



I.P. Sharp Associates

NEWSLETTER

In This Issue

Corporate News

- Daewoo Represents I.P. Sharp 1
- New Data Centre 2

In-House

- In-House Update 2

Data Bases

- The Deutsche Bundesbank Data Base 3
- PIW Oil Production and Price Data Service 4
- Helsinki Currency Exchange Rates 5
- Australian Economic Statistics 5

Customer Application

- Greiner Engineering 6

Network News

- New Dial Access Numbers 7

Applications Software

- Workspace 8
- Documentation Facility 8

Bulletin Board

- "What James Martin didn't tell you about APL" 9
- Coventry, Edmonton, Saskatoon, White Plains Japan APL Association Reorganizes 10
- 11

Technical Supplement 40

- APL Trees T1
- Searching, Part 4 T4
- Puzzling Facts: Baffled Shuffler Buffaloed T7

새시대새마을

Daewoo Represents I.P. Sharp

The signing of an agreement with Daewoo Corporation opens new markets in the Republic of Korea for I.P. Sharp Associates. This represents another step forward in I.P. Sharp's expansion into the Far East.

Daewoo, with sales of \$2.7 billion U.S., is Korea's largest general trading company, as well as the top construction firm in the Republic. In 1981, Daewoo's exports accounted for nine per cent of Korea's total exports.

Walter Keirstead, managing director of I.P. Sharp's Far East operations, states that "1982 marked the start of Korea's fifth five-year Economic and Social Development Plan. The Plan has been criticized by some as being overly ambitious, but it is our view that Korea will do very well over the next five to seven years. I am very impressed with the efforts of both the public and private sectors to expand the economy."

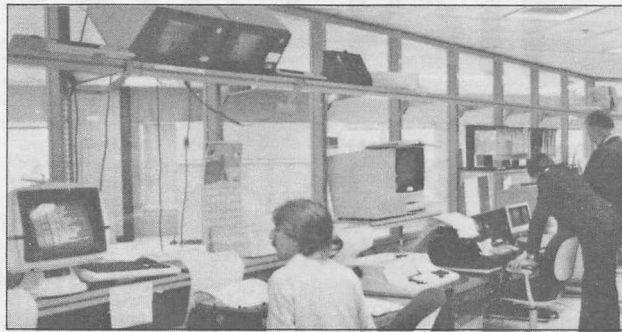
I.P. Sharp Associates has been studying the Korean market for several years. "We found a high degree of awareness of our products and services," says Keirstead. "Daewoo has the experience and knowledge to develop the markets rapidly and efficiently, and we believe they will do an excellent job representing us."

Early in September 1982, SHARP APL timesharing software was installed on an IBM 3032, owned and operated by the Korean Advanced Institute of Science and Technology (KAIST). SHARP APL timesharing services will be available initially within Korea, and a communications link to I.P. Sharp's worldwide network will likely occur next year.

The translation of the Korean calligraphy is "Saemaul in New Age". Saemaul is the new community movement launched in 1970 to encourage cooperative efforts for the growth and development of the Republic of Korea.

New Data Centre

The glass-enclosed "bridge" looks onto the floor where the computers are located. Here, our operations staff keeps an eye on the system.



We are now into our eighth month of operation in the new I.P. Sharp Data Centre. During this time, we have experienced peak usage on the SHARP APL timesharing service of 388 simultaneous tasks.

The new data centre has allowed us to expand our computing facilities considerably. It includes more floor space, additional water cooling capacity, a larger "bridge" for our operations staff, and room for future expansion.

The data centre itself and our corporate offices are located in a new office tower in Toronto's downtown financial district, which provides a high level of physical security.

We were able to move the data centre to the new site without interruption to the timesharing service. This was accomplished through duplication of hardware and testing of equipment before the service was switched to the hardware at the new location.

The SHARP APL timesharing service is now running on an IBM 3081 computer, and uses IBM 3380 disk devices. The new equipment provides considerably faster service and a system which can support a much larger number of users with good response time.

In addition to the 3081 which supports our commercial timesharing service, the new data centre also houses an Amdahl 470/V8 reserved for I.P. Sharp internal use, and an IBM System 370/158 for SHARP APL development.

As most users probably observed, we suffered from unscheduled downtime problems as the result of teething troubles with our new location. The causes were many and varied, ranging from contractors spreading dust to electrical supply problems. We think these are now behind us and the performance of the system during the past few months appears to be back to an acceptable level.

In-House Update

In 1978, we announced that we intended to distribute the SHARP APL timesharing software to run on customer-provided mainframes. Part of the distributed software included code for an IBM 3705 communications controller which would permit the customer's equipment to be attached to our private packet-switched network.

Currently, there are 28 such systems installed, and about half of these are connected to the I.P. Sharp Network. Twenty-one of the installations run under the IBM-provided MVS operating system and most of the remaining seven have plans to convert to MVS at some time in the future. Our future emphasis, therefore, will be on the development of SHARP APL in the MVS environment and an announcement will be made soon about future release schedules for the DOS and DOS/VSE (and either under VM) products.

The hardware used by customers ranges in size from the IBM 4341 group 2 through to the IBM 3081 and includes various sizes of Amdahl and NAS computers. SHARP APL runs under MVS as a single application program. It was, therefore, necessary to make substantial modifications to our software in order to be able to take advantage of the multiple CPUs in the IBM 3081. This work has been completed and the resulting system now constitutes our own production version of SHARP APL, and has also been distributed to the various customer sites which use multiprocessor hardware.

The Deutsche Bundesbank Data Base

Sandra Eadie, Düsseldorf

SHARP APL systems are currently installed in Canada, the United States, Mexico, the United Kingdom, Denmark, Holland, Germany, South Africa, Singapore and Korea. We expect to install ten more systems before the end of this year.

Contrary to our original expectations, the majority of installations of SHARP APL are not for former timesharing customers, but rather for refugees from VSAPL who are primarily interested in high performance, high reliability and enhanced user facilities.

Up-to-date, accurate and detailed information on the economic and financial state of West Germany is now available through SHARP APL.

The Deutsche Bundesbank data base is prepared and published monthly by the West German Central Bank and contains statistics about the German economy, international trade and payments, foreign exchange, money supply, the banking system, and interest rates. In general, the data is available on the SHARP APL service within seven to ten days of its release by the Bank.

The data base is useful for many different types of analysis. Of particular interest to government officials and representatives of organizations wanting to trade with Germany is the information on economic conditions, money supply, and foreign trade and payments.

Institutions acting in international money markets can use the data on interest rates and capital markets in investment decisions.

Within Germany itself, all types of banking institutions and near-banks operating in financial markets will find the ease of access and manipulation of the particular data on money supply, banking and minimum reserve requirements, and interest rates helpful in assessing their relative position and in planning their financial requirements.

The Bundesbank plays a much more important role in its national economy than most other central banks do in theirs. It is an independent institution with special responsibility for keeping the German currency and economy stable. Consequently, it has the power to set interest rates and control the money supply; and as a result, requires timely and accurate information. An extensive information-gathering network collects information daily on banking transactions and thus, on the country's money supply.

In keeping with its economic responsibilities, the bank also receives, as soon as they are available, statistics on price movements, labour force, gross national product and so on, from the Statistisches Bundesamt (Federal Statistical Bureau). The bank then adjusts these numbers for seasonal factors to make them more useful in decision-making.

For more detailed information, in English or German respectively, load 39 *MAGIC* and type *BUNDESBANK 'DESCRIBE'* or *BUNDESBANK 'BESCHREIB'*.

BUNDESREPUBLIK DEUTSCHLAND
DURCHSCHNITTLLICHE KREDITE AN INLAENDISCHE UNTERNEHMEN
AUSERWAELTE WIRTSCHAFTSBEREICHE
(MRD DM)

UND VERAENDERUNG GEGEN VORJAHR IN %

		VERAEN- DERUNG IN %	VERAR- BEITENDES GEWERBE	VERAEN- DERUNG IN %	HANDEL	VERAEN- DERUNG IN %
	INSGESAMT					
1972	1223.2	17.9	407.7	14.2	178.3	14.2
1973	1388.1	13.5	439.9	7.9	201.7	13.1
1974	1499.8	8.0	471.4	7.2	213.8	6.0
1975	1542.3	2.8	459.8	-2.5	213.1	-0.3
1976	1637.4	6.2	456.2	-0.8	242.7	13.9
1977	1759.1	7.4	480.1	5.2	272.7	12.3
1978	1879.6	6.8	492.9	2.7	300.5	10.2
1979	2087.1	11.0	539.4	9.4	357.6	19.0
1980	2404.1	15.2	598.8	11.0	424.0	18.5
1981	3013.0	25.3	713.8	19.2	533.3	25.8

QUELLE: DEUTSCHE BUNDESBANK

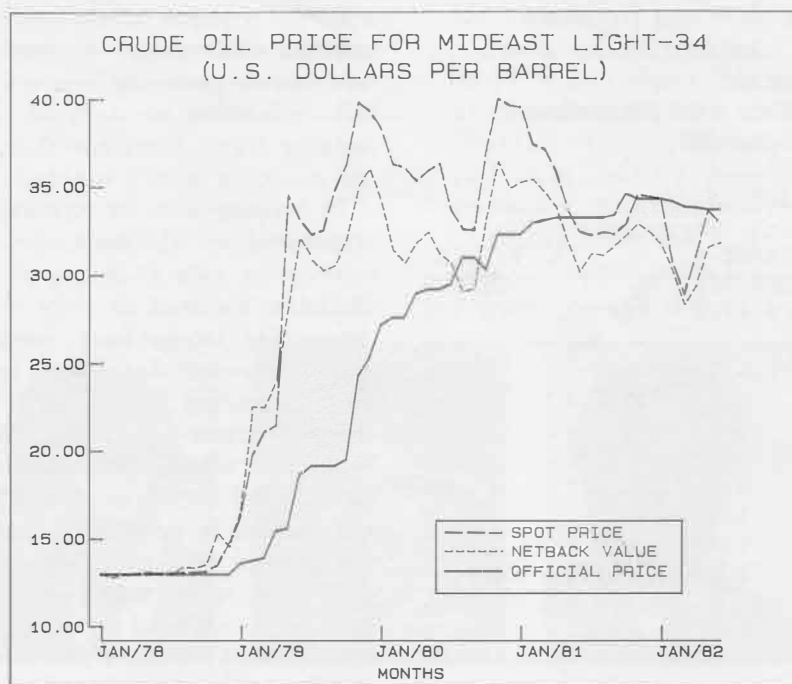
PIW Oil Production and Price Data Service

Ed Feinberg, New York

WEST GERMAN CONSUMER PRICE INDICES
(SEASONALLY ADJUSTED, 1976=100)

	ALL GOODS			FOOD		
	INDEX VALUE	ANNUAL PERCENT CHANGE	MONTHLY PERCENT CHANGE	INDEX VALUE	ANNUAL PERCENT CHANGE	MONTHLY PERCENT CHANGE
JAN/81	120.3	5.8	0.3	113.2	5.2	0.9
FEB/81	121.0	5.5	0.6	113.5	5.2	0.3
MAR/81	121.6	5.5	0.5	114.3	5.7	0.7
APR/81	122.3	5.6	0.6	114.9	6.0	0.5
MAY/81	122.8	5.6	0.4	115.4	6.1	0.4
JUN/81	123.4	5.6	0.5	115.2	5.0	-0.2
JUL/81	124.2	5.8	0.6	115.3	4.5	0.1
AUG/81	124.9	6.0	0.6	115.2	3.2	-0.1
SEP/81	125.8	6.5	0.7	116.6	4.8	1.2
OCT/81	126.4	6.7	0.5	118.0	6.3	1.2
NOV/81	127.0	6.5	0.5	119.0	6.4	0.8
DEC/81	127.4	6.3	0.3	119.0	6.1	0.0
JAN/82	127.9	6.3	0.4	119.1	5.2	0.1
FEB/82	128.1	5.9	0.2	119.8	5.6	0.6
MAR/82	128.1	5.3	0.0	120.4	5.3	0.5
APR/82	128.5	5.1	0.3	120.7	5.0	0.2
MAY/82	129.3	5.3	0.6	122.1	5.8	1.2
JUN/82	130.6	5.8	1.0	123.0	6.8	0.7

SOURCE: DEUTSCHE BUNDESBANK



SOURCE: PETROLEUM INTELLIGENCE WEEKLY (C) COPYRIGHT 1982

Petroleum and Energy Intelligence Weekly—known throughout the oil industry simply as *PIW*—has introduced a new online oil production and price service on the I.P. Sharp system. *PIW*, the authoritative international energy publication, publishes information on international petroleum markets, people, production, and prices. Its staff of veteran oil journalists and experienced analysts evaluate the impact and meaning of fast-paced developments in the international oil industry for top executives. *PIW* generates a broad spectrum of data on oil prices and world crude oil products.

Its recent entry into online dissemination of data is a major event in the energy information business, and a significant addition to I.P. Sharp's growing energy data base offerings.

Currently, the *PIW* data base consists of monthly time series on crude oil production, crude oil prices, and spot prices for refined products. *PIW* uses the SHARP APL service to enter and validate data as it becomes available, by entering it into a private data base. This process assists *PIW* in the editing of source information to camera-ready copy, with automatic retrieval of historic data, calculation of derived figures, and preliminary typesetting of the final reports. Simultaneous with the date of issue of a table, the data is transferred to the public data base and made available to I.P. Sharp users.

The computerized access provides a number of valuable features to supplement the

Helsinki Currency Exchange Rates

Australian Economic Statistics

Silva Bassett, Sydney

newsletter. First, the *PIW* data base is an archive of current and historical monthly data. Normally the *PIW* newsletter shows only a few months of data at a time, whereas some of the time series in the online data base begin in 1961. Secondly, all of *MAGIC*'s statistical analysis and graphics tools for analysing and displaying data effectively are available for use with *PIW* data. (See the *SUPERPLOT* which shows the dramatic increase in crude oil prices over the past few years.) Lastly, *PIW* data may be combined with other energy or non-energy data bases already available on the SHARP APL service.

Additional *PIW* services are planned for the I.P. Sharp system. For example, official prices for over 100 specific crude streams should be available in the near future.

Access

Authorized users can gain access to *PIW* data by entering *PIWSET* in workspace 121 *PETROSERIES*. Online documentation for each section of *PIW* data consists of a description (also available via 1 *DATABASES*), a directory (showing *MAGIC* access codes and series labels), a sample *MAGIC* program, and relevant notes (normally part of the footnotes to tables and charts in the *PIW* newsletter).

PIW data is available without surcharge to all subscribers of the *PIW* newsletter.

Buying and selling rates, in Finnish pennies (1/100 of a markka), for 29 currencies is now available in the Currency Exchange Rates (*CURRENCY*) data base. Prices are reported at noon Helsinki time and are posted two to three hours later. Rates are available for each business day beginning 1 January 1981. The rates are obtained from the Bank of Finland and are entered by the Osuuspankki Bank.

In addition to the recently added Finnish exchange, currency exchange rates for major world currencies on the Copenhagen, London, Melbourne, New York, Paris, Toronto, and Vienna exchanges are available in the *CURRENCY* data base. The currencies and the amount of historical data vary among exchanges. The rates are given in the currency of the country where the exchange is located.

A new *Currency Exchange Rate* manual is available.

The Australian Economic Statistics (*AES*) data base consists of a wide range of statistics on the Australian economy for the period 1948/49 to 1978/79. Data for 1979/80 and 1980/81 will be available shortly.

The Australian series include:

- Balance of payments
- Government finances
- Finances of banks and other financial institutions
- Finance of the private sector
- Flows of capital
- Gross domestic product (by person and by industry)
- Expenditure and income
- Population, labour market and earnings
- International comparisons

This economic data is of interest to both commercial and governmental institutions, including banks, financial organisations, large industrial companies, researchers, econometricians, statisticians and others.

The data can be used for a variety of analyses and economic studies, including econometric modelling, linear relationships identification and expression, and other similar tasks depending on individual requirements.

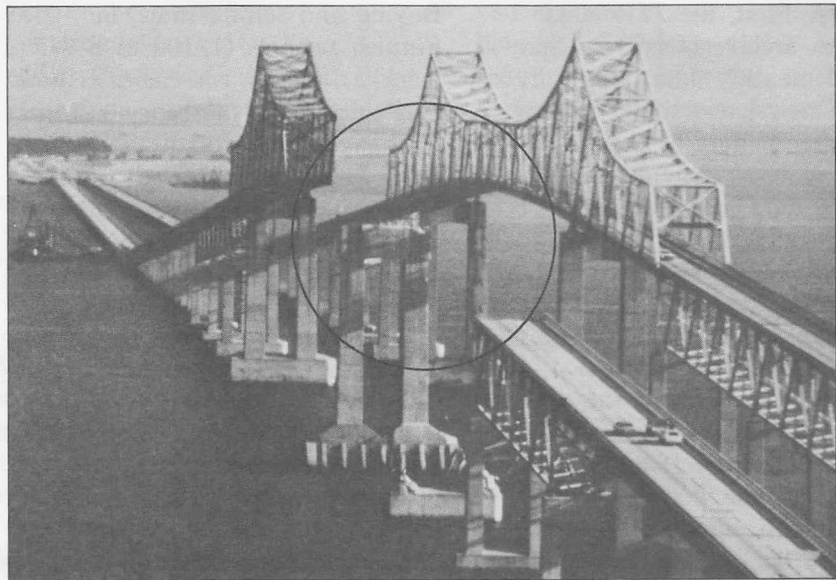
The data comprises over 1 000 time series which relate to the Australian economy and about 150 series of international data. The Australian series are annual for fiscal years ending 30 June. Data in the international supplement relates to calendar years. For ease of comparison with other countries, Australia has also been included in this section.

Greiner Engineering Risk Analysis System

The Sunshine Skyway bridge is located at the south end of Tampa Bay, Florida, and connects St. Petersburg to Sarasota. In May 1980 the Summit Venture, a freighter carrying phosphate, took a detour and hit the Skyway. Thirty-five people died and the southbound main span of the bridge collapsed. To avoid any more tragedies, the Florida Department of Transportation (FDOT) contracted Greiner Engineering Sciences, Inc. in Tampa, Florida, to investigate pier protection alternatives for use on the Sunshine Skyway bridge.

"One aspect of our investigation was to evaluate the vulnerability of the new bridge to ship collisions," says Mike Knott, a Greiner project engineer. "Also, we had to evaluate various alternatives to pier protection based on their cost-effectiveness, so that the FDOT could decide how best to protect the bridge and its users from future accidents."

The number of alternatives is enormous and includes: the size and type of ships and barges travelling under the bridge; the frequency of automobile traffic on the bridge; the number of piers that need to be protected and the multiple combinations of the varieties of protection possible. It also covers the effectiveness of motor safety warnings and navigation safety alerts; the impact force of a range of vessels using the waterway beneath the bridge; the probability of an accident with any given vessel; and the potential damage to the bridge.



The Sunshine Skyway bridge is shown above with its southbound main span missing, after it was hit by a freighter.

To fully assess the cost effectiveness of the myriad alternative protection methods would be a lengthy and major undertaking. As luck would have it, Peter Zent, a computer applications specialist with Greiner, read about computer risk analysis in the *I.P. Sharp Newsletter*.

"The existing software allowed for detailed risk analysis of various threats for computer security and nuclear power plant security applications," Zent explains. "I figured that if the same application of the risk analysis principles could be tailored to ship collision risks, it would help us solve our analysis problem and provide a better product for our client."

Within a month, with the assistance of David Bonyun, manager of Security Systems, Special Systems Division, in I.P. Sharp's Ottawa office, the program was written and ready to run. Using I.P. Sharp's software and the unique methodology developed by Greiner Engineering and Bonyun, engineers were able to evaluate the ship collision problem.

There were three major steps to threat analysis. "First, we had to identify what we were trying to protect, which involved drawing up an inventory of each span and pier of the bridge," explains Bonyun. "Then, we had to identify what the perils were. For a bridge, the perils are ship collisions with a bridge pier, or collisions resulting in environmental damage from oil or cargo spills. Finally, we had to draw up a series of countermeasures which were then evaluated using cost/benefit

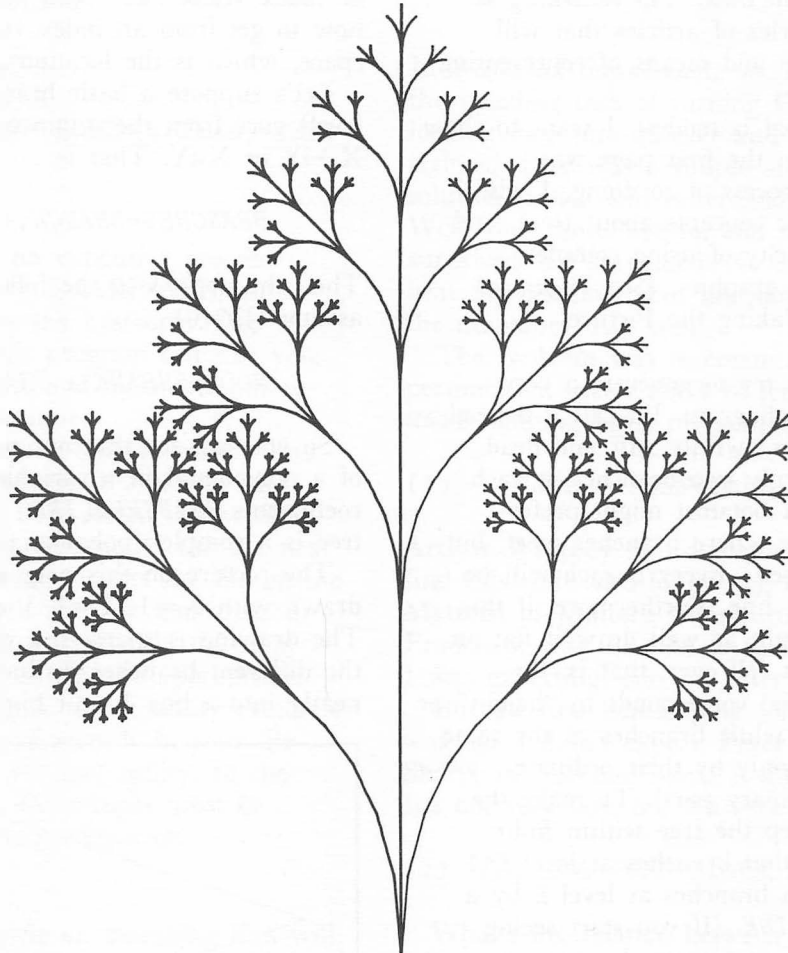


I. P. Sharp Associates

NEWSLETTER

technical supplement 40

APL Trees



Or an Arithmetical Orchard

Clement Kent, Toronto

A well-known poem by Joyce Kilmer asserts the Deity's copyright, and construction monopoly, for trees. However, as most programmers are aware, the building and manipulation of trees is an important part of computing. In APL we, perhaps, don't think as much about trees because of an inability to see them in the forest of arrays.

But as early as 1962, in Ken Iverson's book *A Programming Language*, trees had been defined within APL and some functions and operators extended to them. I'm returning to this topic with a series of articles that will discuss fruitful ways and means of representing and using APL trees.

This time, my goal is modest. I want to show you how the tree on the first page was generated. In the process of so doing, I will introduce some basic concepts about trees, and I'll show the simplicity of using complex numbers to do 2-D graphics. (See *Technical Supplement 39*, "Making the Picture Complex".)

As a starter, let's try to generate a very simple kind of tree diagram. I'll call it a dead tree, because it looks awfully stiff and dead. We'll let there be only two branches at each node of the tree. (A botanist might prefer "groin" as the place where branches meet, but this is a family paper.) Every branch will be drawn as a straight line. Furthermore, if this tree is dead, we might as well draw it flat on the ground where it fell over; that is, the abscissa (or real axis) corresponds to "height" or "level" in the tree, while branches at the same level are separated only by their ordinate coordinate (or imaginary part). To make the picture tidy and keep the tree within finite bounds, I'll decree that branches at level $L+1$ will be shorter than branches at level L by a multiple called *SHRINK*. (If you start seeing imaginary ordinate coordinates, I suggest you call *SHRINK*; at least the beastly little things will get smaller.)

How can we identify a particular node or branch? The easiest way is to give the path you'd have to follow to get from the root, or trunk, to that node (or leaf ... or apple). In this binary tree, a simple "left" or "right" at each intersection would be all you'd need to know. For instance, to get to an apple on the sixth level, you might follow the path LLRLRR: go left twice, right, left, and right twice.

The worm in this binary apple turns in base two; i.e., we could replace L by 0 and R by 1, and then our path would be 0 0 1 0 1 1. (No programmer can resist the temptation to chop a problem up into little bits.) Let's call this path or index vector *IND*. Now we have find out how to get from an index vector to a point in space, which is the location of the node.

Let's suppose a basic branch (at the first level) goes from the origin to one of the points $X+iY$ or $X-iY$. That is:

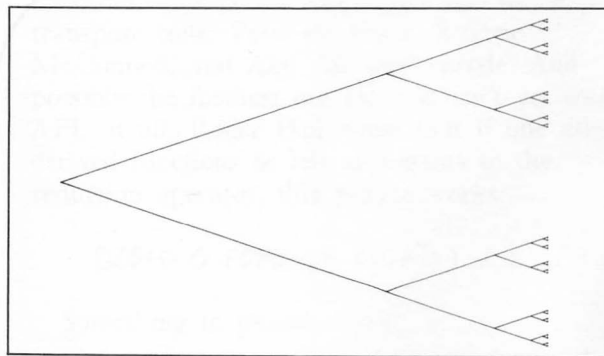
$$BRANCHES \leftarrow BRANCH, +BRANCH \times X + Y \times OJ1$$

Then the node is at the following point (I assume $\square IO \leftarrow 0$):

$$NODE \leftarrow SHRINK \perp BRANCHES[\phi IND]$$

So you can see that our nodes are the values of a polynomial in a variable *SHRINK* with coefficients *BRANCHES*[*IND*]. This dead simple tree is a complex polynomial!!

The picture on this page shows this tree drawn with $X=1$, $Y=1/3$, and $SHRINK \leftarrow \div 3$. The drawing is sparse but clear—you can see the different branches distinctly, and it all fits neatly into a box 1 unit high by 1.5 units wide.



Can you prove that this must be so? How big would the box be if we set $SHRINK \leftarrow 2$?

This dead tree nonetheless bears some fruitful ideas. As a digression, here are some of them:

(1) If the tree is drawn "to completion"; that is, with an infinite number of levels (don't try it, you won't like the bill next month), all the branches "end" at a set S of points with common abscissa 1.5. There are as many points in this set as in the whole plane, but paradoxically the length or content of S (how much space it takes up on the line $X=1.5$) is infinitesimal. If we add $1/2$ to the ordinates of each point in S , the new set of ordinates is called the Cantor set, after the great mathematician Georg Cantor. He used this set to demonstrate some paradoxes about infinities: it was one of the first examples of an uncountable infinite set that fit in a finite interval and yet had no length. The dimension of this set isn't even integral; it's a fraction.

(2) If drawn to completion, what's the total length of all the branches on the tree? Is it infinite or finite? If you simply try summing the length of the branches on a few levels, you'll find the length is greater than six. Is it greater than twice pi (2π)? If so, how many levels up must you go to get a length greater than 2π ? (Hint: There's a way to answer all these questions using paper and pencil only.)

The incredible shrinking tree

For the tree on the first page, I wanted gracefully curving branches that nonetheless obeyed the shrinking rule: branches on level $L+1$ are $SHRINK \times$ the size of branches on level L . I could only think of two curves that fit together nicely when shrunk. The first, a rather trivial one, is a straight line. The second was the logarithmic spiral. If you know the parameters R and $THETA$ governing the shrinkage and twisting rate of the spiral, form the complex number $\omega \leftarrow R + 0J1 \times THETA$. Then at any time α the spiral is at a position:

$$LOGSPIRAL \diamond * \omega \times \alpha$$

If R is negative, the spiral shrinks toward 0. If

it's positive, the spiral expands outwards. If it's 0, you get a circle. In constructing my tree, I decreed that a basic branch would be a segment of a *LOGSPIRAL* from time 0 to 1. To satisfy the shrinking rule, I set $R \leftarrow SHRINK$. The segment of spiral wasn't pointing the right direction, and started at 1—so I translated it to 0 and rotated it until it was tangent to the real axis. (Can you figure out how to do this? I'll give the answer next issue.) The tangent to the $\alpha=1$ end of the branch then made an angle ANG with the real axis. (How could you determine this?) Let's call the complex vector of points that I got, *BRANCH*.

BRANCH is the path that I follow if I turn left at a node. $+BRANCH$ is the right-hand path. If I follow one of these paths, I will have turned through an angle ANG or $-ANG$. To join the branches at the next level smoothly to this one, I have to shrink them and rotate them by this angle. Of course, this just means to multiply the basic branch vector by a single complex number that does the shrinkage and rotation. If I'm M levels up in the tree, I must sum the M angles of twist of the branches below me. Define *TWISTS* to be the complex numbers corresponding to rotating $+$ or $-ANG$ radians, *SHIFTS* to be the end points of the two basic branches (or the translations required at level 1) and *BRANCHES* to be a matrix, of which each row is a branch. Thus:

$$\begin{aligned} TWISTS &\leftarrow 0J1 \times ANG \times 1^{-1} \\ BRANCHES &\leftarrow BRANCH, [.5] + BRANCH \\ SHIFTS &\leftarrow ,2^{-1} + BRANCHES \end{aligned}$$

Then if I've climbed to a node at level M along a path defined by index vector *IND*, I'll end up at a position P . (Again, $\square IO \leftarrow 0$):

$$P \leftarrow (SHRINK \times TWISTS[IND]) \perp SHIFTS[IND]$$

with the end of my branch pointing at an angle equivalent to the complex rotation $ROT \leftarrow \times / TWISTS[IND]$.

What should the next left branch be? We must shrink the branch, rotate it by *ROT*, and translate it to P . So our new branch B is:

$$B \leftarrow P + BRANCH \times ROT \times SHRINK \times M$$

If I generate a matrix *LFIND* in which each row is the index vector leading to a given leaf, then the positions of all the nodes can be found in the matrix *MPOS*:

```
TRANSFORM←x\1, 0-1←SHRINK×TWISTS[LFIND]
MPOS←+SHIFTS[LFIND]×TRANSFORM
```

Of course, every column of *MPOS* but the last is highly redundant, as it repeats the position of a node once for each leaf on its branches.

If we are dealing with a binary tree of height *M*, a simple origin 0 expression to generate the index matrix is:

$$LFIND \leftarrow \bigotimes (M \rho 2) \tau 1 2 \rho M$$

which is equivalent to (see Puzzle 13 results in this issue):

$$LF \quad IND \leftarrow \bigotimes (M \rho 2) \tau, \bigotimes (M \rho 2) \rho 1 2 * M$$

If you watch a plotter draw each path to a leaf in the order given by *LFIND*, you'll see a graphic demonstration of how multidimensional transpose works.

Although the above is fairly neat, it's inefficient for large trees and obscures the essential simplicity of the algorithm. A much simpler recursive function makes the point clearer:

```

VDRBINTREE M;LEFT;RIGHT;IO
[1]  AGLOBALS: SHRINK, BRANCHES, ANG
[2]  ADRWS BINARY TREE TO LEVEL M RECURSIVELY
[3]  RIGHT←LEFT←SHRINK×*0J1×ANG ⋄ IO←1
[4]  0 1 DTREE2 M,1 ADRAW LEFT HALF
[5]  0 1 DTREE2 M,2 ADRAW RIGHT HALF
      V

VPA R DTREE2 LΔI;Y;I
[1]  APAR IS POSITION, ROTATION
[2]  A LΔI IS LEVEL, INDEX
[3]  AUSES COMPLEX DRAW FUNCTION DRAWCOMP
[4]  Y←PAR[1]+PAR[2]×BRANCHES[I+LΔI[2];]
[5]  DRAWCOMP Y ⋄ Y←1+Y
[6]  +(L+LΔI[1])≠0 AEXIT IF AT TREETOP
[7]  PAR←Y,PAR[2]×(LEFT,RIGHT)[I]
[8]  L←L-1 ADRAW BRANCHES AT NEXT LEVEL
[9]  PAR DTREE2 L,1 ⋄ PAR DTREE2 L,2
      V
```

So go forth, be fruitful and multiply ... and add ... and build your own APL trees.

Searching, Part IV

Robert Metzger, Rochester

In the first three parts of this series, we have explored algorithms for searching vectors, matrices, and texts. The objects being searched for all existed in the active workspace. In this article, we will analyse ways of searching for data in files.

4) Searching files

When we search arrays in a workspace, the main cost comes from the comparisons made. In general, the fewer the comparisons, the lower the cost. When we search files, the number of file operations done is also very important. We want to do as few of these as possible.

In this article, we will present six methods for searching files. We will work with one particular kind of data stored on file, but the methods can easily be generalized. These methods are:

1. Sequential search of a random file
2. Sequential search of an inverted file
3. Binary search of a sorted file
4. Directory search of an sorted file
5. Sequential search of a buffered file
6. Algorithmic search of a hashed file

The type of files we will consider are those frequently occurring in business applications—sets of fixed length records. Each record contains one or more fields. Each field is a scalar or vector. One or more of the fields are used to identify the records. These are called the **keys**.

There are a number of ways to store the records. Unless the records are very long, several will be blocked together in each file component. If all of the fields are of the same data type, then blocks of records can be stored in a single file component. Each row contains a record, and each field is represented by one or more columns. If some of the fields are of different data types, all of the fields of the same data type can be stored in a single component. This could require up to four components for each block of records—one each for Boolean,

character, fixed point numbers, and floating point numbers. (Note that this type of structure was popular before the introduction of complex numbers, which would require yet another component.)

Sometimes this method is modified so that there are just two components—numeric and character. The numbers will all be stored in the same representation. This will be the one which requires the most space. Another variation on this approach is to convert the numeric fields to characters, or encode the characters as numbers. Then a single component can be used for each block.

Before package variables were first introduced in SHARP APL, many applications stored fixed length records as described above. After package variables became available, two new options appeared. The groups of two, three, or four components could be consolidated into a single package variable with two, three, or four arrays in it. Or even better, each field could be put in its own array in the package variable. This is the way MABRA stores data on file.

After enclosed arrays were released on SHARP APL, two further options became possible. File components could contain vectors of enclosed arrays. Each enclosed array would contain one field. Or, going back to the original approach, the block of records could be a single matrix. In this case, each row contains a record, and each column, a field. The new representation of enclosed arrays (slack representation) used in the fall 1982 release of SHARP APL makes this practical, both in terms of workspace used and CPU units consumed.

We are now back to the single matrix approach. Therefore, the remainder of this article will be devoted to this type of structure.

The functions discussed in this article have several common characteristics:

- 1) The right argument is a numeric vector with at least two elements. The first is the file tie number. The second is the column in which the key field is found.
- 2) The left argument is a scalar which identifies the desired records. It may be an enclosed array, if the functions are modified to use ϵ for comparing keys.
- 3) The result is 0 0 0 if no records are found.
- 4) The algorithms which do not require an ordered file (1, 2, and 5) will work on arrays of any data type. Algorithms 3 and 4 require numeric data, unless the data can be well-ordered. Algorithm 6 requires that the key field be numeric, or be transformable into a single number.
- 5) The algorithms presume that all of the records having the same value in a field to be searched will occur in the same component.

Sequential search of a random file

A sequential search of a file occurs when you start at one end of a file and search each component in turn, until you find the record you are looking for. No assumptions are made about the order of the records in the file. As you might suspect, it can be rather slow. On average, if you have n file components, this approach will require $n \div 2$ file reads. The following function implements such a search.

```

▽ RECORDS←KEY FILESEARCH1 CONTROL
;CTR;LMT;TIE;BLOCK;FIELD
[1] TIE←CONTROL[1]
[2] FIELD←CONTROL[2]
[3] RECORDS← 0 0 0
[4] CTR← 1+1+⌊SIZE TIE
[5] LMT← 1+1+1+⌊SIZE TIE
[6] LOOP:CTR←CTR+1
[7] →(LMT<CTR)ρEND
[8] BLOCK←⌊READ TIE,CTR
[9] →(∼KEY∈BLOCK[;FIELD])ρLOOP
[10] RECORDS←(BLOCK[;FIELD]=KEY)⌊BLOCK
[11] END: ▽

```

If this method is slow, why would you use it? The cost to build this file structure is minimal. If you almost never search the file, the extra cost in the search is far outweighed by the savings in building and maintaining the structure.

Sequential search of an inverted file

An inverted file is one in which data is blocked by field, rather than by record. If your records have six numeric fields, you might block all fields at 400 records per file component. If you have 10 000 records, and you want to look at all the values in one field, you will read 25 components. Alternatively, you might separate each field, and block the values of one field at 2 500 values per component. If you have the same 10 000 records, and you want to look at all the values in one field, you only need to read four components (even though there are 24 components in all). It is possible to invert fields in MABRA in order to cut costs.

There are two problems with inverted fields. The first is that if you really want to read whole records, which is usually the case, inverting a file will actually cause more file reads. MABRA handles this by storing your data both ways. Data in the master file is blocked by record. Inverted fields are stored in a separate file and are blocked by field.

The second problem is the question of growth. If the first four components of your inverted file contain field A, the second four contain field B, and so on, what happens when you have more records than will fit in four components? MABRA handles this by setting a maximum number of records, and providing a means of reblocking the file to provide more space. An alternative is to put each field in its own file.

How do you search an inverted file? The following function shows one method. Once again, no assumptions are made about the order of the records in the file.

```
▽ RECORDS←KEY FILESEARCH2 CONTROL
;TIE;FIELD;FINDCTR;FINDLMT;READCTR
;READLMT;DATA;SELECT;NUMBEROFBLOCKS
[1] TIE←CONTROL[1]
[2] FIELD←CONTROL[2]
[3] NUMBEROFBLOCKS←CONTROL[3]
[4] RECORDS← 0 0 ρ0
[5] FINDCTR←NUMBEROFBLOCKS×FIELD-1
[6] FINDLMT←NUMBEROFBLOCKS+FINDCTR
[7] FINDLOOP:FINDCTR←FINDCTR+1
[8] →(FINDLMT<FINDCTR)ρFINDLOOPEND
[9] DATA←□READ TIE,FINDCTR
[10] →(~KEY∈DATA)ρFINDLOOP
[11] SELECT←DATA=KEY
[12] RECORDS←((+/SELECT),0)ρ0
[13] READCTR←(1+NUMBEROFBLOCKS|-1
+FINDCTR)-NUMBEROFBLOCKS
[14] READLMT←-11+1+1+□SIZE TIE
[15] READLOOP:READCTR←READCTR
+NUMBEROFBLOCKS
[16] →(READLMT<READCTR)ρREADLOOPEND
[17] RECORDS←RECORDS,SELECT/
□READ TIE,READCTR
[18] →READLOOP
[19] READLOOPEND:
[20] FINDLOOPEND: ▽
```

When should you use an inverted file search? If you have a lot of unsorted records, which you search frequently, you can save a good deal of CPU time over the simple sequential search.

Binary search of a sorted file

Up to this point we have assumed that the records would be stored in the file in a random order, perhaps with the most recent additions at the end. We can achieve substantial savings if we can search a sorted file.

A binary search of a file occurs when you search by splitting the records under consideration into two groups. After this, you determine which of the groups will contain the records you want. You repeat this process until you have found the desired records. On average, if you have n file components, this search will require $2\log n$ file reads. The following function implements the binary search.

Puzzling Facts:**Baffled Shuffler Buffaloed**

```

▽ RECORDS←KEY FILESEARCH3 CONTROL
    ;START;MIDDLE;END;TIE;BLOCK;FIELD
[1] TIE←CONTROL[1]
[2] FIELD←CONTROL[2]
[3] RECORDS← 0 0 ρ0
[4] START←1+⌊SIZE TIE
[5] END←1+1+1+⌊SIZE TIE
[6] LOOP:→(START>END)ρEXIT
[7] MIDDLE←⌈0.5×START+END
[8] BLOCK←⌊READ TIE,MIDDLE
[9] →(KEY∈BLOCK[;FIELD])ρFOUND
[10] →(KEY<BLOCK[1;FIELD])ρHI
[11] →(KEY>BLOCK[1+ρBLOCK;FIELD])ρLO
[12] →EXIT
[13] LO:START←MIDDLE+1
[14] →LOOP
[15] HI:END←MIDDLE-1
[16] →LOOP
[17] FOUND:RECORDS←(BLOCK[;FIELD]=KEY)
    /BLOCK
[18] EXIT: ▽

```

File sorting can be an expensive process. Fortunately for users of SHARP APL, an interface is provided to the host operating system sort utility. This program can sort your APL files inexpensively, and make the binary search a practical technique.

When should you use a binary search? You must use it on a sorted file. But if you can keep your file permanently sorted on the field you wish to search on, you should use the next method described. If you can sort the file on the field you wish to search, but for some reason you can't keep the file permanently stored in that order, binary search may be appropriate. Obviously, you must have a sort utility available to you. The data type of records in your file must be acceptable to the sort utility. In the case of SHARP APL, the records must be strictly numeric or strictly character.

Editor's note: This article on searching files will be continued in *Technical Supplement 41*. In the next issue, Robert will discuss the three remaining methods and then compare the six approaches to searching files.

Clement Kent, Toronto

Transvestite transpose exposed says, "What's the Matter(n)?"

Author accused of putting on Airs, says 'tis base misrepresentation.

Master of one-liner Art cares not a Whit(ney),

Says square old loop Breeds quicker solution.

Puzzled author sent back to Schueler,

Says "He Hui hopes for speed begs for demotion

And flirts with exile to the outer products."

In *Technical Supplement 38*, I left readers with the puzzling task of turning a rather ugly looking loop into quicker and simpler code. I claimed there was a simple and fast one-line solution, and a still faster and simpler loop. Well, the results are in, and as usual, I am surprised and surpassed. No one, fortunately, beat my one-liner; but my loop went the way of the dinosaurs. Ah, well.

The problem was to create an origin 0 permutation vector *POSN* of length $N=2*M$. For $M \leq 12$, nothing beats my simple one-liner:

(1) $\square IO \leftarrow 0 \diamond POSN \leftarrow \Phi(M\rho 2)\rho 1N$

Arthur Whitney in Edmonton was the first to find this. (He also found (2), below). **Don Mattern** of Mattern Construction, in San Francisco, was the first to find the two solutions I was expecting, one day after Arthur.

But the first non-looping solution submitted was one I didn't expect. **Peter Airs** of I.P. Sharp, in Toronto, found it almost immediately. It's not very fast, but it's very interesting:

(2) $\square IO \leftarrow 0 \diamond POSN \leftarrow 2 \downarrow \Theta(M\rho 2) \uparrow 1N$

What's the relation between (1) and (2)? When else do they give the same result? **Paul Gauthier** of the International Air Transport

Association, in Switzerland, proposed a generalization in which the 2's in both solutions are replaced by B , an arbitrary positive integer. This would be useful, since the fast Fourier transform of which this algorithm is a part, could then be extended to data sets of size $B \times M$. Can you prove whether this would work? What if B were a vector of (possibly different) positive integers of length M ? Would it still work?

Here's a hint. Suppose we had a rank 3 array Y and its transpose Z . By definition, $Y[I;J;K] \equiv Z[K;J;I]$. In origin 0, the position of this element in Y is given by $(\rho Y) \downarrow I, J, K$. The corresponding position in Z is $(\phi \rho Y) \downarrow \phi I, J, K$. Can you see now why (1) and (2) are the same?

Now, back to loops. I expected the simple:

```
(3)      →(N≤1)ρPOSN←,0
        LP: POSN←T,1+T←2×POSN
        →(N>ρPOSN)ρLP
```

or the variant:

```
(4) LP: POSN←,0 1 ◦.+ POSN×2
```

Roger Hui contributed an analysis of this version of the algorithm, based on his function *DTRANS* in workspace 880 *HUIΔUTIL*, which models dyadic transpose. Both **Larry Breed**, of IBM's Palo Alto Research Center, and **J. Henri Schueler** of I.P. Sharp, in Calgary, found an unexpected speedup of this outer product loop.

Let's define a function F such that $POSN \leftarrow F M$. If we use a subfunction G :

```
(5)      G◇ ,α◦.+ω×ρα
```

We can see from (4) that:

```
(6)      (F M) ≡ 0 1 G 0 1 G ... 0 1 G ,0
```

What Breed and Schueler took advantage of was the fact that G is associative: you can

shuffle the parentheses around. So:

```
(7)      (A G B) G C ≡ A G (B G C)
```

which means:

```
(8)      (F I+J) ≡ (F I) G (F J)
```

Specifically, $(F 2 \times I) \equiv (F I) G (F I)$. Note that the functions F and G are analogous to exponentiation and multiplication. Now, if you wanted to calculate the eighth power of X , you could do seven multiplications; but three will do.

```
(X*8) ≡ Y×Y←Y×Y←Y×Y←X
```

So it only takes 2×8 steps to do the exponentiation using repeated squaring. Schueler simply made his loops square, as you can see in the modified version of his entry below.

```
      ∇R←F M;X
[1]      R←1+X←0 1
[2]      LP: →(2|M)+SKIP
[3]      R←X G R
[4]      SKIP: →(M←|M÷2)+0
[5]      X←X G X ◇ →LP
      ∇
```

Line [5] "squares" X ; the j th time through the loop, X is $F 2^j$. This algorithm, and a related one submitted by Breed, were the only ones to beat the transpose solution, and then only for large arguments.

Many other readers submitted solutions. Ed Stubbs, Morten Kromberg, Jim Golding, Ray Chantler, and David McKnight also found the transpose code. Peter de Jager, Eugene McDonnell, and Akif Ali used encode. And possibly the farthest out solution isn't yet correct APL at all. Roger Hui noted that if one allows derived functions as left arguments to the reduction operator, this phrase works:

```
□IO←0 ◇ POSN←, > (◦.+"">) / 0 ,""> 2*1M
```

Something to puzzle over!!

New Dial Access Numbers

Chicago

30 cps 312 726-5450
120 cps 312 726-5094

Los Angeles

30 cps 213 879-4141
120 cps 213 879-2999

Minneapolis

30 cps 612 338-4916

San Francisco

120 cps 415 885-1337

Vienna

30 cps 0222 66 38 21

Wayne, N.J.

30 cps 201 785-2370

White Plains

120 cps 914 328-0408

analysis. The countermeasures or pier protection mechanisms decided on for that particular bridge were islands and dolphins around the piers, navigation improvements, and a motorist warning system."

Once all the variables are entered, the program evaluates the risk exposure as an expected loss in dollars per year. Protection of the Sunshine Skyway piers is estimated at a cost of \$17 million to \$30 million, depending on the options chosen by the FDOT. Describing it as a "sophisticated system", Zent believes the detailed analysis provided by the program greatly assisted the risk evaluation of the new bridge. "I am unaware of any equivalent cost effectiveness analysis system available in the market today, apart from the one I.P. Sharp has developed," he says.

Greiner Engineering sees the system as a tool with great business potential. The company recognizes that very few of the bridges in the world are adequately protected and that on the average, one major bridge is seriously damaged by a ship collision each year somewhere in the world, indicating that this service should be more in demand in the future.

The costs of a bridge collapse go far beyond the replacement of the structure itself, Knott points out. Other factors to be reckoned with include the number of lives lost; the cost of closing the port to commercial traffic; the cost of local business interruptions since cars can no longer use the bridge; the cost of demolishing the damaged

structure; and possible environmental damage caused by oil or cargo spills. His company believes that more port and river authorities will prefer to pay the cost of providing bridge protection systems rather than risk the social, financial and economic consequences of a major bridge collapse.

With the Sunshine Skyway experience and the I.P. Sharp Risk Analysis System in hand, Zent projects that Greiner is "well on the way to developing an important new sector for our industry."

Workspace Documentation Facility WSDOC

A new workspace documentation facility—Wsdoc—is now available on the SHARP APL service. The new Wsdoc is much improved over its predecessor and offers many new features, including:

- A cross-reference table for each function, or of all identifiers used in the workspace
- A tree-structure diagram of the workspace's calling hierarchy
- A listing of every package's objects, including all nested packages

With WSDOC, you can analyse a workspace, and produce a report that:

- Gives the workspace's environment
- Lists the identifiers of all functions, variables, groups, and group members
- Gives a one-line description of all functions, variables, groups, and packages
- Lists every function and variable

WSDOC also allows you: to suppress or select any of this information; to limit analysis to certain objects; or to exclude certain objects from analysis.

You can use WSDOC as a *debugging tool* or as part of a system's *permanent documentation*. While you're building a system,

you can use Wsdoc reports to help you find bugs and redundancies. After you've finished a system, the Wsdoc report can show the system's maintainers and users how the system works internally. Both these uses of Wsdoc depend on Wsdoc's effective display of a workspace's objects and their interrelations.

WSDOC is easy to use. You store an image of the workspace you want to document on a **source file**. Then you specify how you want the workspace documented by resetting the Wsdoc **state**, and storing it on the source file. Then you run the Wsdoc.

You can get quite useful Wsdoc reports by simply using the default

NAME AND SYNTAX	LINES	SIZE	COMMENTS
Z+A PMAXS1 B	2	140	A PARTITIONED MAX SCAN
Z+A PNOT N	14	992	A RETURNS THE PERMUTATION VECTOR TO PERFORM A PARTITIONED N-ROTATE ON
Z+A PNOT B	3	216	A PERFORMS A PARTITIONED 1-ROTATE OF THE PARTITIONS IN AS DESIGNATED
Z+A PORR B	3	216	A PARTITIONED OR REDUCE
Z+A PPLR B	3	224	A PARTITIONED PLUS REDUCE
Z+A PPLS B	3	236	A PARTITIONED PLUS SCAN
Z+A PREVERSE B	3	268	A REVERSES THE ORDER OF THE PARTITIONS IN AS DESIGNATED BY <A>.

```

▽ Z+A PNOT N;I;J
[1] A RETURNS THE PERMUTATION VECTOR TO PERFORM A PARTITIONED N-ROTATE ON
[2] A A VECTOR WHOSE PARTITIONS ARE DESIGNATED BY <A>.
[3] A <N> MUST SATISFY (p,N) ∈ 1,+/A AND WILL BE SCALAR-EXTENDED IF NECESSARY.
[4] A EXAMPLE:      V ← 12?12 ⋄ A ← 1 0 0 0 0 1 0 0 1 0 0 0
[5] A              V ⋄ V[⋄+A PNOT 3 2 1]
[6] A              11 3 6 5 7 8 2 10 1 9 4 12
[7] A              4 5 1 2 3 7 8 6 10 11 12 9
[8] A              5 7 11 3 6 2 10 8 9 4 12 1
[9] A GLOBALS: (NONE)
[10] I ← (1⊕A)/⊖A ⋄ J ← +\A
[11] I ← I - 1 ⊕ 0, I ⋄ Z + ⋄ J + (I|N)[J] ≥ +\1 - A - 1 ⊕ 0, I
[12] A OR, IF N[I] ∈ I + . = +\P FOR ALL I ∈ I + P:
[13] A I ← A/⊖A ⋄ J ← +\A
[14] A Z ← ⋄ J + (,N)[J * p N] ≥ +\1 - A - I - 1 ⊕ 1, I
▽

```

A	LA	0	10	10	10	11		N	RA	0	11
I	LV	0	10+	11+	11	11	11	Z	RS	0+	11+
J	LV	0	10+	11	11						

Excerpts from sample WSDOC reports.

"What James Martin didn't tell you about APL"

Glenn McDavid and
Larry Mysz, Chicago

state. Or you can store your favourite state settings as a personal profile that takes automatic effect every time you use WSDOC.

You can use WSDOC not only to document any workspace, but also to document any arbitrary collection of functions, variables, groups, and packages.

WSDOC also lets you chain or merge several workspace image source files together. WSDOC can operate on a nearly full workspace.

Complete control of content and format

You can specify to WSDOC exactly what aspects and elements of a workspace it is to analyse, and the level of detail of its report.

You can also tell WSDOC how the report is to be formatted. You can control page width and depth, the number of columns, pagination, the sort sequence, page titles, and some important details of formatting. For instance, you can have function lines that contain statements separated by diamonds broken out onto several lines.

WSDOC normally produces its output at your terminal. However, you can have WSDOC printed on the highspeed printer as well.

You can also tell WSDOC to run in the background, as an N or B task. This frees your terminal for other work, and can cost substantially less.

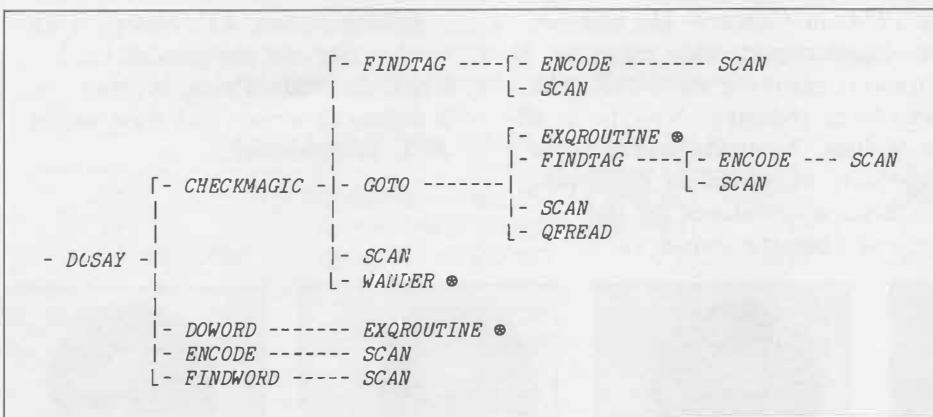
For more information on WSDOC, contact your I.P. Sharp representative; or load workspace 7 STATE, and type DESCRIBE.

Dr. Kenneth Iverson, founder of APL, addressed a luncheon on 15 June, hosted by the Chicago office of I.P. Sharp. Approximately a hundred people heard his talk "What James Martin didn't tell you about APL".

James Martin is one of the most prolific authors and best attended lecturers in the field of data processing. In his recent book *Application Development without Programmers*, he argues that traditional programming languages, such as COBOL and PL/I, are inadequate for meeting the current demand for computer software applications. Instead, new tools, offering a vast increase in productivity for end users as well as application developers, must be used. One of the newer tools he proposes is APL.

Dr. Iverson said that while Martin's discussion of APL was favourable, a much stronger case can be made for APL as a *general-purpose language*. Many companies are already using APL for large systems. In lists found in the proceedings of the various APL conferences are such familiar names as IBM, Xerox, Upjohn, and Fireman's Fund.

He also cited the I.P. Sharp public data bases as an example of APL's being used in a major application. I.P. Sharp's collection of public data bases is among the largest in the industry. However, the I.P. Sharp data base group is about a quarter of the size of any of its competitors. This is possible only because of the increased productivity of APL.



Tree-structure diagram

Coventry

Mike Butler, acting manager for Coventry, is now branch manager. He studied at Birmingham University in England, where he earned a B.Sc. in mathematics, a Certificate in Education, and a M.Sc. in mathematical statistics.

After teaching mathematics at the secondary school level, he worked as a statistician for Perkins Engines (a division of Massey-Ferguson). He then worked for Ford Motor Company as a statistician for their engineering and research centre in Basildon.

Mike initially joined I.P. Sharp's London office over three years ago, and then transferred to the Birmingham office which later moved to Coventry. Mike has been involved extensively in a variety of systems tailored to the needs of the motor industry in Coventry.

Edmonton

Don Williams, the new branch manager in Edmonton, hails from the Vancouver office where he worked for two years on a variety of applications.

He has a B.Sc. in mathematics and computing science from Simon Fraser University. But his studies have not been limited to this field, as he has also studied at the School of Architecture, University of Toronto. Don's interests include economics and financial applications with SHARP APL.

Frank Arthur has been the manager of the Edmonton office since 1977. He earned a B.Ed. degree at the University of Alberta, majoring in business and computing science. He later earned his M.B.A.

Frank started with I.P. Sharp in 1974, in Calgary. He has developed considerable expertise in financial planning models for the petroleum industry. Now he is off to Sydney, Australia where he is especially interested in developing financial applications for the natural resource industry.

Saskatoon

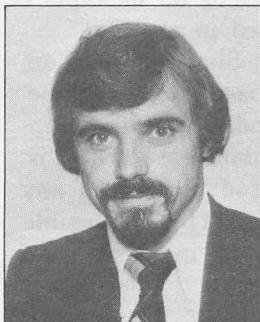
Ernie Briginshaw, acting manager for Saskatoon since January of this year, is now branch manager for that office. He graduated from the University of Waterloo with a Bachelor of Mathematics, majoring in computer science. Under the cooperative program, he worked in the Ottawa office as a student in the summer of '78.

Since joining I.P. Sharp, Ernie has concentrated primarily in financial modelling using STARS.

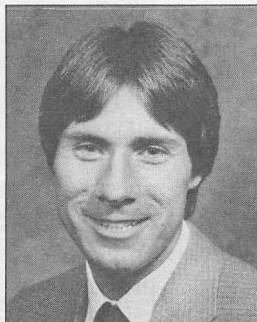
White Plains

Tom Hanlon is the new branch manager in White Plains. He received his B.A. and M.A. in English literature from the State University College at Potsdam and Stony Brook University, both in New York State.

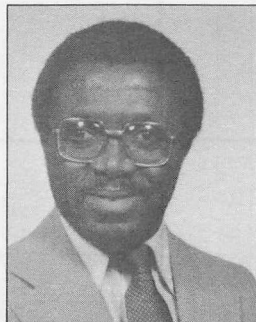
Before joining I.P. Sharp, Tom worked for the investment bank Blyth Eastman Paine Webber, as a technical writer and then as an APL programmer.



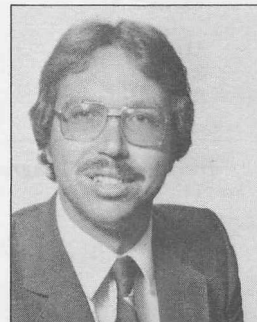
Mike Butler



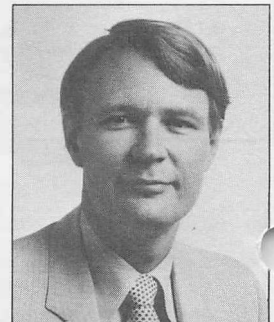
Don Williams



Frank Arthur



Ernie Briginshaw



Tom Hanlon

Japan APL Association
Reorganizes
JAPALA

The original Japan APL Association (JAPAL) was established in 1978 by a group of people who favoured APL. On 1 July 1982, this association was expanded and reorganized as an affiliate of the Foundation for Advancement of International Science (FAIS). FAIS is supported jointly by the Japanese Industrial Society and various universities.

The purpose of the new Japan APL Association (JAPALA) is to promote APL in Japan through the study, training and practical use of the language. Their activities include hosting conferences and seminars on APL, distributing information on APL, communicating with other related

groups in Japan and outside the country, and getting involved with other activities to achieve their goals.

The Board of Directors of JAPALA are:

Honorary Chairman:

K.E. Iverson, I.P. Sharp,
Toronto

Chairman:

J. Osada, Professor, Tokyo
University of Science

Vice-Chairmen:

J. Takeuchi, Instructor, Keio
University

M. Imai, Nikkei Newspaper Co.

Directors:

F. Tahara, Sekitsui Chemical
Engineering Co.

Y. Fujiwara, Professor, Tsukuba
University

K. Matsudaire, Nippon Systemix

K. Aoi, Intec, Inc.

S. Sakai, Micro Soft Associates

T. Kobayashi, Century Research
Center

T. Kubota, FAIS

If you wish to contact JAPALA,
the address is:

JAPALA
c/o Foundation for Advancement
of International Science
Kashiwaya Building, Nishi-
Shimbashi
Minato-ku, Tokyo
Japan

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.
- ☐ Please send me information about your courses in

Name: _____

Title: _____

Company: _____

Address: _____

(City) _____ Telephone: _____

The Newsletter is a regular publication of I.P. Sharp Associates. Contributions and comments are welcome and should be addressed to: I.P. Sharp Newsletter, Suite 1900, 2 First Canadian Place, Toronto, Canada M5X 1E3.

Irene Shimoda, *Editor*
Mary Kopfensteiner, *Circulation*

Printed in Canada
September 1982
ISSN 0226 854X



I.P. Sharp Associates Head Office:

Suite 1900, 2 First Canadian Place, Toronto, Canada M5X 1E3 (416) 364-5361

International Offices

Aberdeen

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

Amsterdam

Intersystems B.V.
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

Bristol

I.P. Sharp Associates Limited
5 Whiteladies Rd., Clifton
Bristol BS8 1NN, England
(0272) 732 918

Brussels

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

Calgary

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

Canberra

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

Chicago

I.P. Sharp Associates, Inc.
Suite 3860
55 W. Monroe
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, England CV1 1EA
(0203) 56562

Dallas

I.P. Sharp Associates, Inc.
Suite 1148, Campbell Center
8350 Northcentral 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
Leostrasse 62A
4000 Düsseldorf 11
West Germany
(0211) 57 50 16

Edmonton

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

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)
Kalevankatu 33 A
Mail to: P.O. Box 452
SF-00101 Helsinki 10, Finland
(9) 0-6946344

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

Kitchener/Waterloo

I.P. Sharp Associates Limited
3 Menno St.
Waterloo, Ont. N2L 2A4
(519) 884-5420

London, Canada

I.P. Sharp Associates Limited
Suite 510, 220 Dundas Street
London, Ontario N6A 1H3
(519) 434-2426

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 S.A.
(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) 221612

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
Mail to: 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
(415) 327-1700

Paris

I.P. Sharp Sarl.
Tour Neptune, Cedex 20
20 Place de Seine
92086 Paris-la-defense
France
(1) 773 57 77

Philadelphia

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

Phoenix

I.P. Sharp Associates, Inc.
Suite 503
3033 N. Central Avenue
Phoenix, Arizona 85012
(602) 264-6819

Rochester

(United States Headquarters)
I.P. Sharp Associates, Inc.
1200 First Federal Plaza
Rochester, N.Y. 14614
(716) 546-7270
Telex: 0097 8473
0097 8474

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

San Jose

I.P. Sharp Associates, Inc.
3028A Scott Boulevard
Santa Clara, California 95050
(408) 727-9446

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 Namdaemoo-Ro
Jung-G (CPO Box 2810)
8269 Seoul, Korea.
771-91/2

Singapore (Far East H.Q.)

I.P. Sharp Associates(S) Pte. Ltd.
Suite 1003, PiL Building
140 Cecil Street
Singapore 0106
Republic of Singapore
2210494
Telex: RS 20597 IPSAS

Singapore

Singapore International
Software Services Pte. Ltd.
Suite 1501, CPF Building
79 Robinson Rd.
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)
37-18, 3-Chome, Hatagaya
Shibuya-ku, Tokyo 151
Japan
(03) 320-2020

Toronto

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

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
Rennasse 4
A-1010 Wien, Austria
(0222) 66 42 48

Warrington

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

Washington

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

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 A.G.
Fortunagasse 15
8001 Zurich, Switzerland
(01) 211 84 24

SHARP APL Communications Network

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

Our private, packet-switched network connects with the Value Added Networks - Datapac. PSS. Telenet. and Tymnet - to provide access from the 35 countries listed below:

Argentina • Australia • Austria • Bahrain • Belgium • Bermuda • Canada • Chile • Denmark • Dominican Republic • England • Finland • France • Germany • Hong Kong • Ireland • Israel • Italy • Japan • Luxembourg • Mexico • The Netherlands • New Zealand • Norway • The Philippines • Portugal • Puerto Rico • Scotland • Singapore • Spain • Sweden • Switzerland • Taiwan • United Arab Emirates • U.S.A.

SHARP APL is accessible from over 500 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 Rochester, New York and Amsterdam nodes.