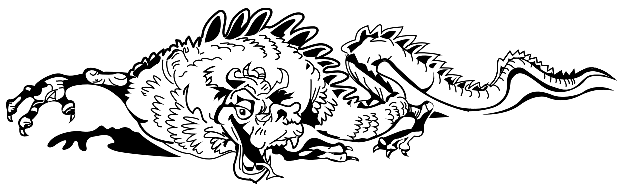
The Icon Newsletter

No. 47 – June 15, 1995



Contents

New Area Code

Most of Arizona, including Tucson, now has a new area code, 520. Until July 23, 1995, our old area code, 602, also will work. After that, you'll have to use 520 to reach us.

A word of warning: Some automated switchboards only can dial area codes that have a second digit that is a 0 or 1. If you try to call us at 520 and are unable to reach us, that may be the problem.

Also note that we have a new telephone extension for inquiring about Icon and for placing orders for Icon material.

Status Report

We are continuing to concentrate on implementations of Version 9 of Icon with graphics for additional platforms as well as the completion of the graphics programming book.

Clint Jeffery has a 32-bit implementation of Version 9 for Windows in beta testing. We hope to have a general release this fall.

Cheyenne Wills has provided us with a preliminary implementation of Version 9 of Icon for OS/2. This implementation does not support mutable colors or the GIF image file format, but it otherwise is substantially complete. You can download it from our FTP site: `cd /icon/alpha/os2` and check out what's there. We hope to have a general release later this summer.

Clint Jeffery's student who started an implementation of Version 9 for the Macintosh is no longer working on the project. The present state of this implementation is best described as "pre-alpha-test". We're looking for a volunteer with the time and resources to complete this implementation, which uses THINK C 6.0. If you're interested, contact us.

The graphics programming book still is behind schedule, but we've made a lot of progress. We gave the book a "field test" in an undergraduate course on graphics programming during the spring semester (see the article starting on page 4). The book worked out quite well in the course, but as expected, numerous problems surfaced. We have revised and reorganized some parts of the book as a result, as well as adding a few chapters and appendices. The result certainly will be a better book — and a larger one (a dubious virtue).

We hope to have the book complete in draft by the end of the summer.

Icon Analyst

The Icon Analyst is entering its sixth year of publication. When we started, we had no expectation that it would last so long and had some trepidation when a charter subscriber signed up for five years.

Those of you who subscribe to the Analyst probably realize that it takes a lot of time and effort to produce an issue. Sometimes we worry that we'll miss a publication deadline, but so far, we haven't. In spite of the work involved, we enjoy doing the Analyst and hope to continue for some time.

If you're not a subscriber to the Analyst and are interested in what material has appeared in past issues, you can download a table of contents from our FTP site: cd /icon/doc and get iatoc.doc. We also will send printed copies of the contents as well as sample copies of the Analyst on request.

All back issues still are available and there's a package price for a complete set.

With the next issue of the Analyst, we're starting a series of articles on building visual interfaces in Icon: how to build interfaces that use buttons, menus, scroll bars, and so forth. Then we'll go on to the organization of programs that communicate with users through visual interfaces. If you're interested in this kind of thing and don't already subscribe to the Analyst, now would be a good time to start.

Downloading Icon Material

Most implementations of Icon are available for downloading via anonymous FTP:

ftp.cs.arizona.edu (cd /icon)

Diskette Distribution

Some of the program material we distribute has grown in size to the point where it's no longer practical to provide it on low-density diskettes. Fortunately, high-density diskette drives are now standard on most personal-computer platforms.

We therefore no longer offer some material on low-density diskettes. (We may, however, use low-density diskettes when a specific distribution happens to fit on one.) This change affects some source code and program library distributions. We also no longer provide MS-DOS on 5.25" diskettes.

We apologize to anyone who may be inconvenienced by these changes.

Icon Program Library

The Icon program library continues to grow. It presently contains 321 complete programs, 9 program packages, and 2,080 procedures to link with programs. And more material is on the way.

We plan to release a new version of the library this fall. Subscribers to the library update service get new material three times a year and don't have to wait for new versions of the library, which usually are only released once every few years.

In our experience, many Icon programmers do not make good use of the library. Part of the reason for this probably is a start-up problem: installing the library, setting up environment variables, and thinking in terms of the library when programming. A more serious problem is finding things in the library, especially useful procedures.

We provide keyword index listings for files in the library, but the information in these listings is limited to what appears in file subject heading lines. In many case there is no clue in these listings about individual procedures.

When we use Icon in courses, we provide source code for the library and the index listings. Resourceful students have made good use of these resources. Other students have spent much time and effort "reinventing the wheel", sometimes incorrectly, which we call "reinventing the flat tire".

We are working on a project to provide index listings of individual procedures and hope to have it ready for the next version of the library. This should make the library significantly more accessible and useful.

We know from experience, however, that nothing will completely solve the problem. In a recent course, one student spent a considerable amount of time and effort implementing Julian date conversion — incorrectly. And in the last class before a major project was due, another student asked if library had any procedures that dealt with Julian dates. Both of these incidents happened despite the existence of a file named julian.icn in the library and the availability of both on-line and printed index listings clearly identifying this file. We were left with a hopeless feeling.

Many persons have contributed to the Icon program library, both with new material and improved versions of existing material. We appreciate such contributions; if you have something of interest that you are willing to share, please do. We usually are able to get new submissions into the next library update so that subscribers can benefit from them as soon as possible.

It will help us and also get your submissions into the library more promptly if you adhere to a few commenting conventions that we rely on for library maintenance and to generate index listings. Except for a new form of comment that we plan to use for producing index listings of individual procedures, you can look at any file in the library for guidance. What's important is the stylized header.

To get the information for indexing procedures, we plan to use a brief comment following each procedure declaration in the procedure portion of the library. To distinguish these comments, a colon immediately follows the sharp sign, as in

```
procedure dopen(s)    #: open file on DPATH
```

The comment must be short; otherwise the lines in index listings are overly long or are truncated.

If you want more details on conventions for Icon program library, you can get them from our FTP site: cd /icon/doc and get libsub.doc. We'll also send a printed copy on request. Ask for IPD151.

You can send submissions to us on MS-DOS or Macintosh diskettes, by e-mail, or upload them to /incoming at our FTP site. 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, use UNIX compress; MS-DOS lharc, lha, or zip; or, for the Macintosh, a binhexed self-extracting archive or a format Stuffit can handle. If you'd like to use another format, check with us before uploading to be sure we can deal with it.

Icon on the Web

Information about Icon is available on the World Wide Web at

http://www.cs.arizona.edu/icon/www/

The Icon Newsletter

Madge T. Griswold and Ralph E. Griswold
Editors

The Icon Newsletter is published twice 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.

voice: (520) 621-6613

fax: (520) 621-4246

Electronic mail may be sent to:

icon-project@cs.arizona.edu

or

...uunet!arizona!icon-project

THE UNIVERSITY OF ARIZONA ®
TUCSON ARIZONA

and

Bright Forest Publishers
Tucson Arizona

© 1995 by Madge T. Griswold and Ralph E. Griswold
All rights reserved.

Contributing Articles to the *Newsletter*

We welcome articles from readers for inclusion in this *Newsletter*. Such articles should be directly related to some aspect of Icon. Material about applications, programming techniques, and experiences in using Icon are of particular interest to our readers.

Because of the format of the *Newsletter*, articles should not exceed four pages in the *Newsletter* format (approximately 3,200 words), and shorter articles both are preferred and more likely to appear promptly.

We can handle images in any of the commonly used formats but allow for the space they will occupy.

If you would like to contribute an article, please contact us well in advance so that we can advise you on the suitability of what you plan, what our publication schedule is, and when we would need the article.

To avoid potential misunderstandings, our policy on contributed articles follows:

> Contributed articles must be the author's property and free of any restrictions, including copyright. Articles published in the *Newsletter* are covered by its copyright, but authors are free to use their own material elsewhere as they see fit.
>
> There is no remuneration for contributed articles published in the *Newsletter*.
>
> Full credit is given to authors.
>
> Articles are edited as necessary to conform to the standards and style of the *Newsletter*. Drafts of edited articles are provided to authors for review prior to publication.
>
> The editors reserve the right to determine what is suitable for publication in the *Newsletter*.

Graphics Programming Course

In spring semester we taught a new undergraduate course in graphics programming. This course was motivated by the view that graphics should be an integral part of programming, not just an esoteric subject reserved for a few specialists — that graphics should be used in programs for displaying data and communicating with users. We were careful to distinguish this course from traditional computer graphics courses that focus on algorithms and photo-realistic rendering of three-dimensional images.

The course in graphics programming had two primary focuses: using graphics in programming and providing visual interfaces for communication between the user and the program. (We prefer "visual interface" to the more commonly used "graphical user interface" – GUI.)

Students were given copies of the Icon graphics programming book draft and supplementary handouts. Although the book is not designed to be a text, it served well as a reference and the course generally followed the order of presentation in the book.

After a general introduction to windows, the course covered various kinds of drawing, text input and output, handling low-level user events from the keyboard and mouse, color, a more detailed look at windows, and images. The low-level treatment of user events laid the foundation for a higher level of abstraction in which users communicate with a program using visual interfaces that provide buttons, menus, scroll bars, and other interface tools. Considerable attention was paid to the principles of interface design and the cognitive aspects of human-computer interaction.

The course started with 29 students. Icon was, of course, the language used for programming. All students had some experience with Icon from other courses, but most students were not skilled Icon programmers. We spent a couple of weeks reviewing Icon with an emphasis on those features that are needed in graphics programming.

The students ran Icon under UNIX, for which the underlying graphics system is X. The department computer facility for students is a two-processor Sparc 10 (to be replaced by a four-processor Sparc S1000 with a lot more memory than the Sparc 10). Various workstations and X-terminals were available as graphics servers.

The Sparc 10 was overloaded and sometimes unreliable. Network traffic compounded the problem. Several students already had or acquired Linux to run on 386/486 platforms they had at home and only used the department's facilities when they had to turn in their work.

In the early part of the course, students did several homework assignments of modest scope: graphing functions, writing screen savers, producing visually interesting geometric images of their own design, and doing simple interactive animations. There were two midterms that were designed to test student knowledge and help evaluate the course. The main focus of student work was a project in lieu of a final examination.

Each student selected and carried out a project individually. The only requirements for projects were that they have a graphics component and a visual interface.

There were four arcade games (Billiards, Bomber Men, Earth War, and Space War), three datebook / information manager applications, four puzzles (Cube Arrangement, Tangram, Tetris, and Vision), three text editors (one for TEX), two drawing programs, two stock-market programs, a gradebook program, a calculator, a Montessori-style education program, a recipe manager, a visual interface for RCS (a UNIX resource control system), a three-dimensional space navigation program, a multimedia scripting application, and an image browser.

Most of the projects came out well and several were very impressive. The best projects were the Vision puzzle, the multimedia scripting application, the three-dimensional navigation program, Bomber Men, Space War, and the Montessori-style education program. No project failed completely, although a few were short on functionality and robustness. Only two projects had serious problems.

We were impressed by how much the students accomplished with programs of modest size. The average program size was about 1,000 lines (not counting comment lines and blank lines). The largest program was the TEX editor (2,561 lines). The smallest was Billiards (313 lines), which was short on functionality.

The poor performance of departmental computer facilities was a significant handicap for many students, especially for those manipulating three-dimensional objects and doing animation. Almost all students were astonished by the snappy performance of their projects when demonstrated on an unloaded 233 Mhz Alpha workstation with no intervening network. One student even had to modify his program so that it wouldn't run too fast to be demonstrated.

Our impression, confirmed by our standard student course-evaluation forms, was that the course went very well. Many students did more work than was expected, both for homework and the final project — definitely a good sign. The drop rate also was low: Only one student dropped the course, although two others withdrew from school altogether.

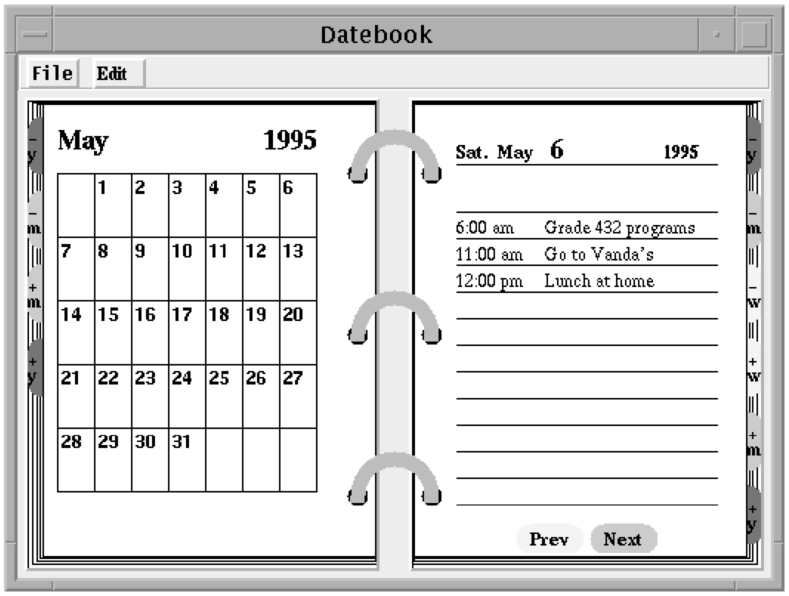
In our view, Icon and its supporting graphics library served well for the course. We were able to cover much more material than if we had used, for example, C or C++. A couple of students felt the course was too "Icon-centric" and wanted more practical training in using graphics in C and C++. A large majority, on the other had, was enthusiastic about Icon and several students already have used it for other applications. Some have used Icon as a prototyping language with the intent of eventually converting to C. Others have written large applications with the full intent of sticking to Icon.

As with all first-time offerings, the course required more effort than teaching an established course. For example, we prepared over 200 transparencies during the semester. In addition, the draft of the Icon graphics programming book was substantially revised and extended during the course, and students got several hundred pages of supplementary handouts. And as problems with software surfaced, we scrambled to fix them and also added new facilities that we had not anticipated would be needed.

Despite the extra effort involved, teaching the course was a gratifying and enjoyable experience. We can do it better and with less effort another time.

We'd like to express our appreciation to Gregg Townsend, who cheerfully provided extensive support for the course — fixing old software, writing new software, providing additional material for the book, answering technical questions, and helping students with problems.

On the following pages, we've included snapshots from some of the projects. We picked images based on visual interest, not project quality, although there was a clear correlation between the two. Unfortunately, we had to leave out some of the best images because they wouldn't reproduce well the way the *Newsletter* is printed. In any case, you'll have to imagine colors.



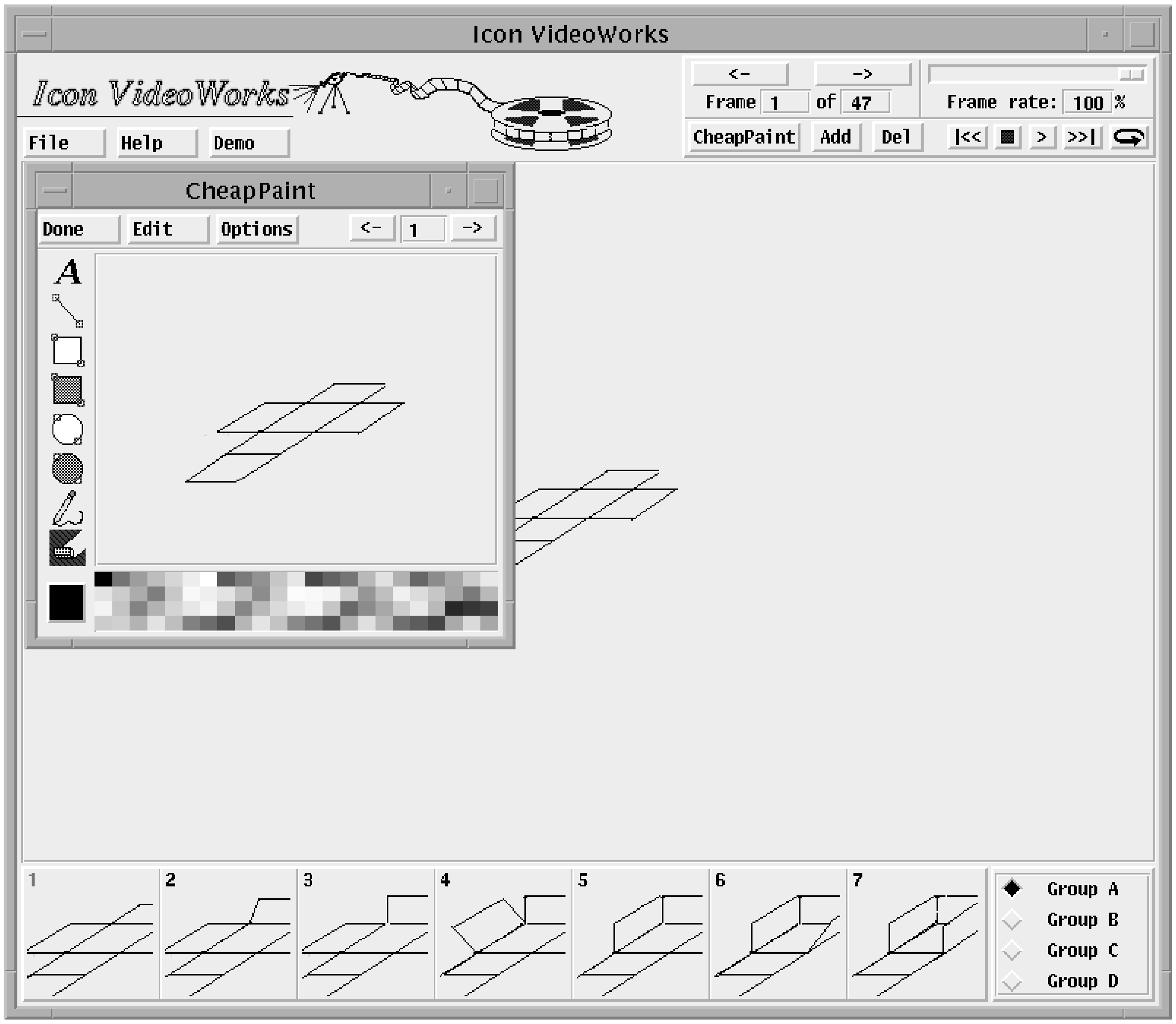
May 1995

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Sat. May 6 1995

6:00 am Grade 432 programs
11:00 am Go to Vanda's
12:00 pm Lunch at home

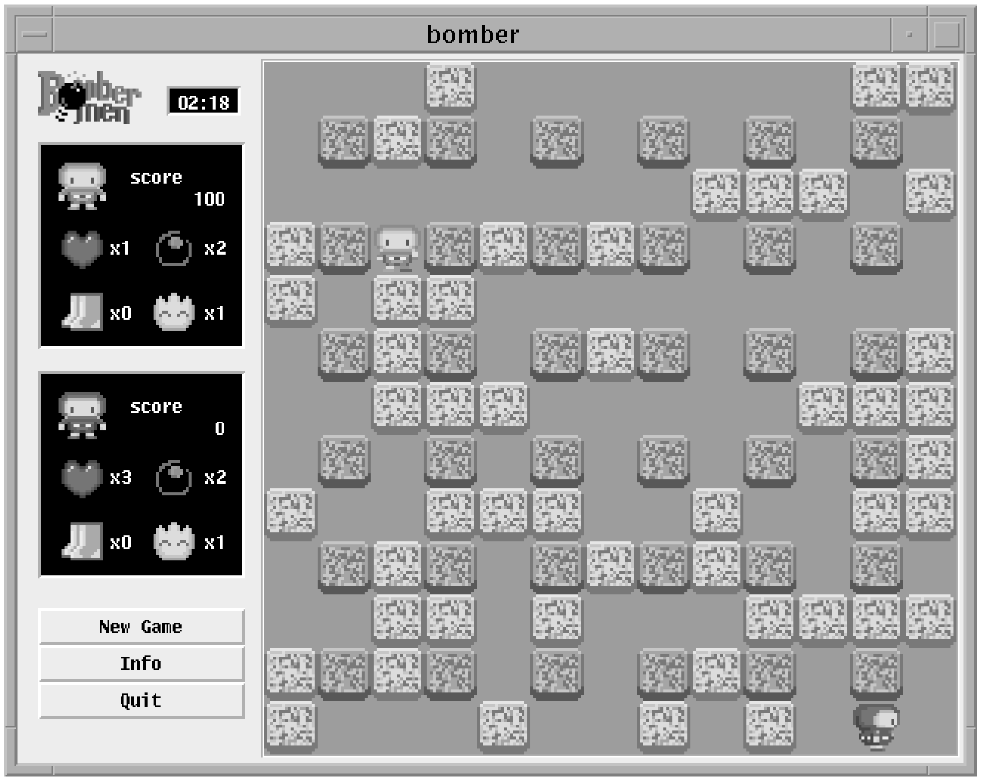
Date and Address Book by James Gundy



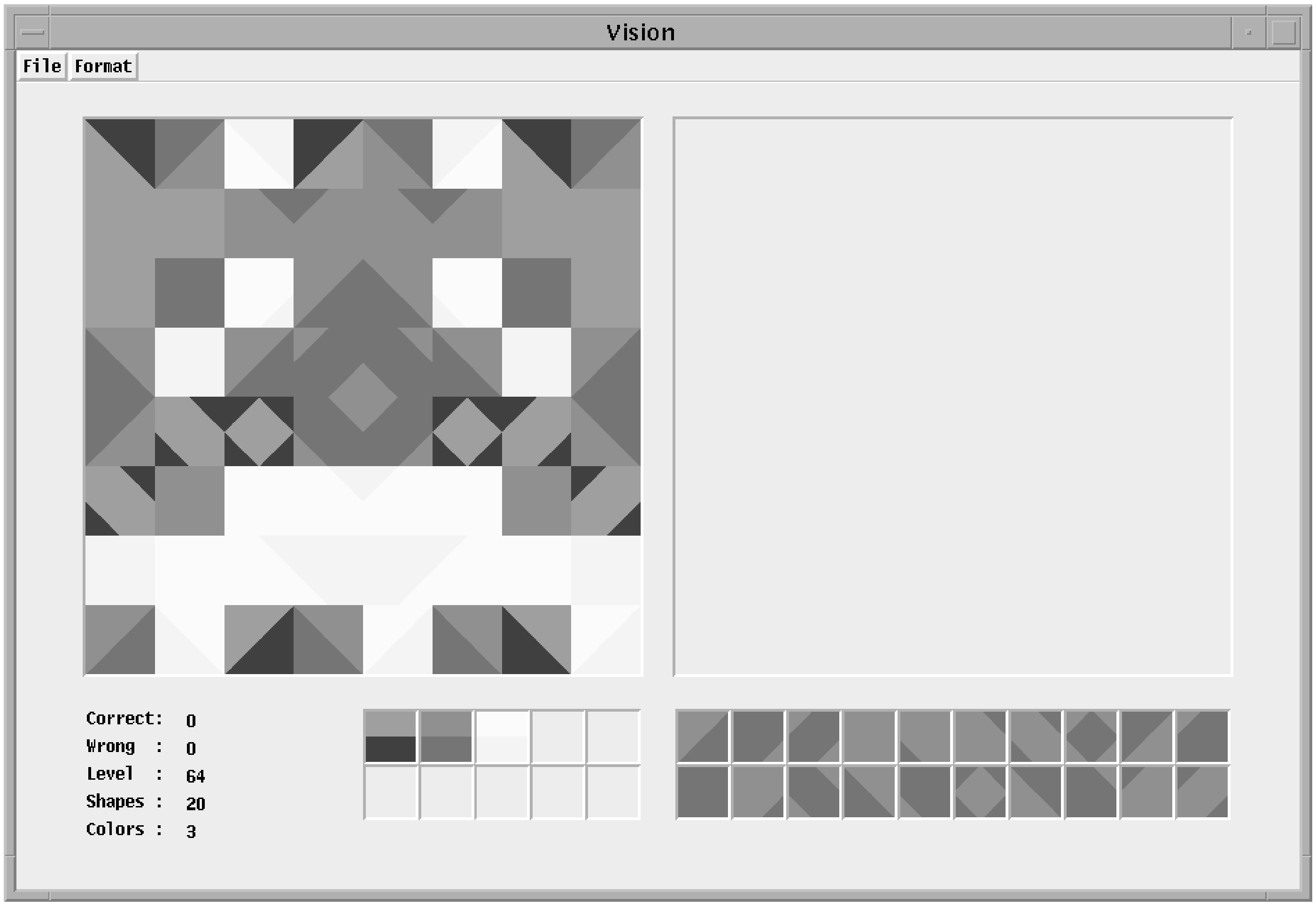
Frame 1 of 47

Frame rate: 100 %

Multimedia Scripter by Brad Traweek

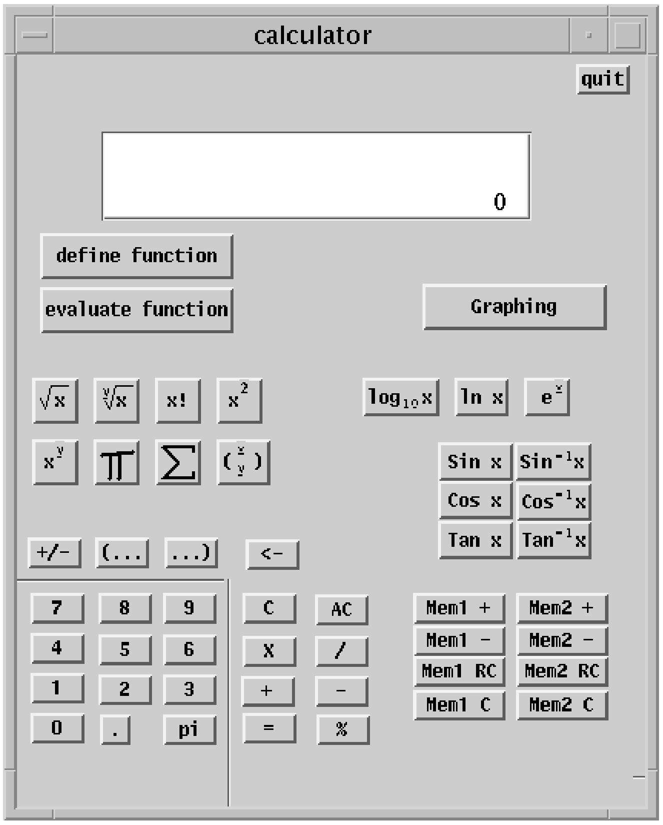


Bomber Men by Kwok-wai Hui



Correct: 0
Wrong : 0
Level : 64
Shapes : 20
Colors : 3

Vision Puzzle by Michael Shipman



Calculator by David Ganote

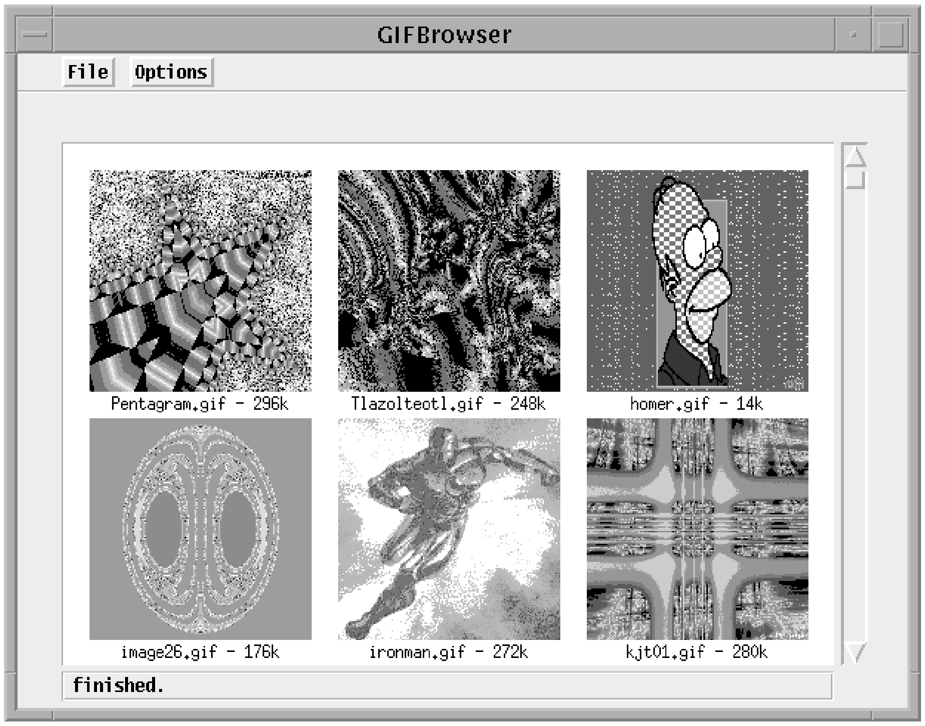


Image Browser by Doug Baerd

Ordering Icon Material

See the latest *Newsletter* for current ordering information.