# INTERNATIONAL STANDARD

# ISO/IEC 10279

First edition
1991-10-15

# Information technology — Programming languages — Full BASIC

*Technologies de l'information — Langages de programmation — Full BASIC*

Reference number
ISO/IEC 10279:1991(E)

## 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 10279 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Annexes A and B are for information only.



ISO/IEC Copyright Office • Case Postale 56 • CH-1211 Genève 20 • Switzerland
Printed in Switzerland

# Information technology — Programming languages — Full BASIC

## 1 Scope

This International Standard specifies the programming language Full BASIC and is derived from the American National Standard X3.113-1987. For details of the syntax and semantics see ANSI X3.113-1987 which specifies

- the syntax of programs written in BASIC, including *core* BASIC and various extensions thereto;
- the formats of data and the minimum precision and range of numeric representations and the minimum length and set of characters in strings that are acceptable as input to an automatic data processing system being controlled by a program written in BASIC;
- the formats of data and the minimum precision and range of numeric representations and the minimum length and set of characters in strings that can be generated as output by an automatic data processing system being controlled by a program written in BASIC;
- the semantic rules for interpreting the meaning of a program written in BASIC;
- errors and exceptional circumstances to be detected and also the manner in which such errors and exceptional circumstances are to be handled

This International Standard also refers to ECMA-116 for the specification of *mini-graphics*. Note: ECMA-116 is based on ANSI X3.113-1987.

This International Standard specifies its own conformance subsets, which include those specified in ANSI X3.113-1987 and ECMA-116.

This International Standard is designed to promote the interchangeability of BASIC programs among a variety of automatic data processing systems.

## 2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and of ISO maintain registers of currently valid International Standards.

ANSI X3.113-1987: *Information systems - programming languages - full BASIC.*

STANDARD ECMA-116: 1986, *BASIC.*

# 3 Technical content

## 3.1 Principal technical content

Apart from the conformance rules the technical content of this International Standard is defined in part by Sections 3 to 16 of the ANSI X3.113-1987 (which will be referred to a ANS BASIC). The definition of *reserved word* in ANS BASIC Section 3.2 is for information only with respect to this International Standard.

The technical content for the *mini-graphics* module of this International Standard is defined by Section 13 of ECMA-116.

## 3.2 Additional reserved words

For the *subset core,* insert the following paragraph after paragraph 6 of Subsection 4.4.2 of ANS BASIC.

"The keywords ACCESS, AND, ANGLE, AREA, ARITHMETIC, ASK, AT, BASE, BEGIN, BREAK, CALL, CASE, CHAIN, CLEAR, CLIP, CLOSE, COLLATE, COLOR, DATA, DATUM, DEBUG, DECIMAL, DECLARE, DEF, DEGREES, DEVICE, DIM, DISPLAY, DO, ELAPSED, ELSEIF, END, ERASE, ERASABLE, EXIT, EXTERNAL, FILETYPE, FOR, FUNCTION, GO, GOSUB, GOTO, GRAPH, IF, IMAGE, INPUT, INTERNAL, IS, LENGTH, LET, LINE, LINES, LOOP, MARGIN, MAT, MISSING, NAME, NATIVE, NEXT, NUMERIC, OFF, ON, OPEN, OPTION, OR, ORGANIZATION, OUTIN, OUTPUT, POINT, POINTER, POINTS, PROGRAM, PROMPT, RADIANS, RANDOMIZE, READ, RECSIZE, RECTYPE, REST, RESTORE, RETURN, SAME, SELECT, SEQUENTIAL, SET, SETTER, SIZE, SKIP, STANDARD, STATUS, STEP, STOP, STREAM, STRING, STYLE, SUB, TAB, TEXT, THEN, THERE, TIMEOUT, TO, TRACE, UNTIL, USING, VARIABLE, VIEWPORT, WHILE, WINDOW, WITH, WRITE, and ZONEWIDTH shall not be used as identifiers."

# 4 Conformance

There are two aspects of conformance to a set of modules in this International Standard: conformance by a program written in the BASIC language, and conformance by an implementation which processes such programs. The conformance requirements are structured so that any program conforming to a set of modules will produce the same results when executed by any implementation conforming to the same or an encompassing set of modules (though certain implementation-dependent features are noted in ANS BASIC Appendix C).

## 4.1 Modules

The programming language defined by this International Standard is organized in a modular fashion. Conformance to this International Standard is defined with respect to particular sets of the following fifteen modules and combinations thereof:

a) A *core* module, which encompasses all programs whose syntax conforms to ANS BASIC Sections 4 to 10, parts of ANS BASIC Section 11 (excluding internal-format record and native-format record files), and ANS BASIC Section 12.

b) A *subset core* module, which encompasses all programs whose syntax conforms to ANS BASIC Sections 4 to 10 (except that a substitute definition of *reserved word* applies -- see 3.2), parts of ANS BASIC Section 11 (excluding enhanced-internal and enhanced-native files), and ANS BASIC Subsection 12.2. (The *subset core* module is identical to the *core* module except that the list of reserved words is larger and exception handling is excluded.)

c) An *enhanced-internal file* module, which encompasses all programs whose syntax conforms to the enhanced production rules in ANS BASIC Section 11 (lacking the prefix "N"), together with the *core.*

d) An *enhanced-internal file subset* module, which encompasses all programs whose syntax conforms to the enhanced production rules in ANS BASIC Section 11 (lacking the prefix "N"), together with the *subset core.*

e) An *enhanced-native file* module, which encompasses all programs whose syntax conforms to the enhanced native production rules in ANS BASIC Section 11 (indicated with the prefix "N"), together with the *core*.

f) An *enhanced-native file subset* module, which encompasses all programs whose syntax conforms to the enhanced native production rules in ANS BASIC Section 11 (indicated with the prefix "N"), together with the *subset core*.

g) A *graphics* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 13, together with the *core*.

h) A *graphics subset* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 13, together with the *subset core*.

i) A *mini graphics* module, which encompasses all programs whose syntax conforms to ECMA-116 Section 13 (see Annex B), together with the *core*.

j) A *mini graphics subset* module, which encompasses all programs whose syntax conforms to ECMA-116 Section 13 (see Annex B), together with the *subset core*.

k) A *real-time* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 14, together with the *core*.

l) A *real-time subset* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 14, together with the *subset core*.

m) A *fixed decimal* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 15, together with the *core*.

n) A *fixed decimal subset* module, which encompasses all programs whose syntax conforms to ANS BASIC Section 15, together with the *subset core*.

o) An *editing* module, which encompasses all unsorted programs and editing commands whose syntax conforms to ANS BASIC Section 16.

In addition, the Conformance Modules shall include those described in ANS X3-113, Section 2, and in ECMA-116, Section 2. (Note: See Annex A for details on the relationship between these conformance modules and the conformance modules defined in ANS X3-113 and ECMA-116.)

## 4.2 Program conformance

A program conforms to a set of modules in this International Standard only when

- the program and each statement or other syntactic element contained therein is syntactically valid according to the syntactic rules specified by this International Standard as belonging to that set;

- the program as a whole violates none of the global constraints imposed by this International Standard on the application of the syntactic rules.

## 4.3 Implementation conformance

An implementation conforms to a set of modules in this International Standard only when

- it accepts and processes all programs conforming to that set of modules in this International Standard;
- it reports reasons for rejecting any program which does not conform to that set of modules in this International Standard;
- it interprets errors and exceptional circumstances according to the specifications of this International Standard;
- it interprets the semantics of each statement of a conforming program according to the specifications in this International Standard;
- it interprets the semantics of a conforming program as a whole according to the specifications in this International Standard;
- it accepts as input, manipulates, and can generate as output numbers of at least the precision and range specified in this International Standard;
- it accepts as input, manipulates, and can generate as output strings of at least the length and composed of at least those characters specified in this International Standard;
- it is accompanied by documentation available to the user that describes the actions taken in regard to features referred to as "undefined" or "implementation-defined" in this International Standard;
- it is accompanied by documentation available to the user that describes and identifies all enhancements to the language defined in this International Standard.

This International Standard makes no requirement concerning the interpretation of the semantics of any statement or program as a whole that does not conform to this International Standard.

In addition, an implementation conforms to the editing requirements of this International Standard if it accepts and processes unsorted programs and editing commands according to the specifications in ANS BASIC Section 16.

## 4.4 Errors

This International Standard does not include specific requirements for reporting syntactic errors in the text of a program. Implementations conforming to a set of modules in this International Standard may accept programs written in an enhanced language without having to report all constructs not conforming to that set of modules.

Whenever a statement, or other program element, does not conform to the syntactic rules given herein, and that statement, or program element, does not have a clear, well-documented implementation-defined meaning, an error shall be reported. Errors shall be reported in a clear and well-documented way, and whenever feasible the implementation should indicate the erroneous statement and the position of the error within the statement.

## 4.5 Exceptions

An exception is a circumstance arising in the course of execution of a program when an implementation recognizes that the semantic rules of this International Standard cannot be followed or that some resource constraint is about to be exceeded. All exceptions described in this International Standard shall be detected, reported, and processed when they occur, unless some mechanism provided in ANS BASIC Subsection 12.1 or in an enhancement to this International Standard has been invoked by the user to handle exceptions.

In the absence of programmer-specified recovery procedures, exceptions shall be handled by the recovery procedures specified in this International Standard. If no recovery procedure is specified in this International