



SEPTEMBER/OCTOBER 1983

Volume 11 Number 5

I.P. Sharp

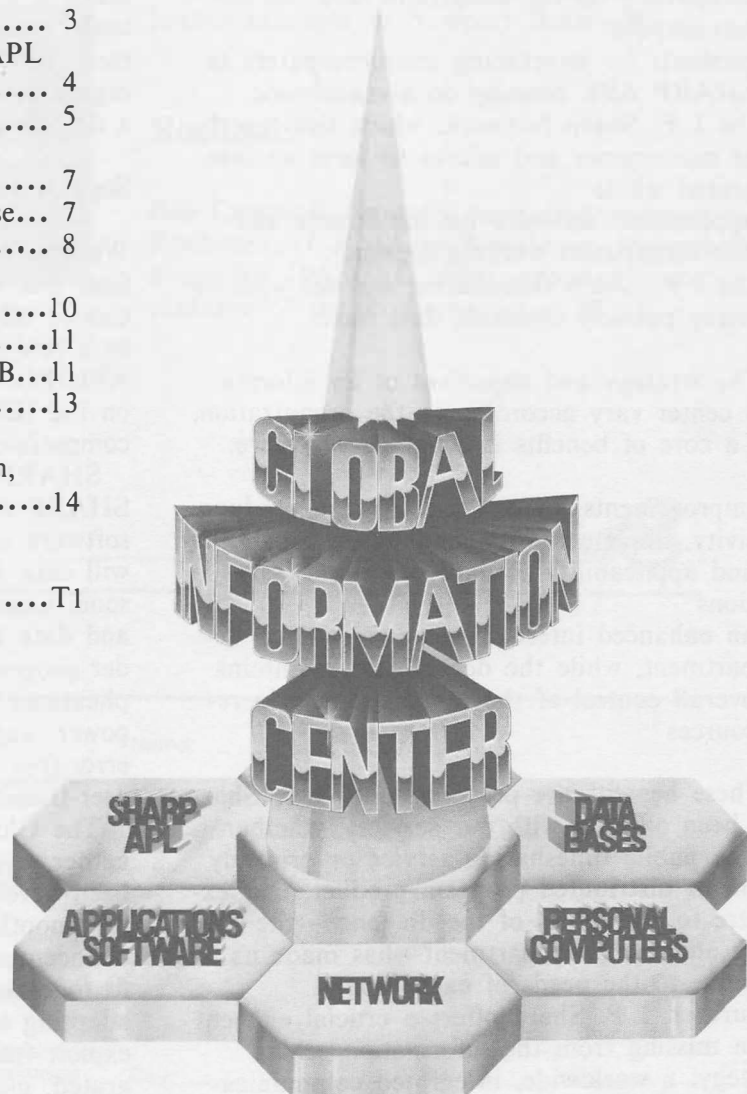
newsletter

in this issue

The Global Information Center.....	1
SHARP APL for the IBM Personal Computer	3
Unbundling of Mainframe SHARP APL Prices.....	4
INFOMAGIC	5
SES Marketscan: A New Global Information Service.....	7
Citibase Economic Forecast Data Base...	7
New U.S. Banks Data Base.....	8
The MEDEX Insurance and Medical Information System.....	10
Third Party Software.....	11
Electronic Spreadsheet Tablet: ESTAB...	11
Workspace 1 <i>INFO</i>	13
Toronto, Paris, Atlanta, Chicago, Houston, Los Angeles, Newport Beach, Rochester	14

Technical Supplement 46

Making Things Fit, Part 2.....	T1
--------------------------------	----



THE · SHAPE · OF
· THE · FUTURE ·

The Global Information Center

The **Global Information Center** is the shape of the future—not just for I. P. Sharp, but for any organization trying to make optimum use of today's array of powerful data processing technologies.

I. P. Sharp's Global Information Center products include:

- the SHARP APL language, implemented compatibly on the mainframe and the microcomputer
- products for interfacing microcomputers to SHARP APL running on a mainframe
- the I. P. Sharp Network, which ties together mainframes and micros to form an integrated whole
- applications software for mainframe and microcomputers working together
- the I. P. Sharp timesharing service, with its many publicly available data bases

The strategy and objectives of an information center vary according to the organization, but a core of benefits is sought everywhere:

- improvements for end users in the productivity, implementation timescales, capability, and applicability of data processing solutions
- an enhanced internal image for the dp department, while the department maintains overall control of the organization's dp resources

These benefits are precisely what I. P. Sharp has been offering with our services, whether on the public timesharing service or privately with our distributed program product. Our exposure to both sides of the dp fence—the users and the dp department—has made us sensitive to the needs of each group.

Further, I. P. Sharp offers a crucial element often missing from the information center strategy: a worldwide, integrated communications network. The I. P. Sharp Network forms the Global Information Center's context, by making it possible, even easy, for mainframe-based SHARP APL timesharing services, microcomputers running SHARP APL, dumb terminals, and microcomputer-based workstations to communicate and process information at the appropriate level of independence and

interdependence.

Recognizing that many of our customers use some I. P. Sharp products in their "information centers" while perhaps not being aware of our total capabilities, we have for some months been critically examining our position as provider of many of the elements essential to the successful information center. It seems that while the *concept* of the information center is widely accepted, the marketplace lacks the software, communications, and support to make it *happen*. I. P. Sharp has the products, tools, and experience you can use to implement the Global Information Center for your organization smoothly, economically, and with a degree of integration not possible before.

Support for microcomputers

Work is now well advanced on forging the final link in I. P. Sharp's Global Information Center offering: support for microcomputers. In a few months, we shall offer SHARP APL/PC, an implementation of SHARP APL on the IBM Personal Computer, absolutely compatible with mainframe SHARP APL.

SHARP APL/PC will maintain mainframe SHARP APL's ability to interface with other software and data bases. At the same time, it will take advantage of the features of the Personal Computer. The ability to move software and data between a mainframe and a PC under program control will encourage hybrid applications that use the mainframe for stability, power, and public data bases; the network for error-free communication; and the PC for user-friendly, front-end processing.

The Global Information Center brings together several technologies that I. P. Sharp has been developing for 14 years. During the next few months, we shall be making several announcements of how our products and services fit together in this context. We shall also be adapting our existing applications software to exploit fully the potential of this new, integrated, global SHARP APL environment.

At the same time, we are keen to take full account of our customers' requirements and strategies as we develop new software for the Global Information Center. We welcome your comments and ideas, and we shall be actively seeking opportunities to discuss how the Global Information Center meets your needs and wants. ■

SHARP APL for the IBM Personal Computer

Development strategy

I. P. Sharp will soon offer SHARP APL for the IBM Personal Computer. SHARP APL/PC—now under development—is a crucial element of I. P. Sharp's commitment to the **Global Information Center** announced in this issue.

SHARP APL/PC

APL is the programming language that for 15 years has greatly reduced the cost of developing computer applications. At the same time, APL has given managers and analysts at every level of enterprise the power to compute exactly what they want to compute.

Until very recently, APL was available only on large mainframe computers. Since APL automatically takes care of operations that require detailed coding in other languages, these extra capabilities made the task of providing APL on a microcomputer harder than bringing you BASIC or PASCAL. But it is being done. Soon it will be possible even on a microcomputer to enjoy the advantages of SHARP APL.

APL on a desktop Personal Computer exists in an environment very different from APL on a timesharing service, where many simultaneous users share a large computer. Certain aspects of timeshared APL just don't apply to a PC. But whenever you execute an APL program, or type a line of SHARP APL on your PC's keyboard, your results will be *exactly* what they would be on the mainframe system because *SHARP APL/PC uses exactly the same code as the mainframe SHARP APL interpreter*.

Choice of strategies

When a new family of machines is invented, with a new architecture and a new instruction set, the people who build language processors such as SHARP APL have two main alternatives:

1. Write the APL system all over again for the new machine, making use of the new

instruction set and the new architecture—a *native system*.

2. Make the new machine behave like the machines for which APL systems are already built—an *emulation*.

The SHARP APL/PC development team—headed by Roger Moore and Eric Iverson—chose the second strategy. The following paragraphs describe the advantages and disadvantages of each, and explain why we believe the second will bring you a better APL system for the Personal Computer.

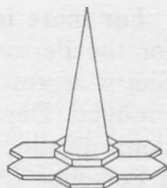
A native PC system

The big advantage of writing a new APL system in the PC's own machine language is speed. The system designers can tailor their code to what the machine does best.

The major disadvantage is that everything must be done anew, from scratch. Moreover, there's a conflict between picking techniques best suited to the PC and preserving consistency with the mainframe version. For example, the easiest way to store numbers in the PC would give them less precision than they have on the mainframe; algorithms that depend on that precision then give slightly different results. Many native implementations on the IBM PC have made such compromises.

SHARP APL on mainframes is the product of many years of enhancement, development, and evolution. Its 82 000 lines of System/370 Assembler language represent years of development, debugging, and refinement. It would be no small matter to duplicate the quality of SHARP APL on a brand-new system written in a new language.

More important, SHARP APL continues to develop. The language offered on the I. P. Sharp timesharing service and to in-house customers is being extended vigorously. Having an independent APL system for the PC would mean that all development would have to be done in parallel for two quite different APL implementations.



...SHARP APL/PC

An emulator system

Emulator is computer jargon for something that makes one machine behave like another. An emulator may be a program (that is, software) which interprets the machine instructions intended for a different machine, or it may be hardware (for example, an installable chip). I. P. Sharp has developed for the IBM Personal Computer a software emulator that permits the PC to execute a very large and sufficient subset of the machine instructions of System/370.

The big advantage of an emulator is that it lets you run on the PC the same interpreter that is used on mainframes. As new releases of SHARP APL are made for mainframes, they're equally appropriate to the PC. There's no need for a complete new system, and for continued separate maintenance and development of SHARP APL on the PC.

The disadvantage of an emulator—especially a software emulator—is that it is often slower than a native implementation. The emulator may require a dozen PC instructions to duplicate exactly the effect of one System/370 instruction, whereas a single PC instruction could do something similar but not quite the same.

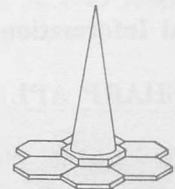
The emulator's low speed doesn't apply across the board. Depending on what the particular instruction is, or the size of the data array involved, the speed with which the emulator can handle an APL statement may approach or exceed the speed of a native system. As well, improvements in hardware (such as emulator chips and accelerators) are inevitable.

The strategy chosen

Given the alternatives, it wasn't difficult to choose a strategy for our development of SHARP APL on the PC. Consistency with the mainframe implementation of SHARP APL dominated over all other considerations. SHARP APL/PC has been built to be *absolutely compatible*, feature for feature, with SHARP APL on the mainframe. This compatibility is essential to I. P. Sharp's conception of the Global Information Center.

For more information about SHARP APL for the Personal Computer, contact your I. P. Sharp representative, or the Microcomputer Products Department at I. P. Sharp's international headquarters. If you're planning to buy a Personal Computer in the immediate future, we can tell you what configuration of the PC will allow you to run SHARP APL. ■

Unbundling of Mainframe SHARP APL Prices



As part of I. P. Sharp's **Global Information Center** strategy, SHARP APL for mainframes has been "unbundled". This means that the "base" SHARP APL timesharing system is available separately from twenty or so optional components.

SHARP APL has been available since 1978 as a software product that customers can run on their own in-house mainframes. But SHARP APL is much more than a language processor; it is a *complete timesharing system*, and has many features not available on competing systems. I. P. Sharp charged a single fee for this complete system, but not every installation needed every feature. Now, with the unbundled product, each installation gets exactly what it needs, and no more. When an installation needs an optional component, it is available for only the incremental licence fee.

The base system comprises the SHARP APL interpreter and supervisor, the file system, the SHARP APL version of TSIO, support for the IBM 3270 display system, and the online user number creation system. The optional components, currently numbering twenty in all, are each available and priced separately. They include: interfaces to OS data sets; auxiliary processors for use with IMS, ADABAS, or IDMS; performance monitors; source code; and maintenance. This list grows continually.

When you install SHARP APL on your mainframe and on your Personal Computers, you are assured of complete compatibility. When you get a new release of the mainframe software, it will also function correctly and compatibly on your PCs.

For more information about mainframe SHARP APL, contact your local I. P. Sharp representative. ■

INFOMAGIC

I. P. Sharp Associates is pleased to announce a new service for data base users—INFOMAGIC. Less experienced and casual data base users can now obtain useful reports, at a predictable price, with little or no special training. More experienced users can use INFOMAGIC for many common data base problems, such as searching through a directory for a particular series code.

INFOMAGIC is a conversational data base access system for quickly retrieving standard reports from the public data bases available on the I. P. Sharp timesharing service.

The hourly rate schedule for INFOMAGIC takes the guesswork out of the cost for a particular standard report. In North America, for example, the rates are:

\$60/hour at 300 baud (30 characters per second)

\$120/hour at 1200 baud (120 characters per second)

These rates include all telecommunication charges, but do not include any applicable data base royalties. However, there are no royalty fees or surcharges associated with usage for most I. P. Sharp data bases.

INFOMAGIC is well suited to 80-character screens, such as those available with most microcomputers. INFOMAGIC operates in conjunction with MICROCOMM, I. P. Sharp's communications diskette for the IBM Personal Computer.

As a data base user on the I. P. Sharp sys-

tem, you have access to one of the world's largest collections of time series—over 45 million in all. Until now, you have been able to take advantage of this data in three ways (in decreasing degrees of difficulty):

- 1) By using APL and a data retrieval language such as RETRIEVE
- 2) By using MAGIC and SUPERPLOT to retrieve and analyse the data. This method has proved to be immensely popular, but it can still be difficult for the casual user to master.
- 3) By asking your local I. P. Sharp representative to prepare the report for you, at standard consulting rates

Now you can access the data with the easy-to-use INFOMAGIC system. A subset of the data bases is available through INFOMAGIC, but the number of data bases and reports available for each data base is increasing daily.

If you have any reports that you think should be included in INFOMAGIC, please give your suggestions to your I. P. Sharp representative.

If you wish to take advantage of INFOMAGIC's fixed hourly rate structure, you must first be registered as an INFOMAGIC user. Contact your local I. P. Sharp office for further information, and for a copy of the INFOMAGIC contract.

The following example shows a session with INFOMAGIC retrieving reports from the CANSIM data base. (CANSIM is a registered trademark of Statistics Canada.) ■

```

)LOAD 39 INFOMAGIC
SAVED 15.51.42 06/01/83

WELCOME TO INFOMAGIC. ENTER DESCRIBE FOR DOCUMENTATION.

ENTER DATABASE: CANSIM
*** CANSIM DATABASE ***

ENTER CATEGORY NUMBER(S) OR REPORT NUMBER(S): HELP
THESE CATEGORIES AND REPORT NUMBERS ARE AVAILABLE FOR THE CANSIM DATABASE.

      CATEGORIES                                REPORT NUMBERS
1. ECONOMIC INDICATORS AND FORECASTS - 101 TO 103
2. LABOUR STATISTICS - 201 TO 208
3. NATIONAL ACCOUNTS - 301 TO 306
4. CONSUMER PRICE INDEX - 401 TO 417
5. FUEL POWER AND MINING - 501 TO 503

ENTER CATEGORY NUMBER(S) OR REPORT NUMBER(S): 208

```


... INFOMAGIC

29 JUL 1983

WAGES AND SALARIES BY INDUSTRY
BASED ON THE 1960 STANDARD INDUSTRIAL CLASSIFICATION
(MILLIONS OF DOLLARS)

	LATEST		PERCENT		PERCENT
	AVAILABLE		CHANGE		CHANGE
	MONTH:	PREVIOUS	FROM	A	FROM A
	MAR 83	MONTH	MONTH	YEAR AGO	YEAR AGO

RAW

AGRICULTURE (D5249)	107.7	91.3	17.9	101.5	6.1
FORESTRY (D5252)	121.3	125.6	-3.4	121.0	0.2
MINING (D5253)	453.3	448.1	1.1	520.2	-12.9
MANUFACTURING (D5254)	3275.9	3215.3	1.9	3290.3	-0.4
CONSTRUCTION (D5255)	792.7	796.7	-0.5	919.7	-13.8
TRANS/COMM/UTILITIES (D5256)	1822.7	1786.0	2.1	1719.3	6.0
TRADE (D5257)	1929.5	1912.9	0.9	1924.4	0.3
FIN./INS./REAL ESTATE (D5260)	1186.9	1176.6	0.9	1138.0	4.3
SERVICE (D5261)	4501.6	4333.0	3.9	4254.0	5.8
PUBLIC ADMIN./DEFENCE (D5266)	1627.9	1578.5	3.1	1488.6	9.4
TOTAL WAGES / SALARIES (D5248)	15819.4	15464.1	2.3	15477.1	2.2
SUPPL. LABOUR INCOME (D5271)	1680.4	1639.7	2.5	1547.0	8.6
TOTAL LABOUR INCOME (D5247)	17499.9	17103.9	2.3	17024.1	2.8

SEASONALLY ADJUSTED

AGRICULTURE (D5274)	158.9	158.9	0.0	149.2	6.5
FORESTRY (D5277)	139.7	142.9	-2.2	139.8	0.0
MINING (D5278)	452.5	448.1	1.0	519.3	-12.9
MANUFACTURING (D5279)	3299.7	3266.0	1.0	3315.0	-0.5
CONSTRUCTION (D5280)	917.1	940.5	-2.5	1062.3	-13.7
TRANS/COMM/UTILITIES (D5281)	1863.0	1822.9	2.2	1757.9	6.0
TRADE (D5282)	1963.3	1954.0	0.5	1958.0	0.3
FIN./INS./REAL ESTATE (D5285)	1172.2	1172.7	0.0	1123.8	4.3
SERVICE (D5286)	4482.7	4331.0	3.5	4209.9	6.5
PUBLIC ADMIN./DEFENCE (D5291)	1654.6	1611.9	2.6	1512.5	9.4
TOTAL WAGES / SALARIES (D5273)	16103.7	15848.7	1.6	15747.6	2.3
SUPPL. LABOUR INCOME (D5296)	1709.9	1679.4	1.8	1573.8	8.6
TOTAL LABOUR INCOME (D5272)	17813.5	17528.1	1.6	17321.4	2.8

SOURCE: STATISTICS CANADA CANSIM DATABASE

NOTE: 1. AGRICULTURE INCLUDES FISHING AND TRAPPING

2. PUBLIC ADMIN./DEFENCE EXCLUDES MILITARY PAY AND ALLOWANCES

SES Marketscan: A New Global Information Service

SES Marketscan, which provides data on 350 companies listed on the Stock Exchange of Singapore (SES), is now available on the I. P. Sharp timesharing service. SES has appointed Singapore International Software Services Pte. Ltd. (SISS) and affiliated I. P. Sharp companies and agents, as the exchange's exclusive agent to distribute Marketscan outside Singapore.

The SES Marketscan data base comprises two files—the historical stock price file and the corporate statistics file. The stock price file is updated daily by the SES and dates back to 1975. The corporate file, dating back to 1973, is updated monthly by the National University of Singapore's Faculty of Accountancy and Business Administration.

The system is menu-driven. You simply sign on to the I. P. Sharp timesharing service, load the workspace, choose a report type, and give a stock/company code.

SES introduced Marketscan on 16 June 1983 at its tenth anniversary celebrations, with 600 financial and banking executives in attendance. SES member brokers and banks and financial institutions have had access to the service since 1 August.

To use SES Marketscan, there is an initiation fee of S\$1 000 (approximately US\$500), and an annual charge of the same amount. For more information on this service, contact your local I. P. Sharp representative.

The Singapore stock market has daily trading volumes that range up to 20 million units, with trading values of US\$25-million.

During its ten-year operating history, SES has provided a solid framework for the raising of capital for the expansion of commerce and industry. A total of US\$3.6-billion has been raised from the investing public. ■

Walter Keirstead, Singapore

Citibase Economic Forecast Data Base

The Citibase Economic Forecast (CITIFORECAST) data base contains time series data for over 230 U.S. economic indicators. The data is supplied by the Economics Department of Citibank, which is based in New York City.

Data is currently available quarterly, from the beginning of 1983 through to the end of 1988. Quarterly updates to the data base will keep extending it five years into the future. Among the series available are breakdowns of:

- Balance of payments
- Energy consumptions and costs
- Interest rates
- Personal consumption expenditures
- Implicit price deflators
- Government expenditures
- Industrial productions

Each series in the data base has a corresponding historical time series available in the CITIBASE data base. You can use the CITIFORECAST access function to retrieve these historical series, and combine them with the forecast data to produce a single long-term time series.

The data is available through the workspace 39 *MAGIC* by subscription only. The subscription fee is US\$2 500 annually, or US\$1 000 for a quarter trial period.

For more information on the data base, or for a directory of available series, load the workspace 39 *MAGIC* and type:

```
CITIFORECAST 'DESCRIBE'  
or CITIFORECAST 'DIRECTORY'
```

You do not need to be a subscriber to access the online description or the directory.

To obtain a contract for a subscription, type:

```
CITIFORECAST 'CONTRACT'
```

in 39 *MAGIC*, and a contract will be printed at your terminal. You may then fill out the contract, and send it to your nearest I. P. Sharp representative.

... CITIFORECAST

For further information on the CITIFORECAST data base, contact your local I.P. Sharp representative or direct your questions to the MAILBOX group, CITIQ.

The following example shows forecast income and employment figures for 1983 and 1984, using data from the CITIFORECAST data base. ■

John Larmour, Toronto

INCOME AND EMPLOYMENT

	TOTAL PERSONAL INCOME	DISPOSABLE PERSONAL INCOME	TOTAL CIVILIAN LABOR FORCE	UNEMPLOYED CIVILIAN LABOR FORCE
1ST/83	2,648	2,256	110,528	11,439
2ND/83	2,699	2,303	111,156	11,222
3RD/83	2,767	2,363	111,130	10,700
4TH/83	2,836	2,425	111,330	10,200
1ST/84	2,879	2,464	111,700	10,100
2ND/84	2,946	2,517	112,110	9,900
3RD/84	3,013	2,571	112,440	9,600
4TH/84	3,080	2,623	112,740	9,200

PERSONAL INCOME IS EXPRESSED IN BILLIONS OF CURRENT DOLLARS.
LABOR FORCE IS EXPRESSED IN THOUSANDS.

New U.S. Banks Data Base

The USBANKS data base contains balance sheet and income statement data for over 15 000 commercial banks in the United States. With the comprehensive scope of this data base, you can do a detailed analysis of individual banks as well as peer group comparisons in such areas as market share, profitability, asset and liability structure, and capitalization. Analysts in both the investment and banking communities will find this new data base extremely useful.

The balance sheet and income statement data is taken from the Consolidated Report of Condition (Call Report), the Consolidated Report of Income, and the Quarterly Report of International Banking Facility Accounts as filed with the Federal Deposit Insurance Corporation (FDIC), each calendar quarter, by all commercial banks. The data is in thousands of dollars and is available for the most recent eight quarters.

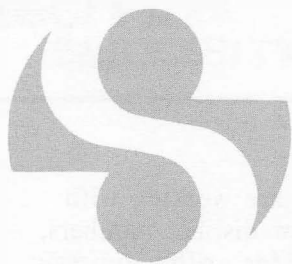
In addition to the FDIC data, USBANKS also contains 60 key ratios for each bank in the data base. Ratios are stored with three implied decimal points and are available as time series for the same timeframe as the FDIC data.

USBANKS also contains 30 static facts for each bank including such items as the bank name, city, county, state, Standard Metropolitan Statistical Area (SMSA) code, charter class code, and FDIC certificate number.

FDIC data is released by the U.S. Federal Reserve Board in both preliminary and final versions. The preliminary data is available 90 to 120 days after the end of the relevant quarter, while final data is released four to six months after the quarter end. There is no surcharge for accessing the data.

USBANKS is updated with the preliminary and final data as soon as the data is released. To determine the latest data available, load the workspace 39 *MAGIC* and type:

TIMECHECK 'USBANKS'



I. P. Sharp

newsletter

SEPTEMBER/OCTOBER 1983

Volume 11 Number 5

technical supplement 46

Making Things Fit, Part 2

In the first part of this series, we considered the problems involved in fitting non-rectangular data into rectangular arrays. In this article, we will consider methods for fitting large amounts of data in finite workspaces.

Making things fit in finite workspaces

APL systems manage memory in two basic ways. Some systems provide fixed size workspaces. When such a workspace is being executed by an APL language processor, it is all in real memory. Other systems provide workspaces whose size can vary up to some limit. When such a workspace is being executed by an APL language processor, only parts of it are in real memory; most of it will be stored on disk.

Fixed workspace sizes typically range from 100 000 to 500 000 bytes. Variable workspace sizes typically range from 500 000 to 8 000 000 bytes. (SHARP APL products can provide either type of environment.) Though the second is larger than the first, both are finite: given enough data or programs, either can be filled with data. So, in the end, most APL programmers have to contend with fitting things in finite workspaces. There are a couple of APL implementations that allow extremely large workspaces, but these implementations limit other concurrent users, and degrade performance.

How do we deal with the problem of finite workspace size? The first part of this series

suggested the following strategies for making things fit:

- 1) Volume alteration
- 2) Segmentation
- 3) Prestidigitation
- 4) Dimension distortion

We can use the first three of these to make things fit in finite workspaces.

Volume alteration

How can we alter the volume of objects in the active workspace? There are several means.

One of the chief causes of *WS FULL* is objects which have been promoted from one internal representation to another. Boolean numbers, which are normally stored using just one bit, are sometimes converted to the fixed-point representation, which takes four bytes (32 bits). Or integers, which are normally stored using the fixed-point representation, are sometimes converted to the floating-point representation, which takes eight bytes (64 bits).

When do such promotions occur? Arithmetic operations (like $+$ $-$ \div) on boolean numbers will promote them. Or if any of the elements become non-boolean, the array will be promoted. Integers can be promoted to the floating-point representation either because at least one of the elements has a fractional part, or at least one of the elements is too large to store in the four-byte fixed-point format. Any integer which is greater than 2^{31} or less than -2^{31} will be stored in eight-byte floating-point form, even though it is an integer.

... Making Things Fit

If an array has been promoted, how can you demote it? First, you must remove the offending elements, if there are any. Then you can apply the following function to the array:

```

∇R←DEMOTE X
[1] R←(x/ρX)ρ0 ∘ R[]←,X ∘ R←(ρX)ρR ∇

```

This function demotes arrays because indexed assignment tries to preserve the data type of the array being indexed. The function creates a boolean array of the same shape as the argument. Then it inserts all of the old elements into the new array. If they are not of the same data type, indexed assignment will attempt to convert them to that type. If the conversion is possible, the new array will have the same values, in a smaller representation. If it is not possible, the array being indexed will be promoted to the necessary form.

As a more expensive alternative, you could test to see which is the smallest possible representation of a number, and use expressions like $X←X=1$ (convert to one-bit booleans), $X←⌊X$ (convert floating-point to fixed-point), and $X←X⌊X$ (convert complex to real).

A second way to alter the volume of arrays in the workspace is to use joint representation efficiently. SHARP APL keeps only a single copy of identical arrays in certain circumstances. If you do $A←15 ∘ B←1 2 3 4 5$, you will have two copies. But if you do $A←15 ∘ B←,A$, you will have only one copy. The chief benefits of this scheme are that arguments are passed as pointers (no copies are made), and enclosed arrays can take less space. If you know that some of the elements of an enclosed array may be identical, and you want to make sure that only a single copy is stored, apply the following function:

```

∇R←JOINT X;∅CT;T
[1] ∅CT←0 ∘ T←,X ∘ R←T[T1X] ∇

```

This function only condenses the array at the first level. It would have to be applied recursively, if there are multiply enclosed objects in the argument.

A third approach to reducing the volume of an array is to take into consideration how the

data will be used. If you are working with many occurrences of a few distinct numbers, which are not being used for arithmetic, you can store the numbers as characters. Each character represents the index of the desired value in a special vector. Only 256 distinct numbers can be represented using this scheme, but the space requirements are reduced at least four times. The following functions convert between numeric values and the character substitutes.

```

∇N←REALNUM C
[1] N←NUMBERS[∅AV1C] ∇

∇C←PSEUDONUM N
[1] C←∅AV[ NUMBERS1N] ∇

```

If you are enclosing literal vectors representing words with 11 characters or less, you can save space by encrypting the words as numbers instead. Every uniquely enclosed literal vector takes at least 28 bytes to store. Words of six or fewer letters (using an alphabet of 34 or fewer letters) can be encrypted into four-byte fixed-point numbers. Words of 11 or fewer letters can be encrypted into eight-byte floating-point numbers using the same alphabet. See *Technical Supplement 38*, May/June 1982, for more information on encrypting character arrays.

If you are using integers to represent choices (such as those made on a questionnaire), you can also reduce the volume by representing the numbers as boolean vectors. Each boolean vector will have as many elements as there are possible choices. If an element of the vector is 1, then the position of the 1 is the choice that was made. If there are eight possible choices for each question, storing the responses in boolean form will take one-quarter as much space as storing the choices as integers. This boolean form will also make it much easier to crosstabulate the results of the questions.

If you want to reduce the volume of defined functions in the active workspace, you can do any of the following:

- 1) Put more than one statement on a line using \diamond . This saves 12 bytes per line eliminated, in SHARP APL.
- 2) Remove comments.

... Making Things Fit

- 3) Reduce all names to three or fewer letters. This saves $(12 + \text{length of name})$ bytes per unique name, in SHARP APL.
- 4) Standardize all local names used in functions. This will save eight bytes in the symbol table per unique name eliminated, in SHARP APL. You can regain this space by reducing the size of the symbol table.
- 5) Store frequently used constants in variables. Each occurrence replaced by a variable will save approximately as many bytes as the number of characters needed to display the constant. Of course, the copy in the variable will require its own space.
- 6) Put sections of code common to several functions in separate subroutines.

Except for the last two methods, changing programs as suggested here reduces their maintainability. If you must use these methods, save a readable maintenance copy. Then run a program to condense your code. You will probably want to lock the condensed versions to prevent doing maintenance work on the production copy.

Segmentation

How can we segment data so that it fits into a finite workspace? The most common way is to take a data structure which is too large, split it into parts, store the parts in a file, and process the separate components by looping. The next article in this series will address fitting things into files, and so we will postpone considering this approach for now. There are also ways of splitting data in the workspace in order to make things fit.

When the amount of space or time required to process any array grows faster than linearly, the process may be a candidate for segmentation. An example is the function *PARTNENCL* listed in *Technical Supplement 39*, July/August 1981 (or see workspace 777 *SEARCH*). The amount of space required grows on the order of $.5 \times N \times 2$. Therefore it loops, processing part of the argument on each iteration. This ensures that a *WS FULL* doesn't occur, and also minimizes CPU usage.

Another example of this approach is the pair of functions for taking n things from a set of m things (combinations), which appeared in *Technical Supplement 31*, March/April 1981. One program was fast for small results, but used a lot of storage, and reached *WS FULL* quickly. The other program was slower, but could handle larger results. It broke up the process into pieces by using recursion. Because it segmented the problem, the program saved space in the active workspace.

Prestidigitation

How can we make more information appear to be in the workspace than is actually there? There are two approaches. The first is not to store certain types of data, but merely an indication of its presence or absence. The second is to move information between workspace and files, ensuring that the data is always in the workspace when you need it.

Certain types of data contain lots of redundant information. A text containing lots of blanks, or a matrix containing many zeros are common examples. It is possible to store such information in a condensed form, in order to make more things fit in a finite workspace. The two functions listed below show one way to do this. They only reduce the storage required for blanks and zeros. The result of the first function is an enclosed array which contains the following:

- 1) A boolean array which is the same shape as the original array. 1's mark the positions of non-fill elements, 0's mark the positions of fill elements.
- 2) A data vector containing the non-fill elements.

If the argument to *CONDENSE1* is:

```
1 0 0 2
0 0 0 3
0 0 0 0
4 0 0 0
```

... Making Things Fit

the result will contain the following two arrays:

```
1 0 0 1      1 2 3 4
0 0 0 1
0 0 0 0
1 0 0 0
```

Note the use of the link (>) function to enclose and concatenate the parts of the result. The second function restores the original array.

```
▽ SMALL←CONDENSE1 BIG;FILL
[1] FILL←BIG*1+0ρBIG
[2] SMALL←FILL>(>FILL)/,BIG ▽

▽ BIG←RESTORE1 SMALL
[1] BIG←(ρ>SMALL[1])ρ(>SMALL[1])\>SMALL[2] ▽
```

The next two functions deal with the problem of duplicate data in a different way. They will reduce multiple consecutive occurrences of any element to a single occurrence. The result of the first function is an enclosed array which contains the following:

- 1) A boolean array which marks groups of consecutive occurrences of the elements
- 2) Single copies of the repeated data elements

If the argument to *CONDENSE2* is:

```
AABC
CCDD
DDEF
FFGG
```

then the result will contain the following arrays:

```
1 0 1 1      'ABCDEFGF'
0 0 1 0
0 0 1 1
0 0 1 0
```

The *RESTORE2* function takes the condensed version and returns the original.

```
▽ SMALL←CONDENSE2 BIG;UNIQUE;SHAPE
[1] SHAPE←ρBIG
[2] BIG←,BIG
[3] UNIQUE←~1+1,BIG≠1φBIG
[4] SMALL←(SHAPEρUNIQUE)>UNIQUE/BIG ▽

▽ BIG←RESTORE2 SMALL;UNIQUE
[1] UNIQUE←>SMALL[1]
[2] BIG←(>SMALL[2])(ρUNIQUE)ρ+\,UNIQUE ▽
```

These first two approaches to handling sparse data make it possible to reduce the amount of space needed to store sparse arrays when they aren't being used. Such condensed arrays can be kept in the workspace, or stored on file in condensed form to conserve disk storage.

But what if your sparse array won't fit in the workspace in its full form? The next technique can even handle sparse arrays of this type. The data structure used in this method stores the indices and values of the non-zero elements in a matrix. This matrix has as many rows as non-zero elements. It has one column for each dimension of the array being represented, plus one for the data values. The first row contains the shape and rank of the array being represented. If the matrix listed on the left were stored in this form, it would be represented by the matrix on the right. In this case, there aren't enough zeros to save any storage.

10	0	0	20	5	4	2
0	30	0	0	1	1	10
0	0	0	0	1	4	20
40	0	50	0	2	2	30
0	0	0	0	4	1	40
				4	3	50

Since the entire array cannot be directly manipulated, functions are needed for doing indexed fetch and store. The functions listed below will fetch and store either rows or columns of a sparse matrix stored in this form. In each case, the left argument is the representation of the sparse matrix as described above. The right argument contains the axis being processed (row or column), the index along that axis, and for a store operation, the data values to store.

... Making Things Fit

```

▽ ROW←SPARSE FETCH AXISΔINDEX
;SELECT;EXPAND;AXIS;INDEX;OTHER
[1] AXIS←AXISΔINDEX[1]
[2] INDEX←AXISΔINDEX[2]
[3] OTHER←2 1;AXIS
[4] SELECT←0,1+SPARSE[;AXIS]=INDEX
[5] EXPAND←(1;SPARSE[1;OTHER])←SELECT/SPARSE[;OTHER]
[6] ROW←EXPAND\SELECT/SPARSE[;3] ▽

▽ SPARSE←SPARSE STORE AXISΔINDEXΔVALUES
;SELECT;NEW;COMPRESS;AXIS;INDEX;VALUES
[1] AXIS←AXISΔINDEXΔVALUES[1]
[2] INDEX←AXISΔINDEXΔVALUES[2]
[3] VALUES←2+AXISΔINDEXΔVALUES
[4] COMPRESS←0+VALUES
[5] NEW←COMPRESS+(AXIS=2)ΦINDEX,[1.5] 1;COMPRESS),VALUES
[6] SELECT←1,1+SPARSE[;AXIS]=INDEX
[7] SPARSE←(SELECT/SPARSE),[1] NEW ▽

```

If the matrix listed above were assigned to *MAT*, these functions would work as follows:

```

MAT FETCH 1,4 A MAT[4;]
40 0 50 0
MAT←MAT STORE 2,1,0 0 0 99 0 A MAT[;1]←0 0 0 99 0

```

Functions to assign individual elements can be created by calling these two functions, or they can be written directly. Functions to do scalar operations (like addition) can be coded to operate efficiently by grouping the values in the arguments in three categories: non-zero value only on the left, non-zero value only on the right, both values non-zero. Functions to do reductions and scans are implemented without looping, by sorting the argument along the axis to be processed, and then applying the appropriate partitioned reduction or scan.

The second approach to making more information appear to be in the workspace than is actually there involves moving information back and forth between workspaces and files. This is the idea behind "virtual memory" operating systems. This approach can be used at the application level as well.

There are two ways to move information between workspaces and files. With the first method, your programs must explicitly retrieve functions or variables they need from file. In the second, the required objects are implicitly retrieved by an event trap when they are referred to. While the second method is only easily accomplished on SHARP APL, the functions which are called implicitly under this scheme can be used explicitly in the other. Therefore, we will only consider the second method because it requires less work by the programmer.

When you use SHARP APL event trapping, one of the possible responses to an event is to execute an APL statement, and then resume executing the original statement *at the point at which the event occurred*. The trap listed below will call *GETOBJECT* with an argument of the name of the missing object whenever a *VALUE ERROR* occurs. If *GETOBJECT* defines the missing object, the original statement will resume precisely where it left off. (If you want to understand how it works, read *SATN-44*.)

```

[TRAP←'▽ 6 I GETOBJECT [ER[4;]] '

```

How does *GETOBJECT* work? It calls subroutines to find, retrieve, and log the needed objects.

```

▽ GETOBJECT NAME
[1] NAME FETCHOBJECT NAME FINDOBJECT NAME ▽

```

As the name suggests, *FINDOBJECT* locates the needed object and returns the file tie and component number where it is stored. *FETCHOBJECT* reads the component from file, and defines the object into the workspace, and records the fact that it is now in the workspace.

```

▽ LOCATION←FINDOBJECT NAME;ROW
[1] ROW←INDEX[;1]1<(NAME≠' ')/NAME
[2] LOCATION←ROW,>INDEX[ROW;2] ▽

▽ NAME FETCHOBJECT LOCATION
[1] NAME [PDEF [READ FFILE,LOCATION[2]
[2] INDEX[LOCATION[1;3]←<1 ▽

```

FINDOBJECT assumes a directory of objects on file (*INDEX*) is brought into the workspace when it is loaded. The first column of the index has enclosed names of objects, and the second has the corresponding component number. The third column indicates whether the object is in the active workspace.

There are many useful variations on these functions. *FINDOBJECT* could compute the location of function or variable by transforming the name into a number, rather than looking up the name. This approach, known as **hashing**, can be quicker than a table lookup. Since many names might be transformed into the same number, hashing could also be

... Making Things Fit

slower for retrieval, since many unwanted functions might be read in. If the lookup approach were kept, *FETCHOBJECT* could be changed to bring in all of the subroutines and variables needed by that function, all in a single operation. All you need to do is store all of these related objects in the same package, and remove the left argument to *⊠PDEF* in *FETCHOBJECT*.

What happens when the workspace has been filled with functions and variables brought in from file? *WS FULL*! But we can take care of that. Since we have been recording the names of objects brought in, we can call a function to remove them, if we trap the *WS FULL*. The function listed below does just that.

```
⊠TRAP←⊠TRAP,'∇ 1 E MAKESPACE ◇ →⊠LC'  
∇ MAKESPACE;⊠TRAP;⊠PS;SPACE;TEMP  
[1] ⊠PS← 0 0 1 1  
[2] ⊠TRAP←'∇ S' A PREVENT ENDLESS LOOP  
[3] SPACE←4 [WS,∇INDEX[:1]  
[4] TEMP←(50000≤+SPACE)1  
[5] SPACE←6 [FD,∇INDEX[TEMP;1]  
[6] INDEX[:3]←(TEMP<0),TEMP+INDEX[:3] ∇
```

This function will only remove as many objects as are necessary to create 50 000 bytes of free space.

Memory management has always been an important programming technique to make applications work. By using the techniques described in this article, you can make your applications more effective by making your functions and variables fit in the workspace. ■

Robert Metzger, Rochester

... USBANKS

You can access the data through MAGIC or RETRIEVE, and you can obtain an online description by typing:

USBANKS 'DESCRIBE' in 39 MAGIC
or ΔVIEWDOC 'USBANKS' in 55 RETRIEVE

To obtain a directory of facts and banks, type:

USBANKS 'DIRECTORY' in 39 MAGIC
or ΔDIRECTORY 'USBANKS' in
55 RETRIEVE

By using the screening facility of USBANKS, you can select the banks meeting your specified criteria on any of the over 900 facts in the data base. Then you have the particular

facts of interest for these banks for immediate examination, further analysis, or graphic display.

In the following example, the constraint screens for banks with fact 2 (city name) equal to Chicago, fact 31270 (consolidated total assets) greater than \$50-million, and fact 505 (ratio of municipal securities to total assets) greater than .2. Note that since dollar values are stored in thousands, \$50-million is represented as 50000. A detailed description of the screening facility is given in the online description.

For further information on the USBANKS data base, contact your local I. P. Sharp representative or direct your questions to the MAILBOX group, MONQ. ■

Chris Taylor, Toronto

RESETOPTIONS

QUARTERLY DATED AT 2 82 4 82

DECIMALS 3 ◇ AUTOLABEL

CONSTRAINT+ '(F2 EQ CHICAGO) AND (F32170 GT 50000) AND (F505 GT .2)'
'H' DISPLAY CONSTRAINT USBANKS 32170 505 502 503

2 FOUND. PROCEED? (Y/N): Y

	2ND/82	4TH/82
COSMOPOLITAN NB OF CHICAGO		
CON TOTAL ASSETS	69,021	71,479
MUNICIPAL SECURITIES ÷ TOTAL ASSETS	0.239	0.219
PRE-TAX INCOME ÷ TOTAL ASSETS	0.014	0.024
NET INCOME ÷ TOTAL ASSETS	0.011	0.020
LINCOLN NATIONAL BANK		
CON TOTAL ASSETS	174,454	185,415
MUNICIPAL SECURITIES ÷ TOTAL ASSETS	0.245	0.221
PRE-TAX INCOME ÷ TOTAL ASSETS	0.005	0.010
NET INCOME ÷ TOTAL ASSETS	0.007	0.013

The MEDEX Insurance and Medical Information System

If you have an accident or become ill when travelling abroad, chances are that MEDEX and I. P. Sharp will help you. MEDEX International Corporation provides a worldwide travel assistance and claims service to insurance companies. With the appropriate travel insurance coverage, you can contact MEDEX directly or through your own insurance company. MEDEX will recommend the best medical care, pay your incurred emergency medical expenses, and arrange air evacuation and repatriation, if necessary. In some cases, particularly for medical emergencies in remote parts of the world, MEDEX will have a specialist (in the United Kingdom or the United States, for example) "talk" to the local doctors on the spot via MEDCOM.

Where does I. P. Sharp come in? MEDEX offices around the world exchange insurance and medical information for about 10 million people each year—and it must be accurate. To maintain this information, MEDEX has developed MEDCOM, a data file management system with integral communications capabilities. The system uses the I. P. Sharp international packet-switched network and a SHARP APL program, EASYMAIL, developed in I. P. Sharp's London, England, office to link all MEDEX offices.

Twenty-four locations now use the MEDCOM system which runs on minicomputers. Each office collects the medical and insurance data and organizes it on a minicomputer, using MEDCOM's file management and report generation facilities. Then a MEDEX assist-

ance operator transmits the information to selected destinations. MEDEX uses an autodial system for automatic dial-up and sign-on by the minicomputer to the I. P. Sharp Network via local dial-up or leased line connection. Teletype or telex terminals can also be used to access the network.

A special option allows new or revised computer programs for the minis to be sent out for quick and thorough system updates.

Although it was originally developed for medical purposes, MEDCOM is a general package and can be used for many other applications. Louis Russell, vice-president of Operations, says, "We're finding so many uses for MEDCOM, and it's bringing our costs down considerably. It is turning into a worldwide management control system." MEDEX is now transmitting other types of corporate information via MEDCOM, such as marketing information, confidential corporate information, and financial information from various branches to the Brighton, England, office for consolidation.

For more details on the MEDCOM system, contact MEDEX at:

MEDEX International Corporation
13773 North Central Expressway
Suite 1102
Dallas, Texas 75243

Telephone: 214 661-1420

For information on the I. P. Sharp facilities, please contact your local I. P. Sharp representative. ■

Margaret Joachim, London, England

Third Party Software

With the increased use of SHARP APL, an impressive library of **third party software** has developed; that is, software specifically designed, or adapted, to run in a SHARP APL environment—but not written, supported, or warranted by I. P. Sharp personnel. Many of the third party software packages evolved from non-proprietary applications software written by customers of the I. P. Sharp timesharing service, or at customer sites running SHARP APL on their own in-house mainframes, where the author of the software saw the potential for more generalized use.

In order to make this software available to other SHARP APL users, I. P. Sharp has established a Third Party Software Library for distribution of these packages. Typically, the author of the software will continue to own, maintain, and document the package. The I. P. Sharp timesharing service will simply be a delivery mechanism.

As applications packages are added to the Third Party Software Library, they will be featured in the *I. P. Sharp Newsletter*. The following article describes one such package, ESTAB. Eventually, all the packages will be documented in a Third Party Software catalogue.

If you have an applications package you would like considered for inclusion in the Third Party Software Library, contact Lib Gibson, manager of Applications Software, at I. P. Sharp's international headquarters. ■

Electronic Spreadsheet Tablet: ESTAB

Are you currently using ledger sheets or columnar worksheets to develop financial plans and analyse operating results manually? Now you can use the Electronic Spreadsheet Tablet program, ESTAB, to help you organize, record, and present related time series. You will find this general-purpose program useful for budget and market planning, and analysis.

ESTAB has been designed by Wm. R. Gerathy Associates of Milford, Connecticut, for financial planning and control, and for any situation in which you must organize and record related numeric data for successive events or calendar periods. You do not need prior experience in APL programming or knowledge of computer concepts to make effective use of ESTAB.

System concepts

Imagine a blank spreadsheet on which you can define up to 366 data columns and an unlimited number of lines. Each spreadsheet can be categorized as either **temporary** or **filed**. You use a temporary spreadsheet to organize and report data that you won't need after the data is reported. You use a filed spreadsheet when you must save the data and access it later on.

You define the spreadsheet columns to represent successive relative events, such as manufacturing periods, or specific calendar periods such as days, months, or years.

Spreadsheet lines represent spreadsheet data and calculated values. You do not need to predefine spreadsheet lines, but can add them as required. You can identify and reference individual lines using one of two modes—a **relative-id** mode or a **coded-id** mode. When you use the relative-id mode, each line is assigned a sequence number which identifies the line and determines its position on the spreadsheet. In the coded-id mode, each line is assigned a descriptive alphanumeric code to identify the line, which you then position on the spreadsheet.

How to use ESTAB

ESTAB is a prompted system; you can type *HELP* at specified points for further information on how to proceed. The more experienced user can answer ahead by entering multiple successive responses to a single prompt.

In response to a *COMMAND* prompt, you select the operation you require from one of:

...ESTAB

)LOAD 6625 ESTAB

INITIALIZE
ADD
DELETE
UPDATE
CHANGE
LIST
REPORT

For further information and the *ESTAB*
Operating Instructions manual, please contact:

Wm. R. Gerathy Associates
861 Naugatuck Avenue
Milford, Connecticut 06460

Telephone: 203 878-4296

In addition to these commands, there are additional ones to assist in the management of spreadsheets and spreadsheet data.

ESTAB will soon be available to all I. P. Sharp timesharing customers. You will be able to use the program by entering:

In the following example, an annual spreadsheet spanning five years was created using ESTAB. ■

DATE: 08/11/83						PAGE 1
INCOME STATEMENT						
	<u>1983</u>	<u>1984</u>	<u>1985</u>	<u>1986</u>	<u>1987</u>	
<u>REVENUE:</u>						
PRODUCT SALES	22175	24100	25925	27105	29750	
RETURNS	554	602	648	678	744	
NET SALES	21621	23498	25277	26427	29006	
OTHER REVENUE	<u>783</u>	<u>649</u>	<u>709</u>	<u>721</u>	<u>828</u>	
TOTAL REVENUE	22404	24147	25986	27148	29834	
<u>EXPENSES:</u>						
COST OF SALES	10866	11809	12703	13281	14578	
SELLING EXPENSE	2120	2195	2250	2305	2880	
SG&A	6175	6280	6395	6460	6485	
DEPRECIATION	1745	1832	1924	2020	2121	
OTHER EXPENSES	<u>445</u>	<u>435</u>	<u>451</u>	<u>454</u>	<u>463</u>	
TOTAL EXPENSES	21351	22551	23723	24521	26527	
	=====	=====	=====	=====	=====	
NET PROFIT	1053	1595	2263	2628	3308	
NET PERCENT	4.7	6.6	8.7	9.7	11.1	

Workspace 1 INFO

Workspace 1 *INFO* is now the main reference source for up-to-date information about public workspaces on the I. P. Sharp timesharing service; it replaces 1 *PUBLICLIBS*. You can use the functions in *INFO* to list workspaces by library, to search for workspaces or functions by name or name fragment, or to describe a public workspace.

The main functions in *INFO* are:

LIB
FINDWS
FNS
FINDFN
WSWITH
DESCRIBEWS

Each of these is self-contained; that means you can copy it into other workspaces and use it alone. The data base accessed by these functions is updated daily.

LIB lists workspaces by library number. For example:

LIB 85 86
85 *APLSORT*
85 *MABRAUTIL*
86 *MABRA*
86 *MAXPAK*

FINDWS searches for a workspace by name. For example:

FINDWS 'MAGIC'
39 *MAGIC*
539 *MAGIC*

The asterisk is used with *FINDWS* to indicate any match, so that a search for *MAGIC** lists all workspaces whose names begin with *MAGIC*. A search for **MAGIC** lists all workspaces whose names contain the string *MAGIC*. For example:

FINDWS 'MAGIC*'
39 *MAGIC*
39 *MAGICSTORE*
539 *MAGIC*
539 *MAGICR*
539 *MAGICSTORE*

FINDWS '*MAGIC*'
39 *BAREMAGIC*
39 *INFOMAGIC*
39 *MAGIC*
39 *MAGICSTORE*
539 *MAGIC*
539 *MAGICR*
539 *MAGICSTORE*
539 *OLDMAGIC*
539 *TMAGIC*
585 *MABRAMAGIC*

FNS lists all the functions saved in a specified workspace. For example:

FNS '1 *INFO*'
DESCRIBEWS
FINDFN
FINDWS
FNS
LASTUPDATED
LIB
WSWITH

FINDFN searches the public workspaces for the specified function. For example:

FINDFN 'LIB'
LIB
LIBDOC
LIBRARIES
LIBRARY
LIBΔREPORT

WSWITH lists the public workspaces in which a specified function can be found. For example:

WSWITH 'LIB'
1 *INFO*

As with *FINDWS*, an asterisk following the workspace name can also be used to indicate a string search. However, *WSWITH* will *not* accept names *beginning* with an asterisk. For example:

WSWITH 'LIB*'
1 *INFO*
12 *SHAREDVARS*
503 *APS*
503 *APS2*

...1 INFO

DESCRIBEWS describes the specified workspace. For example:

```
DESCRIBEWS '3 SUPERPLOT'
```

```
3 SUPERPLOT
```

THIS WORKSPACE CONTAINS SUPERPLOT: A BUSINESS GRAPHICS PACKAGE THAT PRODUCES PLOTS ON A WIDE VARIETY OF PLOTTING DEVICES AND TERMINALS...

For a function to be found by *FNS*, *FINDFN*, or *WSWITH*, it must actually be saved in the workspace. That is, you are able to use *)COPY* to copy the function.

All of *INFO*'s functions return an explicit result, and they can be combined into meaningful expressions. For example:

```
DESCRIBEWS FINDWS '*PLOT*'
```

describes every workspace whose name includes *PLOT*.

For more information on the workspace 1 *INFO*, load 1 *INFO* and enter *DESCRIBE*.

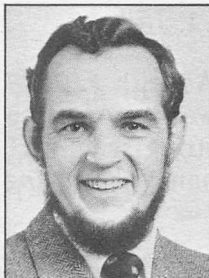
This workspace forms part of the public library maintenance system. If you have any questions or comments, contact the system librarian at I. P. Sharp's international headquarters in Toronto or *SYSL* in the I. P. Sharp MAILBOX. ■

Chris Burke, Toronto

**bulletin
board**

Toronto

Mike Powell has been appointed manager of Microcomputer Products and is busy coordinating the implementation of SHARP APL on the IBM Personal Computer. He joined I. P. Sharp in 1977 and was branch manager in Victoria until his recent appointment.



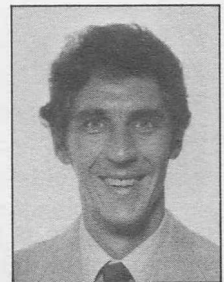
Mike is a graduate of Cambridge University and the University of Toronto. ■

Paris

Malcolm Hyde, formerly technical manager of the Paris office (*I. P. Sharp Newsletter*, March/April 1982), has assumed responsibilities for the European Data Base Group. Malcolm remains based in Paris. ■

Atlanta

Alec Nicholls has been appointed branch manager of the Atlanta office. He emigrated from England to the United States in 1977. He earned his B.S. degree in accounting and an M.B.A. in finance at the State University of New York at Binghamton.



Upon completion of his studies, Alec joined STSC, Inc., in Rochester. Then he joined I. P. Sharp's Rochester office where he was involved with conversions and MABRA projects. He has been with the Atlanta office since January of this year. ■

Chicago

Larry Mysz is the new branch manager in Chicago. He joined I. P. Sharp in June 1982, as marketing manager for Chicago.

His long association with APL began at Science Research Associates. There he designed course materials in chemistry and physics for use in high schools and colleges.

Larry then opened an APL service bureau in Chicago, where for 14 years he was involved extensively with a wide variety of time-sharing applications. ■

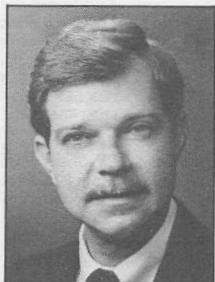
Houston

Ken Bell, formerly branch manager of the Atlanta office (*I. P. Sharp Newsletter*, November/December 1982), has been appointed branch manager in Houston. ■

Los Angeles

Ralph Morrison is the new branch manager in Los Angeles. A native of Canada, he earned a bachelor of mathematics degree from the University of Waterloo. Ralph joined I. P. Sharp in 1969, the year the company started offering APL services.

During his 14 years of APL experience, Ralph has been involved with programming



and analysis, consulting, marketing, customer support, education and management in the Toronto, Ottawa, and San Francisco branch offices. ■

Newport Beach

Barbara Siebert, formerly branch manager of the Rochester office, (*I. P. Sharp Newsletter*, January/February 1983), has been appointed branch manager in Newport Beach. ■

Rochester

Bob Campbell, formerly marketing manager in Rochester, (*I. P. Sharp Newsletter*, November/December 1982), has been appointed branch manager of the Rochester office. ■

mailing request

- ☐ Please amend my mailing address as indicated.
- ☐ Please add the following name(s) to your Newsletter mailing list.
- ☐ Please send me a Publications Order Form.
- ☐ Please add my name to the Aviation Newsletter mailing list.
- ☐ Please add my name to the Energy Newsletter mailing list.
- ☐ Please add my name to the Financial and Economic Newsletter mailing list.
- ☐ Please add my name to the PROMIS Newsletter mailing list. PROMIS is a computer-aided manufacturing system for microelectronics.

Name: _____

Title: _____

Company: _____

Address: _____

☐ Please send me information about your courses in:

City: _____ Tel.: _____

The *I.P. Sharp Newsletter* is published by I.P. Sharp Associates, Suite 1900, 2 First Canadian Place, Toronto, Canada M5X 1E3.
Your comments and contributions are welcome.

Editor: Irene Shimoda
Circulation: Mary Kopfensteiner
Printed in Canada
September 1983
ISSN 0226 854X

Aberdeen

I.P. Sharp Associates Limited
5 Bon Accord Crescent
Aberdeen AB1 2DH
Scotland
(0224) 25298

Amsterdam

InterSystems BV
Kabelweg 47
1014 BA Amsterdam
The Netherlands
(020) 86 80 11
Telex: 18795 ITS NL

Atlanta

I.P. Sharp Associates, Inc.
1210 S. Omni International
Atlanta, Georgia 30335
(404) 586-9600

Boston

I.P. Sharp Associates, Inc.
1 Liberty Square
Boston, Massachusetts 02109
(617) 542-2313

Brisbane

I.P. Sharp Associates Pty. Ltd.
6th Floor, Dalgety House
79 Eagle Street
Brisbane, Queensland 4000
Australia
(07) 229-8330

Brussels

I.P. Sharp Europe SA
Boulevard de la Cambre 36, bte 5
1050 Bruxelles
Belgium
(02) 649 99 77

Calgary

I.P. Sharp Associates Limited
Suite 550, Bow Valley Square 4
250-6th Avenue S.W.
Calgary, Alberta T2P 3H7
(403) 265-7730

Canberra

I.P. Sharp Associates Pty. Ltd.
16 National Circuit
Barton, A.C.T. 2600
Australia
(062) 73-3700

Chicago

I.P. Sharp Associates, Inc.
Suite 3860
55 West Monroe Street
Chicago, Illinois 60603
(312) 782-3177

Copenhagen

I.P. Sharp ApS
Østergade 24B
1100 Copenhagen K
Denmark
(01) 11 24 34

Coventry

I.P. Sharp Associates Limited
7th Floor B Block
Coventry Point, Market Way
Coventry CV1 1EA England
(0203) 56562

Dallas

I.P. Sharp Associates, Inc.
Suite 1148, Campbell Center
8350 North Central Expressway
Dallas, Texas 75206
(214) 369-1131

Denver

I.P. Sharp Associates, Inc.
Suite 416
5680 South Syracuse Circle
Englewood, Colorado 80111
(303) 741-4404

Dublin

I.P. Sharp Associates Limited
Segrave House
Earlsfort Terrace
Dublin 2, Ireland
(01) 763605

Düsseldorf

I.P. Sharp GmbH
Kaiserswertherstrasse 115
4000 Düsseldorf 30
West Germany
(0211) 45 20 52

Edmonton

I.P. Sharp Associates Limited
2358 Principal Plaza
10303 Jasper Avenue
Edmonton, Alberta T5J 3N6
(403) 428-6744

Frankfurt

I.P. Sharp GmbH
Rosserstrasse 15
am Palmengarten
6000 Frankfurt 1
West Germany
(0611) 72 47 73

Halifax

I.P. Sharp Associates Limited
Suite 706, Cogswell Tower
2000 Barrington Street
Halifax, Nova Scotia
B3J 3K1
(902) 423-6251

Helsinki

TMT-Team Oy (Agent)
Westendintie 1
SF-02160
Espoo 16, Finland
(0) 452 1155

Hong Kong

I.P. Sharp Associates (HK) Limited
Suite 606, Tower 1
Admiralty Centre, Hong Kong
5-294341

Houston

I.P. Sharp Associates, Inc.
Suite 375, One Corporate Square
2600 Southwest Freeway
Houston, Texas 77098
(713) 526-5275

London, Canada

I.P. Sharp Associates Limited
Suite 510, 220 Dundas Street
London, Ontario N6A 1H3
(519) 673-4060

London, England

(European Headquarters)
I.P. Sharp Associates Limited
132 Buckingham Palace Road
London SW1W 9SA, England
(01) 730-4567
Telex: 8954178 SHARP G

Los Angeles

I.P. Sharp Associates, Inc.
Suite 1230
1801 Century Park East
Los Angeles, Ca. 90067
(213) 277-3878

Madrid

I.P. Sharp Associates Limited
Serrano 23, Piso 8
Madrid 1, Spain
(91) 276 70 54

Melbourne

I.P. Sharp Associates Pty. Ltd.
520 Collins St., 7th Floor
Melbourne, Victoria
3000, Australia
(03) 614-1766

Mexico City

Teleinformatica de Mexico SA
(Agent) Mail to:
Arenal N 40, Chimalistac
Mexico 20 D.F., Mexico
(905) 550-8033

Miami

I.P. Sharp Associates, Inc.
Suite 240
15327 N.W. 60th Avenue
Miami Lakes, Florida 33014
(305) 556-0577

Milan

Informatical Society Italia Srl
(Agent)
Via Eustachi 11
20129 Milan, Italy
(02) 221 612

Montreal

I.P. Sharp Associates Limited
Suite 1610
555 Dorchester Boulevard W.
Montreal, Quebec H2Z 1B1
(514) 866-4981

New York

I.P. Sharp Associates, Inc.
Suite 210
230 Park Avenue
New York, N.Y. 10169
(212) 557-7900

Newport Beach

I.P. Sharp Associates, Inc.
Suite 1135
610 Newport Center Drive
Newport Beach, Ca. 92660
(714) 644-5112

Oslo

I.P. Sharp A/S
Postboks 486 Sentrum
Dronningens gate 34
Oslo 1, Norway
(02) 41 17 04

Ottawa

I.P. Sharp Associates Limited
Suite 600, 265 Carling Ave.
Ottawa, Ontario K1S 2E1
(613) 236-9942

Palo Alto

I.P. Sharp Associates, Inc.
Suite 201, 220 California Ave.
Palo Alto, Ca. 94306-1683
(415) 327-1700

Paris

I.P. Sharp SARL
9 Rue du Cirque
75008 Paris
France
(1) 225 98 20

Philadelphia

I.P. Sharp Associates, Inc.
Suite 604, 437 Chestnut St.
Philadelphia, Pa. 19106
(215) 925-8010

Rochester

(United States Headquarters)
I.P. Sharp Associates, Inc.
1200 First Federal Plaza
Rochester, N.Y. 14614
(716) 546-7270

Rome

Informatical Society Italia Srl
(Agent)
Piazza Della Rotonda 2
00100 Rome, Italy
(06) 656-5925

San Francisco

I.P. Sharp Associates, Inc.
Suite C-415, 900 North Point St.
San Francisco, Ca. 94109
(415) 673-4930

Saskatoon

I.P. Sharp Associates Limited
Suite 303, Financial Bldg.
230-22nd St. E.
Saskatoon, Saskatchewan
S7K 0E9
(306) 664-4480

Seattle

I.P. Sharp Associates, Inc.
Suite 223, Executive Plaza East
12835 Bellevue Redmond Road
Bellevue, Washington 98005
(206) 453-1661

Seoul

Daewoo Corporation (Agent)
541, 5-Ga, Namdaemoon-Ro
Jung-Gu (CPO Box 2810)
8269 Seoul, Korea.
771-91/2
Telex: 23341-5/24295 DAEWOO K

Singapore (Far East H.Q.)

I.P. Sharp Associates(S) Pte. Ltd.
Suite 1601, CPF Building
79 Robinson Road
Singapore 0106
Republic of Singapore
2230211
Telex: 20597 IPSAS RS

Singapore

Singapore International
Software Services Pte. Ltd.
(Agent)
Suite 1601, CPF Building
79 Robinson Road
Singapore 0106
Republic of Singapore
2230211

Stockholm

I.P. Sharp AB
Kungsgatan 65
S111 22 Stockholm, Sweden
(08) 21 10 19

Sydney (Australian H.Q.)

I.P. Sharp Associates Pty. Ltd.
8th Floor, Carlton Centre
55 Elizabeth St.
Sydney, New South Wales 2000
Australia
(02) 232-6366

Tokyo

INTEC, Inc. (Agent)
Dai 23 Mori Bldg. 4th Floor
1-23-7 Toranomon
Minato-ku, Tokyo 105
Japan
(03) 502 8781
Telex: 2322008 INTECA J

Toronto

(International Headquarters)
I.P. Sharp Associates Limited
Suite 1900
2 First Canadian Place
Toronto, Ontario
M5X 1E3
(416) 364-5361
Telex: 0622259 I P SHARP TOR

Vancouver

I.P. Sharp Associates Limited
Suite 902, 700 West Pender St.
Vancouver, B.C. V6C 1G8
(604) 687-8991

Victoria

I.P. Sharp Associates Limited
Chancery Court
1218 Langley Street
Victoria, B.C. V8W 1W2
(604) 388-6365

Vienna

I.P. Sharp Ges.mBH
Renngasse 4
A-1010 Wien, Austria
(0222) 66 42 48

Warrington

I.P. Sharp Associates Limited
1-3 Dolmans Lane
Warrington, Cheshire
WA1 2ED England
(0925) 50413/4

Washington

I.P. Sharp Associates, Inc.
Suite 305, 2033 K Street N.W.
Washington, D.C. 20006
(202) 293-2915

Wayne

I.P. Sharp Associates, Inc.
Suite 303
155 Willowbrook Blvd.
Wayne, New Jersey 07470
(201) 785-8050

White Plains

I.P. Sharp Associates, Inc.
Suite 312 West
701 Westchester Avenue
White Plains, New York 10604
(914) 328-8520

Winnipeg

I.P. Sharp Associates Limited
Suite 208
213 Notre Dame Avenue
Winnipeg, Manitoba R3B 1N3
(204) 947-1241

Zurich

I.P. Sharp AG
Fortunagasse 15
CH-8001 Zurich, Switzerland
(01) 211 84 24

I.P. Sharp Network

APL OPERATOR VOICE 416 363-2051 COMMUNICATIONS 416 363-1832

Our private, packet-switched network connects with the Value Added Networks—Datapac, Datex-P, PSS, Telenet, Telepac, Transpac, and Tymnet—to provide access from the 46 countries listed below:

• Argentina • Australia • Austria • Bahamas • Bahrain • Barbados • Belgium • Bermuda • Brazil • Canada • Chile • Denmark
• Dominican Republic • Finland • France • French Antilles • Germany • Great Britain • Hong Kong • Hungary • Ireland
• Israel • Italy • Japan • Korea • Kuwait • Luxembourg • Mexico • The Netherlands • New Zealand • Norway • The Philippines
• Portugal • Puerto Rico • Qatar • Republic of Singapore • Republic of South Africa • Saudi Arabia • Scotland • Spain
• Sweden • Switzerland • Taiwan • United Arab Emirates • U.S.A. • Virgin Islands

SHARP APL is accessible from over 600 places via a local telephone call. Please ask your nearest I.P. Sharp office or representative for a complete list of access points and access procedures. Our private network also connects with the worldwide Telex network via the Amsterdam and Rochester, New York nodes.