

The Icon Newsletter

No. 35 – March 1, 1991



Newsletters

As mentioned previously, a second newsletter, *The Icon Analyst*, is now being published. The *Analyst*, which contains material of a more technical nature than *The Icon Newsletter*, has been very well received. The fourth issue of the *Analyst*, which appeared in February, contained articles on programs that write programs, string scanning, large integers, and memory utilization. The fifth issue, which appears in April, will have long articles on string scanning and pattern matching, as well as some advice on debugging. All back issues of the *Analyst* are still available and new subscribers may start with the first issue if they wish.

With the *Analyst* now covering the technical aspects of Icon, the *Newsletter* is being reduced in size to eight pages. We plan to continue to publish the *Newsletter* three times a year in order to provide topical information in a timely manner.

The Icon Program Library

We've come to realize that the Icon program library is a valuable resource for Icon programmers, but one that often is overlooked. For example, we frequently get requests for information on how to do something that's already provided in the library.

The library is not only a useful collection of programs and procedures, but it also provides many examples of programming techniques and styles — useful to novices and experts alike.

Last fall we started a subscription service whereby Icon programmers can get updates to the library automatically. The first update, issued in November, included corrections and improvements to the basic library and many new programs and procedures, including a browser for locating material in the library

and an on-line Icon help facility. The next two updates will include a new version of Idol, the object-oriented version of Icon written in Icon, many new utilities, procedures for controlling terminal input and output, and all sorts of other interesting things. In addition to program material, the word list from Webster's Second is being distributed in updates as space permits.

If you're really interested in Icon, whether you're a novice or an expert, we encourage you to get the Icon program library and to subscribe to the updates. Subscriptions to the updates start with the first one, unless otherwise requested, since each update builds on the preceding one.

See the end of this *Newsletter* for information on ordering the library and updates.

Icon in Your Pocket

Bob Alexander provides this interesting note on a "new frontier" for Icon.

The Atari Portfolio is a tiny MS-DOS machine, almost pocket sized, about 4" x 8" x 1" folded up. It seems to be intended for use as a fancy personal organizer, but its application programs are regular MS-DOS programs. Normally they are developed on a PC and transferred to the Portfolio.

We thought it would be fun to see if we could get Icon running on it and, who knows, maybe even useful. Ken Wallich, my friend who owns a Portfolio, provided the hardware and much of the motivation.

The main problem was, naturally, the memory size. The Portfolio has 128K on-board memory and a solid state "diskette" slot that is 32, 64, or 128K. There is also an external solid state drive, and a memory expansion unit that adds 256K.

We wanted something that could run without any external boxes, so I dug back in my archives to version 5.9 of Icon for the MS-DOS (because it's much smaller than current versions). We quickly found out that the start-up dynamic memory requirements were too much for the Portfolio's small memory, so we severely reduced the configured region sizes, etc. We were able to

run the animal game (from the Icon program library) with the following environment variable values:

```
NOERRBUF=1
STKSIZE=100
NSTACKS=0
NBUFS=2
STRSIZE=1024
HEAPSIZE=1024
```

The Portfolio's on-board memory is partitionable between main memory and a C: drive ramdisk — our configuration was 120K main and 8K disk (the minimum configurable ramdisk). We had to use the largest available ram-diskette card (128K) so it would be big enough to hold the iconx file.

Icon programs that are run on this no-external-devices configuration had to be translated and linked on another machine (Sun workstations in our case) and the icode files were transferred to the Portfolio's ram-diskette.

Using a 256K external memory expansion unit, we were able to perform translation and linking on the Portfolio, too. For our tiny development environment we configured the unit with its internal memory as a 128K ramdisk, and used the expansion unit's 256K as main memory.

We consider our experiment with tiny Icon a success, mainly because it was fun, but also because we have been able to write some small but useful programs for the pocket-sized Portfolio — and probably created the most diminutive Icon programming environment yet on the planet.

Bug in Version 8 of MS-DOS Icon

In the first release of Version 8 of Icon for MS-DOS, a small typographical error in the source code caused the untranslated mode in opening files to be ineffective. As a result, all input and output is translated. That's no problem for text files; in fact it's what you want. But if you read a binary file, any carriage-return linefeed combinations are converted to line feeds and conversely on output. That's not so bad unless you're modifying the data, since output translation reverses the effect of input translation. But the control-Z character (hex 1A) is treated as an end of file. If there happens to be such a character in a binary file, it terminates input.

We corrected this error last fall, but many copies of MS-DOS Icon had been sent out by then. Check the creation date for ICON.ARC; if it's before September 9, 1990, it has the bug.

If this bug is a problem for you (it only applies to reading and writing *binary* files), you can get a new copy via FTP or our RBBS. If that's not convenient, and you purchased MS-DOS Icon from the Icon project,

send us the diskettes you received from us and we'll provide a free replacement.

Note: This bug exists only in basic version of MS-DOS Icon; it is not present in MS-DOS Icon for the 386 or in any other implementation of Icon.

For what it's worth, here is a little program to test whether or not the untranslated mode works:

```
procedure main()
  output := open("char.tmp", "wu") |
    stop("*** cannot open char.tmp")
  every writes(output, !&cset)
  close(output)
  input := open("char.tmp", "ru") |
    stop("*** cannot re-open char.tmp")
  count := 0
  while reads(input) do
    count += 1
  if count < 256 then
    write("untranslated mode not supported")
  else write("untranslated mode supported")
end
```

Update on the Icon Optimizing Compiler

The Icon optimizing compiler is now operational. We've successfully installed it on several UNIX platforms and are in the beta-test phase.

Preliminary testing indicates that most Icon programs run 2 to 4 times faster when compiled than when interpreted. And a compiled Icon program runs stand-alone; it does not require the separate Icon run-time system that an interpreted program does.

We hope to have a version of the Icon compiler available for UNIX by the end of the summer, although it's too early to be confident of that. When it's ready we'll announce it in this *Newsletter* and the Icon news group.

New Company Formed to Provide Commercial Support for Icon

Iconic Software, Inc. (ISI), is a new company founded by a core group of computer scientists, together with professionals in computer marketing and technical customer service.

ISI's goal is to become a major supplier of software development tools, languages, and high-value applications for the UNIX system environment. ISI founders have considerable expertise and experience with Icon and with UNIX-based systems, much of it gained while working at AT&T Bell Laboratories.

The company will initially focus on high-quality voice processing systems. "Voice response systems are

applications in which persons interact with computers over their touch-tone phones to access and update databases, hear prerecorded information and leave recorded messages, and perform other transactions" said ISI founder, Jerry Nowlin. "We've already developed special Icon utilities that will allow us to deliver thoroughly tested systems in a fraction of the time that would be required if C were used."

An exciting prospect for ISI is commercial support for the Icon compiler, which is currently under development at the University of Arizona. Although the initial UNIX version of the compiler will be available to the public, ISI will extend and support a commercial UNIX version. Ken Walker, who designed and implemented the compiler and is presently completing his PhD, is joining ISI to work on this project. ISI is also investigating the feasibility of an MS-DOS implementation. ISI hopes to become known as the "Icon Compiler Company" and sees a significant opportunity in bringing Icon to professional developers of commercial software products.

ISI's administrative and marketing offices are currently located in West Lafayette, Indiana. Development facilities are located in Plano, Illinois. For more information about ISI's products and services contact Roger Fonorow: (317) 463-9269.

ICEBOL5

As mentioned in the last *Newsletter*, ICEBOL5, officially titled the *Fifth International Conference on Symbolic and Logical Computing*, will be held April 18-19, 1991. For more information, contact

Eric Johnson
114 Beadle Hall
Dakota State University
Madison, SD 57042
U.S.A.

electronic mail: ERIC@SDNET.BITNET

or call (605) 256-5270.

We'll be there. We hope you'll be able to come also.

ProIcon Version 2.0

The ProIcon Group is shipping Version 2.0 of ProIcon, an enhanced version of Icon for the Macintosh.

Version 2.0 of ProIcon includes all the features of Version 8 of Icon and well as additional facilities for working in the Macintosh environment. Notable among new features in ProIcon 2.0 is an external function interface that provides access to HyperCard XCMDs

and XFCNs, and well as "stand-alone" code resources that can be used to add extensions to Icon proper.

The Version 2.0 distribution also includes an application for viewing Icon allocation histories in color.

The price of Version 2.0 of Icon is \$175 plus shipping. Educational discounts and site licenses are available. Owners of Version 1.0 can upgrade for \$35 plus shipping. Persons who purchased Version 1.0 after August 1, 1990 only pay shipping.

To order ProIcon 2.0 or get more information, contact

Catspaw, Inc.
P.O. Box 1123
Salida, CO 81201-1123
U.S.A.
(719) 539-3884

The Icon Newsletter

Madge T. Griswold and Ralph E. Griswold
Editors

The Icon Newsletter is published three times a year, at no cost to subscribers. To subscribe, contact

Icon Project
Department of Computer Science
Gould-Simpson Building
The University of Arizona
Tucson, Arizona 85721
U.S.A.

(602) 621-8448

FAX: (602) 621-4246

Electronic mail may be sent to:

icon-project@cs.arizona.edu

or

...{uunet,allegro,noao}!arizona!icon-project

THE UNIVERSITY OF
ARIZONA
TUCSON ARIZONA

and



The Bright Forest Company
Tucson Arizona

© 1991 by Madge T. Griswold and Ralph E. Griswold
All rights reserved.

Programming Corner



Syntax

Steve Wampler sent us this question: "Why doesn't the following code produce a syntax error? I know why, given the absence of a syntax error, it's an infinite loop, but I don't understand why

there isn't an error on the missing left hand side of the assignment operator ..."

The code fragment he sent surely looks syntactically wrong:

```
while next := read() do
  write(next)
```

Ken Walker provided the answer: Assignment in Icon has the form

```
expr1 := expr2
```

where *expr1* and *expr2* can be any expressions. The control structure *next* is an expression, so the program fragment above is syntactically correct, with *expr1* being *next*. Of course, when this expression is evaluated, it transfers control back to the beginning of the while loop.

Mistakes like the one above are easy enough to make — *next* is a perfectly reasonable mnemonic for the intended use. And it's easy to forget that Icon is an expression-based language — no statements, just expressions. That's useful in many situations, but it allows some ridiculous expressions to be syntactically correct.

A Prime Number Generator

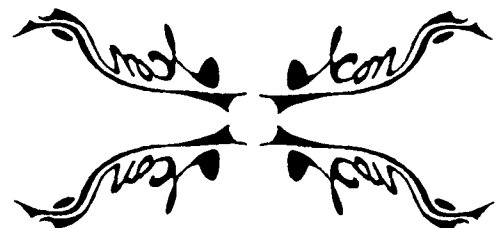
Viktors Berstis sent us this interesting program for generating prime numbers. The technique is due to John Conway and uses the FRACTRAN language, which is interpreted here.

This procedure doesn't get very far without large integer arithmetic. Otherwise, the program does not terminate of its own accord. It stops generating results when the primes exceed the powers-of-two-table, but it keeps on computing primes until the program runs out of memory or something breaks.

One interesting thing about this technique is that each prime is computed from the preceding one.

A word of caution: the technique also is very slow.

```
record fract(numer,denom)
procedure main(arg)
  limit := integer(arg[1]) | 30
  every write(genprime(limit)) # typical usage
end
procedure genprime(limit)
# This is a list of 14 fractions that encode a
# FRACTRAN program for computing prime
# numbers.
f := list(14)
f[ 1] := fract(17,91)
f[ 2] := fract(78,85)
f[ 3] := fract(19,51)
f[ 4] := fract(23,38)
f[ 5] := fract(29,33)
f[ 6] := fract(77,29)
f[ 7] := fract(95,23)
f[ 8] := fract(77,19)
f[ 9] := fract( 1,17)
f[10] := fract(11,13)
f[11] := fract(13,11)
f[12] := fract(15, 2)
f[13] := fract( 1, 7)
f[14] := fract(55, 1)
# Construct a table of the powers of two.
pot := table()
power := 1
every n := 0 to limit - 1 do {
  pot[power] := n
  power *= 2
}
# Generate the primes.
s := 2
repeat {
# Multiply s by the first fraction that produces an
# integer result.
  x := !f & (s * x.numer) % x.denom = 0
  s := (s * x.numer) / x.denom
# The exponent is prime if s is a power of two;
# if so, produce it.
  suspend \pot[s]
}
end
```



From Our Mail

Why can't I get updates to the source code for Icon and the program library on Macintosh disks?

Every different distribution format creates work for us in assembling, packaging, copying, and inventory.

We simply don't have the resources to provide all Icon material in all possible formats. Most of our users either run MS-DOS or have ways to convert MS-DOS diskettes to their own format. So far, we've had only two requests for other formats for source and library code. If we get a significant number of requests for another format, we'll consider adding it.

If you're going to distribute updates to the Icon program library on MS-DOS diskettes, why don't you compress them with ARC? You could get a lot more on each update that way.

Programs for handling compressed files in ARC format are widely available for MS-DOS and also for many other computers. We use ARC for source updates, since the number of diskettes otherwise would be unmanageable for us. This seems to work reasonably well because most persons who want to compile Icon can handle ARC format.

In the case of the program library, we don't have the same motivation for compression. 360K, which fits on a single 5.25" DD MS-DOS diskette, is a lot of space for library material — most Icon programs are not very long. It takes quite a while to accumulate and prepare this much material for library updates; it's not like we have megabytes on hand. It seems that a good balance between timeliness and quantity allows us to put out a library update every three months or so without using data compression.

I'd like to get a copy of the mailing list for your Newsletter so I can send out promotional literature on our new database product. How much do you charge?

We don't sell our subscription list. Occasionally we provide mailing labels for a purpose specifically related to Icon or SNOBOL4, such as the upcoming ICEBOL conference.

I tried to call The Bright Forest Company about ProIcon, but I couldn't find their number. How can I get in touch with them?

ProIcon is a product of The ProIcon Group, a joint venture of The Bright Forest Company and Catspaw, Inc. Catspaw markets ProIcon and if you want to get product information or place an order, you should contact them:



Catspaw, Inc.
P.O. Box 1123
Salida, CO 81201-1123
(719) 539-3884

You can reach The Bright Forest Company as follows:

The Bright Forest Company
P.O. Box 12076
Tucson, AZ 85732-2076
(602) 325-3948

I read the material about the Charles Babbage Institute in the last issue of the Newsletter with interest. Can you give me their address so that I can ask them for more information?

Sure. It's

Charles Babbage Institute
103 Walter Library
University of Minnesota
Minneapolis, Minnesota 55455

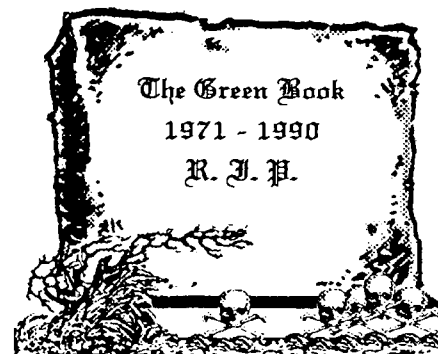
SNOBOL4 Corner

The End of an Era?

Sales of *The SNOBOL4 Programming Language* (also known as the "Green Book") have declined in recent years, so it was no surprise to us when we received a letter from its publisher, Prentice-Hall, telling us that the book was no longer profitable and that no new copies would be printed. The letter assured us, however, that sale of the remaining copies would continue until they were exhausted. Only two weeks later we learned that Prentice-Hall instead had destroyed the remaining copies without giving us or anyone else the chance to buy them.

The Green Book was published in 1971. As technical books go, it had a long life, even if its end was ugly.

We know that many of you have used SNOBOL4 and that some of you still prefer it to other programming languages. In fact, SNOBOL4 always had a loyal following, sometimes fiercely so.



But time has passed. With the death of the book, an era in computing for some of us has ended, at least symbolically.

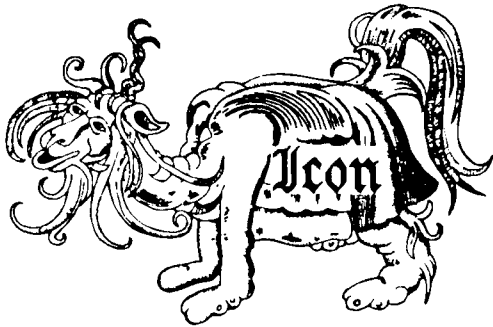
Ordering Icon Material

What's Available

There are implementations of Icon for several personal computers, as well as for CMS, MVS, UNIX, and VMS. Source code for all implementations is available. Program material is accompanied by installation instructions and users' manuals in printed and machine-readable form.

There also is a program library that contains a large collection of Icon programs and procedures, as well as an object-oriented version of Icon that is written in Icon.

In addition to users manuals that are included with program material, there are three books and two newsletters about Icon.



Icon Program Material

The current version of Icon is 8. All the program material here is for Version 8.

All program material is in the public domain except the MS-DOS/386 implementation of Icon, which is a commercial product that carries a standard software license.

Personal Computers: Executables, source code, and the Icon program library for personal computers are provided in separate packages. Each package contains documentation in printed and machine-readable form. *Note:* Icon for personal computers requires at least 640KB of RAM; it requires more on some systems.

CMS and MVS: The CMS and MVS packages contain executables, source code, test programs, the Icon program library, and documentation in printed and machine-readable form.

UNIX: The UNIX package contains source code but not executables, test programs, related software, the Icon program library, and documentation in printed and machine-readable form. UNIX Icon can be configured for most UNIX systems. *Note:* executables for Xenix and the UNIX PC are available separately.

VMS: The VMS package contains object code, executables, source code, test programs, the Icon program library, and documentation in printed and machine-readable form.

Porting: Icon source code for porting to other computers is distributed on MS-DOS format diskettes. There are two versions, one with a flat file system and one with a hierarchical file system.

Update Subscriptions: Updates to the Icon source code and the Icon program library are available by subscription.

Source-code updates are distributed on MS-DOS diskettes in ARC format for hierarchical file systems, and are suitable for compilation under MS-DOS or for porting to new computers. Each update usually provides a completely new copy of the source. A source-code subscription provides five updates. Updates are issued about three times a year.

Icon program library updates are distributed on MS-DOS diskettes in plain ASCII format. A library subscription provides four updates. Updates are issued about three times a year.

Documentation

In addition to the installation guides and users' manuals included with the program packages, there are three books on Icon. One contains a complete description of the language, the second describes the implementation of Icon in detail, and the third is an introductory text designed primarily for programmers in the Humanities.

There are two newsletters. *The Icon Newsletter* contains news articles, reports from readers, information of topical interest, and so forth. It is free, and is sent automatically to anyone who places an order for Icon material. There is a nominal charge for back issues of the *Newsletter*.

The Icon Analyst contains material of a more technical nature, including in-depth articles on programming in Icon. There is a subscription charge for the *Analyst*.

Payment






Payment should accompany orders and be made by check, money order, or credit card (Visa or MasterCard). The minimum credit card order is \$15. Remittance *must* be in U.S. dollars, payable to The University of Arizona, and drawn on a bank with a branch in the United States. Organizations that are unable to pre-pay orders may send purchase orders, subject to approval, but there is a \$5 charge for processing such orders.

Prices

The prices quoted here are good until May 1, 1991. After that, prices are subject to change without further notice. Contact the Icon Project for more current pricing information.

Ordering Instructions

Media: The following symbols are used to indicate different types of media:

-  9-track magnetic tape
-  DC 300 XL/P cartridge
-  360K 5.25" diskette
-  400K 3.5" diskette
-  800K 3.5" diskette
















All cartridges are written in raw mode. All diskettes are written in MS-DOS format except for the Amiga, the Atari ST, and the Macintosh.

CMS and MVS tapes are available only at 1600 bpi. When ordering UNIX or VMS tapes, specify 1600 or 6250 bpi (1600 bpi is the default). When ordering diskettes that are available in more than one size, specify the size (5.25" is the default).












Shipping Charges: The prices listed include handling and shipping by parcel post in the United States, Canada, and Mexico. Shipment to other countries is made by air mail only, for which there are additional charges as noted in brackets following the price. For example, the notation \$15 [\$5] means the item costs \$15 and there is a \$5 shipping charge to countries other than the United States, Canada, and Mexico. UPS and express delivery are available at cost upon request.

Ordering Codes: When filling out the order form, use the codes given in the second column of the list to the right (for example, AME, ATE, ...).












Icon Executables

Amiga	AME		\$15	[\$5]
Atari ST	ATE		\$15	[\$5]
MS-DOS	DE	 (2) or 	\$20	[\$5]
MS-DOS/386	DE-386	 or 	\$25	[\$5]
Macintosh (MPW)	ME		\$15	[\$5]
OS/2	OE	 or 	\$15	[\$5]
UNIX PC	UE	 or 	\$15	[\$5]
Xenix	XE	 (2) or 	\$15	[\$5]
Xenix/386	XE-386	 or 	\$15	[\$5]










Icon Source

Amiga	AMS		\$15	[\$5]
Atari ST	ATS		\$15	[\$5]
MS-DOS and OS/2	DS	 (2) or 	\$20	[\$5]
Macintosh (MPW)	MS		\$15	[\$5]
Porting (flat, ASCII)	PFS	 (5) or  (2)	\$40	[\$8]
Porting (hier., ARC)	PHS	 (2) or 	\$20	[\$5]
Source Updates (5)	SU	 (2) or 	\$50	[\$15]

Icon Program Library

Amiga	AML		\$15	[\$5]
Atari ST	ATL		\$15	[\$5]
MS-DOS and OS/2	DL	 or 	\$15	[\$5]
Macintosh (MPW)	ML		\$15	[\$5]
Porting (ASCII)	PL	 (2) or 	\$15	[\$5]
UNIX (cpio)	UL	 (2) or 	\$15	[\$5]
Library Updates (4)	LU	 or 	\$30	[\$12]

Complete Systems

CMS	CT		\$30	[\$10]
MVS	MT		\$30	[\$10]
UNIX (cpio)	UT-C		\$30	[\$10]
UNIX (cpio)	UC-C		\$45	[\$10]
UNIX (cpio)	UD	 (9) or  (4)	\$40	[\$8]
UNIX (tar)	UT-T		\$30	[\$10]
UNIX (tar)	UC-T		\$45	[\$10]
VMS	VT		\$30	[\$10]

Books

<i>The Icon Programming Language</i> (2nd ed.)	LB	\$34	[\$13]
<i>The Implementation of Icon</i> + update	IB	\$45	[\$14]
<i>Icon Programming for Humanists</i> + disk	HB	\$30	[\$10]

Newsletters

<i>The Icon Newsletter</i> (all back issues, 1-33)	INC	\$15	[\$5]
<i>The Icon Newsletter</i> (single issues, each)	INS	\$1	[\$0]
<i>The Icon Analyst</i> (1 year, 6 issues)	IA	\$25	[\$10]

Order Form

Icon Project • Department of Computer Science
Gould-Simpson Building • The University of Arizona • Tucson AZ 85721 U.S.A.

Ordering information: (602) 621-8448 • Fax: (602) 621-4246

name _____
 address _____

 city _____ state _____ zipcode _____
 (country) _____ telephone _____

check if this is a new address

qty.	code	description	price	shipping*	total

<p>Make checks payable to The University of Arizona</p> <p>The sales tax for residents of the city of Tucson is 7%. It is 5% for all other residents of Arizona.</p> <p>Payment <input type="checkbox"/> Visa <input type="checkbox"/> MasterCard <input type="checkbox"/> check or money order</p>	subtotal sales tax (Arizona residents) extra shipping charges* purchase-order processing other charges total	
---	---	----------------------------------

I hereby authorize the billing of the above order to my credit card: (\$15 minimum)

card number _____

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

exp. date _____

--	--	--	--



name on card (please print) _____

signature _____



*Shipping charges apply only to addresses outside the United States, Canada, and Mexico