パーソナル・コンピュータのベーシック (4)

小林竜一

本回でパーソナル・コンピュータのベーシックは連載の第4回になる．ここで，マイクロ・コンピュータをベースとしたパーソナル・コンピュータ市場に新たにHHCという強力なライバルが現われたことをご紹介しよう．

HHCとは，hand-held-computerの頭文字をつづったものであり，HHCの始めというべきものは本連載でもすでに紹介ずみのSHARPのPC-1210/1211ポケット・コンピュータであるが，これにヒューレット・パッカードの41Cポケット電卓も仲間に入れて考えてもよいだろう．これらのポケット・コンピュータはその中に4ビット程度のマイクロ・コンピュータを1～2個もっており，基本的な構造はマイクロ・コンピュータを使ったパーソナル・コンピュータと変わりのないものである．

このたび米国で発表されたPanasonic/QuasarのHHCというものは8ビットCPUの6502というLSIチップで低電圧低電流で動作可能のものが製作されたためにできたものであって，コンピュータ本体のほかに周辺機器が完備しており，プログラマブルメモリー拡張装置，カラーTVインターフェイス，I/Oドライバー，音響カプラー，ポータブルプリンタ，カセットインターフェイス，ROM拡張装置という多彩さである．

これらHHCを使って大型コンピュータに電話回線を介してアクセスすることも可能である(参考文献[13])．

なお今回発表されたHHC(Panasonic/Quasar)はFORTHという言語(マイコン用言語)に近い言語SNAPを使ってプログラムするので，本講座の対象ではない．しかしこのようにマイクロ・コンピュータの世界は革新の連続であることを読者の皆様にご報告しておくことが必要であろう．

本号ではTRS-80ベーシック(Tandy Radio Shack)とシャープmz-80シリーズ・ベーシックの特長をご紹介する．

7. TRS-80ベーシック

まず，許される変数名としては英字で始まる，2字以内の英数字のつながりである．そこで，

AA, BC, ……

というような変数名が許されることに注目されたい．2字を超える部分は読み飛ばされる．そこで，

SUM, SUB, SU

は同じものと見なされる．

変数の型の宣言ができる．それらは表1に示される．

表1 変数の型宣言

変数の型	定義文字	例	その値の例
整数 −32768から +32767まで	%	A%, B9%	−30, 123, 3, 6000
単精度 有効数字6桁	I!またはE (Eは値の例を見よ)	A!Z1!	1, −50, .1234 1.23E−5
倍精度 有効数字16桁	#	A#, ZZ#	−300.12345678 3.141592653589
倍精度 (科学表示)	D	A#=1.2345678901D+12	$1.2345678901\times10^{12}$
文字型 255文字まで.	$	A1$	"JOHN Q. DOE" "1+2=?"

(注) A$とA%とA!とA#とはお互いに皆異なった記憶場所となる．

DIM宣言については特に制約がない．(注)PETのベーシックではDIMに要素の数が255以下という制約があったが，PETのベーシックでも新しい版ではこの制約はなくなっているとのことである．

乗べきは↑で表わされる．ただし機械によってはこれが[という文字になっているものがある．

論理演算子ではAND, OR, NOTが使える．以下の例を見よ．

```
50 IF Q=13 AND  R2=0 THEN PRINT
                              "READY"
100 Q=(G1<0)AND(G2<L)
```

こばやし　りゅういち　立教大学

G1<0 でかつ G2<L のとき Q=−1 となる．
その他のケースは Q=0

200 Q=(G1<0)OR(G2<L)
G1<0 または G2<L なら Q=−1 となる．
G1≧0 かつ G2≧L のとき Q=0 である．

300 Q=NOT(C>3)
C≦3 のとき Q=1 となる．
C>3 のとき Q=0 となる．

文字列に対する演算として次のようなものが使える．

表 2 文字列に対する演算

記号	意味	例
<	辞書的順序づけ	"A"<"B"
>	〃	"JOF">"JIM"
=	等しい	B$="WIN"
<>	不等	IF A$<>B$ THEN PRINT A$
<=	順序づけ	IF A$<=B$ THEN……
>=	順序づけ	IF A$>=B$ THEN……
+	文字列の接続	A$="TRS-"+"80" (これでA$の中身は "TRS-80" となる)

ビデオ・ディスプレイ上の位置を 0〜1023 の間の数字で指定して（指定場所と数字の関係は表として与えられている．［2］付録 C/7 TRS-80 Video Display Worksheet）そこから印刷を始める命令がある．これはTVディスプレイの画面の任意の場所に文字を画くのに便利である．

100 PRINT @ 550, "LOCATION␣550"

とすると 550番目と約束された場所から LOCATION 550という字が表示される．

なお PRINT 文の最後にセミコロンをつければ復帰改行が行なわれない．

PRINT TAB(5) "TABBED␣5"; TAB(25) "TABBED␣25" という印刷文が使える（PET参照）．

PRINT USING 文を使って任意の書式を指定して印刷することができる．そのすべてを完全に説明するのは無理なので，以下に例を 2，3 示し，あとは［2］の 3/4 頁から3/8頁を参照していただくことにする．

```
10 INPUT A$, A
20 PRINT USING A$, A
30 GOTO 10
```

このプログラムを書いてRUNコマンドでこのプログラムを走らせると，画面に？ という入力促進記号がでるので ##. ##, 12.12 と入力すると，

12.12

という表示が得られ，###, ##, 12.12 と入力すると，

␣12.12

という表示が表われ，##, ##, 121.12 と入力すると，

% 121.12 （%は桁が不足を表わす）

と表示される．

この他にいろいろの表示法を指定することができる．たとえば勘定書きに左側の字に＊印をずっと打つ必要があるが，そのような印刷も指定できる．

PRINT# 文があり，これで装置番号を指定してその装置に印刷命令を出すことができる．また，これで書かれたものを入力するために INPUT# 文がある．たとえば，PRINT#-1, A1, B$, "THAT'S␣ALL" という文でカセットテープ装置に A1, B$ の内容，文字列 "THAT'S␣ALL" を書くことができ，これを INPUT #-1, C1, D$, E$ という入力文で読むことができる．

DEFINT 文がある．この使用法を例で示そう．

10 DEFINT A, I, N

とすると，10番以降の行で現われるA，I，Nで始まるすべての変数は整数型として処理される．

DEFSNG 文がある．

100 DEFSNG I, W-Z

とすると，この行のあとで現われるIまたはWからZまで英字をはじめとする変数はすべて単精度の変数となる．

DEFDBL 文がある．

10 DEFDBL S-Z, A-E

これでSからZまでとAからEまでの英字を先頭にもつ変数は倍精度の変数として取り扱われる．

DIM文で以下のような使い方ができる．

```
40 INPUT N
50 DIM A(N, 2)
```

これでAという配列の大きさが自由に変えられる．もしも配列の大きさをプログラムの途中で変更するときにはCLEAR 文を使ってから行なわないとエラーとなる．

ON GOTO 文，ON GOSUB 文があることは PET ベーシック，ALTAIR ベーシックと同じである．

ON ERROR GOTO文がある．これをプログラムの最初の部分に書いておくと，プログラム実行時にエラーがおこると指定のところへブランチするようになる．たとえば，

```
 5 ON ERROR GOTO 100
10 C=1/0
```

とすると，10のステートメントで0の割算でエラーとなり，100番の行に飛ぶ．そこで100番以下にエラー時の処理方法をプログラムしておけばよい．

RESUME 文がある．

RESUME 10

と書くと行番号10の行から再実行を始める．

RESUME NEXT

とすると，エラーのおこった行の次の行から実行を始める．

IF…GOTO 文がある．もちろん IF THEN 文もある．ELSE 文がある．

200 IF A>0.001 THEN B=1/A : ELSE 260

という書き方ができる．

IF THEN ELSE 文はネスト（重ねて使う）することができる．

文字列を取り扱う関数として ASC, CHR$, LEFT$, LEN, MID$, RIGHT$, STR$, VAL がある．これらはすでに ALTAIR ベーシックとＰＥＴベーシックで紹介ずみであるが，それに加えて FRE, INKEY$, STRING$の３つの関数がある．

FRE(A$) と使うと A$ の中に入っている文字列によって占められていないスペースが何字分残っているかを調べて返してくれる．たとえば，

C=FRE(A$)

と使う．Ｃの中に３が入ればあと３文分 A$ の中に余地のあることを示す．300 PRINT FRE(A$)とも使える．

INKEY$文はＫＢを１回打つと１文字を帰してくる．たとえば，

```
10 CLS (CLSは CLEAR SCREEN でスクリーンを暗くする命令)
100 PRINT @ 540, INKEY$ : GOTO 100
```

と書くと，このプログラムをＲＵＮさせると，ＫＢから何か入れてやると，その文字がスクリーンの 540番の場所（既出）に現われ，次のキーが押されるまでその状態が続く．この文の使用法をもう１つ示す．

```
100 A$=INKEY$ : IF A$=" "THEN 100
    ELSE PRINT A$;
110 B$=INKEY$ : IF B$=" "THEN 110
    ELSE PRINT B$;
120 C$=INKEY$ : IF C$=" "THEN 120
    ELSE PRINT C$;
130 D$=A$+B$+C$
```

これで３文字が D$ の中に入れられる．

STRING$(30,"*")とすると＊印を30字作り出す．つまり，

"**********"（30個）

と同じである．

ＶＡＬはすでに出てきたが，ここでくわしい解説を加えると次のとおりである．A$の中に "12" が，B$の中に "34" が入っているとすると，

VLA(A$+"."+B$) は 12.34

を意味する．

また VAL(A$+"E"+B$) は12E34，つまり12×10^{34}を表わす．また，文字列の中に英数字が入っていると少し変わった対応をする．たとえば，

VAL("100 DOLLARS") は100

を意味する．

次のプログラムは X$ の中に長い文字列を読み込み，Y$ の中に短い文字列を読み込み，短い文字列と同じものが長い文字列の中にあるかどうか，あればどこに存在するかを調べて印刷するプログラムである．

プログラミングの例題としておもしろいので示すことにする．

```
 5 CLEAR 1000 : CLS
10 INPUT X$
20 INPUT Y$
30 GOSUB 1000
40 IF I= 0 THEN 70
50 PRINT Y$; "IS␣A␣SUBSTRING␣OF
   ␣"; X$
55 PRINT "STARTING␣POSITION:"; I
60 PRINT "ENDING␣POSITION:"; I+
   LEN(Y$)-1
65 PRINT : PRINT : GOTO 10
70 PRINT Y$; "IS␣NOT␣CONTAINED␣
   IN␣"; X$
80 GOTO 10
99 END
1000 FOR I=1 TO LEN(X$)-LEN(Y$)+1
1010 IF Y$=MID$(X$, I, LEN(Y$))RETURN
1020 NEXT : I=0 : RETURN
1030 END
```

ただしここで CLEAR 1000とあるのは，以下のプログラムで文字列には1000文字まで入ることを予約するのである．つまりX$やY$には1000字まで入るようにメモリーを取ることになる．

代数関数については標準のものの外にCSNG, CDBL, RANDOM, CINT, FIXが加えられている．

CSNG(x)は倍精度の数値 x を単精度に変換する（丸め方は４捨５入）関数である．CDBL(x) は単精度の数 x を倍精度で表わす関数である．CINT (x) は数学で使うガウスの記号と同じもので，x を超えない最大の整数を与える．

ただし−32768と32767の間にかぎる．たとえばCINT (1.5)は１であり CINT(−1.5)＝−２である．CINT と INT の違いは CINT は−32768から32767の間にかぎられるが，INT はその制限がないことである．

FIX(x)は単純に小数点以下を切り捨てる関数であって，FIX(1.5)は1，FIX(−1.5)は−1となる．

RANDOMは関数というよりは宣言であって，これが文の最初の部分で宣言されているとRNDという関数で与えられる乱数がいつもまったく違った初期値から始まるようになるのである（他のコンピュータの RANDOMIZE 命令に相当）．

RND (0) は区間 (0, 1) での一様乱数を作る．また RND (55) は区間 (0, 56) での一様乱数を作る．さらに RND (55.7) でも区間 (0, 56) での一様乱数を作る．つまり引数は整数部分のみに切捨てをして使われる．

グラフィック機能を活用するために SET (x, y), RESET (x, y), CLS (既出), POINT (x, y) という命令が使える．画面は横に 128，縦に 48個に区分されているので，$0 \leq x < 128$, $0 \leq y < 48$という制限の下で SET (x, y) と書くとxとyで指定される部分が光るようになる．たとえば，

```
10 SET (0, 0)
```

とすれば画面の左上隅の小ブロックが光る．

```
20 SET (127, 47)
```

とすれば画面の右下隅の小ブロックが光る．

また，RESET(0, 0) で画面の左上隅のブロックが光らなくなる．

POINT (x, y)

は x, y で指示された小ブロックがもっか光っているときは値が −1 となり，光っていないときは値が 0 となる関数である．使用例を示そう．

```
100 IF POINT(50,28) THEN PRINT "ON"
      ELSE PRINT "OFF"
```

この命令で $x=50$, $y=28$ で指定される画面の部分が光っていればON，光っていないときOFFが印刷される．

次にERLという変数がシステムに用意されており，ON ERROR GOTOステートメントが実行されると，この変数の中にエラーのおこった行番号が入るようになっている．それでエラーの後処理をするプログラムのところでERLの中の数値を調べてしかるべき後始末のプログラムの所に飛ばすようにプログラムすることができる([14] 8/3頁)．また，ERRという変数もシステムに用意されており，ERR/2+1とすればエラー・コードが与えられることになっている．

また関数 INP(i) があって，iとしてポートの番号を指定すると（ポートとは入出力信号の通る電気回路のことである．ポートの番号は 1〜255 であるがTRS 80ではカセットレコーダのポートとして255を使用している）そのポートを通じて信号を1バイト分読みとる．使い方はたとえば，

```
100 C=INP(255)
    ⋮
200 PRINT INP (255)
```

というようなものである．

MEMはメモリーの中で使われていない部分の大きさをバイトで表わした数値が入っている変数名である．

```
100 IF MEM<80 THEM 900
110 DIM A(15)
```

というような使い方ができる（DIMはプログラムの途中に入れて使用するメモリーを追加することがある）．

OUT文がある．OUT250, 10とすると250番というポートに“10”という値を送る．ポートは0から 255まで可能性としてありうる（マイクロコンピュータのCPUチップの設計でそうなっている）．

PEEK関数がある．A=PEEK(15876)と書けば15876番地にある数値がAという記憶場所に入れられる． POKEがある．POKE X, 191 と書くと記憶場所 Xの中に入っている数字で示される番地の記憶場所の中に 191 という値を代入する．POSという関数があり，ディスプレイのカーソル（KBから入力される文字の位置を示すアンダーライン）が現在どの位置にあるかを0〜63の数字で返す．POS(x) という形で使うが，xはダミーの引数であって，形式的に書くだけで何の意味もない．

また機械語で書かれたサブルーチンへ飛んでいって実行してくる関数としてUSR(x)関数がある．この使用法はハードの知識もからむので省略する．VARPTRという関数があって，その引数に変数名を書くと，その変数名のしまわれている記憶場所（インタープリータの管理下の）と，その変数の割り当てられた場所の番地がわかる．これも詳細は省略する．

なお，テキスト・エディター用にいろいろの便利なコマンドが用意されている．これも使用者にとって重要なことである．

8. シャープ MZ-80 BASIC

参考文献[14]は計算機についている説明書であるが，非常にわかりやすい解説書になっていて，計算機の予備知識は零で読むことができる．別売りで販売されてもよいと思われる．

有効数字8桁の計算精度である．関数で常用対数をLOG(x)で表わし，自然対数をLN(x)で表わしている．INPUT文に任意の入力促進文をつけられる．たとえば，

```
INPUT“ カレノ␣トシハ”; A
```

とすればディスプレイに“カレノ␣トシハ”と表示が出る．片仮名も使える．：をつかって1行に何行もの論理行をかける．たとえば，A=0：B=1：C=Dというよ

うに．PRINTの代わりに？ を使える．たとえば，IF SGN(X)＝－1 THEN？ “マイナス”とできる．

CURSOR コマンドがある．

CURSOR X, Y

と押してX(水平軸)，Y(垂直軸)を指定するとカーソルを任意位置に移動することが可能である．

LEFT$, MID$, RIGHT$, LEN, ASC, CHR$, VAL, というストリングを取り扱う関数が使える．

ON GOTO文，ON GOSUB文が使える．関数RNDの引数については0または負のときは一定の初期化が行なわれ，いつも同じ一連の乱数が得られ，引数が正だと毎回別の乱数の系列が得られることになっている．

SET 文，RESET 文があってディスプレイの指定した場所を光らせたり消したりできる．

GET文を使える．TI$ という変数名があり，この中に時刻（6桁の数字）が入っている．たとえば以下のようになる．

```
10 TI$="102634"
20 PRINT TI$
30 TI$="256471"
40 PRINT TI$
50 END
RUN
102634
020511(256471→25+1−2,64+1−60, 71−60)
```

なお，TI$ の中の数は1秒ごとに1つずつ増していき，時刻を示す（上2桁は時，中間の2桁が分，下の2桁が秒）．

音楽をプログラムできる．TEMPO x で（x=1～7）テンポを指定できる．MUSIC 文を使う．ドレミファソラシドをCDEFGABで表わし，半音上ることをその文字（C, D, E etc）のまえに＃をつけて表わす．無音をRで表わす．

MUSIC “CDERFGAB□C”

（ただしここで □ は高音を意味する。）

と書く．また，この文の前に TEMPO 文でテンポを指定しておくことが必要である．

プログラムをMTに記録するのにSAVE，MTからプログラムを読み出すのに LOAD，読み出したプログラムが正しく読み出したかチェックするために VERIFY のコマンドがある．データをMTに書く命令としてはWOPEN文（データをMTに記入する準備を命令する）とPRINT/T文（データをMTに記入する）がある．またROPEN文（データをMTから読み出す準備をする）と，INPUT/T文（データをMTから読み出す）がある．CLOSE 文（WOPEN したら ROPEN するまえに，ROPEN したらWOPENする前にこの文を必ず使用する）がある．

上述の外にCLR文（変数の値をすべてクリア），IF GOTO文，IF GOSUB文，SET文，RESET文，SIZE文（メモリーの空き部分をバイトで示す），PRINT/P文(プリンタへの出力文)，INP 文，OUT 文，PEEK 関数，POKE 文，USR文（機械語へコントロールを移す），SPC(x)関数（カーソルの位置からx個のスペースを表わす，TAB関数のように使う）などがある．

また LIMIT 文があって，BASIC で使うエリアを制限することができる．

LIST/Pというコマンドでリストをプリンタに出力することができる．

なおシャープのMZ-80 BASICでは変数名は最初の2文字で判別が行なわれる．なお，最初の文字は英字でないといけない．

以上のように，もっか各社のパーソナル・コンピュータのベーシックについて報告中であるが，これらはすべてマニュアルによる報告であることにご注意ねがいたい．また，数社のベーシックのインタープリータに虫のあるという報告がなされている([16])ことも報告しておく．もちろんこれらは直ちに訂正されることを期待している．

参考文献

[13] Gregg Williams, Rick Meyer and Friends Amis The Panasonic and Quasar Hand-Held Computers, BYTE Jan. 1981 Vol. 6 No. 1 McGraw-Hill Co. pp. 34—45.

[14] LEVEL II BASIC Reference manual, Radio Shack(adivision of TANDY CORPORATION) One Tandy Center Fort Worth, Texas 76102.

[15] BASIC mz-80 SERIES SHARP COPORATON.

[16] W. D. Mauer, A Bug in BASIC, BYTE No. 1 Vol. 6, JAN. 1981. pp. 188～196.