the I.P. Sharp newsletter

November / December 1979
Vol. 7 Number 6

TURING AWARD

Dr. Kenneth Iverson was recently honoured by the Association for Computing Machinery. He was selected by the ACM to receive the Turing Award, which is the highest award for technical contributions to the computer community. Dr. Iverson was cited for his pioneering efforts in programming languages and mathematical notations. Our sincerest congratulations.

APL TALK

Workshop at Asilomar promotes informal contacts among designers and implementors of APL systems

Fifty-nine people involved in the further development of APL got together November 22-26 for four days of discussion. They met at Asilomar, a conference centre located in a wooded setting on the Pacific coast. Participants came to Asilomar from 7 universities and 19 private companies located in five countries, but primarily from North America. IBM sent the greatest number of participants, accounting for about a quarter of the attendance; there were 11 from I.P. Sharp Associates.

The workshop was one in a series of invitational meetings sponsored originally by Garth Foster of Syracuse University, and later by Mike Jenkins at Queens University, Ontario, and STAPL. These workshops have provided a neutral ground at which individuals working on APL can get together with their counterparts at other institutions to exchange views. They provide a forum in which ideas about extensions to APL can be broached, and subjected to the scrutiny—often clamorous—of friends and rivals. Primarily, they provide an opportunity (for those who care to use it) to circulate proposals while they're still in their formative stages. This is especially important in the APL community in which there is a widely-shared commitment to advancing APL as a common language, mutually intelligible across a wide range of independent or even competing implementations.

IN THIS ISSUE

ASILOMAR

At this year's gathering, the only topics about which there was explicit agreement were the resolution to hold another session next year, and to send delegates to represent STAPL at such work as may be done on an international standard for APL, now being considered in Europe by the International Standards Organization.

Several sessions were devoted to one or other aspect of nested arrays; over the years, there has gradually been emerging a clearer consensus regarding the form that nested arrays should take and the functions necessary to work with them. Even now, sharp disagreements remain, and there are many points still to be settled. But it seems increasingly likely that nested array implementations will be attempted before long, and that their designers will have benefitted from discussions such as these.

One of the few topics on which there has emerged something close to unanimity is the extension of APL to complex numbers. Even there, some unresolved details persist, one of which is how complex values should be entered or displayed.

At one of the sessions, Ken Iverson, the original architect of APL, gave an advance presentation of the lecture he was to deliver a few days later while accepting the Turing Award, presented annually by the ACM in honour of an outstanding contribution to computer science.

STAPL EXECUTIVE

Newly Elected Officers of STAPL, (ACM's Special Technical Committee on APL), include two from I.P. Sharp Associates:

On the executive board: **Lib Gibson**
Chairman: **Gene McDonnell**

Other new officers are:

Ray Polivka, of IBM Poughkeepsie, well-known as author of an APL text, elected Vice-Chairman.

Marilyn Pritchard, of STSC, Bethesda, Md, and former editor of APL Quote-Quad, as Secretary-Treasurer.

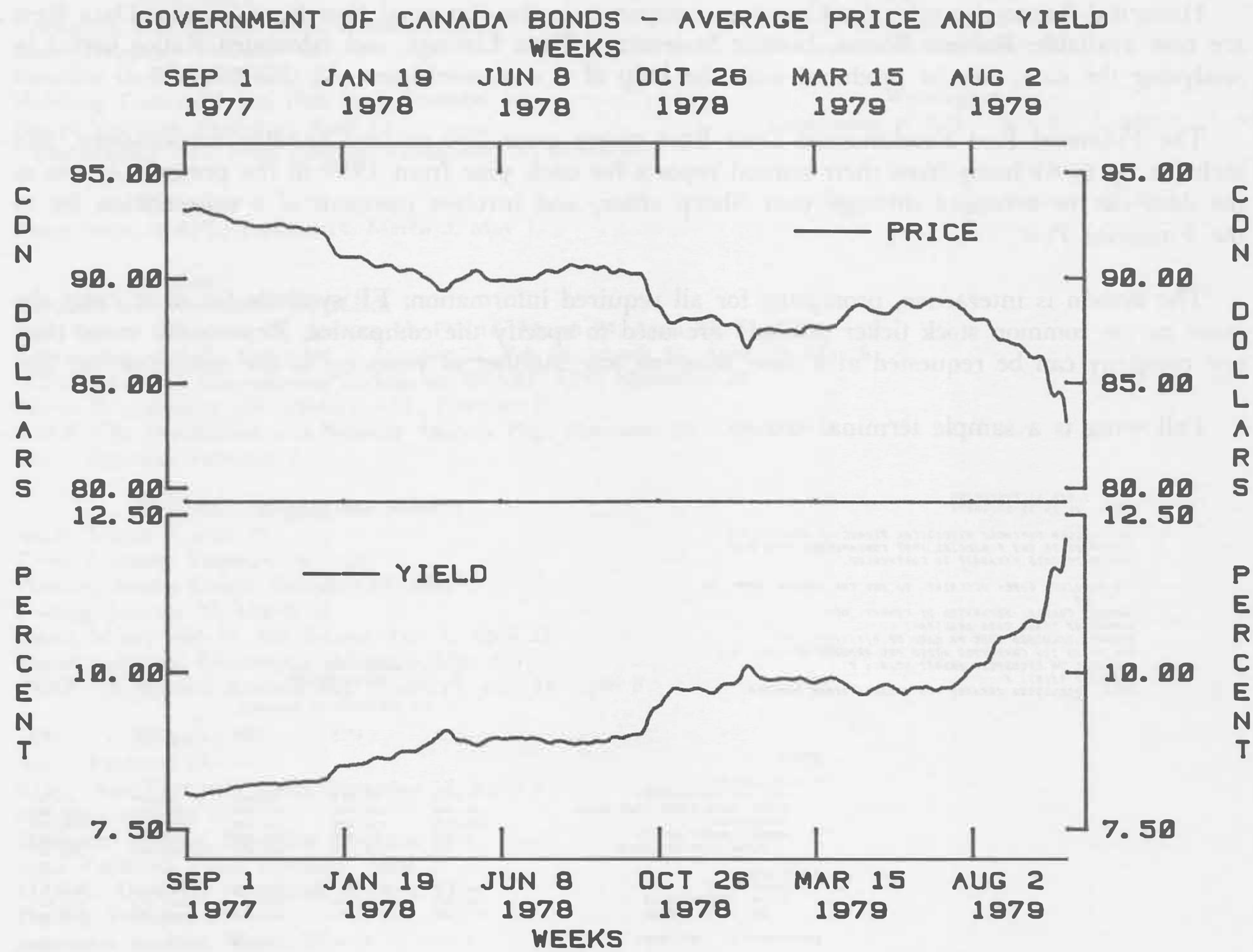
Garth Foster, of Syracuse University, and founder and first chairman of STAPL, as member of the executive board.

Bob Smith, of STSC, Bethesda, Md, currently leading their work in the area of nested arrays, and well-known as former problems editor of APL Quote-Quad and as 'Boolean Bob', as member of the executive committee.

Gene, as Chairman, sees work on an APL Standard as the most important thing to be accomplished over the next two years, and also wants to see more publications produced which serve the younger user of APL. Anyone with ideas as to how this can be done should communicate with him at the I.P. Sharp office in Palo Alto.

BOND YIELD VS BOND PRICE

Marc Odho, Toronto



The plot above graphically illustrates the effect of the recent increases in the Bank Rate by the Bank of Canada on the Canadian Bond Market. Bond yields have soared and prices have dropped. When interest rates rise, bond prices as a group decline, and consequently yields rise. As a result of the high interest rate, short and mid-term bonds are yielding a higher return than long-term bonds, contrary to the usual trend.

Current and historical weekly **bond price and yield statistics** are now available on SHARP APL. Data is available for approximately 900 Canadian Government, Provincial, Provincial Guaranteed, Municipal, Corporate and Foreign Government Bonds. Included are convertibles, extendibles, retractables, Euro-Canadian and Euro-U.S. issues. Historical price and yield data is available from September 1977 to the present. Matured and delisted bonds remain on the system but are no longer updated. The source for this data is Wood Gundy Limited.

The bond data base can be accessed through RETRIEVE, (the SHARP APL data base retrieval system), under the view name *BONDMARKET*. Bonds can be selected by characteristics such as coupon, maturity, issuer, issuer type, currency codes, and extendible, retractable and conversion privileges. Please refer to the View Directory (available from your local Sharp representative), or on line via the *RETRIEVE* function *ΔVIEWDOC*. For a complete description of RETRIEVE, refer to "RETRIEVE - A SHARP APL Data Base Retrieval System".

FINANCIAL STATEMENTS NOW AVAILABLE IN LIBRARY 51

Linda Zetterstrand, Toronto

Historical financial reports for Canadian companies in the Financial Post Fundamental Data Base are now available. Balance Sheets, Income Statements, Data Listings, and calculated Ratios useful in analyzing the data, can be produced with the help of a new workspace, 51 *FPREPORTS*.

The Financial Post Fundamental Data Base covers some 250 major Canadian corporations, and includes up to 81 items from their annual reports for each year from 1959 to the present. Access to the data can be arranged through your Sharp office, and involves payment of a subscription fee to the Financial Post.

The system is interactive, prompting for all required information. FP symbols (in most cases the same as the common stock ticker symbol) are used to specify the companies. Reports for more than one company can be requested at a time, covering any number of years up to the maximum on file.

Following is a sample terminal session:

```
      )LOAD 51 FPREPORTS
SAVED  21.36.32 10/26/79

THIS SYSTEM PRODUCES HISTORICAL FINANCIAL REPORTS FOR
COMPANIES IN THE FINANCIAL POST FUNDAMENTAL DATA BASE.
ACCESS TO THIS DATABASE IS CONTROLLED.

FOR DETAILS, TYPE: DESCRIBE. TO RUN THE SYSTEM, TYPE: GO.
      GO
COMPANY CODE(S)-SEPARATED BY COMMAS: MCL
NUMBER OF YEARS DATA REQUIRED: 5
REPORTS REQUIRED (ONE OR MORE OF B/I/L/R):  B
USE 80 OR 132 CHARACTER WIDTH FOR REPORTS(80/132): 132
TERMINAL OR HIGHSPEED PRINT? (T/H): T
PROCEED? (Y/N): Y
FILE: BALS103002 CREATED FOR BALANCE SHEET REPORTS.
```

MOORE CORP. LTD.
BALANCE SHEET

(IN THOUSANDS OF DOLLARS)

	1974	1975	1976	1977	1978
ASSETS					
CURRENT ASSETS					
CASH AND EQUIVALENTS	31,835	87,528	98,428	96,329	57,783
ACCTS. RECEIVABLE LESS ALLOW.	198,548	180,046	197,383	231,137	268,945
INVENTORIES	208,748	152,698	152,125	168,292	188,366
OTHER CURRENT ASSETS	4,262	3,833	5,039	5,927	5,911
TOTAL CURRENT ASSETS	443,393	424,105	452,975	501,685	521,005
FIXED ASSETS					
GROSS PLANT	432,517	460,038	475,559	497,008	519,617
ACCUM. DEPRECIATION	155,155	172,449	190,247	207,032	221,577
NET FIXED ASSETS	277,362	287,589	285,312	289,976	298,040
INVESTMENT IN AFFILIATES	3,842	7,298	7,926	9,203	11,530
INTANGIBLES				10,690	10,406
OTHER ASSETS	16,616	18,161	18,049	8,323	8,417
TOTAL ASSETS	741,213	737,153	764,262	819,877	849,398
LIABILITIES					
CURRENT LIABILITIES					
SINKING FUND AND DEBT	6,431	7,210	1,830	1,974	2,307
CURRENT LOANS	37,715	22,971	18,498	58,802	16,561
OTHER CURRENT LIABILITIES	137,171	106,205	116,904	135,260	160,110
TOTAL CURRENT LIABILITIES	181,317	136,386	137,232	196,036	178,978
MINORITY INTEREST	60,078	59,811	59,674	8,581	9,565
DEFERRED INCOME TAXES	38,571	37,263	40,158	42,286	48,825
LONG-TERM DEBT	93,248	92,082	90,417	90,780	96,614
OTHER LIABILITIES	9,601	14,333	12,642	18,120	7,038
TOTAL LIABILITIES	382,815	339,875	340,123	355,803	341,020
SHAREHOLDERS' EQUITY					
CAPITAL STOCK					
COMMON STOCK CAPITAL	33,698	33,706	33,770	33,770	33,178
CONTRIBUTED SURPLUS	(14,608)	(14,608)	(14,608)	(17,841)	
RETAINED EARNINGS	339,308	378,180	404,977	448,145	475,200
TOTAL SHAREHOLDERS' EQUITY	358,398	397,278	424,139	464,074	508,378
TOTAL LIABILITIES AND SHAREHOLDERS' EQUITY	741,213	737,153	764,262	819,877	849,398

As indicated above, on-line documentation is available in the variable *DESCRIBE*. For more details, please read the publication "Financial Post Data Base", available from your local Sharp office.

NONLINEAR PARAMETER ESTIMATION

Mike Powell, Victoria

A new capability has been added to the SHARP APL statistical package - nonlinear parameter estimation. Existing software in the public libraries deals fully with the many techniques for regression with linear models (e.g. 32 *REGRESSION*) and with single equation nonlinear models (e.g. 32 *NONLINEAR*). The workspace 32 *NPE* goes a step further and deals with nonlinear multi-equation models.

A distinguishing characteristic of such models is that the equations are "linked" - usually because one of the unknown parameters appears in several of the equations simultaneously. A simple example might be,

```
GASALTA[T]   =  P[1]×GASALTA[T-1]×(TEMPALTA[T]÷TEMPALTA[T-1])*P[3]
GASONT[T]    =  P[2]×GASONT[T-1]×(TEMPONT[T]÷TEMPONT[T-1])*P[3]
```

which relates this year's consumption of natural gas in two Canadian provinces (*GASALTA* and *GASONT*) to last year's consumption taking into account some measure of the coldness of the winter (*TEMPALTA* and *TEMPONT*). The model has three unknown parameters, *P*[1 2 3], of which the third appears in both equations (the effect of temperature on gas consumption is the same in both provinces). If *P*[3] were estimated separately in each of the two equations then, almost certainly, we would end up with two different results! Such equations must therefore be estimated simultaneously and the correct theoretical technique for such models is known as **maximum likelihood** estimation (which is not always the same as least squares).

The workspace 32 *NPE* provides a reliable and efficient method of dealing with such problems and interested readers should contact their local Sharp office (or Mike Powell, I.P. Sharp Associates, Victoria, Canada) for complete documentation or further information.

THE USE OF APL IN AIRLINE OPERATIONS

Roberto S. Skertchly
Joey K. Tuttle
Palo Alto

The use of the term "airline operations" is intended here to include most of the decision processes used in operating an airline. A very simplified view of airline management might involve the following steps in an iterative process:

Market analysis
Flight scheduling
Daily crew work assignments
Monthly crew work schedules
Crew tracking and reporting
Operations reporting
Management reports
Market analysis
.
.
.

This sequence is repeated typically on a monthly basis, although some of the steps may be performed more, or less, frequently. Many other aspects of operating an airline are directly related to one or more of these steps. The published passenger schedule is one form of the results of flight scheduling. The equipment flow chart used to plan flight crew requirements is another form of the same data. The published schedules provide the data for passenger reservation systems. The daily equipment and crew assignments are used to file flight plans with Air Traffic Control centers. One result of crew tracking is input to the payroll process for flight crews, and so on.

We have been working with Air California to provide decision support systems for certain aspects of the management process. In particular, a data gathering and reporting system was created by David Osekavage and Ettie Ettinger of the Newport Beach office, and has been in use for some time now. The reporting system is used to collect operational statistics and prepare management reports to aid in market analysis and flight schedule planning.

Crew Scheduling

We have also been studying the manual process of crew scheduling used at Air California. During the initial study we spent a week observing the activity of crew schedulers while they prepared a new quarterly work schedule. As a result of the study, plans were made for a set of tools to assist crew schedulers in obtaining a machine solution to the problem of scheduling work for crews. Parts of a decision support system have been implemented and results from the system were used in preparing the September 1979, and subsequent, schedules at Air California. The system has already demonstrated potential for significant cost savings. For example, more efficient work schedules are produced in much shorter times.

The design of the crew scheduling decision support system has evolved around the simplified description of airline management given above. The problem is similar in many respects to more general scheduling problems. However, the constraints on solutions are more complicated and restrictive. The main constraints are provided by the crew contract (management - crew work agreements) and Federal Air Regulations (FARs). Both sources provide lengthy and rather complicated interrelated definitions of "legal" work scheduling rules.

A bit of research into the history of computer solutions to this problem shows that large amounts of programming effort and machine resources have been devoted to solutions with varying success. Computers are widely used in report and inquiry oriented aspects of crew tracking and operations reporting. Various tools for checking the validity of manual solutions to the work assignment problem are in use and some major airlines make varying amounts of use of more complete machine solutions.

A well tested approach to solving the problem of creating legal and minimum cost work sequences or "crew rotations" is discussed by Rubin [1]. The approach taken by Rubin is to start with an initial solution to the scheduling problem and then apply techniques of linear programming to arrive at an optimal (minimum cost) solution. Additional refinements to the techniques for solving the problem were included in a presentation to the Airline Group of the International Federation of Operational Research Societies (AGIFORS) by Zob [2].

Zob observes that the traditional complete set-covering solutions require large amounts of machine resources. Furthermore, the set-covering solutions have shortcomings in being unable to adapt to the complicated constraints of the problem. Zob makes the very interesting observation that the problem can be partitioned into smaller problems by considering one domicile (home base) at a time and solving each such subset of the entire problem separately. The insight that each domicile can be treated separately would seem to be based on an analogy to physical conservation laws and might be stated as **the law of conservation of crews**. The definition of a domicile is that a set of crews originate and terminate their work sequences there. In the case of Air California, this was not a part of the problem since they operate from only one domicile.

The problem of interest to Zob was international "long haul" air carriers. That interest results in the basic assumption that flights visit each airport relatively infrequently. The operation at Air California is very much the opposite. Currently they serve about 15 airports with 12 aircraft and the longest scheduled flight is just over one hour. All of the airports served are visited several times daily. However, the idea of breaking the scheduling problem into steps that could be solved by heuristic methods encouraged us to seek further ways to view the Air California problem.

Our original work on rotation building followed the traditional approach of an initial "rough" solution followed by optimization. We found (not surprisingly) that this approach resulted in time-consuming calculations. Since we work with APL and prefer to construct highly responsive and interactive solutions to problems, we tried to reduce the amount of computation required to solve the problem. New insight into the crew scheduling problem came as another analogy to conservation laws: **the conservation of aircraft...** The point here is that the flight schedule for daily flight activities in a short route and high activity situation can be viewed as a collection of "equipment rotations" where each aircraft is likely to end an operating day at a domicile airport from time to time during the schedule period. If the scheduling problem is first divided into several such equipment rotations then each rotation can be considered for possible crew rotations within it.

We abandoned the original two-part approach to the problem and turned instead to the techniques of Dynamic Programming [3], applied to the equipment rotations. Taking into account the work rules dictated by the crew contract and the FARs, all of the viable solutions are generated for each subset of the larger problem. After the table of solutions has been generated, a solution which meets the goals of optimization can be selected. That optimal solution is then added to the cumulative results of the solution. The complete collection of crew rotations (called a Crew Finder at Air California) is built up in this manner. The current optimization goal at Air California is to generate crew rotations which minimize the amount of non-productive crew usage or "pay credits."

By using dynamic programming techniques after dividing the problem into equipment rotations, the solution time is approximately linearly related to the number of rotations generated to satisfy the schedule. Set-covering linear programming techniques generally take solution times which increase exponentially with the size of the problem.

In addition to the advantages of APL, dynamic programming provides several key advantages such as very easy ways to include any sort of constraint, multiple solutions from one series of computations, and greatly reduced computational time and storage resource requirements. The process of breaking the crew scheduling problem into small manageable parts and applying dynamic programming techniques provides adaptable solutions at very modest costs.

There are typically several cases where more than one aircraft ends an operating day at a non-domicile airport. In those cases a choice can be made as to which aircraft should be used to continue the equipment rotation the following day. Currently, the aircraft which ends the day at the earliest time is the one leaving earliest the next day. This assumption is optional and a different solution to the entire problem will result if other choices are made. During early experimentation, we observed that our current default choice consistently resulted in a more optimal total solution than other options. However, since our choice was based on empirical observations, we included facilities in the system to experiment with other choices. This flexibility has proved valuable (allowed us to produce more efficient solutions) in more recent schedules.

Work is in the final stages for combining the crew rotations into monthly work schedules for the crew members. The technique originally planned for use in that problem was the Ford-Fulkerson "out-of-kilter" algorithm and network models described by Nicoletti [4] and also by Zob. The problem of work schedules is a little more straightforward than rotation building and involves selecting the work to be accomplished on a given day in such a way that will assist in meeting end-of-month goals, e.g. work hours, days off, weekend work time, and overnight trips. The entire set of work schedules is presented to the flight crews and they "bid" for the monthly work schedule of their choice.

The success in applying dynamic programming to the crew rotation problem led us to pursue the same technique to assign rotations to the monthly work schedules. This approach proved to be unsatisfactory. Our final approach was a specialized one developed with the help of Dantzig and the methods set forth in his book [5] for optimal assignment. One phase in this process is the construction of the cost matrix used for assignment. Each cost weights the advisability of pairing a crew-member with a given rotation. The cost is a function of a set of objectives and constraints. We have found that identification of **the** appropriate cost function is very important. The assignment process is extremely sensitive to this function.

It is our goal to create an interactive crew tracking and operation data collection system which will allow closing the loop back to the market analysis and schedule planning steps. We would like to emphasize that the decision support systems we are constructing are quite different from the traditional data processing approach used in most airlines. The intent is to provide a collection of computerized tools for the end users (schedulers, dispatchers, and planners as well as management) to use directly. We are avoiding the usual data processing staff and operations in this approach. In view of that, it should be noted that extensive use of on-line files makes for a very flexible system in the ways that past data can be integrated into current work and changes can be introduced manually into the machine solutions. In short, we are attempting to provide a real-time decision support system instead of the more traditional batch job oriented systems usually used for this kind of work.

The primary consideration in our design of airline operations decision support systems is to provide a set of interactive computer tools to be used by people with experience and intuition to better manage the operation of their airline. It is very important that changing goals and operating conditions can be easily incorporated into the decision support systems. APL provides an ideal vehicle for constructing adaptable, dynamic, and interactive systems.

References

[1] Rubin, Jerrold, "**A Technique For The Solution Of Massive Set Covering Problems, With Application To Airline Crew Scheduling**," IBM Philadelphia Scientific Center Technical Report No. 320-0334, September 1971

[2] Zob, Almos P., "**International Airline Crew Scheduling Model**," Boeing Computer Services Company, AGIFORS Symposium, Washington D.C., May 1979

[3] Larson, Robert E. and Casti, John L., "**Principles of Dynamic Programming, Part 1**," ISBN 0-8247-6589-3, Marcel Dekker, Inc., 1978

[4] Nicoletti, Bernardo, "**Automatic Crew Rostering**," Transportation Science, Vol. 19, 33-42, February 1975

[5] Dantzig, George B., "**Linear Programming and Extensions**," ISBN 0-691-08000-3, Princeton University Press, 1963, pp.404-412

Technical Supplement–23

BRANCHING IN SHARP APL, PART II

Robert Metzger, Rochester

In the first part of this series we explained some reasons why we frequently look for means to avoid branching. We also looked at the concept of 'boolean conditional' statements as a means of expressing ALTERNATIVE logic without branching. In this article we explore two further methods for avoiding the use of branching.

The second method for avoiding branching is:

USE COMPRESSION

The result of a compression operation, and the result of a conditional branch statement both depend on boolean expressions, ie. expressions which return 0's and 1's. Thus, it is often possible to replace branching with compression.

Often, it is necessary to print messages based upon a test. This can be done with branching, or more directly with compression. A statement of the form:

If test, Then print 'MESSAGE1', Else print 'MESSAGE2'

can be constructed with branches as the following:

```
IF:→(~TEST)/ELSE
THEN: ⎕←'MESSAGE1'
→IFEND
ELSE: ⎕←'MESSAGE2'
IFEND:
```

can be much more easily constructed as:

```
⎕←(⍲\2⍴TEST)⌿(⎕PW↑'MESSAGE1'),[.5] ⎕PW↑'MESSAGE2'
```

(Note that `⍲\1 1` is `1 0` and `⍲\0 0` is `0 1`).

In a similar manner, selection and assignment of data based upon a boolean expression can be expressed in terms of compression, rather than branching. A statement of the form:

If test, Then `VAR[...;;;INDEX;;;...]←SCALAR`

can be constructed with branches as the following:

```
IF:→(~TEST)/IFEND
VAR[...;;;INDEX;;;...]←SCALAR
IFEND:
```

but can be much more easily constructed as:

```
VAR[...;;;TEXT/INDEX;;;...]←SCALAR
```

An example of the use of such a statement would be:

```
MATRIX[(MATRIX[;1]<0)/⍳1↑⍴MATRIX;]←0
```

which would set to 0 all rows of a matrix in which the first column was negative. Here we make use of the fact that assigning a value into an array indexed by an empty vector causes no change. If none of the rows contains a negative number in the first column, nothing will be changed.

The third method for avoiding branching is:

USE THE EXECUTE FUNCTION

The execute function (⍎) is very powerful. It makes it possible for a program to construct character arrays which represent APL statements, and then use the execute function to do the statements represented by the array.

One of the most common ways to use execute instead of branching is with a substitute for a simple ALTERNATION of the form

If test, Then do statement(s)

which would be written with branches as

```
IF:→(~TEST)/IFEND
STATEMENT1
STATEMENT2
     .
     .
     .
IFEND:
```

but is written more concisely with execute as the following:

```
⍎(TEST)/'STATEMENT1 ◇ STATEMENT2 ◇...'
```

The execute function does the APL statement(s) that the character vector argument represents. The convention is adopted that giving ⍎ an empty vector means 'do nothing'. So this statement either executes all of the character vector containing the statements, or it does nothing. The advantage to this type of construct is that it keeps the condition upon which the execute depends right next to the statements which will be executed. This prevents the bad habit that some programmers have of letting the statements related to a test drift away from the test itself. It has the disadvantage that we can't put in statements to be executed if the test turns out to be false.

An example of the use of such a statement would be

```
      ∇RESULT←LEFTARG FUN RIGHTARG
[1]   ⍎(0=⎕NC 'LEFTARG')/'LEFTARG←DEFAULT∆VALUE'
```

This statement would be the first executable statement in a user-defined function which could be called either monadically or dyadically (See SATN 25). If the function was called monadically, *LEFTARG* would be undefined, and it would be assigned its default value.

If it is necessary to provide statements to be executed when a test is false, as well as if it is true, it is necessary to use some simple defined functions.

```
      ∇C←X IF C
[1]   ⍎(0≠⎕NC 'X')/'C←C/X'
      ∇

      ∇EXPR←TEST THEN EXPR
[1]   EXPR←,(⍲\(⍴⍴EXPR)⍴TEST)⌿EXPR
      ∇

      ∇EXPR←TRUE ELSE FALSE;WIDTH
[1]   WIDTH←(¯1↑⍴TRUE)⌈(¯1↑⍴FALSE)
[2]   EXPR←(WIDTH↑TRUE),[.5] WIDTH↑FALSE
      ∇
```

Using these functions, it is possible to write:

```
⍎IF test THEN 'STATEMENT1 ◇ STATEMENT2 ◇...'
```

or

```
⍎ IF test THEN 'STATEMENT1 ◇...' ELSE 'STATEMENT101 ◇...'
```

These functions are also used in branching, when it is truly necessary, and their use is further explained in a later article in this series.

There is one drawback to this method - it is limited to those ALTERNATIVE structures which can fit on one program line. It should be noted that some APL users employ another means of executing IF-THEN-ELSE constructs, which has an even worse drawback.

```
⍎8 ¯11[⎕IO+FLAG=1]↑'TRUE←123 FALSE←45678'
```

will set `TRUE←123` if `FLAG=0`, or `FALSE←45678` if `FLAG=1`. The problem with this construct comes when you wish to modify the statements. Not only will the assignment need to be modified, but also the two numbers (`8 ¯11`) which control the take operation. Since these statements require twice the effort to maintain, and are subject to human error in counting characters, they are rarely a good substitute for branching.

Not only is it possible to build ALTERNATIVE structures using execute, but also simple ITERATIVE structures can be created. The function listed below will execute the statements represented by the character vector right argument as many times as specified by the left argument.

```
      ∇TIMES EXECUTE EXPR
[1]   ⍎(TIMES×1+⍴EXPR)⍴EXPR,'◇'
      ∇
```

Thus, doing:

```
10 EXECUTE '(⍳0) ⎕APPEND 99'
```

would append 10 empty vectors to the file tied to 99. This method has the drawback that there is no loop counter to refer to, and no means of initializing variables used in the loop is possible.

Considering the number of possible uses of the execute function for branch substitutes, one can say that in APL, unlike chess, the pawn symbol (⍎) is one of the most powerful pieces.

We have considered three methods for writing program logic without branching: boolean conditional statements, compression, and the execute function. Future articles will deal with subfunctions, recursion, and other more advanced programming techniques for building control structures.

TEESON'S TEASER

Peter Teeson, Toronto

The published answers to **Teaser 1** prompted a number of you to write (July/August 79, TS 21, page T5). Your responses were especially interesting in their comments on elegance". Here are a few samples:

> ...(6) is elegant because it only refers to *V* once ..."
> ... (2) is most elegant because it looks like what it is ..."
> ... (2) is most elegant because it's as cheap as (1), **and has fewer symbols.** "

P.S. Solution (4) had an error and should have read: `0 ¯1 ↓ V,[⎕IO+.5]V`

Teaser 2 (Vector insertion, see July/August, TS 21 page T5) generated quite a bit of mail, some even outside 666 *BOX*.

Everybody seemed to come up with solutions using ⍋ and of the form:

(1) `(S,V)[⍋((⍴S)⍴I+⎕IO-1),⍳⍴V]   ⍝ BEFORE`

but they all broke down when tested at the boundary conditions of first or last element.

One of you sent in:

(2) `(-I)⌽S,I⌽V`

and a little more time might have led him to this strange table:

(3)

	Before *I*	**After** *I*
`⎕IO←1`	`(-¯1+I)⌽S,(¯1+I)⌽V`	`(-I)⌽S,I⌽V`
`⎕IO←0`	`(-I)⌽S,I⌽V`	`(-1+I)⌽S,(1+I)⌽V`

Upon further reflection we realize that we can generalize two of these answers:

```
(4a)   (⎕IO-I)⌽S,(I-⎕IO)⌽V     ⍝ BEFORE
(4b)   (-I+~⎕IO)⌽S,(I+~⎕IO)⌽V  ⍝ AFTER
```

and SJT came up with the similar solution of:

```
(5a)   (X↑V),S,(X←I-⎕IO)↓V    ⍝ BEFORE
(5b)   (X↑V),S,(X←I+~⎕IO)↓V   ⍝ AFTER
```

Too bad the general answers look so ugly.

TEASER 3

All of us must be familiar with the idiom:

`((V⍳V)=⍳⍴V)/⍳⍴V   ⍝ V IS A VECTOR`

(a) Is there a better way?

(b) Is there an elegant solution extendable to a matrix
(1) where we want unique values for each row?
(2) where we want unique rows?

(c) Extend (b) to any array.

(d) What is your definition of the problem when extended to rank ≥2 arrays?

Thanks to C.J.J. Beart of Lloyds London for the idea.

By the way, the Teaser is an occasional column, and your comments and suggestions are solicited and welcomed.

SYSTEM REVIEW II — SYSTEM SECURITY

SHARP APL provides several resources to help you maintain the security and integrity of your systems and data, and to help you monitor your usage. On-line information is available, so that you can print reports on system reliability, User Log reports that help to monitor security violations, Sign-off History reports, and Usage reports for usage control.

System security is a major concern of time-sharing coordinators. It is still subject to misunderstanding and neglect. Popularly, computer security involves considerable protection against violent skulduggery. Such threats do exist, and protection is important. However, in addition to these more easily understood threats, a number of others also exist. When these other threats are mundane more than exciting, we sometimes ignore them.

Conventional protection at the I.P. Sharp data centre includes an ionization detection system, restricted physical access, duplication of equipment, extensive system backup, off-site storage, and continuous staffing. The SHARP APL system provides for passwords on user numbers for sign-on and passwords for access to workspaces. It also provides extensive facilities for the control of access to files. Using these access controls, a file owner can specify precisely how each other user may access the file or workspace (see "The SHARP APL File System" manual, and the flyer "System Security" for details).

Many software security facilities, especially those enabling the user program-control of access, are unavailable in conventional time-sharing systems. Their effective use requires proper planning and programming in an application system. They are essential for the protection of highly sensitive data and programs.

Monitoring Sign-on — USERLOG

SHARP APL records each unsuccessful attempt to sign on or to access a workspace with an incorrect password, and each failure to access a file due to insufficient permission. The log containing these records is known as the USERLOG. We consider the maintenance and analysis of this log a very essential function of the SHARP APL timesharing system. Remember that USERLOG records no successful access attempts. In theory, a successful security intrusion is undetected. This is not often the case in violent skulduggery, but it can be the case in an intrusion through a software weakness. **Sensitive user application programs** must record each successful access if necessary. This can be very important.

USERLOG is much more than a record of attempted security violations. For the most part, it documents instances of improperly-maintained access control, unfortunate lapses of memory, and poking about by curious employees. These highlight various system weaknesses and sloppiness.

The IPSA data centre will report any large number of USERLOG entries to the local I.P. Sharp office. A typical example might be the dozen attempts made one day to sign on to a particular user number with an incorrect password. (Considerably more attempts of a trial and error sort would be needed for success!) You can arrange to access USERLOG on-line with a request to your local Sharp office.

Protection of User Numbers

When creating a new user number, the I.P. Sharp data centre generates a random number which has not been used previously. Some customers request a set of sequential user numbers. This presents a problem. In this case each user knows many or all of the numbers. It is, of course, easier to guess a user number successfully if you know one of them, and 'guessing games' with 'likely' passwords have a better chance of succeeding. We also discourage the allocation of a single user number to many people for many purposes. This practise can allow a problem in one system to affect another; it can make proper maintenance (such as removal of obsolete files) difficult; can result in the storage of and payment for useless information; and it is a needless security exposure. If you lose track of which user number corresponds to which person or which application, you may ask us to "system lock" the number. No-one will be able to sign on that number, and the user usually quickly contacts the time-sharing coordinator. The number can be enabled again once the problem has been straightened out.

Passwords

Password protection is a common access control device, but is effective only if you change your password from time to time. It may reveal multiple users of the same number who did not know of one another, or show that an unauthorized user exists. Do not use the same password for more than one user number, or attach a sign-on password to a workspace as well. Each of these has been a source of needless trouble, including unauthorized access. Passwords should not be easy to guess. The name of your spouse, employer, or your mailbox code should probably not be a password. A password which is easily remembered is sometimes easily guessed.

The only purpose of a password on a user number is to enable sign-on. Likewise the only purpose of a password on a workspace is to enable access to it. No user under any circumstances should reveal a password for any other reason. **Carelessness in revealing passwords is very dangerous.** If you forget or lose track of a password - which is more likely if you change passwords frequently - contact your local IPSA branch with a request for a new one. Verification of the request can cause delays of hours, even days. Security can sometimes be inconvenient! Breaches of security can, however, be much more inconvenient and cause much greater delays.

File Access

It is just as important to record successful accesses to **sensitive** files as it is to record successful accesses to systems. If the cost of doing this can not be justified, it is important to at least keep track of which users have what access permissions to the file.

At times, someone may need access to a sensitive file just for a little while, say during the developing and testing stage. It is important to remove access permission when a new system goes into production. It is easy to add permission for subsequent modifications when, and only when, needed. It is good practise to review the access permission for sensitive files on a regular basis. Permission should be based on the need to use the file, and should only include the exact permission required. If there is some doubt about whether a particular access is essential, we recommend that you do not grant it. This usually results in a good explanation of the reasons for access if there are any. Overly broad permission is a constant source of serious problems. In systems which do not record successful accesses, there are often users with access permission who do not in fact make access to the file. The practise of removing access permission which is not obviously necessary is a good way of weeding these out.

Security Breaches

It is not possible to completely eliminate unauthorized access. Breaches do sometimes occur. If you do come across a violation, please **promptly** notify your local IPSA branch and the data centre. If it seems likely that a security breach might have occurred, even if there is no proof, we suggest you treat it as a security breach until the evidence shows otherwise.

NOW READ.....

Announcements of new data bases, applications packages, or improvements to existing ones, are usually followed by "... for further information, please read ....".

A complete list of titles, dates and prices appears on the **Publication Order Form**. Ask your local Sharp representative for a copy. The information is also available on-line in workspace 707 *MANUALS*.

User documentation at IPSA is available in several formats:

Fact sheets sum up one topic at a glance.

Revised recently:
Box-Jenkins
Financial Analysis
International Offices
Packages
Software/Distributed Program Product Prices

New:
Energy Data Bases

Brochures outline the scope of the package without too much technical detail.

New:
OAG
Distributed Program Product

Revised:
I.P. Sharp Corporation

View Directories for data bases associated with the *RETRIEVE* system.

New:
Bond Market

Manuals give a great deal of technical detail: all you need to know to use (and tailor) the software.

New:
Commodities Data Base
AGDATA

Revised:
Currency Exchange Rates Data Base
Report Formatting

.....THE BOOK.

The **SHARP APL Reference Manual**, by **Paul Berry**, is an attractive, illustrated, 349-page book that answers questions about the use of APL for a wide range of people. Its intended audience includes both the occasional user of APL and the professional programmer, and people at various levels in between. It will be a handy source for looking things up when you have a question about how something works. A description of each of the functions primitive to SHARP APL is contained in the manual. Among the many topics covered are:

- Syntax of APL
- APL Primitive Functions and Operators
- The Workspace Environment
- Structure of Data
- Shared Variables
- SHARP APL File System
- Event Trapping
- System Functions for Managing Files
- System Functions for Function Definition
- System Functions for Manipulating 'Package' Data
- Meaning of Error Messages and Trouble Reports
- Line Editing
- Modes of Interaction at the Terminal
- Arbitrary Input and Output at Terminals

Updates to the manual are on-line in workspace 1 *REFERENCE*.

SHARP PEOPLE

IN THE MIDLANDS

Support and service in the midlands will now be supplied from COVENTRY:

I.P. Sharp Associates Limited,
7th Floor, B Block,
Coventry point, Market Way,
Coventry CV1 1EA
Telephone: Coventry (0203) 21486

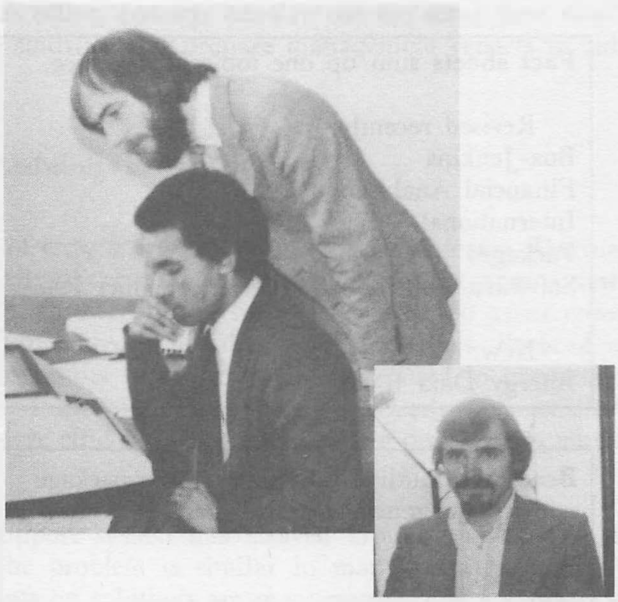
Midlands computer dial-up numbers:

Birmingham	021	- 359 - 6161
Coventry	0203	- 555201
Oxford	0865	- 724031

Bernard Perkins, the Branch Manager, is an honours graduate in electrical and electronic engineering. After spending some time investigating the use of APL in hardware design he joined IPSA and is now interested in more general applications of APL.

Mike Butler, Consultant, joined Sharp in January. He became interested in APL while working as a statistician at Perkins Engines. He has a B.Sc. in mathematics and an M.Sc. in statistics. He is interested in the field of reliability statistics and is working on a reliability package.

Abder Guessoum joined Sharp 4 months ago as a consultant. He has an M.Sc in transportation from Birmingham University as well as a B.Sc in ecomomics. He is trilingual and can offer consultation in English, French and Arabic.



Perkins, Guessoum and Butler

NORWAY

Carol Aitchison

As of November 1, **Carol Aitchison** is the new Branch Manager of the OSLO office. She takes over from David Bonyun, who opened the office last summer and has now returned to Ottawa. Carol has a degree in Mathematics from the University of Waterloo. After graduating, she joined what is now T.I.L. Systems where she spent two years working on the computerization of Stock Market information. For the past five years, Carol has been with I.P. Sharp Associates in Vancouver. Her primary involvement has been in customer support and applications programming.

Visitors are welcomed to the office, which is situated at Kronprinsesse Marthas Plass, 1, (4.Etasje). Mail should be addressed to Postboks 1470, Vika, Oslo 1.

SEMINARS and SPECIAL COURSES

Aberdeen
Appreciation of APL, February 21, April 8

Birmingham
Formatting with SHARP APL, November 30
Event Trapping, (half-day) May 10
Handling Output, (half-day) February 9
Handling Terminal Input, (half-day) December 1
Shared Variables, (half-day) April 12
Using SHARP APL Files, (half-day) December 17, March 8

Gloucester
Appreciation of APL, January 14, March 3, May 12

London
Advanced APL, November 29, March 17
APL Review, December 12, January 15, Feb 18, March 24, April 23, May 27
Appreciation of APL, December 3, January 21, Feb 26, March 21, April 10, May 8
CONSPACK - A Consolidation Package on SHARP APL, November 26
Linear Programming with SHARP APL, February 29
SNAP - An Introduction to a Network Analysis Pkg., November 28
Use a Terminal, February 7

Montreal, (English and French)
AIDS, March 4, May 20
Event Trapping, February 26, April 29
Planning System Design, February 19, May 6
Plotting, January 22, March 18
Saving Money with N- and B-tasks, Feb. 5, April 22
Shared Variables, December 4, March 6, May 8
SNAP - A Network Analysis Pkg. (halfday), Jan. 23, April 8

Ottawa
AIDS, February 19
Batch (Non-Terminal) Tasks, December 18, April 22
Box-Jenkins, June 19
Highspeed Printing, November 22, April 24
Input Validation (Idiot Proofing), April 23
MAGIC, December 20, March 18, June 17
Plotting, February 21
Regression Analysis, March 20

Rochester
APL Utilities, (4 half-days) February 7, 14, 21, 28

Toronto
Advanced APL Coding Techniques for Actuarial Applications, Dec. 10, Feb. 7, May 1
Advanced APL Topics, Jan. 15-17/21-24, April 8-10/14-17
(enrol for all or part of 7 sessions)
AIDS - Introduction, December 4-5
APL for Managers, Jan. 14, Feb. 25, Apr. 4, May 19
Batch Input Validation, Jan. 17, April 10
Blessings of Booleans, Jan. 22, April 15
Box-Jenkins, Jan. 10, March 6, May 8
Data Base Design, December 20
Data Representation, Jan. 24, April 17
Data Structures/Shared Variables, Jan. 15, April 8
Debugging Techniques, Jan. 23, April 16
Event Trapping, (halfday), Jan. 16, April 9
Forecasting Methods, Jan. 11, March 7, May 9
MAGIC for Time Series Analysis, Feb. 11, April 3
Regression Analysis, December 14, Feb. 13, April 23
Report Formatting, December 28, Feb. 21, April 25
Restartability & Secure Systems, Jan. 21, April 14
Saving Money with N- and B-tasks, December 6, Feb. 12, April 24
SUPERPLOT & MAGIC, Feb. 15, April 11
Text Editing, February 26-27, May 15-16
The Sharp System: A Practical Approach, Jan. 17, Feb. 14, March 13

Vienna (German)
Packages als Datentyp, November 21
MABRA: Ein Management-Informationssystem, Nov. 28
Event Trapping, December 5
APL 1, December 11-13
Die αω-notation, December 19

Warrington
Appreciation of APL, December 5, March 13, May 30

Washington DC
Data Base Design, December 14 (1 day)

INTRODUCTION TO APL

Aberdeen
(3 day)
March 5-7
May 21-23

Birmingham
(3 day)
December 11-13
January 22-24
March 18-20
May 20-22

Gloucester
(3 days)
February 4-6
April 28-30

London
(3 days)
December 5-7
January 7-9
February 12-14
March 10-12
April 15-17
May 13-15

Montreal
(English)
Jan. 15-17
March 11-13
May 13-15
(French)
December 11-13
February 12-14
April 15-17

Ottawa
(non-programmers)
December 4-6
February 5-7
April 1-3
(programmers)

January 8-10
March 4-6
May 6-8

Toronto
(3 days)
December 3-5
December 17-19
January 7-9
January 21-23
February 4-6
February 18-20
March 3-5
March 17-19
Mar.31,April 1-2
April 14-16
April 28-30
May 12-14
May 27-29

Vancouver
(3 day)
January 23-25
February 20-22

Vienna
(APL 1, German)
December 11-13

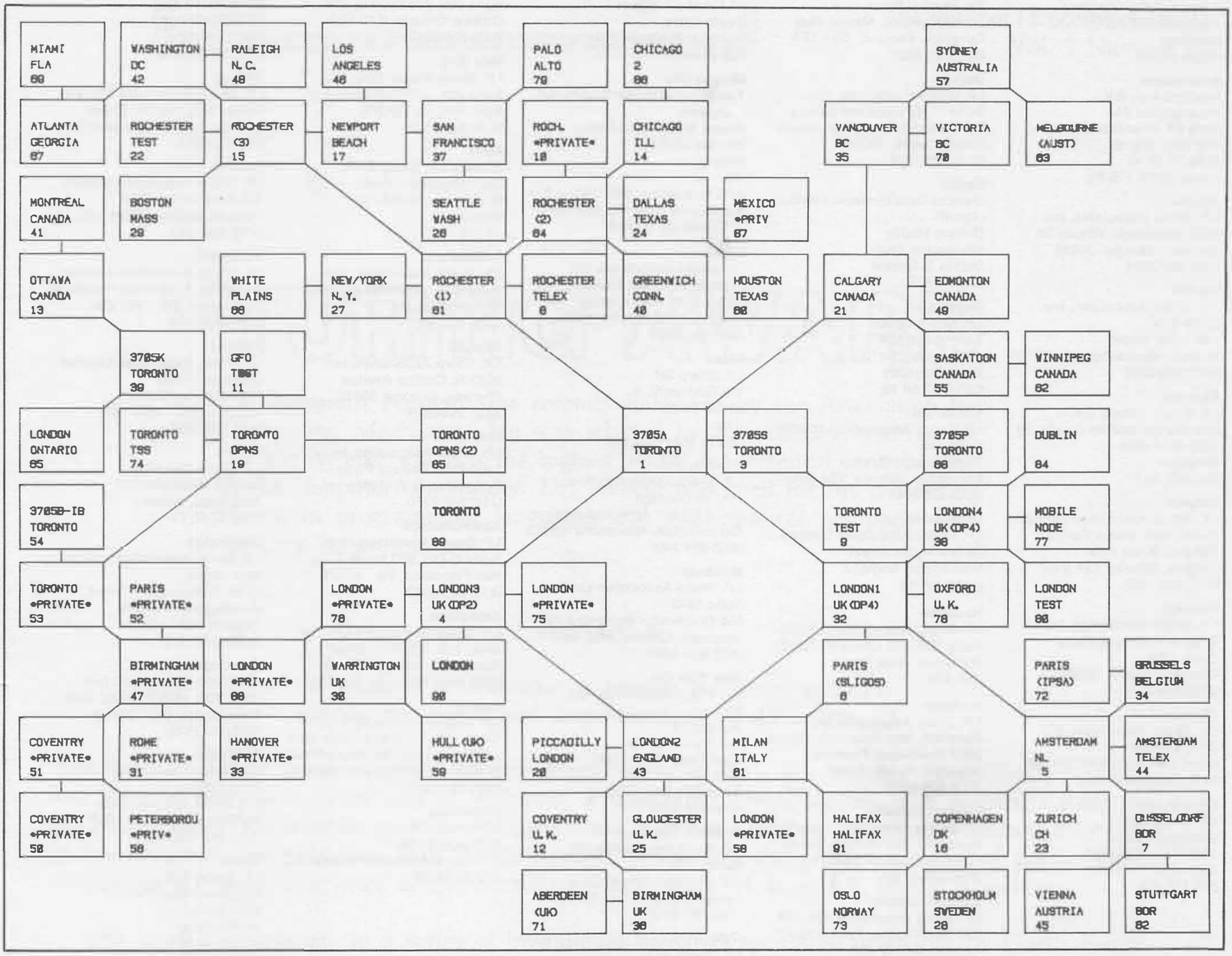
Warrington
(3 day)
December 11-13
February 25-27
April 9-11

Zurich
(German)
(APL Einfuhrung I)
December 3-5
January 14-16
February 11-13
March 10-12
(Wiederholung Einf I)
December 10
(APL Afbaukurs/Einf 2)
December 11-12
January 28-31
February 25-27
March 17-19

INTERMEDIATE APL

Gloucester, December 3
London, December 13, Jan 16, Feb 19, March 25, April 24, May 28
Montreal, (English or French), Nov. 27-29, Jan. 29-31, March 25-27, May 27-29
Ottawa, January 22-23, May 20-21
Rochester, (4 days) November 26,27, Dec 3,4 and March 17,18,24,25
San Francisco, December 4
Toronto, Jan. 28-30, March 10-12, April 21-23
Washington DC, November 29-30 (half-days)
Zurich, Dec. 11-12, Jan. 28-31, Feb. 25-27, Mar. 17-19

Network Topology



UPDATE

Label number: ______

☐ Please amend my mailing address as indicated.

☐ Add to your mailing list the following name(s).

☐ Send me a SHARP APL publications order form.

Name: ______________________

Co.: ______________________

Address: ______________________

Printed in Canada
December 1979

I.P. Sharp Associates **Head Office:** 145 King Street West, Toronto, Canada M5H 1J8 (416) 364-5361

International Branch Offices

Aberdeen
I.P. Sharp Associates Limited
5 Bon Accord Crescent
Aberdeen AB 12DH
Scotland
(0224) 25298

Amsterdam
Intersystems B.V.
Herengracht 244
1016 BT Amsterdam
The Netherlands
(020) 24 40 50
Telex: 18795 ITS NL

Atlanta
I.P. Sharp Associates, Inc.
5000 Snapfinger Woods Dr.
Decatur, Georgia 30035
(404) 987-2301

Boston
I.P. Sharp Associates, Inc.
Suite 415
148 State Street
Boston, Massachusetts 02109
(617) 523-2506

Brussels
I.P. Sharp Europe S.A.
Avenue General de Gaulle, 39
1050 Bruxelles
Belgique
(02) 649 99 77

Calgary
I.P. Sharp Associates Limited
Suite 2660, Scotia Centre
700-2nd Street S.W.
Calgary, Alberta T2P 2W2
(403) 265-7730

Chicago
I.P. Sharp Associates, Inc.
2 North Riverside Plaza
Suite 1736
Chicago, Illinois 60606
(312) 648-1730

Cleveland
I.P. Sharp Associates, Inc.
(216) 431-6861
(local call, switched through to Rochester office.)

Copenhagen
I.P. Sharp ApS
Østergade 24B
1100 Copenhagen K
Denmark
(01) 112 434

Coventry
I.P. Sharp Associates Limited
7th Floor B Block
Coventry Point, Market Way
Coventry, England CV1 1EA
(0203) 21486/7

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

Dublin
Gamma Data Systems Limited
(Agent)
Dollard House
Wellington Quay
Dublin 2, Ireland
(01) 711877

Düsseldorf
I.P. Sharp GmbH
Leostrasse 62A
4000 Düsseldorf 11
West Germany
(0211) 57 50 16

Edmonton
I.P. Sharp Associates Limited
Suite 505
10065 Jasper Avenue
Edmonton, Alberta T5J 3B1
(403) 428-6744

Gloucester
I.P. Sharp Associates Limited
29 Northgate Street
Gloucester, England
(0452) 28106

Hamilton
I.P. Sharp Associates Limited
Suite 308, 237 Locke St. South
Hamilton, Ontario
L8P 4T4

Houston
I.P. Sharp Associates, Inc.
Suite 925, 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) 434-2426

London, England
I.P. Sharp Associates Limited
132 Buckingham Palace Road
London SW1W 9SA
England
(01) 730-0361

Melbourne
I.P. Sharp Associates Pty. Ltd.
36 Elizabeth Street
South Yarra
Victoria, Australia 3141
(03) 244-417

Mexico City
Teleinformatica de Mexico S.A.
(Agent)
Arenal N 40, Chimalistac
Mexico 20 D.F.
Mexico
Mail to:
A/C Despachos des Centro S.A.
Aeropuerto Internacional de la Ciudad de Mexico

Miami
I.P. Sharp Associates, Inc.
Suite D, Kennedy Building
14560 N.W. 60th Avenue
Miami Lakes, Florida 33014
(305) 556-0577

Milan
I.P. Sharp Srl
Via Eustacchi, II
20129 Milan
Italy
(2) 271 6541/221 612

Minneapolis
I.P. Sharp Associates, Inc.
Mail to Chicago
Suite 1371, 1 Appletree Square
Bloomington, Minnesota 55420
(612) 854-3405

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

New York City
I.P. Sharp Associates, Inc.
Mail to:
Suite 2004
200 Park Avenue
New York, N.Y. 10017
Also at:
Suite 242, East. Mezz.
(212) 557-1200

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
4 Etasjc
Kronprinsesse Mārthas Plass 1
Mail to:
Postboks 1470, Vika
Oslo 1, Norway
(02) 41 17 88

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
Société I.P. Sharp SARL
Tour Neptune - Cedex No.20
92086 Paris-la-defense
France
(1) 773 57 77

Philadelphia
I.P. Sharp Associates, Inc.
Suite 407, 1420 Walnut Street
Philadelphia, PA. 19102
(215) 735-3327

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

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

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 208, 135 21st Street E.
Saskatoon, Sask. S7K 0B4
(306) 664-4480

Seattle
I.P. Sharp Associates, Inc.
Suite 217, Executive Plaza East
12835 Bellevue-Redmond Road
Bellevue, Washington 98005
(206) 453-1661

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

Stuttgart/Boeblingen
I.P. Sharp GmbH
Schafgasse 3
7030 Boeblingen
West Germany
(070 31) 2 30 14

Sydney
I.P. Sharp Associates Pty. Ltd.
Suite 1342, 175 Pitt Street
Sydney, N.S.W., Australia 2000
(02) 232-5914

Toronto
I.P. Sharp Associates Limited
145 King Street West
Toronto, Ontario M5H 1J8
(416) 364-5361

Vancouver
I.P. Sharp Associates Limited
Suite 604, 1112 West Pender St.
Vancouver, B.C. V6E 2S1
(604) 682-7158

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
Rechte Wienzeile 5/3
1040 Wien, Austria
(222) 57 65 71

Warrington
I.P. Sharp Associates Limited
Paul House
89-91 Buttermarket Street
Warrington, Cheshire
England WA1 2NL
(0925) 50413/4

Washington
I.P. Sharp Associates, Inc.
Suite 400, 1835 K Street N.W.
Washington, D.C. 20006
(202) 293-2915

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.
Badenerstrasse 141
8004 Zurich
Switzerland
(1) 241 52 42

SHARP APL Communications Network: Local Access Cities

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

Local dial access is available in all locations listed above. The SHARP APL Communications Network also provides local dial access in:

• Ann Arbor • Birmingham • Buffalo • Dayton • Des Moines • Detroit • Ft. Lauderdale • Greene (NY) • Greenwich (Ct) • Halifax • Hamilton • Hanover • Hull • Kitchener • Liverpool • Los Angeles • Manchester • Oxford • Raleigh • Regina • Rome • Syracuse • White Plains (NY)

In the United States the SHARP APL Network is interconnected with the networks of TYMNET and TELENET to provide local dial access in more than 100 other cities.

The Newsletter is a regular publication of I.P. Sharp Associates. Contributions and comments are welcome and should be addressed to: Jeanne Gershater, I.P. Sharp Newsletter, 145 King Street West, Toronto, Canada M5H 1J8.
Jeanne Gershater, *Editor*   Ginger Kahn, *Assistant Editor*