INTERNATIONAL STANDARD

ISO/IEC 10967-1

First edition
1994-12-15

Information technology — Language independent arithmetic —

Part 1:
Integer and floating point arithmetic

Technologies de l'information — Arithmétique indépendante de langage —

Partie 1: Arithmétique de nombres entiers et en virgule flottante

Reference number
ISO/IEC 10967-1:1994(E)

Contents

Annexes

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 10967-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

ISO/IEC 10967 consists of the following parts, under the general title *Information technology — Language independent arithmetic*:

— *Part 1: Integer and floating point arithmetic*

— *Part 2: Mathematical procedures*

— *Part 3: Complex arithmetic and procedures*

Additional parts will specify other arithmetic types or operations.

Annexes A to J of this part of ISO/IEC 10967 are for information only.

Introduction

The aims

Programmers writing programs that perform a significant amount of numeric processing have often not been certain how a program will perform when run under a given language processor. Programming language standards have traditionally been somewhat weak in the area of numeric processing, seldom providing an adequate specification of the properties of arithmetic data types, particularly floating point numbers. Often they do not even require much in the way of documentation of the actual arithmetic data types by a conforming language processor.

It is the intent of this part of ISO/IEC 10967 to help to redress these shortcomings, by setting out precise definitions of integer and floating point data types, and requirements for documentation. This is done in a way that makes as few presumptions as possible about the underlying machine architecture.

It is not claimed that this part of ISO/IEC 10967 will ensure complete certainty of arithmetic behavior in all circumstances; the complexity of numeric software and the difficulties of analysing and proving algorithms are too great for that to be attempted. Rather, the requirements set forth here will provide a firmer basis than hitherto for attempting such analysis.

Hence the first aim of this part of ISO/IEC 10967 is to enhance the predictability and reliability of the behavior of programs performing numeric processing.

The second aim, which helps to support the first, is to help programming language standards to express the semantics of arithmetic data types. These semantics need to be precise enough for numerical analysis, but not so restrictive as to prevent efficient implementation of the language on a wide range of platforms.

The third aim is to help enhance the portability of programs that perform numeric processing across a range of different platforms. Improved predictability of behavior will aid programmers designing code intended to run on multiple platforms, and will help in predicting what will happen when such a program is moved from one conforming language processor to another.

Note that this part of ISO/IEC 10967 does not attempt to ensure bit-for-bit identical results when programs are transferred between language processors, or translated from one language into another. Programming languages and platforms are too diverse to make that a sensible goal. However, experience shows that diverse numeric environments can yield comparable results under most circumstances, and that with careful program design significant portability is actually achievable.

The content

This part of ISO/IEC 10967 defines the fundamental properties of integer and floating point numbers. These properties are presented in terms of a parameterized model. The parameters allow enough variation in the model so that most platforms are covered, but when a particular set of parameter values is selected, and all required documentation is supplied, the resulting information should be precise enough to permit careful numerical analysis.

The requirements of this part of ISO/IEC 10967 cover three areas. First, the programmer must be given runtime access to the parameters and functions that describe the arithmetic properties of the platform. Second, the executing program must be notified when proper results cannot be returned (e.g., when a computed result is out of range or undefined). Third, the numeric properties of conforming platforms must be publicly documented.

This part of ISO/IEC 10967 focuses on the classical integer and floating point data types. Later parts will consider common mathematical procedures (part 2), complex numbers (part 3), and possibly additional arithmetic types such as fixed point.

Relationship to hardware

ISO/IEC 10967 is not a hardware architecture standard. It makes no sense to talk about an "LIA machine." Future platforms are expected either to duplicate existing architectures, or to satisfy high quality architecture standards such as IEC 559 (also known as IEEE 754). The floating point requirements of this part of ISO/IEC 10967 are compatible with (and enhance) IEC 559.

This part of ISO/IEC 10967 provides a bridge between the abstract view provided by a programming language standard and the precise details of the actual arithmetic implementation.

The benefits

Adoption and proper use of this part of ISO/IEC 10967 can lead to the following benefits.

Language standards will be able to define their arithmetic semantics more precisely without preventing the efficient implementation of their language on a wide range of machine architectures.

Programmers of numeric software will be able to assess the portability of their programs in advance. Programmers will be able to trade off program design requirements for portability in the resulting program.

Programs will be able to determine (at run time) the crucial numeric properties of the implementation. They will be able to reject unsuitable implementations, and (possibly) to correctly characterize the accuracy of their own results. Programs will be able to extract apparently implementation dependent data (such as the exponent of a floating point number) in an implementation independent way. Programs will be able to detect (and possibly correct for) exceptions in arithmetic processing.

End users will find it easier to determine whether a (properly documented) application program is likely to execute satisfactorily on their platform. This can be done by comparing the documented requirements of the program against the documented properties of the platform.

Finally, end users of numeric application packages will be able to rely on the correct execution of those packages. That is, for correctly programmed algorithms, the results are reliable if and only if there is no notification.

Annex A is intended to be read in parallel with the standard.

Information technology – Language independent arithmetic – Part 1: Integer and floating point arithmetic

1 Scope

This part of ISO/IEC 10967 defines the properties of integer and floating point data types on computer systems to ensure that the processing of arithmetic data can be undertaken in a reliable and predictable manner. Emphasis is placed on documenting the existing variation between systems, not on the elimination of such variation. The requirements of this part of ISO/IEC 10967 shall be in addition to those that may be specified in other standards, such as those for programming languages (See clause 7).

It is not the purpose of this part of ISO/IEC 10967 to ensure that an arbitrary numerical function can be so encoded as to produce acceptable results on all conforming systems. Rather, the goal is to ensure that the properties of arithmetic on a conforming system are made available to the programmer.

Therefore, it is not reasonable to demand that a substantive piece of software run on every implementation that can claim conformity to this part of ISO/IEC 10967.

An implementor may choose any combination of hardware and software support to meet the specifications of this part of ISO/IEC 10967. It is the arithmetic environment, as seen by the user, that does or does not conform to the specifications.

The term *implementation* (of this part of ISO/IEC 10967) denotes the total arithmetic environment, including hardware, language processors, exception handling facilities, subroutine libraries, other software, and all pertinent documentation.

1.1 Specifications included in this part of ISO/IEC 10967

This part of ISO/IEC 10967 defines integer and floating point data types. Definitions are included for bounded, unbounded, and modulo integer types, as well as both normalized and denormalized floating point types.

The specification for an arithmetic type includes

a) The set of computable values.

b) The set of computational operations provided, including

   1) primitive operations (addition, subtraction, etc.) with operands of the same type,