**CLOSE** 174  
 CLOSE statement 174  
*close-stmt* 174  
 collating sequence 37  
 comment 26, 27  
**COMMON** 89  
 common association 90  
 common block 89, 367, 426  
 common block storage sequence 90  
 COMMON statement 89–92  
*common-block-name* 89  
*common-stmt* 89  
 companion processor 20  
 compatibility  
     FORTRAN 77 3  
     Fortran 90 3  
     Fortran 95 3  
**COMPLEX** 35, 63  
 complex type 35  
 COMPLEX type specifier 66  
*complex-literal-constant* 35  
*component-def-stmt* 39  
 components 372  
 computed GO TO statement 159  
*computed-goto-stmt* 159  
 concatenation 36  
 conform 129  
 conformable 18  
 conformance 114  
 connected files 169  
 constant 17, 23, 29  
     character 36  
     integer 32  
     logical 38  
     named 82  
*constant* 23  
 constant subobject 17  
 Construct association 377  
 constructor  
     array 60  
     derived-type 54  
     structure 54  
**CONTAINS** 232, 263

CONTAINS statement 263  
*contains-stmt* 263  
 continuation 26, 27  
**CONTINUE** 159  
 CONTINUE statement 159  
*continue-stmt* 159  
 control edit descriptors 217  
*control-edit-desc* 207  
 conversion  
     numeric 130  
 current record 165  
**CYCLE** 157  
 CYCLE statement 154, 157  
*cycle-stmt* 157

## D

**DATA** 79  
 data edit descriptors 209–216  
 data object 16  
 data object reference 19  
 DATA statement 79, 382  
 data transfer 185  
 data transfer statements 175  
 data type 15, 29–61  
     *see also* type  
 data type of a primary 113  
 data type of an expression 112  
 data type of an operation 113  
 data type  
     concept 29  
*data-edit-desc* 206  
*data-implied-do* 79  
*data-ref* 94  
*data-stmt* 79  
**DEALLOCATE** 104  
 DEALLOCATE statement 104  
*deallocate-stmt* 104  
 decimal symbol 209  
**DECIMAL=** specifier 176, 179  
 declaration 19  
 declarations 63–92  
*declaration-type-spec* 63  
 declared binding 248  
 declared type 68  
**DEFAULT** 148  
 default character 36  
 default complex 35  
 default integer 32  
 default logical 38  
 default real 34  
*default-char-exp* 113

- default-initialized 44
- deferred type parameter 30
- deferred-shape array 71
- defined 19
- defined assignment 244
- defined assignment statement 130
- defined operation 112, 125, 243
- defined-binary-op* 110
- defined-operator* 24
- defined-unary-op* 108
- definition 19
- definition of variables 382
- DELIM=** specifier 172, 198
- derived type determination 50
- derived type type specifier 68
- derived types 16, 38–56
- derived-type-def* 39
- designator* 93
- digit-string* 32
- DIMENSION** 40, 63, 81
- DIMENSION** attribute 70
- DIMENSION** statement 81
- dimension-stmt* 81
- direct access 163
- direct access input/output statement 180
- DIRECT=** specifier 198
- disassociated 18
- DO** 154
- DO** construct 154
- DO** statement 154
- DO WHILE** statement 154
- do-construct* 154
- do-stmt* 154
- DOUBLE PRECISION** 34, 63
- double precision real 34
- DOUBLE PRECISION** type specifier 66
- dummy arguments
  - restrictions 255
- dummy procedure 237
- dummy-arg* 261
- dynamic binding 248
- dynamic type 68

## E

- edit descriptors *see* format descriptors
- effective item 181
- element array assignment (**FORALL**) 138
- ELEMENTAL** 259
- elemental intrinsic procedure 271
- elemental procedure 267
- ELSE** 146

- else-if-stmt* 146
- else-stmt* 146
- ELSEWHERE** 136
- elsewhere-stmt* 136
- END** 231
- END** statement 14
- END=** specifier 203
- ENDFILE** 194
- endfile record 162
- ENDFILE** statement 162, 195
- endfile-stmt* 194
- end-of-file condition 182
- end-of-record condition 182
- end-program-stmt* 231
- entity-decl* 64
- ENTRY** 261
- entry-stmt* 261
- ENUM** 58
- enum-alias-def* 58
- enum-def-stmt* 58
- enumeration 58
- ENUMERATOR** 58
- enumerator 58
- enumerator-def-stmt* 58
- EOR=** specifier 203
- EQUIVALENCE** 87
- EQUIVALENCE** statement 87–89
- equivalence-stmt* 87
- ERR=** specifier 203
- ERRMSG=** specifier 104
- ERRMSG=** specifier 100, 104
- ERROR\_UNIT** 169, 332
- evaluation
  - operations 117
  - optional 118
  - parentheses 119
- executable constructs 145
- executable-construct* 11
- execution control 145–159
- EXIST=** specifier 198
- EXIT** 157
- EXIT** statement 157
- exit-stmt* 157
- explicit formatting 205–220
- explicit initialization 65, 79
- explicit interface 239
- explicit-shape array 71
- explicit-shape-spec* 71
- expr* 110
- expressions 17, 107–128
- extended type 51
- EXTENDS** attribute 39, 51
- EXTENSIBLE** attribute 39, 51
- extensible type 51

extension operation 112, 126  
 extension operator 112  
 extension type 51  
 extent 18  
**EXTERNAL** 63, 245  
 EXTERNAL attribute 73  
 external file 162  
 external procedure 12, 237  
 EXTERNAL statement 245  
 external subprogram 11  
*external-stmt* 245  
*external-subprogram* 9

## F

file access 163  
 file connection 168  
 file inquiry 196  
 file position 165  
 file positioning statements 194  
 file storage unit 166  
**FILE=** specifier 172, 197  
 files  
   connected 169  
   external 162  
   internal 167  
   preconnected 170  
 final subroutines 49  
 finalization 56  
 fixed source form 27  
**FORALL** 138  
 FORALL construct 138  
*forall-construc* 138  
**FORM=** specifier 173, 198  
**FORMAT** 205  
*format* 177  
 format descriptors  
   / 218  
   : 218  
   A 214  
   B 210  
   BN 219  
   BZ 219  
   control edit descriptors 217  
   D 212  
   data edit descriptors 209–216  
   E 212  
   EN 212  
   ES 213  
   F 211  
   G 214, 215  
   I 210

L 214  
 O 210  
 P 218  
 S 218  
 SP 218  
 SS 218  
 TL 217  
 TR 217  
 X 217  
 Z 210  
 FORMAT statement 177, 205  
*format-item* 206  
*format-specification* 205  
*format-stmt* 205  
 formatted data transfer 187  
 formatted input/output statement 177  
 formatted record 161  
**FORMATTED=** specifier 198  
 formatting  
   explicit 205–220  
   list-directed 192, 220–224  
   namelist 192, 224–229  
 FORTRAN 77 compatibility 3  
 Fortran 90 compatibility 3  
 Fortran 95 compatibility 3  
 free source form 25  
**FUNCTION** 258  
 function 12  
 function reference 17, 257  
 FUNCTION statement 258  
*function-reference* 247  
*function-stmt* 258  
*function-subprogram* 9, 258

## G

generic identifier 242  
 generic interface 46, 242  
 generic interface block 241  
 generic name 243  
 generic procedure references 367  
*generic-spec* 240  
 global entities 365  
**GO TO** 159  
 GO TO statement 159  
*goto-stmt* 159

## H

host 13, 232  
 host association 374

host scoping unit 12

## I

ICHAR intrinsic 37  
**ID=** specifier 198  
 IEEE\_ 280, 333–354  
**IF** 146, 147, 159  
 IF construct 146  
 IF statement 147  
*if-construct* 146  
*if-stmt* 147  
*if-then-stmt* 146  
 imaginary part 35  
**IMPLICIT** 83  
 implicit interface 247  
**IMPLICIT NONE** 83  
 IMPLICIT statement 83  
*implicit-stmt* 83  
 implied-DO 60, 79, 180  
**IMPORT** 240  
**IN** 73  
**INCLUDE** 28  
 INCLUDE line 28  
 inheritance association 52, 381  
 inherited 51  
 initial point 165  
 initialization 44, 64, 65, 382  
*initialization* 64  
 initialization expression 116  
*initialization-expr* 116  
**INOUT** 73  
 input/output editing 205–229  
 input/output list 180  
 input/output statements 161–204  
**INPUT\_UNIT** 168, 332  
*input-item* 180  
**INQUIRE** 196  
 INQUIRE statement 196  
*inquire-stmt* 196  
 inquiry function 271  
 inquiry, type parameter 95  
*int-constant* 23  
**INTEGER** 32, 63  
 integer constant 32  
 integer editing 210  
 integer model 272  
 integer type 32–33  
 INTEGER type specifier 66  
**INTENT** 63, 81, 272  
 INTENT attribute 73  
 INTENT statement 81

*intent-spec* 73  
*intent-stmt* 81  
**INTERFACE** 240  
 interface  
   (procedure) 238  
   abstract 241  
   explicit 239  
   generic 242  
   implicit 247  
*interface-block* 240  
*interface-body* 240  
 internal files 167  
 internal procedure 13, 237  
 internal subprogram 11  
*internal-subprogram* 10  
 interoperable 356–362  
*int-expr* 113  
*int-literal-constant* 32  
**INTRINSIC** 63, 233, 247  
 intrinsic 20  
   elemental 271  
   function 271  
   inquiry function 271  
   subroutine 274  
   transformational 271  
 intrinsic assignment statement 129  
 INTRINSIC attribute 75  
 intrinsic data types 31–38  
 intrinsic operation 111  
 intrinsic operations 122–125  
   logical 38  
 intrinsic procedures 280–331  
   *see alphabetical listing, ch. 13*  
 INTRINSIC statement 247  
 intrinsic type 15  
*intrinsic-operator* 23  
*intrinsic-stmt* 247  
**IOSTAT=** specifier 202  
*io-unit* 168  
 ISO 10646 322  
 ISO\_C\_BINDING module 355  
 ISO\_FORTRAN\_ENV module 168, 202, 331

## K

keyword 19, 249  
*keyword* 19  
**KIND** 64, 67  
**KIND** attribute 39  
 KIND intrinsic 32, 33, 35, 36, 38  
 kind type parameter 30, 32, 33, 35, 36, 38  
*kind-param* 32

## L

label 373  
*label* 24  
*language-binding-spec* 70  
**LEN** 67  
length 35  
line 25  
list-directed formatting 192, 220–224  
list-directed input/output statement 177  
literal constant 17, 93  
*literal-constant* 23  
local entities 366  
**LOGICAL** 38, 63  
logical constant 38  
logical intrinsic operations 38, 125  
logical type 38  
**LOGICAL** type specifier 68  
*logical-exp* 112  
*logical-literal-constant* 38  
loop 154

## M

main program 12, 231  
*main-program* 9, 231  
many-one array section 99  
masked array assignment (WHERE) 135  
model  
    bit 272  
    integer 272  
    real 272  
**MODULE** 232  
module 13, 232  
*module* 9  
**MODULE PROCEDURE** 240  
module procedure 12  
module reference 19, 233  
module subprogram 12  
*module-subprogram* 10

## N

name 18  
*name* 23  
name association 374  
**NAME=** specifier  
    in the INQUIRE statement 199  
    in the *language-binding-spec* 70  
named common block 89  
named constant 17, 75, 82, 93

**NAMED=** specifier 199  
*named-constant* 23  
**NAMELIST** 86  
namelist formatting 192, 224–229  
namelist input/output statement 178  
**NAMELIST** statement 86  
*namelist-stmt* 86  
NaN 280, 337  
**NEXTREC=** specifier 199  
**NML=** specifier 178  
**NON\_INTRINSIC** 233  
**NON\_OVERRIDABLE** attribute 42, 53  
*nonblock-do-construct* 155  
**NONE** *see* **IMPLICIT NONE**  
**NONE** (DELIM value) 228  
**NONKIND** attribute 39  
nonkind type parameter 30  
normal 337  
**NULL** intrinsic 40, 46, 64  
**NULLIFY** 104  
**NULLIFY** statement 104  
*nullify-stmt* 104  
**NUMBER=** specifier 200  
numeric conversion 130  
numeric editing 210  
numeric intrinsic operations 122  
numeric sequence type 49  
numeric storage unit 379  
*numeric-exp* 113

## O

object -- *see* data object  
object designator 18  
**ONLY** 233  
**OPEN** 171  
**OPEN** statement 170  
**OPENED=** specifier 200  
*open-stmt* 171  
operations 30  
    character intrinsic 123  
    defined 125  
    logical intrinsic 125  
    numeric intrinsic 122  
    relational intrinsic 123  
**OPERATOR** 240  
operator precedence 126  
operators 23  
**OPTIONAL** 64, 81  
**OPTIONAL** attribute 75  
optional dummy argument 254  
**OPTIONAL** statement 81

*optional-stmt* 81  
**OUT** 73  
**OUTPUT\_UNIT** 168, 332  
*output-item* 180  
 override 52

## P

**PAD=** specifier 173, 200  
**PARAMETER** 17  
**PARAMETER** 64, 82  
**PARAMETER** attribute 75  
**PARAMETER** statement 82  
*parameter-stmt* 82  
 parent data transfer statement 187–192, 204  
 parent type 51  
 parentheses 119  
 partially [storage] associated 380  
*part-ref* 94  
**PASS\_OBJ** attribute 40, 41, 47, 250  
 passed-object dummy argument 47  
**PENDING=** specifier 198  
**POINTER** 40, 64, 82  
 pointer assignment 133  
 pointer association 377  
 pointer association status 377  
**POINTER** attribute 76  
**POINTER** statement 82  
*pointer-assignment-stmt* 133  
*pointer-stmt* 82  
 polymorphic 68  
**POS=** specifier 200  
**POSITION=** specifier 173, 200  
 positional arguments 271  
*position-edit-desc* 207  
 precedence of operators 126  
**PRECISION** intrinsic 33  
 preconnected files 170  
 pre-existing entity 381  
*prefix* 258  
 present (dummy argument) 254  
**PRESENT** intrinsic 75  
 primary 108  
*primary* 108  
**PRINT** 175  
**PRINT** statement 175  
*print-stmt* 175  
**PRIVATE** 39, 69  
**PRIVATE** attribute 69  
**PRIVATE** statement 78, 233  
**PROCEDURE** 240  
**PROCEDURE** 246

procedure 12  
   characteristics of 238  
   dummy 237  
   elemental 267  
   external 237  
   internal 237  
   intrinsic 271–331  
   non-Fortran 264  
   pure 265  
 procedure interface 238  
 procedure pointer 246  
 procedure reference 19, 247  
 procedure references  
   generic 367  
   resolving 368  
*procedure-declaration-stmt* 246  
*procedure-stmt* 240  
 processor 1  
**PROGRAM** 231  
 program 12  
*program* 9  
 program name 231  
 program unit 11  
*program-stmt* 231  
*program-unit* 9  
**PUBLIC** 69  
**PUBLIC** attribute 69  
**PUBLIC** statement 78, 233  
**PURE** 259  
 pure procedure 265

## Q

**QUOTE** (DELIM value) 229

## R

**RANGE** intrinsic 32, 33  
 rank 17, 18  
**READ** 175  
**READ** statement 175  
**READ=** specifier 200  
*read-stmt* 175  
**READWRITE=** specifier 200  
**REAL** 34, 63  
 real and complex editing 211  
 real model 272  
 real part 35  
 real type 33–34  
**REAL** type specifier 66  
*real-literal-constant* 34



**REC=** specifier 180  
**RECL=** specifier 173, 201  
 record 161  
**RECURSIVE** 259, 261  
 relational intrinsic operations 123  
*rename* 233  
 repeat specification 206  
 resolving procedure references 368  
 restricted expression 114  
**RESULT** 258, 261  
 result variable 12  
**RETURN** 263  
 RETURN statement 263  
*return-stmt* 263  
**REWIND** 194  
 REWIND statement 195  
*rewind-stmt* 194  
**ROUND=** specifier 173  
**ROUND= specifier** 201

## S

**SAVE** 64, 82  
 SAVE attribute 76, 79  
 SAVE statement 82  
 saved object 76  
*save-stmt* 82  
 scalar 17, 93  
 scale factor 207  
 scope of names 365  
 scoping unit 12  
*section-subscript* 96  
**SELECT CASE** 148  
 SELECT CASE statement 147  
**SELECT TYPE** construct 150, 373, 377  
*select-case-stmt* 148  
 SELECTED\_INT\_KIND intrinsic 32  
 SELECTED\_REAL\_KIND intrinsic 33  
**SEQUENCE** 39  
 sequence association 253  
 SEQUENCE property 50  
 SEQUENCE statement 38, 49  
 sequence structure 68  
 sequence type 38, 49  
 sequential access 163  
 sequential access input/output statement 180  
**SEQUENTIAL=** specifier 201  
 shape 18  
*signed-int-literal-constant* 32  
*sign-edit-desc* 207  
 size 18  
**SIZE=** specifier 180, 201

specific interface 241  
 specific interface block 241  
 specification expression 114  
 specification function 115  
*specification-expr* 114  
 specifications 63–92  
*specification-stmt* 10  
 standard-conforming program 2  
**STAT=** 100, 104  
 statement 25  
 statement function 265  
 statement label 24, 159  
 statement order 13  
 statements
 

- accessibility 78
- ALLOCATABLE 78
- ALLOCATE 100
- arithmetic IF 159
- assignment 129
- ASYNCHRONOUS 78
- attribute specification 77–92
- BACKSPACE 195
- BIND 78
- CALL 247
- CASE 147
- CLOSE 174
- COMMON 89–92
- computed GO TO 159
- CONTAINS 263
- CONTINUE 159
- CYCLE 157
- DATA 79
- data transfer 175
- DEALLOCATE 104
- DIMENSION 81
- direct access input/output 180
- DO 154
- DO WHILE 154
- END 14
- ENDFILE 195
- EQUIVALENCE 87–89
- EXIT 157
- EXTERNAL 245
- file positioning 194
- FORALL 138, 143
- FORMAT 205
- formatted input/output 177
- FUNCTION 258
- GO TO 159
- IF 147
- IMPLICIT 83
- IMPORT 240
- input/output 161–204
- INQUIRE 196

INTENT 81  
 INTRINSIC 247  
 list-directed input/output 177  
 MODULE 232  
 MODULE PROCEDURE 240  
 NAMELIST 86  
 namelist input/output 178  
 NULLIFY 104  
 OPEN 170  
 OPTIONAL 81  
 PARAMETER 82  
 POINTER 82  
 PRINT 175  
 PRIVATE 78  
 PROCEDURE 240, 246  
 PROGRAM 231  
 PUBLIC 78  
 READ 175  
 RETURN 263  
 REWIND 195  
 SAVE 82  
 SELECT CASE 147  
 sequential access input/output 180  
 STOP 159  
 SUBROUTINE 260  
 TARGET 83  
 type declaration 63–??  
 unformatted input/output 177  
 VOLATILE 83  
 WHERE 135  
 WRITE 175  
**STATUS=** specifier 174, 175  
*stmt-function-stmt* 265  
**STOP** 159  
 STOP statement 159  
*stop-stmt* 159  
 storage associated 379  
 storage association 87–92, 379  
 storage sequence 90, 379  
 storage unit 379  
 Stream access 164  
 stream file 161  
**STREAM=** specifier 201  
 stride 98  
 string  
   - *see* character string  
 structure 16, 54, 68  
 structure component 94  
 structure constructor 54  
 subcomponent 95  
 subobjects 16  
**SUBROUTINE** 260  
 subroutine 12  
 subroutine reference 257

subroutine subprogram 260  
*subroutine-stmt* 260  
*subroutine-subprogram* 9  
*subscript* 96, 138  
 subscript triplet 98  
*subscript-triplet* 97  
 substring 94

## T

**TARGET** 64, 83  
*target* 133  
 TARGET attribute 76  
 TARGET statement 83  
*target-stmt* 83  
 terminal point 165  
**THEN** 146  
 totally [storage] associated 380  
 transfer of control 145, 158, 203, 204  
 transformational function 271  
**TYPE** 38, 39, 63  
 type  
   base 51  
   character 35–38  
   complex 35  
   declared 68  
   derived types 38–56  
   dynamic 68  
   extended 51  
   extensible 51  
   extension 51  
   integer 32–33  
   intrinsic types 31–38  
   logical 38  
   parent 51  
   real 33–34  
 type alias 57  
 type conformance 130  
 type declaration statements 63–??  
**TYPE DEFAULT** 150  
 type equality 50  
**TYPE IN** 150  
**TYPE IS** 150  
 type parameter 30, 32, 33  
 type parameter inquiry 95  
 type specifier  
   **CHARACTER** 67  
   **COMPLEX** 66  
   derived type 68  
   **DOUBLE PRECISION** 66  
   **INTEGER** 66  
   **LOGICAL** 68

REAL 66  
TYPE 68  
TYPE type specifier 68  
**TYPEALIAS** statement 57  
type-bound procedure 46  
type-compatible 68  
*type-declaration-stmt* 63  
*type-spec* 63

*write-stmt* 175

## Z

zero-size array 18, 71, 80

## U

ultimate component 38  
undefined 19  
undefinition of variables 382  
unformatted data transfer 186  
unformatted input/output statement 177  
unformatted record 162  
**UNFORMATTED=** specifier 201  
unit 168  
unlimited polymorphic 68  
**USE** 233  
use association 374  
USE statement 233  
*use-stmt* 233

## V

**VALUE** attribute 64, 77  
value separator 220  
**VALUE** statement 83  
*variable* 93  
variables 17  
    definition & undefinition 382  
vector subscript 99  
VOLATILE statement 83  
*volatile-stmt* 83

## W

wait operation 174, 183, 193, 194  
**WAIT** statement 193  
**WHERE** 135  
WHERE construct 135  
WHERE statement 135  
*where-construct* 136  
*where-stmt* 135  
**WHILE** 154  
**WRITE** 175  
WRITE statement 175  
**WRITE=** specifier 202

