Version 8.7 of the Icon Programming Language

Ralph E. Griswold, Clinton L. Jeffery, Gregg M. Townsend, and Kenneth Walker

Department of Computer Science, The University of Arizona

1. Introduction

The current version of Icon is Version 8.7. The second edition of the Icon book [1] describes Version 8.0. This report is a supplement to that book.

Most of the language extensions in Version 8.7 are upward-compatible with previous versions of Icon and most programs written for earlier versions work properly under Version 8.7. The language additions to Version 8.7 are:

- an optional interface to the X Window system (for platforms that have X)
- new functions and keywords
- several minor changes

There also are changes to the implementation in Version 8.7, including support for multiple storage regions, that provide more capabilities for some users. See Section 3.

2. Language Features

X-Window Facilities

Version 8.7 provides support for X Windows through a large repertoire of functions. These facilities are optional and are not available on all platforms. See [2] for more information.

New Functions and Keywords

The new functions and keywords are described briefly here. At the end of this report there also is a sheet with more complete descriptions in the style of the second edition of the Icon book. This sheet can be trimmed and used as an insert to the book.

There are five new functions:

chdir(s)	Changes the current directory to s but fails if there is no such directory or if the change cannot be made.
delay(i)	Delays execution i milliseconds. This function presently is only supported on UNIX platforms.
flush(f)	Flushes the input/output buffers for file f.
function()	Generates the names of the Icon (built-in) functions.
sortf(X,i)	Produces a sorted list of the elements of X. The results are similar to those of sort(X,i), except that among lists and among records, structure values are ordered by comparing their ith fields.

There are four new keywords:

&allocated	Generates the number of bytes allocated since the beginning of program execution. The first result is the total number of bytes in all regions, followed by the number of bytes in the static, string, and block regions.
&e	The base of the natural logarithms, 2.71828 ...
&phi	The golden ratio, 1.61803 ...
&pi	The ratio of the circumference of a circle to its diameter, 3.14159 ...

The X interface adds additional new keywords [2].

Minor Changes

- The invocable declaration is accepted but ignored by the interpreter to provide source-language compatibility with the Icon compiler. See [3] for a description of this declaration.
- Real literals that are less than 1 no longer need a leading zero. For example, .5 now is a valid real literal instead of being the dereferencing operator applied to the integer 5.
- The identifiers listed by display() are now given in sorted order.
- In sorting structures, records now are first sorted by record name and then by age (serial number).
- The keyword &features now includes either interpreted or compiled to indicate whether the program is interpreted or compiled.
- If X-Window facilities are supported, &features also includes X Windows.
- If multiple storage regions are supported, &features also includes Multiple Regions.
- Error message 101 now reads integer expected or out of range to reflect the fact that not all operations support large integers.
- Error 120 now reads two csets or two sets expected to more accurately reflect the fact that set operations require arguments of the same type.
- Error 125, list or set expected, has been added for sortf().
- Errors 140, window expected, and 141, program terminated by window manager, have been added when X-Window facilities are supported.
- Errors 316, interpreter stack too large, and 318, co-expression stack too large, have been added for 16-bit platforms.

3. Implementation Changes

The implementation of Version 8.7 is different in many respects from the implementation of Version 8.0 [4]. Most of the differences are transparent to the user. The following changes provide additional facilities:

- On platforms that use fixed-sized storage regions (notably MS-DOS), Icon now allocates additional regions if more space is needed. Consequently, some programs that formerly ran out of memory on such platforms no longer do so.

  Memory monitoring is not designed for multiple regions and if an additional region is allocated, memory monitoring is disabled.
- Under MS-DOS, iconx now finds icode files at any place on the PATH specification as well as in the current directory.

4. Limitations, Bugs, and Problems

- Line numbers sometimes are wrong in diagnostic messages related to lines with continued quoted literals.
- Large-integer arithmetic is not supported in i to j and seq(). Large integers cannot be assigned to keywords.
- Large-integer literals are constructed at run-time. Consequently, they should not be used in loops where they would be constructed repeatedly.
- Conversion of a large integer to a string is quadratic in the length of the integer. Conversion of very a large integer to a string may take a very long time and give the appearance of an endless loop.
- Right shifting of large negative integers by ishift() is inconsistent with the shifting of ordinary integers.
- Integer overflow on exponentiation may not be detected during execution. Such overflow may occur during type conversion.
- In some cases, trace messages may show the return of subscripted values, such as &null[2], that would be erroneous if they were dereferenced.
- If a long file name for an Icon source-language program is truncated by the operating system, mysterious diagnostic messages may occur during linking.
- Stack overflow checking uses a heuristic that is not always effective.
- If an expression such as

  x := create *expr*

  is used in a loop, and x is not a global variable, unreferenceable co-expressions are generated by each successive create operation. These co-expressions are not garbage collected. This problem can be circumvented by making x a global variable or by assigning a value to x before the create operation, as in

  x := &null
  x := create *expr*

- Stack overflow in a co-expression may not be detected and may cause mysterious program malfunction.

Acknowledgements

The design and implementation of Version 8.7 of Icon was supported, in part, by National Science Foundation Grants CCR-8713690 and CCR-8901573.

Sandra Miller contributed to the implementation of Version 8.7 of Icon.

References

1. R. E. Griswold and M. T. Griswold, *The Icon Programming Language*, Prentice-Hall, Inc., Englewood Cliffs, NJ, second edition, 1990.
2. C. L. Jeffery, *X-Icon: An Icon Windows Interface*, The Univ. of Arizona Tech. Rep. 91-1, 1992.
3. K. Walker, *Using the Icon Compiler*, The Univ. of Arizona Icon Project Document IPD157, 1991.
4. R. E. Griswold, *Supplementary Information for the Implementation of Version 8.7 of Icon*, The Univ. of Arizona Icon Project Document IPD190, 1992.

chdir(s) : n change directory

chdir(s) changes the directory to s but fails if there is no such directory or if the change cannot be made. Whether the change in the directory persists after program termination depends on the operating system on which the program runs.

Error: 103 s not string

---

delay(i) : n delay execution

delay(i) delays execution i milliseconds. This function presently is supported only on UNIX platforms.

Error: 101 i not integer

---

flush(f) : n flush buffer

flush(f) flushes the input/output buffers for f.

Error: 105 f not file

---

function() : s1, s2, ..., sn generate function names

function() generates the names of the Icon (built-in) functions.

---

sortf(X,i) : L sort list or set by field

sortf(X,i) produces a sorted list of the values in X. Sorting is primarily by type and in most respects is the same as with sort(X,i). However, among lists and among records, two structures are ordered by comparing their ith fields. i can be negative but not zero. Two structures having equal ith fields are ordered as they would be in regular sorting, but structures lacking an ith field appear before structures having them.

Default: i 1

Errors:	101	i not integer
	115	X not list or set
	205	i = 0
	307	inadequate space in block region

&allocated : i1, i2, i3, i4 cumulative allocation

&allocated generates the total amount of space, in bytes, allocated since the beginning of program execution. The first value is the total for all regions, followed by the totals for the static, string, and block regions, respectively. The space allocated in the static region is always given as zero. *Note:* &allocated gives the cumulative allocation; &storage gives the current allocation; that is, the amount that has not been freed by garbage collection.

&e : r base of natural logarithms

The value of &e is the base of the natural logarithms, 2.71828 ... .

&phi : r golden ratio

The value of &phi is the golden ratio, 1.61803 ... .

&pi : r ratio of circumference to diameter of a circle

The value of &pi is the ratio of the circumference of a circle to its diameter, 3.14159 ... .