Icon Program Library Submissions

Ralph E. Griswold

Department of Computer Science
The University of Arizona
Tucson, Arizona

IPD151e
March 5, 1996
http://www.cs.arizona.edu/icon/docs/ipd151.html

The Icon program library consists of programs, collections of procedures, include files, data, documentation, and packages that contain more complex applications. The programs range from simple utilities to sophisticated text-generation applications.

In addition to providing useful programs and procedures, the Icon program library also is an interesting source of examples of programming in Icon, which may be useful to persons learning Icon.

Most of the material in the Icon program library is contributed by users -- the work of several dozen programmers.

The Icon Project accepts contributions to the Icon program library, checks them to make sure they work, and packages them for distribution. Since many of the programs are complicated, it isn't practical for us to test them extensively or to certify that they work entirely as advertised. Nonetheless, many of the programs in the library are in regular use.

Contributions to the Icon Program Library

We welcome submissions of new material to the Icon program library, but there are a few requirements. Look at the examples in the next section.

Contributed programs must be adequately documented and should be accompanied by test data. Test files that are read as standard input and produce results to standard output are best. In some cases, data read from standard input may not be appropriate. Send what you think is best, but realize that our resources are limited and programs that are easy to test are more likely to get into the library and to get there sooner.

Contributions to the Icon program library must be free of any distribution restrictions.

We reserve the right to modify submitted programs to correct errors, to improve code, and to make them conform to library conventions.

Finally, we reserve the right to decide what material to include in the library and to modify it as we feel appropriate. The main considerations are quality, usefulness, and compliance with the library format. Submissions that duplicate functionality that already exists in the library generally are not accepted unless they are substantially better than what's already in the library.

All submissions are acknowledged, but decisions on inclusion in the library may not be made immediately. If you've submitted material in the past that's not appeared in the library, ask us about its status - we have a lot of material from past years that lacks adequate documentation. And in some cases we've lost touch with the persons who submitted material.

Of course, we give full credit to the authors of program material that's included in the library.

Even if you don't have anything to contribute to the library, you may find a bug or make an improvement to the library. By all means let us know about such things.

Example of a Program

Heading information must be in the format shown (it's processed by programs used to maintain the library). Keep the subject on one line and as short as possible. Do not put other information, including version and modification information in the header box; place it below.

A description of the material in the file must follow the heading. It should explain what the program does and be sufficiently detailed so that use is clear. Command-line options should be clearly evident.

A description of the material in the file must follow the heading. It should explain what the program does and be sufficiently detailed so that use is clear. Command-line options should be clearly evident.

The code itself should begin with link, global, and record declarations.

The main procedure should appear first, followed by other procedures, preferably in alphabetical order.

Declare all local identifiers.

```
############################################################################
#
#       File:     deal.icn
#
#       Subject:  Program to deal bridge hands
#
#       Author:   Ralph E. Griswold
#
#       Date:     June 20, 1994
#
############################################################################
#
#     This program shuffles, deals, and displays hands in the game
#  of bridge. An example of the output of deal.icn is
#
#                 S: KQ987
#                 H: 52
#                 D: T94
#                 C: T82
#
#       S: 3                  S: JT4
#       H: T7                 H: J9863
#       D: AKQ762             D: J85
#       C: QJ94               C: K7
#
#                 S: A652
#                 H: AKQ4
#                 D: 3
#                 C: A653
#
```

```
#  Options: The following options are available:
#
#       -p;h n Produce n hands. The default is 1.
#
#       -p;s n Set the seed for random generation to n.  Different seeds give
#            different hands. The default seed is 0.
#
############################################################################
#
#  Links: options, shuffle
#
############################################################################

link options, shuffle

global deck, deckimage, handsize, suitsize, denom, rank, blanker

procedure main(args)
   local hands, opts

   deck := deckimage := string(&letters)        # initialize global variables
   handsize := suitsize := *deck / 4
   rank := "AKQJT98765432"
   blanker := repl(" ",suitsize)
   denom := &lcase[1+:suitsize]

   opts := options(args,"h+s+")
   hands := \opts["h"] | 1
   &random := \opts["s"]

   every 1 to hands do
      display()

end

#  Display the hands
#
procedure display()
   local layout, i
   static bar, offset
         .
         .
         .
```

Example of a Collection of Procedures

Heading information for collections of procedures is similar to that for programs.

A prototype call for each procedure should be given with an example of the results, if appropriate.

Each procedure should have a brief explanatory comment on the line in which the reserved word procedure appears. A colon must immediately follow the sharp sign if the procedure is to appear in the index listing for the library.

```
############################################################################
#
#       File:     numbers.icn
#
#       Subject:  Procedures to format and convert numbers
```

```
#
#       Author:   Ralph E. Griswold, Tim Korb, and Robert J. Alexander
#
#       Date:     May 28, 1995
#
############################################################################
#
#     These procedures format numbers in various ways:
#
#     commas(s)         inserts commas in s to separate digits into groups
#                       of three.
#
#     digred(i)         reduction of number by adding digits until one digit
#                       is reached.
#
#     div(i, j)         produces the result of real division of i by j.
#
#     fix(i, j, w, d)   formats i / j as a real (floating-point) number in
#                       a field of width w with d digits to the right of
#                       the decimal point, if possible. j defaults to 1,
#                       w to 8, and d to 3. If w is less than 3 it is set
#                       to 3. If d is less than 1, it is set to 1. The
#                       function fails if j is 0 or if the number cannot
#                       be formatted.
#
#     roman(i)          converts i to Roman numerals.
#
#     spell(i)          spells out i in English.
#
#     unroman(s)        converts Roman numerals to integers.
#
############################################################################

procedure commas(n)      #: insert commas in integer

   if *n < 4 then return n
   else return commas(left(n, *n –p; 3)) || map(",123", "123", right(n,3))

end
                .
                .
                .
```

Submitting Library Material

Material for the library must be in machine-readable form. You can send library submissions to us on MS-DOS or Macintosh diskettes, by e-mail to icon-project@cs.arizona.edu, or upload them to our FTP incoming area. If you upload a file, notify us by e-mail so we can pick it up before it gets deleted by the automatic garbage-collection process.

If you want to compress your submission, for Unix, use compress; for MS-DOS Lharc, Lha, or Zip; or, for the Macintosh, a BinHexed self-extracting archive or any other format Stuffit Deluxe can handle. If you'd like to use another format, check with us before uploading to be sure we can deal with it.
