The Icon Newsletter

No. 37 – November 1, 1991

Contents

For New Readers

If this is the first issue of the *Newsletter* that you've received, we'd like to welcome you to our readership, which now numbers more than 5,100.

If you're new to Icon, coming in at Issue 36 of the *Newsletter* may be a bit bewildering. If you find this issue interesting, you can get back issues at a nominal cost. See the order form at the end of this *Newsletter*.

If this is your first issue of the *Newsletter* and you didn't ask for it, you may wonder why you got it. Perhaps a friend told us you might be interested. Or maybe you inquired about a commercial version of Icon or SNOBOL4. Of course, if you're not interested, we'll be glad to remove your name from our mailing list.

If what's here looks interesting but you know little or nothing about Icon, we'll be glad to send you a short technical report that describes the main features of Icon. Just ask for the "Icon Overview". You can reach us by telephone, postal mail, electronic mail, or fax as listed in the publication box on page 9.

What's Going On with Icon?

Version 8 of Icon was released early in 1990. You may have been wondering what's been going on since then and what, if anything, we have up our sleeves.

We have several things in the works: the optimizing compiler for Icon, X-Window extensions for Icon, and a "multi-thread" version of Icon that allows several Icon programs to run and communicate under the control of a single interpreter.

The Compiler

The compiler is the biggest of these projects. A preliminary version of the compiler, based on Version 7.6 of Icon, was released for UNIX platforms earlier this year.

There are many differences between the run-time systems for the Version 7.6 compiler and the Version 8 interpreter, not even counting recent changes.

We've been working on bringing the compiler up to date so that persons using Icon could have a functionally compatible interpreter and compiler — the interpreter for fast and easy program development and the compiler for improved performance of finished products.

Merged Run-Time Systems

In order to reduce maintenance effort and to assure future compatibility between the compiler and interpreter, we decided to merge the two run-time systems.

The run-time system for the interpreter is written in C. Although the compiler run-time system is similar to the one for the interpreter, it is written in a specially crafted language called RTL, which contains extensions to C to allow the expression of the semantics of Icon that the compiler needs in order to perform optimizations and generate code. RTL also has provisions for describing run-time operations like type checking and conversion in a convenient way. Code written in RTL is translated into C by a program named rtt ("run-time translator").

We knew we were undertaking a major project. In fact, it's already occupied a good portion of the resources of four persons for over four months.

We now have the combined run-time system working, but some problems remain to be solved and, beyond them, we'll have to put together a new source distribution that is considerably larger and more complex than the former one for the interpreter.

There's a downside to the merged run-time systems for persons working with Icon source code. For the interpreter, there's now an extra step needed to translate RTL code to C code — and of course, there's another program, rtt, to worry about. Persons who have modified or extended the interpreter run-time system will have to retrofit their changes to use the new system, and that requires learning RTL first. The most serious problem is that RTL cannot handle data types that have been added to Icon's standard repertoire. Persons who rely on such extensions are marooned, at least for now, in the earlier Version 8 Icon interpreter.

X-Icon

In *Newsletter 36* we described extensions we've developed to add X-Window facilities to Icon. We call the result X-Icon for want of a better name.

Every so often we come up with minor additions and improvements to these facilities, but the interface and functionality have stabilized.

MT-Icon

The so-called multi-thread capabilities for Icon (going under the name MT-Icon) are less mature at this point, and we do not plan to include them in a public distribution in the near future.

Instead, these facilities will provide the basis for our research on program visualization in Icon, which may lead to a programming environment in which high-level visualization tools will play a major role.

You'll probably hear more about these visualization tools in future issues of the *Newsletter* and Analyst.

Upcoming Releases

Our first priority is to make Version 8 of the Icon compiler available. We hope to have an implementation for UNIX platforms ready later this fall, to be followed by a VMS implementation.

We'd like to bundle the compiler with the interpreter derived from the same RTL code. We might accomplish this for the compiler releases mentioned above, but we're not sure yet whether we'll be able to accompish this.

We're planning to start beta testing X-Icon soon, but we certainly won't have it ready for inclusion with the release of the Version 8 compiler. We hope to have X-Icon available for distribution sometime next spring.

Icon Compiler Documentation

Ken Walker's dissertation on the new Icon compiler is now available as a technical report. It covers topics such as type inference, liveness analysis, code generation, the language he devised for describing Icon's runt-time system, and the over-all structure of the compiler system.

Copies of this 120-page report are available for $6.50, postpaid in the United States, Canada, and Mexico. Add $3.50 for airmail postage to other countries.

To order a copy, just ask for TR 91-16 or "the compiler report" on the order form on page 15.

For a complete list of all Icon-related documents that are available, ask for IPD117. It's free.

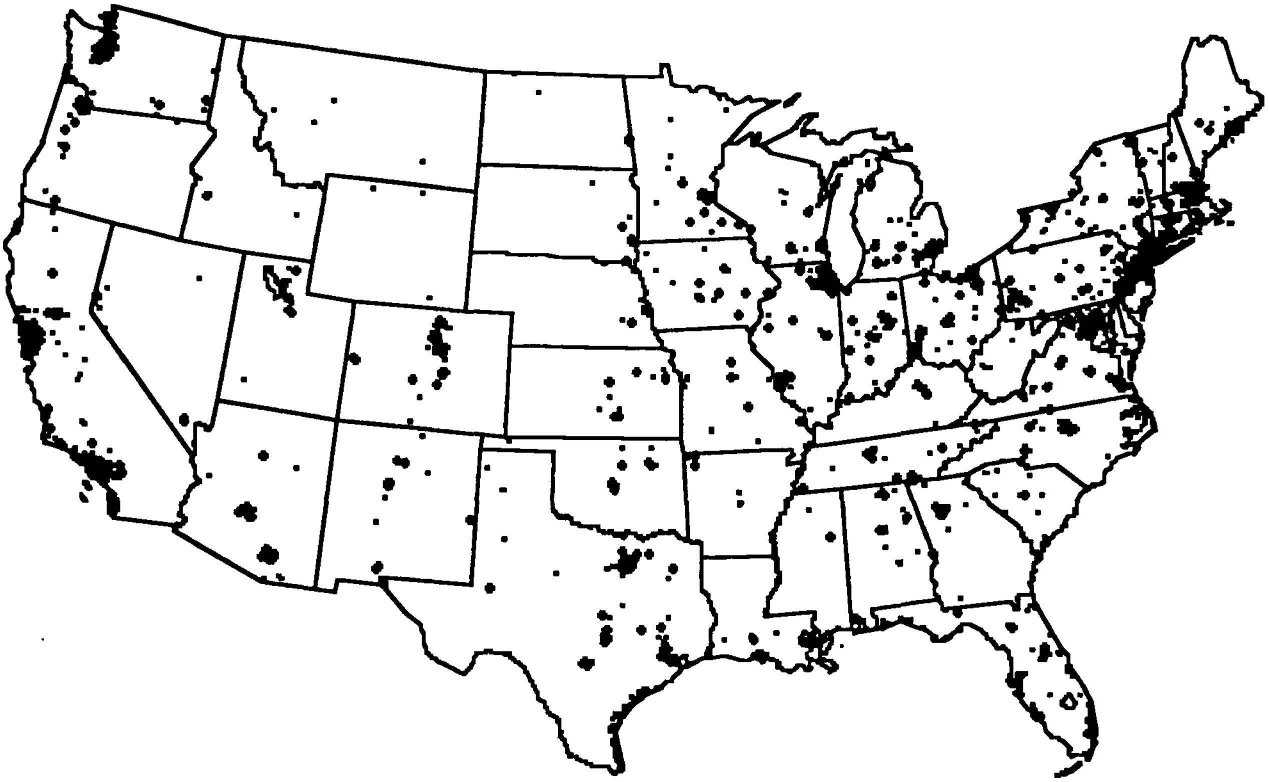
Geographical Distribution of Newsletter Subscriptions

About two years ago, in Issue 31, we published a list of subscribers to the *Newsletter* by country. At that time, we had 3,313 subscribers in 61 different countries. Now we have 5,125 subscribers in 63 different countries. Here's the current list:

United States	3889	Mexico	16
United Kingdom	244	Poland	16
Canada	192	Austria	15
Germany	139	Spain	14
France	84	Belgium	13
Australia	75	Switzerland	13
The Netherlands	73	Czechoslovakia	10
Japan	47	New Zealand	9
Sweden	30	Iran	8
India	24	Portugal	8
Finland	23	Norway	7
Israel	22	Republic of South Africa	7
Italy	17	Singapore	7
Brazil	16	Argentina	6
Denmark	16	Korea	6

Greece	5	Russia	2
Hong Kong	5	Saudi Arabia	2
Ireland	5	Taiwan	2
Philippines	5	Costa Rica	1
Venezuela	5	Cyprus	1
Chile	4	Dominican Republic	1
Iceland	4	Egypt	1
Indonesia	4	Fiji	1
Yugoslavia	4	Kuwait	1
Cuba	3	Luxembourg	1
Hungary	3	Malaysia	1
Thailand	3	Republic of Singapore	1
Algeria	2	Romania	1
Ecuador	2	Turkey	1
Pakistan	2	Uruguay	1
Peoples Republic of China	2	West Indies	1
Peru	2		

It's not surprising that most of our subscribers are in the United States. The map at the bottom of this page shows "push-pins" for subscribers according to their ZIP codes. You really can't get much more than a general idea from this, but it does indicate those parts of the country with dense and sparse subscription.



Newsletter **Subscriptions in the United States**

Here's the breakdown by states and territories for the United States:

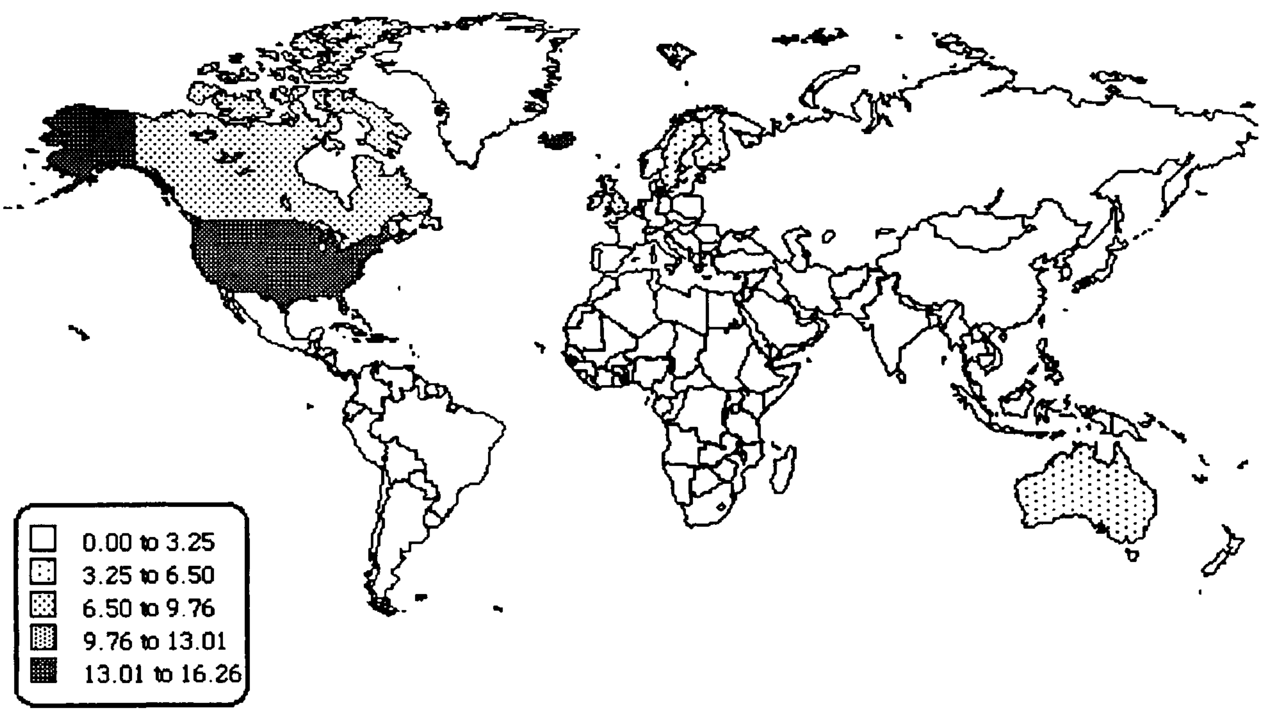
California	602	Utah	35
New York	299	Alabama	33
Illinois	286	District of Columbia	32
Arizona	229	New Hampshire	25
Texas	212	Louisiana	24
Massachusetts	187	New Mexico	24
New Jersey	162	Oklahoma	24
Pennsylvania	141	Kansas	22
Maryland	129	Nevada	21
Washington	123	South Carolina	19
Virginia	115	Hawaii	18
Ohio	101	Kentucky	17
Minnesota	95	West Virginia	13
Colorado	94	Arkansas	9
Michigan	92	Idaho	9
Florida	76	Maine	9
Oregon	71	Rhode Island	8
Indiana	61	Delaware	7
North Carolina	61	Vermont	7
Wisconsin	61	Mississippi	6
Connecticut	56	North Dakota	6
Missouri	45	Puerto Rico	6
South Dakota	45	Montana	4
Nebraska	44	Wyoming	3
Iowa	39	Alaska	2
Tennessee	39	Fed. States of Micronesia	1
Georgia	38	Guam	1

It's more difficult to give a "world view". The data is sparse and graphical techniques have serious limitations. But, for what it's worth, the Miller cylindrical projection of the world at the bottom of this page uses fill patterns to show the density of subscribers per million persons ("mega-capita") by country. Note that these patterns make no account for actual distribution within a country, which is why, for example, Alaska appears so heavily populated with subscribers to the *Newsletter*.

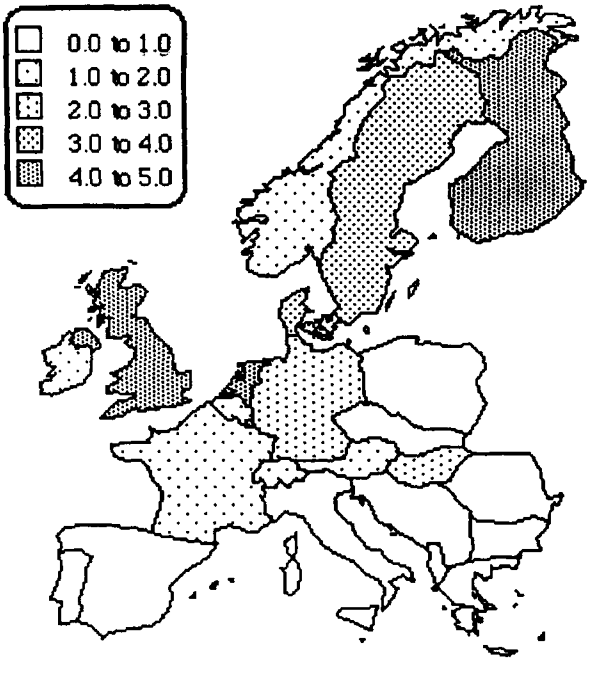
A similar map for Europe on the next page is more interesting.

Yet another view is given by the actual figures for the top few countries according to subscriptions per mega-capita. Countries with a small number of subscriptions have been omitted to avoid unduly distorting the figures — for example, Iceland has the highest subscription rate per mega-capita of all countries; true but unstable and not very meaningful.

United States	15.80
Canada	7.36
Israel	4.97
The Netherlands	4.88
Finland	4.65
Australia	4.55
United Kingdom	4.27



Newsletter **Subscriptions World Wide**



Newsletter **Subscriptions in Europe**

Executable Files for the Icon Interpreter

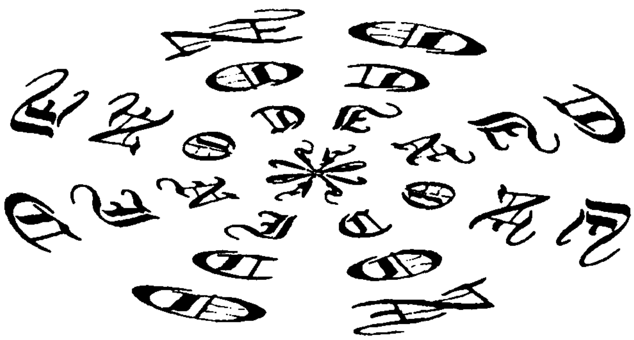
The distribution of the Icon interpreter for UNIX does not include executable binary files, since there are so many different platforms on which UNIX Icon runs.

If you don't want to go to the trouble of building the interpreter from scratch for your UNIX system, you may be able get executable binary files via FTP to cs.arizona.edu. After an anonymous login (See *Newsletter 36*),

```
cd /icon/interpr/binaries
```

and look around.

In addition to executable binary files for UNIX, there also are ones for VMS and several personal computers.



ISIcon Release

Iconic Software, Inc. provides this information about their first Icon product.

The ISIcon/SI Release 1.0 Screen Interface Development System is now available for UNIX/386 systems. ISIcon/SI is upward-compatible with the standard Version 8.0 Icon interpreter and has an enhanced terminal interface and other features unique to ISIcon.

The SI extensions support the rapid development of high-performance, terminal independent, full-screen user interfaces with ETI text windows, menus and forms on character terminals. Intended for professional software development, ISIcon/SI also includes Iconic's new module-level scoping, built-in function tracing, and enhanced translator warnings.

The retail price of ISIcon/SI is $695. ISI is offering ISIcon/SI at a special introductory price of $450 to readers of the *Newsletter* and **Analyst**. This offer expires December 1, 1991.

For additional information on ISIcon/SI or to order, contact ISI at:

Iconic Software, Inc.
P.O. Box 3097
Lisle, IL 60532

uunet!isidev!isi

(708) 429-9166

ProIcon 2.0 Upgrade

Catspaw, Inc. provides this information about an upgrade to ProIcon 2.0.

Version 2.04 of ProIcon has just been released. This upgrade is System-7 compatible, 32-bit clean, and supports TrueType fonts. A new function also has been added to allow the user to set window font-style properties.

The cost of this upgrade is $10 to owners of Version 2.0 of ProIcon and is free to owners who purchased ProIcon 2.0 after May 1, 1991. Owners of ProIcon 1.*x* can upgrade to 2.04 for $35 plus $5 shipping.

Persons who presently do not own ProIcon can purchase Version 2.04 for $175 plus shipping (which varies by country). Educational discounts and site licenses are available.

To order ProIcon 2.04 or the upgrade, contact:

Catspaw, Inc.
P.O. Box 1123
Salida, CO 81201-1123

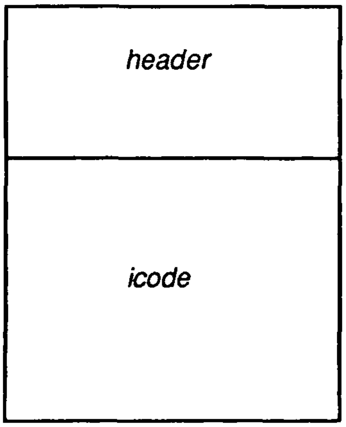
voice: (719) 539-3884
fax: (719) 539-4830

Smaller Icode Files for UNIX-Based Icon Implementations

Editors' note: This article was provided by Bob Alexander of Metaphor Computer Systems.

In this article, a method is discussed that can decrease the size of Icon icode files (the "executable" files produced by icont). The techniques discussed here apply only to Icon implementations that support the *direct execution* feature. Implementations that do not support direct execution, such as MS-DOS, cannot benefit from these techniques. In particular, this discussion applies to UNIX implementations, but the techniques may apply to other platforms (they are currently used in the standard Macintosh Programmer's Workshop implementation).

An icode file consists of a header part and an icode part:



The icode part contains the operations and data specifications that make up an Icon program. The header is the part that allows the icode file to be directly executable as a command, without having to type iconx.

In UNIX implementations of Icon, the header takes the form of a small program that invokes the iconx interpreter to execute the icode part of itself. In some versions of UNIX, the small header program ends up being disturbingly large once all the needed (and unneeded) library code is linked in (something like 16K bytes on a Sun 3!). Since the Icon program is contained in the icode part, the header is just overhead. For that reason, and since the header is replicated in each icode file, it is desirable that the header be as small as possible. This article discusses some alternative approaches to the header that can result in significantly smaller icode files.

The header program as implemented in the current Icon UNIX distribution is a C program. The header itself is the executable code produced by the C compiler and linker. Two other approaches are examined here: (1) using UNIX's shell-invocation format in place of a header, and (2) using a shell script as a header.

Shell Invocation Line Approach

A *shell* program to execute a script file can be specified in most versions of UNIX by beginning the file with a line of the following format:

```
#!shell-command-name
```

When such a file MyScript is executed as a command (it must have execute permission), the system behaves as if the following command had been issued:

```
shell-command-name MyScript
```

Viewing iconx (the Icon executor program) as a shell that executes icode files, the header can consist of a #! line instead of executable machine code:

```
#!/usr/icon/v8/bin/iconx
.........icode.........
```

Advantages:

- It's very small.
- It provides fast execution.

Disadvantages:

- The length of iconx file path name is limited (a maximum of 32 characters in SunOS).
- It doesn't fully support the documentation (for example, it doesn't recognize the ICONX environment variable).
- It requires a full path specification for the iconx interpreter (no variables are permitted).

Shell Script Approach

A shell script as a header can match the documentation, but it is not as streamlined as the shell invocation approach. However, it is the solution that I have adopted, and I've found it to be quite satisfactory in my SPARCstation 2 environment.

Advantages:

- It's small.
- It fully lives up to the documented behavior.
- It finds iconx in PATH specifications.

Disadvantage:

- It's not quite as fast as the #! method.

I wanted the script to be for the Bourne shell, since that is the ubiquitous shell available with all UNIX systems.

Here is a Bourne-shell version that does the job nicely (thanks to Gregg Townsend for suggesting this elegant and compact solution):

```
#!/bin/sh
exec "${ICONX-iconx}" "$0" "$@"
```

If the ICONX shell variable is undefined, the iconx executable file is located via the current PATH variable. A small variation to this script could point to an iconx executor in a specific hard-wired location; change the second line to:

```
exec "${ICONX-/usr/icon/v8/bin/iconx}" "$0" "$@"
```

Be warned: the first line, #!/bin/sh, *must be present* for this script to work properly under all circumstances. Even though UNIX will default to executing the Bourne shell if no #! line is present, the shell behaves somewhat differently depending on whether or not the #! line is included.

Even if you prefer another shell for interactive command entry (such as the C or Korn shell), usage of the above Bourne shell header presents no conflict. However, just for fun, the following slightly longer C-shell script could be used and is functionally equivalent:

```
#!/bin/csh -f
if ($?ICONX) set ICONX=iconx
exec "$ICONX" "$0" $*:q
```

Note the -f option provided to csh in the first line of the C shell script. The -f option (for "fast") bypasses the usual act of executing the .cshrc startup script, making it quicker.

How do I convert my Icon System to Use Small Headers?

The required steps are as follows:

1. Modify the icont makefile to omit construction of the C header.

2. Hand-code a .h file for the script header.

3. Reduce the space reserved in each output icode file for the header.

The header that is copied to every icode file resides in the translator/linker tool, icont. The makefiles provided with the Icon distribution compile the C program that comprises the header portion of normal icode files. A utility program is then executed that converts the resulting executable machine code to a .h text file that is #included in icont.

The makefile for icont (in the src/icont directory) should be modified:

- Omit the compile and link of the C header (ixhdr.c → ixhdr.o → iconx.hdr).

- Omit the conversion of the resulting executable to a .h file (iconx.hdr → hdr.h).

Then a hand-coded hdr.h file is constructed containing the desired script; for example (note that all lines in this example must be flush against the left margin):

```
static char iconxhdr[MaxHdr+1] = "\
#!/bin/sh\n\
exec \"${ICONX-iconx}\" \"$0\" \"$@\"\n";
```

Be careful; if you didn't modify your makefile correctly, the build could clobber your painstakingly hand-written hdr.h file.

Finally, adjust the size of the area reserved for the header in the file src/h/define.h:

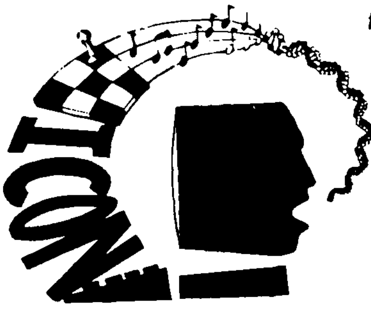
```
#define MaxHdr 256
```

The script I use (the example above) is fewer than 100 characters, but I specify a size of 256 to allow room for, dare I say, future expansion. It's still a lot smaller than 16K!

Now, rebuild icont, and enjoy smaller icode files!

Editors' note: The shell invocation line approach for headers that is described in the first part of this article once was used in UNIX Icon. The limit on the length of the path and the mysterious failures that resulted when the limit was exceeded caused us to abandon that approach.

Programming Corner



Rick Fonorow sent us this amusing tale of a strange encounter:

I have been doing a great deal of C programming lately, but at work I wrote an ICS "smart diff" in Icon. Well, the following lines made me waste over an hour!

```
/*if find (oldstring, line) then break */
if find(newstring, line) then break
```

I thought I was going nuts — until it dawned on me that the first find() was getting executed! (Then I realized I was using the C not the Icon style comment.)

In Icon, these lines parse fine. They test to see whether the size of the first if find() ... expression is null and multiplies that by the result of the second if find() ... expression if it is null.

⌘

Here's a puzzler: Consider a chain of exchanges of the form

```
x1 :=: x2 :=: x3 :=: ... :=: xn
```

Suppose, for concreteness ,that x1 = 1, x2 = 2, x3 = 3, ... xn = *n* before the expression above is evaluated. What are their values afterwards? Why? And can you think of a practical use for this kind of thing?

From Our Mail

In Issue 34 of the Newsletter, you said that the frequency of publication would be reduced to two times a year. But in Issue 35, you say it's three times a year. Which is it, please?

The *Newsletter* doesn't have a fixed publication schedule. We originally said it was published aperiodically (a term that several persons misinterpreted). A couple of years ago we realized that despite the lack of a publication schedule, we were, in fact, getting out three issues a year rather regularly. We decided we could continue with that and we made it official.

When we started the Analyst, we expected to reduce the frequency of publication of the *Newsletter* to twice a year. We discovered, however, that there was news that needed to get out sooner than a semi-annual publication schedule allowed, so we've maintained our earlier rate of publication of three issues a year. On occasion, as we did in Issue 35, we may reduce the number of pages from our ordinary 12 or increase them, as we have in this issue.

Incidentally, since the *Newsletter* is free, we don't feel the same obligation to meet as fixed schedule as we would if subscriptions were paid. We intend, nonetheless, to continue to publish three issues a year.

Is the Icon news group the same as comp.lang.icon?

No, but ... Electronic mail sent to the Icon news group (icon-group@cs.arizona.edu) is automatically distributed to news group subscribers. Most of this mail also appears in the Uunet news group comp.lang.icon, although it occasionally is somewhat delayed. If you subscribed to comp.lang.icon, you probably don't need to subscribe to the Icon news group as well.

I'm curious about The Icon Analyst. *I've thought about subscribing, but I'm not sure I want to pay $25 a year for something I've not seen. What's in it? How many subscribers do you have?*

As we mentioned in *Newsletter* 36, we'll be happy to send a sample copy of the Analyst to interested persons; it's the best way to see what it's like. A copy is in the mail to you.

We have just over 200 subscribers to the Analyst at present. We can't give you an exact figure because we lose a few subscribers who don't renew but we add new ones also.

Incidentally, when we started the Analyst, we hoped we'd get at least 200 subscribers and figured it would be worth the effort if we had only half that number. In our more optimistic moments, we hope to get as many as 500 subscribers.

Does Amiga Icon source compile under SAS 5.1 C (formerly Lattice C)? Has anyone done anything to interface Icon to the Amiga graphical user interface (Amiga Workbench)?

The last version of Lattice C that we used to compile Icon for the Amiga was 5.0.4. We were unable to get co-expressions working under that release because of stack checking. We no longer have access to an Amiga with enough resources to compile Icon.

We're not aware of any work on interfacing Icon to the Amiga Workbench, although we've had several inquiries about it.

Perhaps some of our readers can provide more information.

How about adding regular expressions to Icon? And SNOBOL4 patterns. Maybe Prolog-style unification?

There's an old (?) saying: "You can put only so many warts on a toad. After that, you have to buy a new toad." Icon is pretty well "warted up". You take it from there. (Yes, folks, we make some of these up. We must.)

Why do you waste whole page of the Newsletter with some funny-looking picture on the back cover? I'd rather have more news about Icon.

The *Newsletter* must be an even multiple of four pages because of the way it's printed. A good balance between the amount of material, frequency of publication, and mailing costs is eight or 12 pages. We first put a graphic on the back cover when we had 11 pages of material and didn't want to delay publication until we had more. Since then we've had numerous compliments on the various graphics we've used to add visual interest to the *Newsletter*. When we published *Newsletter 35* without a back-cover graphic we had several complaints. (Perhaps some persons prefer the graphics to the news.) We realize different readers feel differently about this, but on balance we think graphics make the *Newsletter* more attractive. And it's an interesting challenge to continually come up with new Icon-related graphics.

I missed out on the first year of The Icon Analyst. *I want to subscribe, but I'd like to start at the beginning. Can I do that, or do I have to buy back issues?*

We accept two-year subscriptions to the Analyst, starting with the first issue. That's slightly cheaper

Downloading Icon Material

Most implementations of Icon are available for downloading electronically:

BBS: (602) 621-2283

FTP: cs.arizona.edu (cd /icon)

than buying back issues. Be sure to specify on your order that you want your subscription to start with the first issue. And you must, of course, subscribe for two years to cover the issues you've missed.

Does the C source code for Icon require an ANSI-C compliant compiler.

No, but it helps. Support for function prototypes is important except in the simplest situation in which C integers, longs, and pointers are all 32 bits long and pointers are represented by simple integer addresses. In addition, the source code for Icon requires a robust and full-featured C compiler that supports (at least) 32-bit pointers. It will not compile, for example, under the MS-DOS small-memory model.

What are the advantages of the 386 version of Icon over the normal MS-DOS version?

The 386 version of MS-DOS Icon (which also runs on 486 processors) runs in 32-bit protected mode. It's about 3 times faster than the standard MS-DOS version of Icon. Perhaps more important, the 386 version can access all available memory. See the article on page 12 for information about a new version of 386 MS-DOS Icon.

Have you compiled Icon under Borland C++?

No. We have a hard time keeping up with new compilers. We rely on our users for much of this kind of thing. If anyone has Icon compiled and running under Borland C++, we'd appreciate knowing about it.

An article in the last Newsletter expressed concern that the Icon compiler would not run on MS-DOS platforms because it requires so much memory. Frankly, I don't think that is a real concern, and is getting less and less important all the time. So many machines are being equipped these days to run Windows, CASE tools, and a lot of other pieces of software, that MS-DOS machines with lots of memory will soon be the rule rather than the exception.

We were thinking of "standard" MS-DOS platforms when we made our comments. What you say is certainly true. And with 32-bit C compilers and memory extenders, it's possible to access memory in the way Icon needs it. Times are changing, but there will continue to be many older MS-DOS machines with limited memory (we still have persons interested in Icon who do not have even 512K of RAM). All we're saying is that we can't offer any hope to persons running personal computers with limited memory.

In Newsletter 36, Eric Weiss suggested that "inventions must be more than clever, they must be needed." May I propose a modification — "they must either be needed or marketed, or even better, needed and marketed!"

We're reminded how much times have changed. In the early days of SNOBOL, marketing for a new programming language wasn't necessary at all — there was an eagerly waiting clientele.

Can Icon read Microsoft Word files, or can it handle only ASCII text.

Icon can read any file, text or binary. However, it does not provide automatic translation for the many special and proprietary formats used by word processors and similar programs. It has no built-in provision for translating Word-format files to text. If you know the format in which Word stores it's documents, you could, in principle, write a translator in Icon, although that probably would be a formidable job.

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,allegra,noao}!arizona!icon-project

THE UNIVERSITY OF
ARIZONA
TUCSON ARIZONA

and

The Bright Forest Company
Tucson Arizona

Programming Language Archives

Newsletter 35 contained an article about the oral history of Icon initiated by the Charles Babbage Institute, an organization dedicated to the study of the history of information processing.

Bruce Bruemmer, the archivist for CBI, visited the Icon Project in the fall of 1990 to examine its accumulation of material related to programming languages. This material dates from the early work on SNOBOL in 1962 and continues through the subsequent SNOBOL languages, SL5, and Icon. The material includes correspondence (some 16 linear feet of it), electronic mail, books, technical reports, program material, photographs, and all kinds of odds and ends, even T-shirts.

This material is scheduled to go to CBI's archives sometime in the future. Meanwhile, we are continuing to accumulate material (perhaps "pack-rat" is the proper verb) and are attempting to put it in order. We even have an office set up in a storage locker deep inside an massive stone building. It's a strange and inhospitable place to work, but at least there is no telephone or computer to distract.

The historical importance of this kind of material depends in part on its completeness. If you have technical reports, correspondence, photographs, or other material related to the SNOBOL, SL5, or Icon programming languages, we would greatly appreciate your contributing it to the Icon Project. If you suspect we already have a copy of what you have, you can ask us before sending it — we have a data base that catalogs our holdings. Or you can just send it and let us sort it out.

Faculty Positions at the University of Arizona

The Department of Computer Science at The University of Arizona invites applications for faculty positions at all ranks to begin in August, 1992. Applicants must have a doctorate in Computer Science or a closely related field. Applicants for junior positions should show promise of future excellence, while applicants for senior positions should have made substantial research contributions to the field.

There are currently 14 faculty members with research areas spanning the field from software systems to programming languages and theory of computation.

For recruiting purposes, current areas of high priority include graphics, user interfaces, databases, programming languages, compilers, parallel computing, computer architecture, performance evaluation, and computational biology. Exceptionally well-qualified people working in other areas are also encouraged to apply.

The research program is supported by numerous grants to individual faculty as well as a second department-wide infrastructure grant from NSF.

Computational facilities are diverse, including numerous Sun workstations, a Silicon Graphics 4D/340 VGX and two Personal Iris graphics workstations, two DecStation 5000s, a Sequent Symmetry, a NeXT machine and dozens of Macintoshes. Also available are high-resolution color terminals, a QMS color PostScript printer, color scanners, numerous laser printers, and an L-300 image setter.

Send a complete resume and the names of at least three references to:

Richard D. Schlichting
Faculty Recruiting Committee Chairman
Department of Computer Science
The University of Arizona
Tucson, AZ 85721

Applications will be reviewed beginning January 15, 1992, but the positions will remain open until filled. The University of Arizona is an equal opportunity/ affirmative action employer.

Thanks from the Editors ...

Individuals associated with the Icon Project sometimes contribute one way or another to the material that appears in the *Newsletter*. We usually don't explicitly acknowledge these contributions, since that's more are less part of their jobs.

We do, however, want to express special thanks Gregg Townsend and Ken Walker for the contribution they make on a continuing basis in reading the *Newsletter* before it goes to press. They find all kinds of mistakes and often suggest improvements to the content.

Any remaining errors are, of course, the responsibility of the editors.

Icon Seminar

Tom Reid provides this press release about an upcoming seminar.

The Washington, D.C. Chapter of ACM Professional Development Committee presents over 20 one-day tutorials a year taught by many of the world's top computer scientists. During the week of November 18-22, 1991, the PDC will offer 10 tutorials. Of special interest is the tutorial on November 22 on the Icon programming language given by Ralph E. Griswold.

Dr. Griswold is Regents' Professor of Computer Science at The University of Arizona. He has nearly 30 years of experience in the design, implementation, and use of high-level programming languages. He started his work at Bell Laboratories, where he co-authored the SNOBOL series of programming languages for string manipulation. Since 1971 he has continued his programming-language work at Arizona, concentrating on facilities for non-numerical computation. He is the author or co-author of six books on programming languages and their implementation.

Other tutorials given the same week as Dr. Griswold's are:

- Object-Oriented Program Design Using C++ by David Bern and Peer Reviews — Theory and Practice by Richard Cohen on Monday, November 18.
- Code Metrics and Design Metrics by Dr. Wayne M. Zage and Software Reverse Engineering by Dr. Hasan Sayani on Tuesday, November 19.
- Applying Statistical Process Control to Software Development by Barba Affourtit and Software Repository and Bridge Technology by Dr. Robert Arnold on Wednesday, November 20.
- Information Modeling by Dr. Cy Svoboda and Unit Testing During Maintenance by Thomas Bogart on Thursday, November 21.
- Software Maintenance Technology by Nicholas Zvegintzov on Friday, November 22.

The tutorials will be given at the University of Maryland Center of Adult Education in College Park, Maryland. Registration can be by mail or telephone. For a brochure describing all of the courses, please call the PDC's answering machine at 202-462-1215. Phone registration by credit card can be made by calling Ms. Eliane Van Ty Smith at 301-299-4286. Prices for check or credit card are $170 for Washington DC ACM chapter members and $175 for non-members by November 4; $205 after November 4. Purchase order are $230. Full-time students and senior citizens are $60.

If you have questions, please call the PDC answering machine at 202-462-1215 or contact Tom Reid by electronic mail at reid@vtopus.cs.vt.edu.

Icon Program Library Update

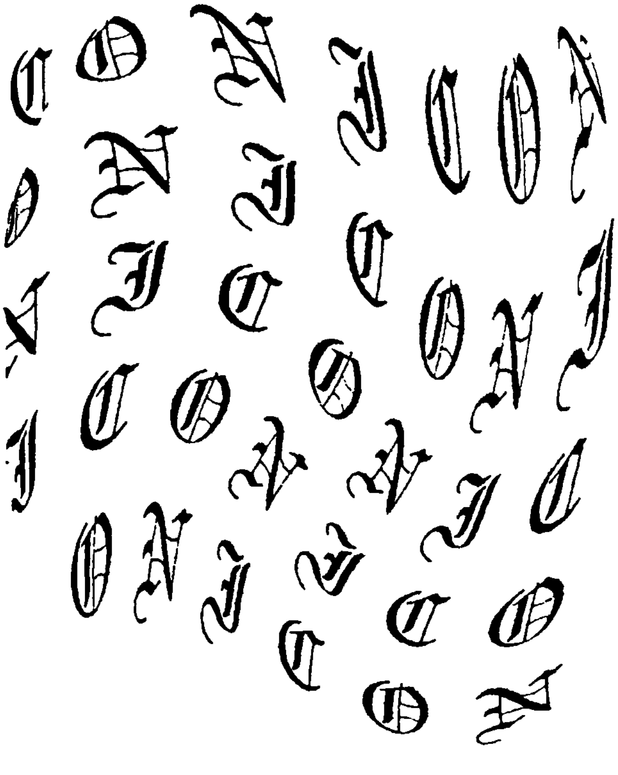
The Icon program library has been updated to Version 8.1. This version contains nearly twice as much program material as the original Version 8 release. Version 8.1 contains 108 complete programs, 114 packages of procedures, the latest release of Idol, the object-oriented version of Icon, and data files.

Although the entire library only gets updated every year or two, there is an update subscription service. Persons subscribing to this service get updates about four times a year. The fourth update was sent out recently and brings subscribers up to Version 8.1 of the library.

In order to clear an accumulation of older material and to handle an increased volume of new material, we're doubling the size of future subscription updates to about 700KB. The price of the subscription updates remains the same.

Subscribers to the update service not only get material more quickly than through re-issue of the entire library, but they also get material that will not be included in official library releases: complicated packages, "grab bags" of interesting material that is not in library format, and large data files, such as word lists.

See page 15 of this *Newsletter* for information on how to order Version 8.1 of the Icon program library or how to subscribe to the update service.



Graphics Credits

The map on page 3 was produced using GeoQuery The maps on pages 4 and 5 were produced using MapMaker and touched up in Adobe PhotoShop 2.0.

The graphics on pages 5, 10, 11 and the back cover were produced using Adobe Illustrator 3.0, Adobe PhotoShop 2.0, and Adobe Streamline 2.0.

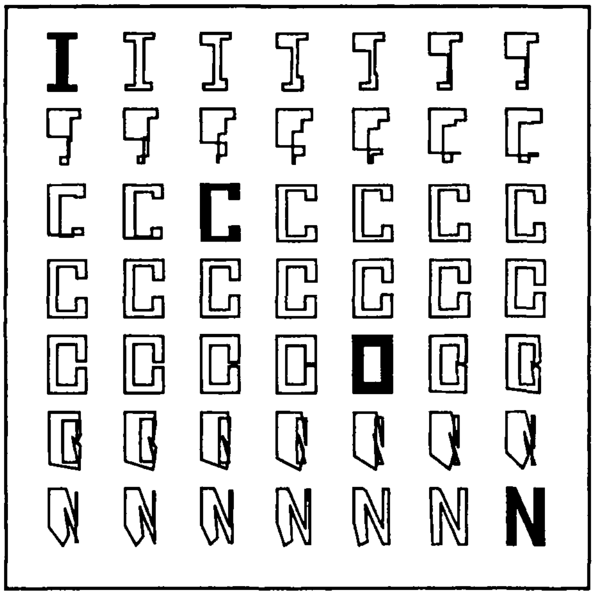
As usual, all work was done on a Macintosh.

Tweening

Tweening is a term used in computer graphics for the process of changing one form to another through a succession of steps.

The transformation of the letters I C O N in the graphics below was done using an X-Icon program written by Steve Wampler for tweening polygonal shapes.

Scan across, row by row, to see the successive polygons.



A Puzzle

What's this?

```
2020 2020 2020 2e0a 2020 2020 202f 0a20
2020 2020 202f 5c0a 2020 2020 2020 5c0a
2020 2020 2f5c 2020 2f5c 0a20 2020 205c
2f20 205c 2f20 205c 0a20 202f 2020 2020
2020 202f 5c20 270a 2e20 5c2f 2020 2020
2020 202f 0a20 5c20 202f 5c20 202f 5c0a
2020 2020 5c2f 2020 5c2f 0a20 2020 2020
2020 5c0a 2020 2020 2020 5c2f 0a20 2020
2020 2020 202f 0a20 2020 2020 2020 270a
```

We'll have the answer in the next *Newsletter*.

Public-Domain 386 MS-DOS Icon

Until recently, the only version of MS-DOS Icon that took advantage of the capabilities of 386 and 486 processors was a commercial product built using Metaware HighC and incorporating the Phar Lap memory extender. Phar Lap licensing requirements prevented this version from being placed in the public domain.

We now have a public-domain version of 386 Icon for MS-DOS built using the new Intel Code Builder, which incorporates its own memory extender. The price of 386 Icon for MS-DOS has been reduced accordingly. See the order form on page 14.

Here's some information taken from the release notes for the new version:

> Icon/386 incorporates a DOS extender product from Intel Corporation. This DOS extender enables a 80386 system to run in 32-bit protected mode with access to extended memory. It operates with 80386, 80386SX, 80486, and 80486SX processors.
>
> The Intel DOS extender operates with MS-DOS Versions 3.0 through 5.0. It runs under native MS-DOS using all extended memory installed, or with hosts providing a DOS Protected Mode Interface (DPMI) Version 0.9 or above. For example, Windows 3.0 provides a DPMI host in a DOS window when Windows is started in enhanced mode. When released, OS/2 2.0 is also expected to provide a DPMI host in a DOS window.
>
> The Intel Extender requires a 387 co-processor or a 486 in order to run under Windows 3.0. These are *not* required for operation under native DOS, where Intel's built-in floating-point emulator works properly.
>
> The Intel DOS extender is not compatible with Virtual Control Program Interface (VCPI) hosts. Therefore, VCPI-compliant expanded memory managers such as QEMM-386 or 386Max cannot be used with this version of Icon/386. VCPI multitasking systems such as DESQView-386 are also not compatible with this DOS extender.
>
> Intel's DOS extender is a new product and may not work in all environments. In particular, it is not compatible with some very old 386 systems using the very first 32-bit BIOSs. (Phoenix BIOS version ".10" and early AMI BIOSs are problems.)

This new version of 386 MS-DOS Icon was provided by Mark Emmer, president of Catspaw, Inc. It is the latest of many contributions Mark Emmer has made to the SNOBOL4 and Icon programming communities. We owe him a great deal for all of his help.

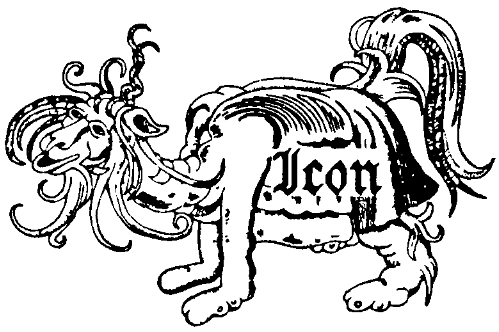
Ordering Icon Material

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 listed here is in the public domain.

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 four 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.

What's New

New and updated releases are identified by the symbol ☛ .

Prices

The prices quoted here are good until December 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 💾	$15	[$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]
Update subscription (5)	SU	■ (2) or 💾	$50	[$15]
Icon Program Library				
Amiga	AML ☛	💾	$20	[$5]
Atari ST	ATL ☛	💾	$20	[$5]
MS-DOS and OS/2	DL ☛	■ (2) or 💾	$20	[$5]
Macintosh (MPW)	ML ☛	💾	$20	[$5]
Generic (ASCII)	PL ☛	■ (4) or 💾 (2)	$35	[$5]
UNIX (tar compressed)	UL ☛	■ (2) or 💾	$20	[$5]
Update subscription (4)	LU	■ (2) 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				
The Icon Programming Language (2nd ed.)	LB		$34	[$13]
The Implementation of Icon + update	IB		$50	[$14]
Icon Programming for Humanists + disk	HB		$30	[$10]
Newsletters				
The Icon Newsletter (all back issues, 1-36)	INC		$15	[$5]
The Icon Newsletter (back issues, each)	INS		$1	[$2*]
The Icon Analyst (1 year, 6 issues)	IA		$25	[$10]
The Icon Analyst (back issues, each)	IAS		$5	[$2*]

* Per order, regardless of the number of issues purchased.

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

subtotal	
sales tax (Arizona residents)	
extra shipping charges*	
purchase-order processing	
other charges	
total	

Make checks payable to The University of Arizona

The sales tax for residents of the city of Tucson is 7%.
It is 5% for all other residents of Arizona.

Payment □ Visa □ MasterCard □ check or money order

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

card number □□□□□□□□□□□□□□□□

exp. date □□□□

MasterCard

VISA

name on card (please print) ______________________________

signature ______________________________

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