

# DATA ABSTRACTION

## A BASIC IMPLEMENTATION FOR PROBLEM SOLVING

*Editor's note: For more information and background reading on data abstraction, see the article by Niklaus Wirth in the August issue, "History and Goals of Modula-2."*

PROGRAMMERS, AND INDEED problem solvers in general, have two basic strategies for attacking new problems: decomposition and abstraction. These strategies offer two different ways of solving a complex problem by simplifying it in some way. Decomposition is the strategy embodied by the time-honored Machiavellian dictum to "divide and conquer"—solving large problems by dividing them into simpler, smaller ones that can be solved independently. Abstraction is the strategy of ignoring certain details about the original problem so as to transform it into a simpler and more general one.

For example, consider the problem of computing the sum of the squares of two numbers, 3 and 4 (that is, compute  $3^2 + 4^2$ ). We first simplify the problem by decomposing it into a sequence of three simpler problems: (1)

compute the square of 3, (2) compute the square of 4, and (3) add the two results together. We assume that the final step of addition is sufficiently fundamental that we need not consider it further. However, the first two subproblems can be restated in more simple terms. "Compute the square of 3" means the same as "multiply 3 by 3," and "compute the square of 4" means the same as "multiply 4 by 4."

We may now apply the principle of abstraction to simplify the problem further. We see that there is something essentially the same about computing the square of 3 and computing the square of 4. By abstracting away the particular details of the 3 versus the 4, both subproblems can be solved by a single more general solution, namely, that of computing the square of  $n$ , where  $n$  can represent any number.

### INFORMATION HIDING AND ABSTRACT DATA TYPES

The essential design methodology for data abstraction is known as *information hiding*. The approach was first proposed by D. L. Parnas in 1972 (see reference 4). He proposed that the behavior of software modules be specified completely in terms of their external effects. Such a module hides a secret, namely, the representation of the data object that the module manages. To the outside user, the module provides a set of access functions that are used to create, alter, or observe instances of the abstract data object. There is no way for anyone or anything but the implementation of the module itself to access the ob-

jects, other than through those access functions.

The type of module that Parnas first described has come to be known as an abstract data type. It is abstract because the details of the concrete representation of the data type are unknown to the user. It has also been called an *encapsulated* data type, since the details of implementation are locked away from the user inside a capsule. The functions that access an abstract data type are now commonly referred to as its *operations*.

An abstract data type, therefore, presents itself to the user not as a data structure, but as a collection of procedural abstractions. These are the operations that allow one to create, observe, or alter objects of the abstract type. The task of implementing an abstract data type then consists of determining a concrete representation for objects of the type and writing the procedural abstractions that operate on objects thus represented.

### THE INFORMAL SPECIFICATION OF ABSTRACT DATA TYPES

Two methods for specifying the behavior of abstract data types have emerged. The first is an informal approach that uses prose statements to describe the effect of each of the operations of the data type. The second is a formal approach that uses algebraic statements that are precise and unambiguous. Both approaches are described in turn.

Barbara Liskov gives a complete example of the informal method in her paper "Modular Program Construc-

Gary F. Simons (7500 W. Camp Wisdom Rd., Dallas, TX 75236) is an International Consultant for Linguistics and Academic Computing with the Summer Institute of Linguistics.

tion Using Abstraction." (See reference 3.) In Liskov's informal approach, the specification of a data abstraction has three parts: a header that names the data abstraction and its operations; a brief description of the data abstraction as a whole; and a specification for each of the operations. Each of the operations is a procedural abstraction. The specification of a procedural abstraction may have four parts: a header; a modifies line; a requires line; and an effect line. The header defines how the procedure interfaces with the outside world, that is, its name, the order and types of its inputs and outputs, and the error conditions it signals. The modifies line defines which of the inputs may be modified by the procedure. The requires line defines any assumptions that are made about the calling environment. The effect line describes what the operation is intended to do. Figure 1 gives some sample specifications for procedural abstractions based on Liskov's approach.

We may now illustrate the specification of a data abstraction. Figure 2 gives a sample specification of a data type called *intset* (set of positive integers). Note the three parts of the specification: the header that names the data type and lists its operations; a brief description of the data type as a whole; and the specifications for each of the operations.

### THE FORMAL SPECIFICATION OF DATA ABSTRACTIONS

The formal approach to specifying data abstractions defines an abstract

(continued on page 414)

```
divide (x,y:real) returns real signals divide-by-zero
  effect if y = 0 then signals divide-by-zero
  else returns x/y

sort (x:array[int])
  modifies x
  effect sorts the elements of x in ascending order

search (x:array[int], y:int) returns int signals no-match
  requires x is sorted in ascending order
  effect returns i such that x[i] = y;
  signals no-match if no such i is found
```

Figure 1: Sample specifications of procedural abstractions (after Liskov).

*intset* is create, insert, remove, isempty, ismember

*Intsets* are sets of positive integers; the maximum size of an *intset* is  $2^{16} - 1$  members. *Intsets* are either created empty (using *create*) or made from other *intsets* with a new member added (using *insert*) or removed (using *remove*). *IsEmpty* tests whether an *intset* has any members; *ismember* tests whether a given integer is a member of an *intset*.

```
create ( ) returns intset
  effect returns an empty intset

insert (s:intset, x:int) returns intset signals no-room
  effect if the size of s union {x} is less than or
    equal to the maximum size,
    then returns s union {x},
    else signals no-room

remove (s:intset, x:int) returns intset
  effect returns s minus x

isempty (s:intset) returns Boolean
  effect returns true if s has no members

ismember (s:intset, x:int) returns Boolean
  effect returns true if x is a member of s
```

Figure 2: Sample specifications of data abstraction (after Liskov).

(continued from page 131)

type in terms of an algebra. In the general theory of algebras, an algebra is a pair  $\langle A, F \rangle$ , where  $A$  is a non-empty set and  $F$  is a family of operations on  $A$ . For instance, the familiar algebra of grade school mathematics is defined by the set of real numbers and the operations of addition, subtraction, and so on. Since an abstract data type consists of a set of objects that carry the type and the operations on those objects, it is easy to see how abstract data types lend themselves to definition in terms of an algebra.

The meaning (or the effect) of the operations is defined as a set of formal axioms that state the relationships among the operations. The reduction of the operations' meanings to a set of axioms makes it possible

to reason formally about the correctness of a design before it is implemented. This is one of the productive ways of using this approach in program design.

The algebraic approach to specifying abstract data types is rigorously defined by John Guttag and J. J. Horning (see reference 1) and consists of two parts: a syntactic specification and a semantic specification. The syntactic specification defines how the type interfaces with the outside world; it defines the name of the type, the names of all its operations, and the types of the domains (inputs) and ranges (outputs) of the operations. Figure 3 illustrates the syntactic specification for the abstract type *intset*.

The operations on any data type fall into two classes: generator operations

and inquiry operations. The generators are those operations that produce an object of the type of interest (for example, *intset*). The inquiry operations focus on objects of interest but produce a result that is of a different type (for example, Boolean). The blank line in the example of figure 3 separates the two classes of operators. Within the set of generators there is a subset called basic generators that are sufficient to generate any object of the type of interest. The basic generators, *create* and *insert*, are marked in figure 3 with a preceding asterisk.

The semantic specification of the operations consists of a set of axioms that define the meaning of the operations by stating their relationships to one another. The axioms are presented as equations in which the left-hand side specifies an expression to be defined and the right-hand side gives its meaning. For the basic generators, no definitions are written; they are assumed as given. Thus we first write axioms that define the meaning of the nonbasic generators (for example, *remove*); the right-hand sides of these equations must eventually be reduced to expressions involving only basic generators. Then

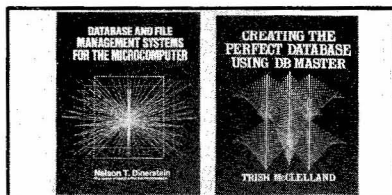
(continued)

Abstract data type: <i>Intset</i>			
Operations:			
* <i>create</i> :		-->	<i>intset</i>
* <i>insert</i> :	<i>intset</i> x <i>int</i>	-->	<i>intset</i>
<i>remove</i> :	<i>intset</i> x <i>int</i>	-->	<i>intset</i>
<i>isempty</i> :	<i>intset</i>	-->	Boolean
<i>ismember</i> :	<i>intset</i> x <i>int</i>	-->	Boolean

Figure 3: Syntactic specification for abstract type *intset*.

## Get Productive!

### With these new how-to-do-it books



#### Apple to IBM PC Conversion Guide

by Richard Steck. The first book to show Apple users how to convert Apple programs and peripherals to IBM PC use. 18047, \$11.95

#### Database and File Management Systems for the Microcomputer

by Nelson T. Dinerstein. A timely, clear introduction to database management from the best-selling author of *dBASE II for the Programmer*. 18088, \$15.95

#### Creating the Perfect Database Using DB MASTER

by Trish McClelland. Outlines a tested process you can use to create a database using DB MASTER on your Apple or IBM PC. 18039, \$17.95

#### The ABCs of Lotus 1-2-3

by Bill Kling. This step-by-step tutorial for beginners helps you put 1-2-3 to work immediately in your business. 15996, \$18.95

To order, contact your local bookstore or computer store, or contact

**Scott, Foresman and Company**  
Professional Publishing Group,  
Dept. BY-2  
1900 East Lake Avenue  
Glenview, IL 60025  
312/729-3000, x2208

For Canadian orders,  
please contact  
Gage Publishing Company  
164 Commander Blvd.  
Agincourt, Ontario M1S 3C7



# PREVENT THE DISASTER OF HEAD CRASH AND DROPOUT.

The war against dust and dirt never ends. So before you boot-up your equipment, and everytime you replace a cassette, disk or drive filter, be sure to use Dust-Off II; it counteracts dust, grit and lint. Otherwise you're flirting with costly dropouts, head crashes and downtime.

Dust-Off II is most effective when used with Stat-Off II. Stat-Off II neutralizes dust-holding static electricity while Dust-Off II blasts loose dust away. There's also the Dual Extender and Mini-Vac for vacuuming dust out of hard-to-reach places.

Photographic professionals have used Dust-Off brand products consistently on their delicate lenses and expensive cameras for over ten years. They know it's the safe, dry, efficient way to contaminant-free cleaning.



Cleaning not provided by liquid cleaners.

Dust-Off II's remarkable pinpoint accuracy zeros in on the precise area being dusted. And you have total control—everything from a gentle breeze for



Stat-Off II neutralizes dust-holding static electricity from media and machines.

delicate computer mechanisms to a heavy blast for grimy dirt.

Don't let contamination disrupt your computer operation.

Stock up on Dust-Off II—the advanced dry cleaning system, at your local computer or office supply dealer.

Or send \$1.00 (for postage and handling) for a 3 oz. trial size and literature today.



## Dust-Off II

The safe dry cleaning system

Falcon Safety Products, Inc., 1065 Bristol Road, Mountainside, NJ 07092

## DATA ABSTRACTION

we provide axioms that define the meaning of applying each inquiry operation to each of the basic generators. Note that since the meaning of any nonbasic generator can be expressed in terms of basic generators alone, there is no need to write axioms defining the application of inquiry operations to nonbasic generators. Figure 4 gives a sample semantic specification for the type intset. The equal sign used in the axioms is to be read as "means the same thing as."

Note that there are three sets of axioms, one for each nonbasic generator and inquiry operation that must be defined. Each is defined in terms of two axioms, one for each of the two basic generators of the type of interest.

The definition of isempty is easy to understand. If an intset, *s*, were generated by create, then the set has no members yet, and the expression empty(*s*) means the same thing as T, or true. If the intset was generated by insert, then the set must have members, and empty means the same thing as F, or false.

Remove and ismember use a basic generator of type Boolean named equal. It tests two integers for equality. In conjunction with this Boolean generator, they use the if-then-else operation that is defined by the following two axioms:

if T then *a* else *b* = *a*  
if F then *a* else *b* = *b*

That is, "if T then *a* else *b*" means the same thing as (or, can be reduced to) *a*, and "if F then *a* else *b*" means simply *b*. In other words, if the Boolean condition reduces to true then the whole expression reduces to the *then* clause, otherwise to the *else* clause.

Both remove and ismember are defined recursively, that is, in terms of themselves. For instance, the second axiom for ismember says that if the integer *i* is not equal to the first item in the intset generated by "insert (*i*,*s*)", then the expression "ismember(*i*, insert(*i*,*s*))" means the same thing as "ismember(*i*,*s*)". We then apply the

(continued)

Let  
inc  
yo  
New N  
for a r  
newes

The wave o  
everything fro  
By 1990, over 1  
Over 25,000

Keeping thi  
understand adv  
estimates call f  
that pay \$25,00  
salaries have n  
Build Your C

Now, you c  
industrial contr  
experience in t  
the most fascin

You need n  
beginning, tak  
right on throug  
controls, serv  
optoelectronica  
give you a prio  
Program An

Designed e  
ments of indust  
troubleshoot us  
job training at

Building th  
art into the ne  
You'll learn

You get and keep  
speech synthesis  
experimentation  
digit LCD reader

# APPLEWARE, INC.

## The Apple Users Group Software Library

For the first time enjoy your Apple to its fullest capacity, using specially packed disks with over 60 outstanding programs each.

(Not available from any other source)

Each packed disk includes an extensive variety of interesting, useful and entertaining programs indispensable to all computerists!

Mixed category packed disks include:

BUSINESS • EDUCATIONAL • DATA BASE • GAMES • UTILITIES • SCIENCE • MUSIC • GRAPHICS • FINANCE  
Library Disks I, II and III are mixed categories. Single category disks are: GAMES • UTILITIES • GRAPHICS • INTEGER • SCIENCE • TECH • MUSIC & AUDIO

Individual disks available at \$59.95 each

Order direct from this ad and Save up to \$150. Buy Library Disks I, II and III and get a special bonus disk FREE - over 260 programs for \$179.95 + \$4 shipping. BUT for the Best Value, receive any 3 disks featuring over 600 of our best programs for only \$59.95 each for a package price of \$389.95. Certified Package plus handling post!

\*Send one-time membership fee of \$15. (no fee charged to institutions) for 1000 + program catalog and gain access to a library of over 10,000 programs at a special 15% discount

(Foreign memberships \$28. U.S.)

For Orders Only Call now

TOLL FREE: 1-800-327-9884

Florida: 1-305-887-8885

Or Write:

Appleware, Inc.

6400 Hayes Street

Hollywood, Fla. 33024

Program Disks compatible with Apple II, II+, IIe, III Emul., Franklin Ace and IBM Quad

PROGRAMS  
65¢  
EACH



Circle 23 on inquiry card.

## C SOFTWARE DEVELOPMENT PCDOS/MSDOS

- FULL C COMPILER PER K&R
  - Inline 8087 or Assembler Floating Point
  - Full 1MB Addressing for Code or Data
  - Transcendental Functions
  - MSDOS 1.1/2.0 LIBRARY SUPPORT
  - Program Chaining using Exec
  - Environment Available to Main
  - c-window™ C SOURCE CODE DEBUGGER
  - Variable Display & Alteration Using C Expression
  - FAST 8088/8086 ASSEMBLER
- Combined Package — \$199**
- Call or write:
- c-systems** Fullerton, CA 92634  
P.O. Box 3253 714-637-5362
- TM c-systems

**BASF**  
FlexyDisks®



5 1/4"

Specify soft, 10 or 16 sector Price 10-90 Price 100 +

Single side double density 1.60 ea. 1.45 ea.

Double side double density 1.85 ea. 1.70 ea.

Hard sectors in Library box only add .15.  
Certified Check - Money Order - Personal Check. Allow up to 2 weeks for personal checks to clear. Add \$3.00 per 100 or part to each order for U.S. shipping charges.  
NJ Residents add 6% sales tax.

**DATA**  
EXCHANGE, INC.

178 Route 206 South, P.O. Box 993  
Somerville, N.J. 08876 • (201) 874-5050

Circle 91 on inquiry card.

## DATA ABSTRACTION

same axioms to this simplified expression to reduce it ultimately to T or F. The recursive application of the axioms is easily illustrated. For instance, imagine an intset of three members: {1,2,3}. In the language of our axioms, this set would be represented by the expression:

insert(1,insert(2,insert(3,create)))

We now want to test if 4 is a member of the set. This test is equivalent to the expression:

ismember(4,insert(1,insert(2,insert(3,create))))

To discover what this expression means, we simplify it by applying the

axioms repeatedly until it can be reduced no further. Figure 5a illustrates this process; we see that the expression ultimately reduces to F, or false. Similarly, we can test if 2 is a member of the set. This is done in figure 5b where we see that the result is T, or true.

These illustrations demonstrate the potential of the method for giving us a rigorous way to reason about the meaning and ultimately the correctness of program designs. The sequences of derivations in figure 5 are actually proofs that "ismember(4, {1,2,3})" means false and that "ismember(2, {1,2,3})" means true. In

(continued)

### Axioms:

For all i, i' of type Int, and all s of type Intset:

remove(i,create) = create  
remove(i,insert(i',s)) = if equal(i,i') then s else insert(i',remove(i,s))

isempty(create) = T  
isempty(insert(i,s)) = F

ismember(i,create) = F  
ismember(i,insert(i',s)) = if equal(i,i') then T else ismember(i,s)

Figure 4: Semantic specification for abstract type intset.

(a) Is 4 a member of the set {1,2,3}?

ismember(4,insert(1,insert(2,insert(3,create)))) =  
if equal(4,1) then T else ismember(4,insert(2,insert(3,create))) =  
if F then T else ismember(4,insert(2,insert(3,create))) =  
ismember(4,insert(2,insert(3,create))) =  
if equal(4,2) then T else ismember(4,insert(3,create)) =  
if F then T else ismember(4,insert(3,create)) =  
ismember(4,insert(3,create)) =  
if equal(4,3) then T else ismember(4,create) =  
if F then T else ismember(4,create) =  
ismember(4,create) =  
F

(b) Is 2 a member of the set {1,2,3}?

ismember(2,insert(1,insert(2,insert(3,create)))) =  
if equal(2,1) then T else ismember(2,insert(2,insert(3,create))) =  
if F then T else ismember(2,insert(2,insert(3,create))) =  
ismember(2,insert(2,insert(3,create))) =  
if equal(2,2) then T else ismember(2,insert(3,create)) =  
if T then T else ismember(2,insert(3,create)) =  
T

Figure 5: Reducing expressions by axiomatic substitution.

Gifford has the simple, fast, secure works. Multiuser based on Digital DOS, the only m operating system designed for net. Users can sl printers transpar also take advanta tiuser features lil record lockout. A has added a buna that makes Multi DOS easy to inst get right to work.

**Our net is Al**  
Multiuser Concv DOS utilizes Da ARCNET, the m lar network hard the industry. It's economical, and you can add user without overload ing the network.

You can net work up to 255 single and multi user systems. You can connect sing multiuser Gifford systems as well a processor Gifford systems can run CP/M or MP/M can run 16 bit C programs as wel MS-DOS applic Lotus 1-2-3.

**Gifford a net**

Our enhance DOS make it po better work don wide features in event calendar, communication accounting and generation, tele user expandabl

Circle 142 on inq



a formal proof, we would also need to list with each reduction a reference to the axiom used.

These examples should make it clear that the axioms are expressed rigorously enough that a computer could help us in the tedious work of reducing expressions as we reason and test our designs. In fact, the ax-

ioms could be implemented directly on a substitution (or reduction) machine as a way of testing the design. Cristoph Hoffman and Michael O'Donnell describe just such a system in their article, "Programming with Equations" (see reference 2). Certainly the specification of data abstractions, whether by formal or informal means,

is a powerful new tool for software designers.

## IMPLEMENTING ABSTRACT DATA TYPES IN BASIC

We now turn our attention to the practical application of data abstraction by implementing two versions of the intset abstraction in BASIC. BASIC is by no means an ideal language for implementing data abstractions. However, in as much as it is the lingua franca of personal computing, there is perhaps no better way of demonstrating the principles of data abstraction to a general audience. Not only should this exercise illustrate the benefits of the data-abstraction technique, it should also demonstrate that with discipline, a programmer can produce good code in BASIC that is maintainable and portable.

BASIC lacks two things that data abstraction needs: parameterized procedure calls (for invoking the operations) and limited scoping of variables (to support information hiding). Neither shortcoming is insurmountable; both are solved by requiring a more careful use of variables. Unfortunately, this puts a greater responsibility on the programmer than would be required by more modern languages and consequently increases the opportunity for programming errors.

The variables used in a BASIC implementation of an abstract data type fall into three categories: variables used to pass parameters to, and return values from, the operations; variables used in the concrete representation of the abstract type; and variables used locally in the implementation of the operations. Unfortunately, all variables in BASIC have a global scope and so nothing can be used without reference to how it is used elsewhere in the program. The first category of variables defines the user's interface with the operations of the data type. These must be known and understood in order to use the data type. The second and third categories need not be understood by the user, but their names must be known.

(continued)

**What do you get when you cross 1200 baud, free on-line time, and extra features at a price Hayes can't match?**

### Data Rate?

The MultiModem gives you a choice—either 1200 or 300 bits per second. So you can go on-line with the information utilities. Check out bulletin boards. Dial into corporate mainframes. Swap files with friends.

### On-Line Time?

With the MultiModem you get CompuServe's DemoPak, a free two-hour demonstration of their service, and up to seven more free hours if you subscribe. You also get a \$50 credit towards NewsNet's business newsletter service.

### Features & Price?

Of course, the MultiModem gives you automatic dial, answer, and disconnect. Gives you the Hayes-compatibility you need to support popular communications software programs like Crosstalk, Data Capture, our own MultiCom PC, and dozens of others. Gives you a two-year warranty, tops in the industry.

### But Better?

Yes. The MultiModem gives you features the Hayes Smartmodem 1200™ can't match. Features like dial-tone and busy-signal detection for more accurate dialing and redialing. Like a battery-backed memory for six phone numbers. All at a retail price of just \$549—compared to \$699 for the Smartmodem.

What do you get? The new MultiModem, from Multi-Tech Systems. Isn't this the answer you've been looking for?

For the name of your local distributor, write **Multi-Tech Systems, Inc.**, 82 Second Avenue S.E., New Brighton, MN 55112. Or call us at (612) 631-3550.

**MultiModem.**



**MultiTech**  
Systems

**The right answer every time.**

Trademarks—MultiModem, MultiCom PC, Multi-Tech Systems, Inc.—CompuServe, CompuServe Information Services, an H & R Block company—NewsNet, NewsNet, Inc.—Crosstalk, Microstul, Inc.—Data Capture, Southeastern Software—Smartmodem, Hayes Microcomputer Products, Inc.

Circle 245 on inquiry card.

Circle 292 on inquiry card. →

the programmer must know what variables are used in the implementation of the data type, in order to avoid abusing the type by inadvertently using one of these variables for a different purpose elsewhere.

In a pure information-hiding environment this would not be the case. When a data abstraction truly hides the concrete representation of a type, then the way one type is represented cannot interfere with how another is. But in BASIC this does not come automatically since all variables have a global scope and the representation of one type can therefore interfere with another if the variables are not unique. The BASIC programmer must therefore be content with a weaker form of data abstraction that hides the meaning and use of the data structures that implement an abstract type but cannot hide their names.

After an abstract data type has been

specified, as discussed in the preceding sections, there are three steps in its implementation: (1) define the user interface, (2) define the concrete representation, and (3) implement the operations. Each of these steps will be considered in turn.

## DEFINING THE USER INTERFACE

The user interface refers to what a person must know in order to invoke the operations of the data type. For each operation, the following items must be defined: the line number where the subroutine begins, the variables in which input parameters are passed to the operation, the variables in which values are returned, and the variables in which exception codes are signaled. All of this information can be conveyed in a one-line REM (remark) statement that serves as a header for the subroutine. (The

version of BASIC used in the listing below allows ' as shorthand for REM.)

Figure 6 illustrates a possible definition of the user interface for the abstract type intset. Note, for instance, the header for the operation "insert." The address of the subroutine is identified as line 1200. The input parameters (a set identifier and an integer to insert into the set) are passed in the variables S and X respectively. The operation returns s as the set identifier of the resulting set. The operation may signal one of three exception codes; this is done with three variables that return a Boolean value. If one of these variables has a value of true when the subroutine returns, then that condition has occurred. Thus the insert operation may signal "not a valid set" in NS, "not a valid integer" in NI, or "no more room" in NR.

(continued)

# Meet The Controllers.



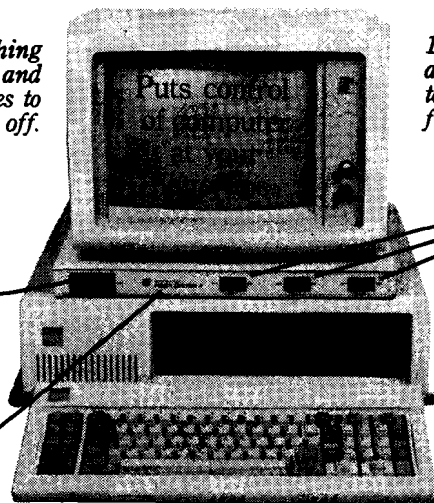
Power Control™ protects computer circuitry and data stored in memory against the damage voltage spikes can cause.

Puts on/off control of your computer, terminal, printer, and more at your fingertips in a slim panel unit sized to fit underneath your computer terminal.

**Control Power, Peripherals, Spikes, and Glitches.**

Contains a master switch (to turn your computer, terminal, printer, a modem or a lamp on or off at the same time) and three additional switches to turn peripherals on or off in any order.

Eliminates reaching over, behind and around devices to turn them on or off.



16" width, 10" depth allows placement under terminal for fingertip control.

Additional switches give individual control over peripherals.

Organizes power wires. 4 cords in- 1 cord out.

Master switch turns computer and all peripherals on or off at same time.

Less than 2" high.

Relax Technology. The company that works so you can relax and get down to business.

## Relax Technology.

To order, phone: 415/471-6112 or mail to: 3101 Whipple Rd., #25, Union City, CA 94587  
\*Calif. Residents add applicable sales tax.  
Prices include shipping.

- ☐ Power Control 1: \$69.95\*
- ☐ Power Control 2: \$89.95\*  
10 amp circuit breaker. RFI noise filtering. IEC power connector.
- ☐ Power Control 3: \$129.95\*  
Cross suppression between all 4 outlets. Illuminated switches. 3-stage RFI filter.

☐ Check for \$\_\_\_\_\_ enclosed.

☐ VISA ☐ MasterCard

Card # \_\_\_\_\_

Exp. Date \_\_\_\_\_ Bank # \_\_\_\_\_

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ St. \_\_\_\_\_ Zip \_\_\_\_\_

Signature \_\_\_\_\_

Note also that figure 6 defines four new operations on type intset: display, kill, intersect, and union. Display prints the contents of a set on the screen. Kill deletes an existing set.

Intersect produces a new set that is the intersection of two existing sets. Union produces a new set that is the combination of two existing sets. These operations provide a fuller en-

vironment for testing our implementations of the intset abstraction.

### USING THE ABSTRACT DATA TYPE

With the design of the user interface in hand, we know enough to write a program that uses the type. Figure 7 gives the overall design for a program to test an implementation of intset. (It overlooks the details of what to do when an exception is signaled.) The test program repeatedly takes a one-letter command and two numerical arguments, executes the named operation, and displays the resulting set.

See listing 1 for the test program. As an example of how to use an operation, consider the use of insert in lines 410 and 420. The operation is invoked by GOSUB 1200, but before calling the subroutine we must pass the parameters. The interface requires

(continued)

```

1000 'abstract data type: INTSET
1100 'create() returns(s) signals(os)
1200 'insert(s,x) returns(s) signals(ns,ni,nr)
1300 'remove(s,x) returns(s) signals(ns,ni)
1400 'isempty(s) returns(b) signals (ns)
1500 'ismember(s,x) returns(b) signals(ns,ni)
1600 'display(s) signals(ns)
1700 'kill(s) signals(ns)
1800 'intersect(s1,s2) returns(s) signals(ns,os)
1900 'union(s1,s2) returns(s) signals (ns,os,nr)

```

Key to exception codes:

ni = not a valid integer  
nr = no more room  
ns = not a valid set  
os = out of sets

Figure 6: User interface for abstract data type intset.

## BYTE CONNECTION INC.

presents MONTHLY PRICE BUSTERS!!!

CALL (714) 778-6496

FALL SPECIALS AT AFFORDABLE PRICES

#### PERSONAL COMPUTERS

IBM PC (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$2499.00

IBM PC (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$2999.00

IBM XT (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$4195.00

ALTOS 856 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$7595.00

ALTOS 586 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$9595.00

COMPAQ PC (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$2895.00

EPSON QX 10 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1995.00

CORONA 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1895.00

ME MOTECH 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$495.00

#### PRINTERS

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$325.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$259.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$449.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$659.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$489.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$299.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$399.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$775.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$629.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$815.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1559.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1995.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1695.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$1695.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
Call for printer prices not listed above

#### MODEMS

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$169.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$369.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
\$369.00

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

EPSON 1000 (think of it as a desktop calculator and monitor)  
1600 1600 Kbytes, 1600 Kbytes, 1600 Kbytes, 1600 Kbytes  
CALL

TEK

mu

100 MHz  
time base  
3.5 ns rise  
sweeps  
to 5 ns/div  
vertical  
accuracy  
2 mV/div  
90 MHz

Now  
faster  
accurate  
The Te  
meter me  
time, f  
voltage  
button  
current  
readout

Its c  
includ  
plus p  
line, T  
normal



## DATA ABSTRACTION

that the set identifier be passed in S and the integer to insert in X, thus the code: "S=P1: X=P2". In line 420, the statement GOSUB 1600 calls display. This too requires a set identifier in S but since insert returns with the current set in S, there is no need for an

assignment statement to pass the parameter. As long as S was not an invalid set identifier, that is, as long as NS=False, we display the resulting contents of set S. It is assumed that all exception conditions will generate an error message when they are dis-

covered by intset. If NI or NR occurs, a message will be given, but we still want to display the current status of S. The use of the other operations follows the same principles.

It is important to note that we were able to write a program using intset before we decided how the type is to be represented or implemented. This is the power of information hiding and data abstraction at work. As long as we stick to the interface definition in figure 6 when we implement the data type, the test program will work.

There is, however, one slight complication brought on by the global scoping of all variables in BASIC. Our test program is guaranteed to work only if we avoid variable conflicts. The test program happens to use three variables of its own that are not part of the intset interface. These are C\$, P1, and P2. The test program is

(continued)

```

PROCEDURE test intset abstraction:
  Initialize abstract data type Intset
  REPEAT until commanded to quit,
    Get a command and two parameters (p1 and p2)
    CASE of command,
      c: display(create())
      d: display(p1)
      e: isempty(p1)
      i: display(insert(p1,p2))
      k: kill(p1)
      m: ismember(p1,p2)
      q: quit program
      r: display(remove(p1,p2))
      u: display(union(p1,p2))
      x: display(intersect(p1,p2))
  
```

Figure 7: Design for program to test an implementation of intset.

Come visit us in our  
New York City Showroom  
226 Sherwood Ave.  
Farmingdale, NY 11735

# Computer Channel

Se Habla Español  
Cable: COMSYSTEC NEWYORK  
Telex: CSTNY 429418

**OUR SPECIALTY: IBM COMPATIBLE PRODUCTS, GRAPHICS, DATABASE, 68000 UNIX, EXPORT**

**IBM PC & COMPATIBLES**

Columbia, Corona, Zenith,  
Leading Edge, Televideo, Sanyo,  
Tava, & IBM PC

**OTHER POPULAR  
COMPUTERS**

Epson, Cromemco, NEC PC,  
Altos, North Star, Dual 68000,  
DEC Rainbow, OSM

**AN AFFORDABLE  
CAD SYSTEM  
FOR ENGINEERS  
& DESIGNERS**

AutoCAD™ is a two-dimensional computer-aided drafting and design system suitable for many applications including drawings for architectural, mechanical, electrical, PCB layout, chemical, structural, and civil engineering.  
For the configuration as shown in the above flowchart,

SPECIAL  
CALL FOR \$5,800.00

(cables included)  
Package with 10 MB hard disk also available  
\*\*\*\*\* CALL FOR DETAILS \*\*\*\*\*

**PRINTERS**

**EPSON, OKIDATA  
full line**

Prism 132	200 cps, 132 col.	1,100
Toshiba P1340	80 col., 160 cps	799
Microprism	110 cps, 80 col. graphic	379
Dataproduct	8000 series	CALL
	8010 180 cps, graphic	545

\*\*\*Letter Quality\*\*\*

NEC 2050	20 cps for IBM PC	840
3550	35 cps for IBM PC	1,610
Citoh F-10	40 cps	999
Juki 6100	18 cps	459
Qume 11/40	w/IBM interface	1,420
Star Power Type	18 cps	399
Diablo 630	ECS/IBM ext. char. set	2,100
Dynax HR35	33 cps	910
Comrex Comwriter III		740
Transtar 315	graphic, color	479

**PLOTTERS/DIGITIZERS**

Amdek	6-pen X-Y Plotter	895
Houston Instrument	DMP-29 8-pen X-Y Plotter	1,795
	DMP-40 1 pen plotter	795
	DMP41, DMP42 22x34", 24x36" plotter	CALL
	DMP-51, DMP-52 22x34", 24x36" plotter	CALL
	HIPAD DT-11AA Digitizer	725
	HIPAD DT-114 4-button digitizer	CALL
Hewlett Packard	7470A 2-pen plotter	940
	7475A 6-pen plotter	1,640
Calcomp M84	8-pen plotter	1,650

**POWERFUL ADD-ON BOARDS**

from AST, PERSYST, PLANTRONIC, TECMAR,  
QUADRAM, HERCULES, TITAN

MORE FOR YOUR IBM PC

**MODEMS**

HAYES	Smartmodem 300/1200 bps	499
	1200B modem for IBM PC	CALL
USR	300/1200 bps w/64K, parallel port	550
	Password 300/1200 bps modem	339
NOVATION	Smartmodem 300/1200 bps modem	415
	PC Cat 300/1200 bps modem	450

**MONITORS**

(TERMINALS: HAZELTINE, ZENITH, WYSE, VISUAL, CALL)

Panasonic amber super	199
Comrex CR6800 14" RGB	489
NEC JC1216 RGB monitor, 640x300 resolution	435
IB1201 20 Mhz green monitor	185
Princeton Graphic HX12 RGB monitor	490
SR12 RGB	630
Amdek 300 12" green monitor	155
Color IV Xtra	710
Zenith ZVM 123 Green Monitor	87
ZVM 122 Amber monitor	135
ZVM 135 RGB monitor for IBM PC	475

Prices subject to change. American Express, Visa/Mastercard add 3%. F.O.B. point of shipment. 20% restocking fee for returned merchandise. Personal checks take 3 weeks to clear. COD on certified check only. N.Y. residents add sales tax. Manufacturers' warranty only. International customers, please confirm price before order. Accept P.O. from Fortune 500, schools and gov't.

**Computer Channel**  
226 Sherwood Ave.  
Farmingdale, NY 11735  
For information CALL (516) 420-0142  
To order CALL 1-800-331-3343

**TELEX:**  
429418  
CSTNY

guaranteed, therefore, only if the implementation of `intset` stays away from these three variable names. Conversely, if we begin with an already implemented data type and want to write a program that uses it, that program may not use any variable as a free variable that is used in the implementation of the data type.

### DEFINING THE CONCRETE REPRESENTATION

The second step in implementing an abstract data type is to define the concrete representation. A straightforward representation for integer sets is to store them in a matrix where each row represents a set and the columns hold individual set elements. A value of `-1` means that the matrix cell is empty; a positive integer is a set element. A value of `-1` in column 0 means that the whole row is unused.

Note that in figure 6, the first oper-

ation is not coded until line 1100. The lines between 1000 and 1100 are reserved for comments that describe the concrete representation, followed by an initialization subroutine that sets up the storage space for the data type as required by the method of representation. This subroutine is the first thing called by the test program of listing 1.

Listing 2 gives a full implementation of `intset` with an underlying matrix representation of 11 sets (in rows 0 to 10) with 10 elements each (in columns 1 to 10). Note lines 1000 to 1040, which are comments describing the method of representation, and lines 1050 to 1080, which define an initialization routine that sets up the storage space for `intsets`.

### IMPLEMENTING THE OPERATIONS

Now we are ready for the third step, implementing the operations. The

headers defined for the user interface (see figure 6) serve as the first lines for the subroutines that implement each of the operations. Given the representation of the data type and the variables specified in the header for parameter, result, and exception code passing, the implementation of the operations falls into place. See listing 2 for the complete implementation of the operations. Note that the implementation makes use of two private subroutines (at lines 2000 and 2100) for checking the validity of parameters `s` and `x`. These do not appear in the list of operations of the data type (figure 6) because they are meant to be used only from within the module, not by outside users.

A new complication presents itself when implementing the operations. That is the problem of local variables. One must ensure that the extra vari-

(continued)

## FREE SHIPPING DISKETTES

West Coast "Call"  
1(800) 621-6221

Central & East "Call"  
1(800) 654-4058

Discounts Starting at 3 Box Quantities

3M	Dysan	maxell	Verbatim
• 5 1/4" •	• 5 1/4" •	3 1/2" CALL	• 5 1/4" Datalife
s-side 17 <sup>95</sup>	s-side 22 <sup>95</sup>	• 5 1/4" •	s-side 18 <sup>95</sup>
d-den. 23 <sup>95</sup>	d-den. 30 <sup>50</sup>	s-side 19 <sup>95</sup>	d-den. 24 <sup>95</sup>
d-den. 27 <sup>50</sup>	d-den. 34 <sup>50</sup>	d-side 25 <sup>95</sup>	d-side 24 <sup>95</sup>
s-side quad 33 <sup>95</sup>	s-side quad 45 <sup>50</sup>	d-side 28 <sup>95</sup>	s-side quad 30 <sup>95</sup>
d-side quad 33 <sup>95</sup>	d-side quad 45 <sup>50</sup>	s-side quad 36 <sup>95</sup>	d-side quad 39 <sup>95</sup>
• 8" •	• 8" •	d-side quad 36 <sup>95</sup>	• 8" Datalife
s-side 21 <sup>50</sup>	s-side 28 <sup>50</sup>	• 8" •	s-side 24 <sup>75</sup>
s-den. 26 <sup>00</sup>	s-den. 30 <sup>95</sup>	s-side 31 <sup>95</sup>	s-den. 26 <sup>95</sup>
s-side 26 <sup>00</sup>	d-den. 30 <sup>95</sup>	d-den. 31 <sup>95</sup>	d-den. 26 <sup>95</sup>
d-side 31 <sup>50</sup>	d-side 34 <sup>95</sup>	d-side 34 <sup>95</sup>	d-side 31 <sup>95</sup>
d-den. 31 <sup>50</sup>	d-den. 34 <sup>95</sup>	d-den. 34 <sup>95</sup>	d-den. 31 <sup>95</sup>
3M	AMARAY MEDIA MATE		Head Cleaners
DC100A...13 <sup>95</sup>	(3 1/2"...1195)	(5 1/4"...1195)	Kits...520
DC300A...1840	DISK MINDERS		Refills...955
DC300XL 2025	(5 1/4"...1675)	(8"...2150)	Analizers 2500
DC600A...2445	BULK PACKED DISKS "CALL"		

Diskettes  
10/Box

the

**Diskette  
Connection™**

Dealer Inquiries  
Welcomed

1(800) 654-4058



OKLAHOMA  
& NEVADA



\*UP's Delivery Only. Add 3% on orders under 35.00 or 20 disk.

Circle 172 for Dealer Inquiries.  
Circle 173 for End-User inquiries.

Circle 106 on Inquiry card.

\*IBM® PC Compatible



E-PROMS — Call! Lowest Prices Anywhere

\*4164-150/200 ..... 480  
 2764-250 ..... 675  
 6116-LP3 ..... 499  
 HMM 256K. .... Call for quote  
 TTL Parts — now avail ..... Call

**Disk Drives:** (F.O.B. Tampa)

\*TM-100-2 or Half-Heights. ... 169<sup>95</sup>  
 \*1 pr. Teac Drives. .... 299<sup>99</sup>

\*10 mb Microscience  
 w/cable & controller ..... 959<sup>99</sup>

Add \$3.95 shipping to all orders • Prices subject to  
 change • P.O.'s on approval • C.O.D. OK • All new, no  
 surplus, no seconds, QUANTITY DISCOUNTS.

4920 Cypress St., Ste. 100, Tampa, FL 33607

In FL and for info, call 813-875-0299

**FOR ORDERS ONLY, 800-237-8910**



8 AM-7 PM EDT

Circle 272 on inquiry card.



**Sure  
 it's insured?**

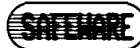
SAFWARE Insurance provides full  
 replacement of hardware, media and  
 purchased software. As little as \$35/yr covers:

- Fire • Theft • Power Surges
- Earthquake • Water Damage • Auto Accident

For information or immediate coverage call:

**1-800-848-3469**

In Ohio call (614) 262-0559



SAFWARE, THE INSURANCE AGENCY INC.

Circle 273 on inquiry card.



		100	20
		Qty	Qty
5 1/4	SS DD 104/1D	1.97	2.10
	DS DD 104/2D	2.66	2.82
8	SS SD 3740/1	2.50	2.65
	DS DD 3740/2D	3.18	3.36



★ Fast  
 Delivery

5 1/4	SS DD MD1	1.74	1.86
	DS DD MD2	2.29	2.44
8	SS DD FD1	2.79	2.95
	DS DD FD2	3.07	3.24

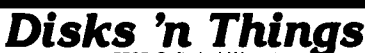


★ Factory  
 Warranty

5 1/4	SS SD M11A	1.40	1.52
	DS DD M14A	2.12	2.26
8	SS SD F11A	1.67	1.80
	SS DD F13A	2.36	2.51

**CALL 818-706-8602**

★ Credit For USA Direct Dial Call  
 With Any Disk Order. ★



5505 Softwind Way  
 Agoura Hills, CA 91301  
 Free Price List Available



Circle 274 on inquiry card.

## DATA ABSTRACTION

### Listing 1: Program for testing the intset abstraction.

```

100 GOSUB 1000 'initialize INTSET storage
110 T = - 1: F = 0 'initialize TRUE and FALSE
200 INPUT C$,P1,P2: IF C$ < "a" THEN C$ = CHR$(ASC(C$) + 32)
250 IF C$ = "c" THEN 260 ELSE 300
260 GOSUB 1100 'create( )
270 IF NOT OS THEN GOSUB 1600 'display(s)
280 GOTO 200
300 IF C$ = "d" THEN 310 ELSE 350
310 S = P1: GOSUB 1600 'display(p1)
320 GOTO 200
350 IF C$ = "e" THEN 360 ELSE 400
360 S = P1: GOSUB 1400 'is-empty(p1)
370 IF NOT NS THEN IF B THEN PRINT "True" ELSE PRINT "False"
380 GOTO 200
400 IF C$ = "i" THEN 410 ELSE 450
410 S = P1: X = P2: GOSUB 1200 'insert(p1,p2)
420 IF NOT NS THEN GOSUB 1600 'display(s)
430 GOTO 200
450 IF C$ = "k" THEN 460 ELSE 500
460 S = P1: GOSUB 1700 'kill(p1)
470 IF NOT NS THEN PRINT "Set";S;"deleted"
480 GOTO 200
500 IF C$ = "m" THEN 510 ELSE 550
510 S = P1: X = P2: GOSUB 1500 'is-member(p1,p2)
520 IF NOT NS AND NOT NI THEN IF B THEN PRINT "True" ELSE PRINT "False"
530 GOTO 200
550 IF C$ = "q" THEN 560 ELSE 600
560 STOP 'quit program
600 IF C$ = "r" THEN 610 ELSE 650
610 S = P1: X = P2: GOSUB 1300 'remove(p1,p2)
620 IF NOT NS THEN GOSUB 1600 'display(s)
630 GOTO 200
650 IF C$ = "u" THEN 660 ELSE 700
660 S1 = P1: S2 = P2: GOSUB 1900 'union(p1,p2)
670 IF NOT NS AND NOT OS THEN GOSUB 1600 'display(s)
680 GOTO 200
700 IF C$ = "x" THEN 710 ELSE 750
710 S1 = P1: S2 = P2: GOSUB 1800 'intersect(p1,p2)
720 IF NOT NS AND NOT OS THEN GOSUB 1600 'display(s)
730 GOTO 200
750 PRINT "Unrecognized command": GOTO 200
    
```

### Listing 2: Matrix implementation of intset abstraction.

```

1000 'Abstract data type: INTSET
1005 '
1010 'Representation:
1015 'Each row of matrix S(10,10) stores a set. There is thus
1020 'a maximum of 11 sets (numbered 0 to 10). If the 0 element
1025 'of a row is - 1, then that set has not been created.
1030 'That leaves columns 1 to 10 for set elements. A cell with
1035 'value - 1 is empty; otherwise the cell contains an element
1040 'of the set
1050 'Initialize the INTSET storage:
1060 DIM S(10,10)
1070 FOR I = 0 TO 10: FOR J = 0 TO 10: S(I,J) = - 1: NEXT J,I
1080 RETURN
1095 '
1100 'create( ) returns(s) signals(os) local(i)
1110 'OS = F
    
```

(continued)

Circle 255 on inquiry card. —

## SINGLE BOARD COMPUTER

### ZINGER ONE

1NS 8073 CPU WITH TINY BASIC INCL.  
2K RAM EXPANDABLE TO 12K ON BOARD  
RAM/ROM INTERCHANGEABLE  
EXPANDABLE TO 56K RAM/ROM OFF BOARD  
EEPROM PROGRAMMER ON BOARD  
24 LINE 3 PORT PARALLEL I/O (8255)  
REQUIRES +5V @ 300 MA ONLY  
ON BOARD SWITCHING POWER SUPPLY  
EIA - RS 232 I/O PORT  
IDEAL FOR CONTROL OR ROBOTICS  
\*\*\*\* \$119.95 ASMBLD & TESTED \*\*\*\*

### ZINGER INTERFACE BOARD

MAY BE PURCHASED WITH ANY OR ALL COMB.  
276 WORD SPEECH SYNTHESIZER  
8 CHANNEL A/D CONVERTER  
REAL TIME CLOCK WITH BATT. BACKUP  
24 LINE 3 PORT PARALLEL I/O (8255)  
64 KEY KEYBOARD INTERFACE  
8 CHAR ALPHA/NUMERIC DISPLAY  
INTERRUPT REGISTER ON ALL DEVICES  
COMMODORE VIC 20  
ADAPTER AVAILABLE

### SCHULZ ENTERPRISES INC.

1285 LAS TUNAS DR.  
SAN GABRIEL, CA 91776  
(818) 287-5067

## SPEED UP YOUR dBASE PROGRAMS

with dB/RA, dBRX,  
dBRx/87,  
dHELPER & RA+

Add arrays, math functions,  
8087 support, syntax  
checking, animation &  
windowing.

Call or write for details!



**GRYPHON**  
microproducts

P.O. BOX 6543 SILVER SPRING, MD. 20906  
(301) 946-2565

See us at COMDEX in Las Vegas

**3M**  
**3M**  
**3M**  
**3M**  
**3M**

THE  
RELIABLE  
ONE

DAY AFTER DAY  
5 1/4 Diskettes

Hard or Soft Sector with  
Reinforced Hub (RH)  
UNLIMITED WARRANTY.  
Sold in Boxes of 10.

Single Sided, Double Density.....\$1.62  
Double Sided, Double Density.....\$2.19  
SS-DD-96TPI-RH.....\$2.45  
DS-DD-96TPI-RH.....\$3.05  
10% Surcharge for quantities less than 50 diskettes.

#### \* Attention \*

\* Software Duplicators \* Bulk - No logo  
Call for Special Pricing on orders of 50 or more only.



**Precision Data Products**  
P.O. Box 8367, Grand Rapids, MI 49508  
(616) 452-3457 • Michigan 1-800-632-2468  
Outside Michigan 1-800-258-0028

MI Residents, Add 4%  
Sales Tax.  
Shipping & Handling,  
Add \$3.00/100  
C.O.D. Add \$2.00

ORDER  
TOLL-FREE



C.O.D.

## DATA ABSTRACTION

```

1120 FOR I=0 TO 10 'find first unused set
1130 IF S(I,0)=-1 THEN S(I,0)=0: S=I: GOTO 1150
1140 NEXT I: OS=T: PRINT "Out of sets"
1150 RETURN 1195 '
1195 '
1200 'insert(s,x) returns(s) signals(ns,ni,nr) local(i,j)
1210 GOSUB 2000: IF NS THEN RETURN
1220 GOSUB 2100: IF NI THEN RETURN
1230 NR=F: J=0 J holds first available cell
1240 FOR I=10 TO 1 STEP -1
1250 IF S(S,I)=X THEN RETURN 'no duplicates in a set
1260 IF S(S,I)=-1 THEN J=I
1270 NEXT I
1280 IF J=0 THEN NR=T: PRINT "No more room in set";S: ELSE S(S,J)=X
1290 RETURN
1295 '
1300 'remove(s,x) returns(s) signals(ns) local(i)
1310 GOSUB 2000: IF NS THEN RETURN
1320 GOSUB 2100: IF NI THEN RETURN
1330 FOR I=1 TO 10
1340 IF S(S,I)=X THEN S(S,I)=-1
1350 NEXT I
1360 RETURN
1395 '
1400 'is-empty(s) returns(b) signals(ns) local(i)
1410 GOSUB 2000: IF NS THEN RETURN
1420 B=T
1430 FOR I=1 TO 10
1440 IF S(S,I)>-1 THEN B=F 'test for a used cell
1450 NEXT I
1460 RETURN
1495 '
1500 'is-member(s,x) returns(b) signals(ns) local(i)
1510 GOSUB 2000: IF NS THEN RETURN
1520 GOSUB 2100: IF NI THEN RETURN
1530 B=F
1540 FOR I=1 TO 10
1550 IF S(S,I)=X THEN B=T
1560 NEXT I
1570 RETURN
1595 '
1600 'display(s) signals(ns) local(i)
1610 GOSUB 2000: IF NS THEN RETURN
1620 PRINT "Set";S;": {"
1630 FOR I=1 TO 10: IF S(S,I)<>-1 THEN PRINT S(S,I);
1640 NEXT I: PRINT "}"
1650 RETURN
1695 '
1700 'kill(s) signals(ns) local(i)
1710 GOSUB 2000: IF NS THEN RETURN
1720 FOR I=0 TO 10
1730 S(S,I)=-1 'every cell becomes unused
1740 NEXT I
1750 RETURN
1795 '
1800 'intersect(s1,s2) returns(s) signals(ns,os) local(i,j,k)
1810 S=S1: GOSUB 2000: IF NS THEN RETURN
1815 S=S2: GOSUB 2000: IF NS THEN RETURN
1820 GOSUB 1100: S3=S: IF OS THEN RETURN 'create new set for result
1830 FOR K=1 TO 10
1840 IF S(S1,K)=-1 THEN 1860 'for each member of first set,
1850 X=S(S1,K): S=S2: GOSUB 1500 'if it is a member of second set,
1855 IF B THEN S=S3: GOSUB 1200 'then insert into result
1860 NEXT K
    
```

(continued)

## The Business Professionalism from Hewlett-Packard —The 6-F

Today, business  
more aware  
of business  
Tomorrow, the  
mendation. H  
that will help

## Make a firm

Truly impressive  
impression of  
lasts. The way  
as important  
where the H  
fessionalsism

## Standard

The technical  
ing quality  
inch, curved  
consistently  
pen to return  
lines and cir

## Compatible in your office popular

The HP 747  
computers y  
Apple™, and  
even have a  
as Lotus 1-2-3  
tivity with t

## Your Choice

While most  
with stande  
adds the ca

## The cost

The HP 747  
able \$1895.  
prepared by  
ment is alm

1-2-3 and Sym

Circle 385

```

1870 S=S3: RETURN
1895 '
1900 'union(s1,s2) returns(s) signals(ns,os,nr) local (i,j,k)
1910 S=S1: GOSUB 2000: IF NS THEN RETURN
1915 S=S2: GOSUB 2000: IF NS THEN RETURN
1920 GOSUB 1100: S3=S: IF OS THEN RETURN 'create new set for result
1930 FOR I=1 TO 10
1935   S(S3,I)=S(S1,I) 'copy first set into result
1940 NEXT I
1950 FOR K=1 TO 10
1960   IF S(S2,K)=-1 THEN 1980 'for each member of second set,
1970   S=S3: X=S(S2,K): GOSUB 1200 'insert it into result
1975   IF NR THEN 1990
1980 NEXT K
1990 RETURN
1995 '
2000 'is-valid-set(s) return(ns)
2010 IF S<0 OR S>10 THEN NS=T: PRINT S;"not a valid set number": RETURN
2020 IF S(S,0)=-1 THEN NS=T: PRINT "Set";S;"not created yet": RETURN
2030 NS=F: RETURN
2095 '
2100 'is-not-integer(x) returns(ni)
2110 IF X<0 OR X<>INT(X) THEN NI=T:PRINT X;"not a valid integer":RETURN
2120 NI=F:RETURN

```

ables used (that is, those not part of the concrete representation or the user interface) do not conflict with those used elsewhere, whether in another abstraction, the calling procedure, or in other operations of the same data type. For this reason it is advisable to add one more item of information to the header for each operation, namely, a list of the additional local variables that it uses. Note that this is done in listing 2.

The local variables are ones that the operations use temporarily. Whatever value they may have had previously is destroyed. When a subroutine returns, their value is undefined and they are free to be used again. Keeping track of local variables becomes tricky (and imperative) when one operation calls another. The implementation of intersect (line 1800 and following) is a case in point. One

(continued)

## Technologically Superior Syntech Data Systems (SD) S-100 Boards at a Discount

including...but not limited to...

- CPU 8/16 - dual processor Z80 & 8088... \$785.00  
Plus many other features
- Dynamic Ram - 256K up to 2 MB... Starting at... \$750.00
- SBC 300 - single board computer-master or slave... \$615.00
- Versafloppy II - controls up to 4 floppy drives... \$330.00
- I/OB-1/04 - 4 or 8 serial channels... Starting at... \$575.00
- Hard Disk Controller - up to 4 winchester drives... \$525.00
- Z80 Starter Kit - Kit form or A&T... starting at... \$469.00

Many More Boards & Systems Available...  
Call for details...

**PLUS**

Concurrent CP/M only \$125.00\*  
\*With any CPU 8/16 purchase

For more information call 301-842-5442  
**MCC** MICRO COMPUTER COMPANY INC.  
101 Wheaton Place North Wheaton, Maryland 20902  
SALES \* SERVICE \* SUPPORT

Mention this ad for FREE FREIGHT

## MEGA-BYTES FOR MICRO-BUDGETS expand your system...shrink your cost.

Why pay more for top quality products when our prices are consistently among the lowest anywhere?  
We invite you to compare prices, then call us.

### MISC. PERIPHERALS

HAYES SMARTMODEM 1200B (IBM-PC).....	\$399.50
HAYES SMARTMODEM 1200 (RS-232) .....	489.50
HAYES CHRONOGRAPH.....	189.50
ROLAND DG XY800 6 PEN PLOTTER.....	799.50
BAUSCH&LOMB DMP-29 PLOTTER .....	1885.00
PENCEPT PENPAD 320 .....	900.00
PRINCETON GRAPHICS SYSTEMS HX-12.....	Call

### EPSON PRINTERS

MX-100 .....	\$475.00	RX-80 .....	\$309.00
FX-80 .....	489.00	RX-80 F/T .....	375.00
FX-100 .....	689.00	LQ-1500 .....	1135.00
DYSAN DISKETTES (Box of 10)		3740/1 8" SSDD .....	\$32.39
104/1 5 1/4" SSDD .....	\$31.20	3740/1D 8" SSDD .....	40.19
104/1D 5 1/4" SSDD .....	32.98	3740/2 8" DSSD .....	40.19
104/2D 5 1/4" DSDD .....	38.99	3740/2D 8" DSDD .....	46.89

### GREAT LAKES (PEGASUS) HARD DISK SYSTEMS

10 MEGABYTE INTERNAL .....	\$1149.00
10 MEGABYTE EXTERNAL .....	1295.00
23 MEGABYTE EXTERNAL .....	1895.00
40 MEGABYTE EXTERNAL .....	2449.00
65 MEGABYTE EXTERNAL .....	3249.00
140 MEGABYTE EXTERNAL .....	4995.00
TAPE DRIVE 23 MEGABYTE INTERNAL .....	950.00
TAPE DRIVE 23 MEGABYTE STAND ALONE .....	1249.00

## COMMERCIAL BUSINESS SYSTEMS

2858 S. ROBERTSON BLVD., LOS ANGELES, CA 90034

ORDERS ONLY 800-858-4810  
IN CALIF. 800-821-6662



INFORMATION  
(213) 559-0596

Phone orders accepted on Visa and MC only. CA residents add 6 1/2% sales tax. No COD. Actual shipping and handling charge added to all orders. Prepaid orders as follows: M.O. or cashier's check—merchandise shipped upon receipt. Personal checks must clear before shipping. 20% restocking fee. Prices and availability subject to change. \$100 min. order.

## Data abstraction allows you to modify a program simply by changing a module's implementation.

would be tempted to use I as the local indexing variable as is done in all other operations. However, intersect calls insert (line 1200 and following), which already uses I and J as local variables; we see this by looking at the local statement in the header of insert. This alerts us to the fact that if we use I in intersect we are in for trouble; every call to insert would destroy its value. Thus, we use a new variable, K. Note also that the local variables for intersect are given as I, J, and K, even though I and J do not appear in the code for the intersect operation. This is because a pro-

cedure always inherits the local variables of any procedure it calls.

### MAINTENANCE AND PORTABILITY

From the perspective of the program code that is outside an abstract data type, we have already seen that an advantage of programming with data abstractions is that you can write programs that use them without knowing how they are implemented on the inside. Now we take the perspective of the program code inside the abstract data type and see that an advantage of programming with data abstractions is that you can modify the implementation without affecting the outside programs that use it. This is a boon for maintaining a program and porting it to other systems.

For instance, suppose we decided that limiting our sets to a maximum of 10 elements (as does the implementation of listing 2) is too restrict-

ing. We decide we want to modify our program to allow for sets of up to 20 elements. Because the information about how the intset abstraction is represented is hidden inside the intset module, any programs that use the abstraction (in this case the test program of listing 1) are not affected. The only changes to make include redefining the matrix dimensions in the initialization procedure (lines 1050-1080) and changing the upper bound of the FOR statements in all the subroutines for the operations. We soon discover that this latter change gets rather tedious and that we would have been better off in the first place to make the maximum size of a set a variable in the concrete representation of intset, and then to use that variable in the FOR loops for all the subroutines. Then changing the maximum size of sets would mean chang-

(continued)

# WOW!



**IBM PC**  
**\$1399.95**  
**OKIDATA 92**  
**\$369.95**

#### PRINTERS

Okidata 92	369	Gemini 10K	234	Juki 6100	359
Okidata 93	569	Gemini 15K	349	Tractor	114
Okidata 82	309	Radix 15	554	Panasonic KXP 1091	269
Okidata 83	524	Radix 10	519	Panasonic KXP 1090	199
Epson RX80 FX	290	Brother HR15	344	Silver Reed EXP 550	369
Epson RX80	239	Brother HR25	569	Silver Reed EXP 500	319
Epson RX100	479	Brother HR35	759	Silver Reed EXP 770	809
Epson FX80	414	Keyboard	129	Itoh Pwrtter 8510	319
Delta 10	339	Tractor	79	Stinwriter F10	869
Delta 15	489	Cut Feed	169	Nec 3550	1389

**800-441-1144**



#### APPLE

2E w/Disk Drive	799
Macintosh	1749
Apple 2C	939
Imagewriter	499
Printer Card	79
CPM 2.2 Card	109
RGB Card	124
Alt. Drives From \$99 & Up	
System Saver Fan	69
Koala Pad	74

#### SANYO

550 S.S.	654
550 D.S.	659
555 S.S.	859
555 D.S.	969

#### MONITORS

Amdek 300 Green	114	Commodore 64	184
Amdek 300 Amber	124	1541 Disk Drive	214
Color 1	234	1702 Monitor	214
Color 2	349	MPS801 Printer	179
Color 4	549	1526 Printer	224
310 Amber	134	1650 Modem	89
BMC Green	89		
Banana Green	74	FRANKLIN	
Taxan 210	209	1000 Pro	679
		1200 OMS	1059

#### COMMODORE

550 S.S.	654
550 D.S.	659
555 S.S.	859
555 D.S.	969

#### MODEMS

Hayes 1200	424
Hayes 1200B	364
Hayes 300	189
Micromodem 2E	209
Access 123	339
Novation Scat	88

## HARMONY VIDEO & COMPUTERS

2357 CONEY ISLAND AVE., BROOKLYN, NY 11223

TO ORDER CALL TOLL FREE

800-VIDEO84 OR 212-627-1000 OR 800-441-1144

## MICRO CONTROLLED DIGITAL DATA RECORDER

### FEATURES:

Microprocessor controlled data buffering • Buffers data in RAM • Data comes in at any standard baud rate, plays back at any baud rate (switchable) • Tape runs only during block record/playback • RS232 input/output 110/220 v ac or 12 v dc • 1.2

MB per tape side • Uses chrome oxide audio cassettes • Has hold-off during playback via CTS line • No data hold-off during record.

### APPLICATIONS:

PROCESS CONTROL • POINT OF SALE • TELEPHONE SWITCH LOGGING (SMDR) • INSTRUMENTATION • DIAGNOSTIC SUPPORT • PROGRAM LOADING • DATA LOGGING.

BUFFERED VERSION MODEL PD1-BF..... \$595.00  
NON BUFFERED VERSION - MODEL PD-1..... \$335.00



TO ORDER, DIAL:  
(201) 356-9200

236 Lackland Drive  
Middlesex, N.J. 08848

When i  
perfo  
your IBM  
best multi  
the Rio Ph  
features of  
multi-fun  
Plus, and  
built into  
Rio Plus II  
one slot a  
greater m  
ports.

"Pack" T

Now, you  
All in one  
a parallel  
Accelerate  
and a par  
(rememb  
additional  
Using

STB Sy

IBM PC, XT and



# MEMOREX

**SAVE 50%  
ON DATA  
RELIABLE DISCS**



Dealer Inquiries Invited

5 1/4"	Specify Soft 10 or 16 Sector	Box/10
3481	1 side/dbl dens.....	\$22.30
3491	2 sides/dbl dens.....	\$30.70
3504	1 side/quad 96 tpi.....	\$30.00
3501	2 sides/quad 96 tpi.....	\$33.30

8"	Specify Soft or 32 Sector	
3062	1 side/sgl dens.....	\$22.10
3090	1 side/dbl dens.....	\$28.90
3102	2 sides/dbl dens.....	\$33.00

Checks-VISA-MC-C.O.D./Add \$2 Shipping  
Call or write for our discount catalog.

**LYBEN COMPUTER SYSTEMS**

1250 Rankin, Bldg. E, Troy, MI 48064  
Phone: (313) 777-7780

**CERTIFIED 100% ERROR-FREE**

## MEMORY MODULES

8Kx8  
CMOS  
RAM

**Radio Shack Model 100  
NEC PC-8201A  
Olivetti M10**

✓ Suggested List \$120.00.  
Purple Price **\$59.95**

- ✓ Low power CMOS design.
- ✓ Simple installation.
- ✓ 30 day satisfaction guarantee or your money back.
- ✓ 1 Year warranty.
- ✓ Next day shipment via UPS included in price.
- ✓ Optional Memory Test program \$15. (Cassette)
- ✓ No frills direct connect Modern Cable - \$9.95

Shipping: From stock. Free UPS surface Cont. USA —  
Add \$4.00 for UPS 2 day Air — Add \$7.00 for Canada  
— Payment: VISA, Master Card, or American Express.  
Checks held 14 days — Tax: 6% (Calif only).

**CALL NOW**

**PURPLE COMPUTING**  
2068 Ventura Blvd. 1 - (800) 732-5012  
Camarillo, CA 93010 Calif (805) 987-4788

VISA  
Master Card  
American Express

## STAND-BY POWER SUPPLY



- 200 WATTS
- 10 MINUTE  
BACKUP TIME
- BATTERY  
INCLUDED

Fits most desktop and portable computers.  
Complete, just plug it in. 4-(3 prong) outlets  
on back protected by line filters. Neat, clean  
appearance on desk. Don't risk damage or  
data loss with POWER FAILURES.

**MODEL BC200-10**



**\$299\***



Units available to 1000 Watts

**312-490-9239**

**SHEPHERD MARKETING**

P.O. BOX 941339

SCHAUMBURG, IL 60194

\*Add \$7 each shipping & handling

## DATA ABSTRACTION

ing only one assignment statement in the concrete representation of the data type. In any case, all program modifications are confined to the intset module.

Unfilled sets waste space, and increasing the size of the matrix to accommodate potentially larger sets has the disadvantage of wasting even more space. One solution is to represent an intset as a linked list of elements so that the set never uses more memory than it needs. With such a scheme, the maximum length

of a set depends on the total length of all other sets, rather than a fixed limit for each set.

Normally such a change would require a complete rewrite. When data abstraction is used, it means replacing only the implementation of the intset module; the programs that use intset change in no way. As an example, listing 3 gives a completely new implementation of the intset abstraction using a linked-list representation. This implementation adheres

(continued)

### Listing 3: Linked list implementation of intset abstraction.

```

1000 'Abstract data type: INTSET
1005 '
1010 'Representation:
1015 ' ST(10) is the "Set Table"
1020 ' ST(I) specifies the address in S of the first element of set I
1025 ' ST(I) = -1 means set I not yet created
1030 ' S(100,1) stores the set elements
1035 ' Column 0 stores a set element
1040 ' Column 1 is link which specifies address in S of next element
1045 ' Column 0 value of -1 marks end of list (and of set)
1050 ' A points to next available cell of S
1055 ' Initially all cells of S are linked together on available list
1060 'Initialize the INTSET storage:
1065 DIM ST(10): FOR I=0 TO 10: ST(I)=-1: NEXT I
1070 DIM S(100,1): FOR I=0 TO 99: S(I,1)=I+1: NEXT I 'link all cells
1075 S(100,0)=-1: A=0 'put entire list of cells on available list
1080 RETURN
1095 '
1100 'create() returns(s) signals (os) local(i,n)
1110 OS=F
1120 FOR I=0 TO 10
1130 IF ST(I)=-1 THEN 1150 'find first unused set
1140 NEXT I: OS=T: PRINT "Out of sets": RETURN 'fail with OS if none
1150 GOSUB 2200: IF NR THEN OS=T: RETURN 'or if no more room
1160 ST(I)=N: S(N,0)=-1: S=N: RETURN 'initialize set
1195 '
1200 'insert(s,x) returns(s) signals(ns,ni,nr) local(i,n)
1210 GOSUB 2000: IF NS THEN RETURN
1220 GOSUB 2100: IF NI THEN RETURN
1230 GOSUB 1500: IF B THEN RETURN 'no duplicates
1240 GOSUB 2200: IF NR THEN RETURN
1250 S(N,1)=ST(S): ST(S)=N S(N,0)=X 'insert the element
1260 RETURN
1295 '
1300 'remove(s,x) returns(s) signals(ns,ni) local(i,j)
1310 GOSUB 2000: IF NS THEN RETURN
1320 GOSUB 2100: IF NI THEN RETURN
1330 I=ST(S): J=0 J will point to cell preceeding one to remove
1340 IF S(I,0)=-1 THEN RETURN
1350 IF S(I,0)<>X THEN J=I: I=S(I,1): GOTO 1340 'find element to remove
1360 IF J=0 THEN ST(S)=S(I,1) ELSE S(J,1)=S(I,1) 'remove it
1370 S(I,1)=A: A=I 'return removed cell to available list
1380 RETURN
1395 '
1400 'is-empty(s) returns(b) signals(ns)

```

(continued)

# Sul Tor

The N  
Center is t  
research,  
evaluation  
warfare an  
rine weap  
tems. We  
high techn  
vital to the  
tactical an  
gic edge  
systems  
combat c  
electroma  
underwat  
ons and t  
weapon l  
face ship  
ranges.

Engi  
join the C  
port, Rho



```

1410 GOSUB 2000: IF NS THEN RETURN
1420 IF S(ST(S),0)=-1 THEN B=T ELSE B=F
1430 RETURN
1495 '
1500 'is-member(s,x) returns(b) signals(ns,ni) local(i)
1510 GOSUB 2000: IF NS THEN RETURN
1520 GOSUB 2100: IF NI THEN RETURN
1530 I=ST(S)
1540 IF S(I,0)=-1 THEN B=F: RETURN 'end of set, not found
1550 IF S(I,0)<>X THEN I=S(I,1): GOTO 1540 'not yet, keep looking
1560 B=T: RETURN 'it is found
1595 '
1600 'display(s) signals(ns) local(i)
1610 GOSUB 2000: IF NS THEN RETURN
1620 PRINT "Set";S;" = {"
1630 I=ST(S)
1640 IF S(I,0)=-1 THEN 1660
1650 PRINT S(I,0);": I=S(I,1): GOTO 1640
1660 PRINT "}": RETURN
1695 '
1700 'kill(s) signals(ns) local(i)
1710 GOSUB 2000: IF NS THEN RETURN
1720 I=ST(S)
1730 IF S(I,0)<>-1 THEN I=S(I,1): GOTO 1730 'find end of set
1740 S(I,0)=0: S(I,1)=A: A=ST(S) 'return cells to available list
1750 ST(S)=-1: RETURN 'mark ST entry unused
1795 '
1800 'intersect(s1,s2) returns(s) signals(ns,os) local(i,j,n)
1810 S=S1: GOSUB 2000: IF NS THEN RETURN
1815 S=S2: GOSUB 2000: IF NS THEN RETURN
1820 GOSUB 1100: S3=S: IF OS THEN RETURN 'create a new set for result
1830 J=ST(S1)
1840 IF S(J,0)=-1 THEN 1870
1850 X=S(J,0): S=S2: GOSUB 1500 'if a member of S1 is in S2,
1855 IF B THEN S=S3: GOSUB 1200: IF NR THEN 1870 'then insert in result
1860 J=S(J,1): GOTO 1840
1870 S=S3: RETURN
1895 '
1900 'union(s1,s2) returns(s) signals(ns,os,nr) local(i,j,n)
1910 S=S1: GOSUB 2000: IF NS THEN RETURN
1915 S=S2: GOSUB 2000: IF NS THEN RETURN
1920 GOSUB 1100: S3=S: IF OS THEN RETURN 'create a new set for result
1930 J=ST(S1) 'insert every element of S1 into result
1940 IF S(J,0)=-1 THEN 1960
1945 X=S(J,0): S=S3: GOSUB 1200: IF NR THEN 1990
1950 J=S(J,1): GOTO 1940
1960 J=ST(S2) 'insert every element of S2 into result
1970 IF S(J,0)=-1 THEN 1990
1975 X=S(J,0): S=S3: GOSUB 1200: IF NR THEN 1990
1980 J=S(J,1): GOTO 1970
1990 S=S3: RETURN
1995 '
2000 'is-not-set(s) returns(ns)
2010 IF S<0 OR S>10 THEN NS=T: PRINT S;"not a valid set number": RETURN
2020 IF ST(S)=-1 THEN NS=T: PRINT "Set";S;"not created yet": RETURN
2030 NS=F: RETURN
2095 '
2100 'is-not-integer(x) returns(ni)
2110 IF X<0 OR X<>INT(X) THEN NI=T:PRINT X;"not a valid integer":RETURN
2120 NI=F: RETURN
2195 '
2200 'get-next-cell() returns(n) signals(nr)
2210 IF S(A,0)=-1 THEN PRINT "No more room": NR=T: RETURN
2220 N=A: A=S(A,1): NR=F: RETURN

```

to the interface of figure 6 and therefore is equivalent to the matrix implementation in its external effects. The test program works equally well, whether one combines it with the intset module of listing 2 or the module of listing 3.

Since nearly any two implementations of BASIC differ in some details, BASIC programs turn out to be of limited portability in actual practice. Writing software in terms of data abstractions is an excellent way to enhance a program's ultimate portability. Information hiding localizes the details of implementation that are likely to be changed, such as variable names, matrix dimensions, and input/output protocols.

Passing parameters with assignment statements and calling GOSUB in most lines of a program can get tedious. It may often seem justifiable on the grounds of efficiency to access a data structure directly without going through the operations. You should avoid such temptations at all costs: you'll pay the price when it comes time to debug, enhance, or port the program. ■

#### REFERENCES

1. Guttag, J., and J. J. Horning. "The Algebraic Specification of Abstract Data Types." *Acta Informatica*, volume 10, 1978, pages 27-52.
2. Hoffman, C. M., and M. J. O'Donnell. "Programming with Equations." *ACM Transactions on Programming Languages and Systems*, volume 4, 1982, pages 83-112.
3. Liskov, B. "Modular Program Construction Using Abstraction." *Abstract Software Specifications*, D. Bjørner, ed. *Lecture Notes in Computer Science*, number 86. Berlin: Springer-Verlag, 1980, pages 354-389.
4. Parnas, D. L. "A Technique for Software Module Specification with Examples." *Communications of the ACM*, volume 15, 1972, pages 330-336. Reprinted in *Communications of the ACM*, volume 26, number 1, 1983.

#### ACKNOWLEDGMENTS

The author is indebted to two of his colleagues of the Summer Institute of Linguistics: to Graeme Costin for many helpful comments on an earlier draft of the article, and to Dr. Joseph E. Grimes (also of Cornell University) for some insights on programming with data abstractions in BASIC.



#### Develop

Avocet cro and user-p use. Ask hundreds of them. Even processor-chance it cross-as

Avocet cro They run computer and the most p

#### Your Com Complete

Avocet has and assem cast it in E

#### VEDIT T

entry a sm TECO-like tasks. Ea gram sup personal keyboard MSDOS, I

#### EPROM

gram, ver EPROMS communic computer modules! menu-ba EPROMS devices r contained RS-232 in Driver so you acce through load cro EPROM

#### Model 7

— Suppo fast "ada programs

#### Model 7

Lower-co PROM ty and 2712 rithm pro