compared to the normal value of 1·354 Å[9-10] for C(sp^2)–F distances. A similar tendency was observed in the microwave studies; when the ring was assumed undistorted (same geometry as that of pyridine), the C–F bond distance was obtained as 1·297 Å. The microwave results were interpreted in terms of the 'caving-in' of the ring. The NMR results are in agreement with the view.

The molecular orientation of 2-fluoropyridine is like that of most of the aromatics in the nematic phases[3]. The molecule orients preferentially with its plane along the magnetic field direction. The principal axis system for the order-parameter tensor is rotated by $-31·0°$ in MBBA and $-29·7°$ in Merck Phase IV solutions. The largest principal S value then almost points along the direction F–H(2) with values of 0·0915 and 0·1168 in MBBA and Merck Phase IV solutions respectively. The largest molecular dimensions are also along the F–H(2) direction.

1. Sharma, S. D. and Doraiswamy, S., *Curr. Sci.*, 1972, **41**, 475.
2. — and —, *Chem. Phys. Lett.*, 1976, **41**, 192.
3. Khetrapal, C. L. and Kunwar, A. C., *Adv. Magn. Resonance*, 1977, **9**, 301.
4. Diehl, P., Khetrapal, C. L. and Kellerhals, H. P., *Mol. Phys.*, 1968, **15**, 333.
5. Thomas, W. A., and Griffin, G. E., *Org. Magn. Resonance*, 1970, **2**, 503.
6. Diehl, P., Henrichs, P. M. and Niederberger, W., *Mol. Phys.*, 1971, **20**, 139.
7. Schumann, C. and Price, R., *Angew. Chem. (Int. Ed.)*, 1973, **12**, 930.
8. Sorensen, G. O., Mahler, L. and Andersen, N. R., *J. Mol. Struct.*, 1974, **20**, 119.
9. Nygaard, L., Bojesen, I., Pedersen, T. and Andersen, J. R., *Ibid.*, 1968, **2**, 209.
10. Yokozeki, A. and Bauer, S. H., *Topics Current Chem.*, 1975, **53**, 78.

A STANDARD SYSTEM OF KEYS FOR PROGRAMMABLE CALCULATORS

M. P. ABDURAZAK

Department of Statistics, College of Agriculture, Vellayani, Trivandrum 695 522 (*India*)

ABSTRACT

A standard system of keys involving 36 keys that are used for programming has been proposed for a calculator.

INTRODUCTION

PROGRAMMABLE calculators (PCs) will become widespread in the near future. These are suitable machines for processing small scale data. Programming in a calculator is a much more challenging task than programming in a higher level language such as FORTRAN. Here all the manipulations have to be made within the very limited capacity of the machine. At present the use of a programme is confined to that particular model alone. One way to get over this difficulty is to choose the most popular brand as standard and then to identify and redesignate the keys of other machines by those of the standard. Sometimes two or more keys have to be identified with a single key of the standard or *vice versa*. If the keys are designated by a single symbol, the programme will look elegant. Unfortunately in the existing PCs many of the keys are characterised by more than one symbol and so are not suitable for treating as standard. Further, not all keys given are necessary. The author discusses a few keys of PCs (each designated by a single symbol) which are capable of solving problems that require versatility of the order of FORTRAN. Once the keys of a PC are identified with those given here, programmes can be written using these symbols. In this manner, calculator programmes, like computer programmes, can be made machine independent. A programme written in this system or a particular calculator will have to be slightly modified in the same way as a computer programme, to run in other machines. In short, this system will serve the purpose of a language for PCs.

MATERIALS AND METHODS

The PC discussed in this paper has the following features.

Programme steps

Each pressing of a key is reckoned as a programme step or simply step for brevity. A step can be stored in what is called a programme memory. There are 1000 programme memories serially numbered in three digits from 000 to 999. A step is always associated with a programme memory where it is stored. We call the serial number of this programme memory as step number. There is a place in the machine, called step counter, which displays the step number of the key next to be executed. A programme is a sequence of steps for solving a specific problem. It can be started from any desired location.

The programme is to be written in such a way that the logic is easily traced and the step numbers readily determined. A programme can be divided into smaller logical units, preferably corresponding to FORTRAN statements, each consisting of a few steps. We may write these steps in a line giving only the step number of the first key in the line. This simplifies the tracing of logic. The step number can easily be obtained by starting with the first key in the line. A logical unit can be written in two or more lines when a line is not sufficient or when there is some reason to do so. Similarly two or more units can be written in a line when they are too short or when there is some reason to do so. A programme written in this manner will have the appearance of a FORTRAN programme.

Registers

Registers are places where numbers can be stored and manipulated. There are four types of registers namely display register, internal register, data register (DR) and index register (IR). The first three work exactly as in ordinary calculators. The display register is used to display data entered and results calculated. The internal register, in association with the display register, performs all calculations. The DR is the same as memory in ordinary calculators. We use the words DR and memory interchangeably. There are 200 DRs serially numbered from 000 to 199. These are called addresses of DRs and these registers are distinguished from one another by their addresses. The address must be stated whenever DRs are considered. Index registers are mainly used to address memories indirectly. This makes the use of subscripted variables possible. Indirect addressing is one of the most important features of calculator programming. There are 10 IRs serially numbered from 0 to 9. These nos. are called the addresses of IRs and must be stated whenever they are referred. The number of DRs, IRs or programme steps will in no way affect our system of keys and may vary from machine to machine. We have given their numbers as 200, 10 and 1000 because it is sufficient to process many of the practical problems. Using this capacity, programmes have been written to analyse data from most of the statistically designed experiments, inversion of a matrix of order up to 13, multiple linear regression analysis for variables up to 17, formation of clusters in D^2, etc.

Output medium

The outputs can be read from the display register or printed on a paper. Only one number can be printed per line.

Mode of operation

The machine should have algebraic mode of operation.

Keys and Their Functions

The various keys and their functions are explained below :

Ordinary Calculator keys

There are 23 keys which function as in ordinary calculators. These are the numeral keys 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; the arithmetic operation keys $+$, $-$, $\times$, $\div$, $=$, $\surd$; the change sign key $\pm$, the clear keys C, G; the parentheses keys (f,); the decimal point key · and the exchange key E. The key C clears display. G is meant to clear the display, internal and data registers along with the arithmetic operation command given to the machine. However, in view of the existing calculators, let us use the combination G4 for this purpose. Similarly, let us use G0 to clear the display and internal registers and arithmetic operation command. The key E exchanges the contents of display and internal registers. The sequence x = is used to find the square of numbers.

The data register control keys

The four keys S, M, Z and F control the operations on DRs. These must be followed by a 3 digit number indicating the address of the DR concerned. The S key stores the display in the respective DR. The M key displays the content of the DR. The Z key interchanges the content of the display register and that of the data register concerned. The F key adds the display to the content of the respective DR. The following table explains the functioning of these keys using DR150 as a typical data register. Suppose 15 is already on display.

Table I

Functioning of S, M, Z and F keys

Keys	Display	Comment
..	15	Already on display
S150	15	Stores 15 in DR150
20	20	20 is entered
F150	20	20 is added to the content of DR150
M150	35	Content of DR150 displayed
25	25	25 is entered
Z150	35	DR150 displayed and 25 stored there
M150	25	Content of DR150

Two or one digit numbers can be used as memory addresses provided the address is followed by a nonnumeral key as in M12 × M0 = F2 =. The = key can be used to terminate memory addresses as in the case of the second = key here.

The index register control keys

The four keys T, R, I, and D control the operations on IRs. These keys must be followed by a single digit integer indicating the address of the IR concerned. T stores the display in the respective IR. R recalls (displays) the content of the IR. I increments and D decrements the content by unity. The most important property of IRs is that they can be used to address the DRs indirectly. This means that the content of IR can be treated as a DR address. This is done by recalling the IR in place of DR address. Suppose IR0 contains 150. Then pressing R0 will display 150. Pressing MR0 will display the content of DR150 since R0 is now equivalent to 150. (Micro 2200 of HCL uses the decimal point key instead of the R key). These points are illustrated in Table II using IR0 as a typical index register. Suppose 15 is already on display.

TABLE II

Functioning of T, R, I and D Keys

Keys	Display	Comment
..	15	Already on display
T0	15	Stores 15 in IR0
I0	15	Increments IR0 by 1
R0	16	Content of IR0
+ 2 =	18	16 + 2 = 18 displayed
SR0	18	Stores 18 in the DR whose address is in IR0. Here 18 is stored in DR16
25	25	25 entered
FR0	25	25 added to DR16
ZR0	43	Exchanges DR16 and Display
MR0	25	Content of DR16
D0	25	Decrements IR0 by 1
R0	15	Content of IR0

The jump keys

Normally a programme is executed step by step. There are two keys which can be used to transfer control to any desired location (step). These are the unconditional jump key →, and the conditional jump key ≤. These keys have to be followed by a 3 digit integer indicating the step number to which jump is desired. → immediately transfers control to the step number following it. ≤ key passes control to the step number following it only if the figure on display is less than or equal to the figure in the internal register, otherwise the jump command is ignored. The ≤ key, in combination with the IR control keys, forms a powerful tool for forming loops.

The printer control keys

These are the P and A keys. The P key prints the number displayed and the A key advances the paper by a fixed amount. Hence this key provides spaces between lines in the print out.

The halt key

The key H halts execution of the programme. The step counter will show the step number of the key immediately following the H key. Manual intervention is necessary to resume execution. This can be done by a resume key. (In Micro 2200, the R/D key performs this). The H key is made use of for entering data.

All the keys described above should only do the functions assigned to them and nothing else. In particular, the keys S, F, T, I, D, P, A, →, and ≤ should not affect the display and internal registers and the arithmetic operation command given to the machine.

LOOPING

Repeated execution of a programme or part of a programme is called looping, and the sequence of keys involved, a loop. The keys → and ≤ are used to form loops. Since the jump in the case of → is unconditional, an infinite loop will be formed if this key is used. The control can never come out of the loop. In the case of ≤, the loop will be executed so long as its condition is satisfied. The control can be brought out of the loop by breaking the condition. So, if a loop is to be executed *n* times, we store this number in some register say DR0. Another register preferably an index register say IR0, is assigned the task of counting the number of repetitions. If it contains 1 initially and increases its content by unity with each execution, the loop can be made to execute so long as the content of IR is less than or equal to the content of DR. That is, the loop will be terminated when the IR value exceeds that of DR. These comparisons are made by means of display and internal registers. These are explained in the example that follows. DR1 is used as an accumulator for total.

Example : 1. Write a programme for finding the total and mean of n values.

Table III

Programme for finding total and mean

Step No.	Keys	Comment
000	AC	Leaves some space and clears display
002	HP S0	n is fed and stored in DR0 and
006	= A	paper advanced
008	0 S1 =	DR1 initialised to 0
012	1 T0	IR0 is initialised to 1
015	HP F1	Loop started. Numbers are added one by one to DR1
019	I0	IR0 incremented
021	M0	n recalled, it is
023	E	stored in the internal register
024	R0	No. of values added plus one displayed
026	$\leq$	and tested whether it is less than or equal to n
027	015	If so, control goes to 015 to add the next value and if not proceeds to 030 below
030	A M1 P	Leaves some space and prints total
034	÷ M0 =	
	PA	Mean printed and paper advanced
040	Halt	End of programme.

The addition is done in a loop. Initially we put 0 in DR1 to start the accumulation. Each execution of the loop (steps 015 to 029) will add a value to DR1 and unity to IR0, so that after each iteration DR1 will contain the total of the values added so far and IR0, the number of values added plus one. Then a test will be made (steps 021 to 029) to ascertain whether all values are added. If not, control goes to step 015 to add the next number. The loop is terminated when all the values are added. Then DR1 will contain the total and IR0, $n + 1$. In the programme all inputs are printed and space provided at appropriate places.

A programme can have more than one loop. Loops can be nested.

The system of keys given here is sufficient to handle very intricate problems eventhough many features of programming are not discussed. When we expand our system to include all these, most of the letters of the alphabet will be used. Then a keyboard similar to that of a typewriter can be designed. Hence we will get a standard keyboard configuration also.

We conclude this paper by giving a programme to evaluate a polynomial side by side with a FORTRAN programme to illustrate the correspondence between them. In both the programmes, all inputs are printed. In the calculator programme spaces have been provided at appropriate places. The comments are left to the reader.

Example : 2. Write a programme to evaluate the polynomial $y = a_0 + a_1 x + \cdots + a_n x^n$ for values of $x = x_1, x_2, \cdots, x_m$.

The polynomial can be rewritten as $y = (\cdots (a_n x + a_{n-1}) x + \cdots + a_1) x + a_0$. It can be evaluated in a loop. The allocation of variables and registers is given below.

Table IV

Allocation of variables and registeres for evaluating a polynomial

Algebraic variable	n	x	y	a_0	a_1	..	a_n	i	j
FORTRAN variable	N	X	Y	A (3)	A (4)	..	A (N + 3)	I	J
Register allocated	DR0	DR1	DR2	DR3	DR4	..	DR (n + 3)	IR0	IR1

TABLE V

Calculator programme to evaluate a polynomial

```
000 AC
002 H S0 =
006 PA
008 0 T0
011 3 T1
014 H SR1
018 P
019 I0
021 I1
023 M0 E R0 ≤ 014
032 D1
034 A R1 T0
039 H S1
042 P
043 MR0 S2
048 D0
050 M1 × M2 + MR0 = S2
062 R0 E 4 ≤ 048
070 M2 P
073 → 034
077 Halt
```

TABLE VI

FORTRAN programme to evaluate a polynomial

```
      DIMENSION A (197)
      READ, N
      PRINT, N
      I = 0
      J = 3
14    READ, A (J)
      PRINT, A (J)
      I = I + 1
      J = J + 1
      IF (I.LE.N) GO TO 14
      J = J - 1
34    I = J
      READ, X
      PRINT, X
      Y = A (I)
48    I = I - 1
      Y = Y* X + A (I)
      IF (4 . LE . I) GO TO 48
      PRINT, Y
      GO TO 34
      END
```

The inputs are to be given in the order n, $a_0, \ldots, a_n$, x_1, $x_2, \ldots, x_m$. After entering an x value, the corresponding y value will be calculated and printed.

From this example, we see that a calculator programme can first be written in FORTRAN, keeping the limitations of the calculator in mind and then translating it into our 'calculator language'. This is a great facility for understanding the logic by others or for the programmer himself on different occasions.

APPENDIX

The correspondence of the keys of our system to that of Micro 2200 is given here. The symbols A, C, D, E, F, G, H, I, M, P, R, S, T, Z, → and ≤ correspond to SPACE, CE, DEX, EX, MFN, GOLD, HLT, INX, MEM, PRINT, RCX, STM, STX, EXM, GTO and X≤Y respectively. Any R key following F, M, S and Z must be replaced by the decimal point key.

1. *Instruction Manual*, Micro 2200, Hindustan Computers Limited, Lucknow.
2. Chirlian, P. M., *Introduction to FORTRAN IV with Timeshare and Batch Application*, Academic Press, New York, 1973.