

第4章

これがBS-BASICだ

もうあなたはBS-BASICの入口まで来てしまった。
この扉の向う側にはベーシックの世界が広がっている
のだ。

この章は、STEP1、STEP2、STEP3と3段階になっていて、どのステップで出てくる命令も基本的でかつ重要なものばかりだ。STEP3の関数の命令の中には、数学で使うちょっと難しいものがあるけど、逃げないで挑戦しよう。

とくにSTEP1、STEP2はしっかりやっておこう。



STEP 1 RX-78との軽いあいさつ

STEP 1は、RX-78との軽いあいさつともいべき、基本的な命令ばかりだ。

簡単な命令だが、プログラムを組むうえで必ず使うものばかりだから、しっかり理解する必要がある。

はじめのうち、少しとまどうところがあるかもしれないが、すぐに慣れていくだろう。まちがいを恐れずに、どんどんためしてみよう。そこから、君とRX-78とのスキンシップが生まれる。

STEP 1 サンプルプログラム

```

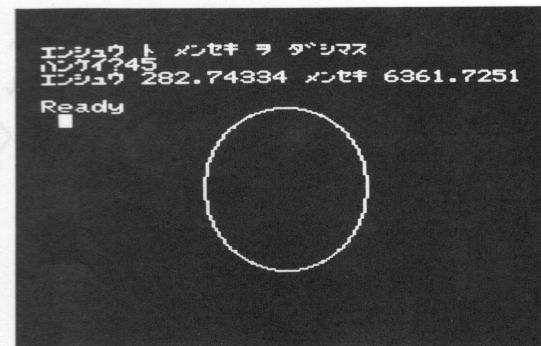
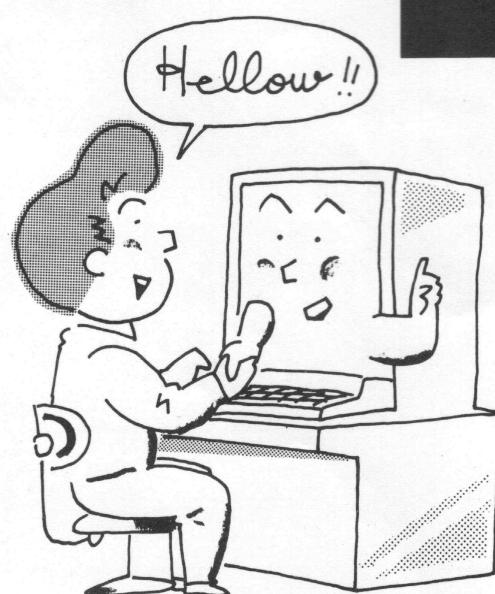
10 REM ** エンノ エンシュウト メンセキ ** ----- 注釈文。
20 PRINT CHR$(6) ----- 画面に出ている内容をすべて消す。
30 PRINT "エンシュウト メンセキヲ タシマス" ----- 「円の円周と面積を出します」と画面に出す。
40 INPUT "ハンケイ?"; R ----- 半径を入力する。
50 C=2*π*R ----- 円周を計算する。
60 S=π*R^2 ----- 面積を計算する。
70 PRINT "エンシュウ"; C; " メンセキ"; S ----- 「円周」、とCの内容、「面積」とSの内容を画面に出す。
80 CIRCLE 95, 90, R ----- 円を描く。
90 END ----- プログラムの終りを示す。

```

〔解説〕

このプログラムは、STEP 1 にでてくる命令を使ったプログラムだ。動かすとつぎのようになる。

- ① RX-78 が円の半径を聞いてくる。
 - ② 君がキーボードから円の半径をいれ、**RETURN** を入力する。
 - ③ RX-78 が円周と面積を計算し、答と円を画面にだしてくれる。
- 円の大きさは、君がいれた半径によって変化する。だから、あまり大きい半径（この場合は90以上）をいれると、円が画面からはみだしてしまう。



半径45の場合

* サンプルプログラム中に出てくる〔 〕はキーボード上で（ ）と同じです。

* べき乗「^」はキーボード上では↑になります。

頭脳をリフレッシュ

ニュー

NEW

使用例

NEW

RETURN

RX-78が覚えているプログラムを
消す。

一般型式

NEW

説明

● RX-78 に新しいプログラムを入力する時に、前に入力してあるプログラムと混乱しないよう、あらかじめプログラムを消しておく命令である。

● NEW RETURN と入力しても、画面にでているプログラムや文字は消えないが、RX-78 の記憶はすべて消されている。



```
LIST
10 A=25
20 B=2
30 C=A-B
40 PRINT C
Ready
```

プログラムを入力し、LISTで確かめたところ、しっかりと入力されている。

```
LIST
10 A=25
20 B=2
30 C=A-B
40 PRINT C
Ready
NEW
Ready
LIST
Ready
```

NEWを入力した後、LISTさせてみると、プログラムが消えてしまって、何もでてこない。

④ NEW RETURN と入力すると RX-78 のなかのプログラムがすべて消えるので注意しましょう。

行番号はAUTOマッチク——**AUTO**

オート

使用例

- ① **AUTO** **RETURN** **RETURN** を押すたびに行番号を10, 20, 30, ……と RX-78 が自動的に10とびに画面にだす。
- ② **AUTO 50** **RETURN** 50, 60, 70, ……と50から10とびに画面にだす。
- ③ **AUTO , 30** **RETURN** 10, 40, 70, ……と10から30とびに画面にだす。
- ④ **AUTO 100, 20** **RETURN** 100, 120, 140, ……と100から20とびに画面にだす。

①

```
AUTO  
10 AA=10  
20 BB=AA+10  
30 CC=BB+10  
40 PRINT AA  
50
```

行番号が10からスタートして、10, 20, 30, 40, 50, ……というふうに10とびに自動的にでてくる。

① 1行目から5行目（行番号40の命令文）までは行の最後で **RETURN**、6行目（行番号50）では **SHIFT STOP** を押しています。

②

```
AUTO 50  
50 A=50  
60 B=A+10  
70 C=B+10  
80  
90
```

行番号が50からスタートして、50, 60, 70, 80, 90, ……というふうに10とびに自動的にでてくる。

② 1行目から5行目（行番号80の命令文）までは行の最後で **RETURN**、6行目（行番号90）では **SHIFT STOP** を押しています。

③

```
AUTO , 30  
10 AA=10  
40 B=AA+30  
70 C=B+30  
100 PRINT C  
130
```

行番号が10からスタートして、10, 40, 70, 100, 130, ……というふうに30とびに自動的にでてくる。

③ 1行目から5行目（行番号100の命令文）までは行の最後で **RETURN**、6行目（行番号130）では **SHIFT STOP** を押しています。

④

```
AUTO 100, 20  
100 AA=10  
120 B=AA+20  
140 C=B+20  
160 PRINT C  
180
```

行番号が100からスタートして、100, 120, 140, 160, 180, ……というふうに20とびに自動的にでてくる。

④ 1行目から5行目（行番号160の命令文）までは行の最後で **RETURN**、6行目（行番号180）では **SHIFT STOP** を押しています。

一般型式 **AUTO** はじめの行番号、行番号の間隔

- 説明** ● **RETURN** を押すたびにプログラムの行番号を自動的に RX-78 がだしてくれる。行番号は1～65,535の整数（1, 2, 3, ……65535）。
- 行番号が自動的にでてくるのをやめるには、**SHIFT STOP** を押す。

④ 後から間に命令文を追加できるように、行番号は10, 20のように間隔をあけてつけましょう。

物語はハッピーEND

エンド

END

使用例

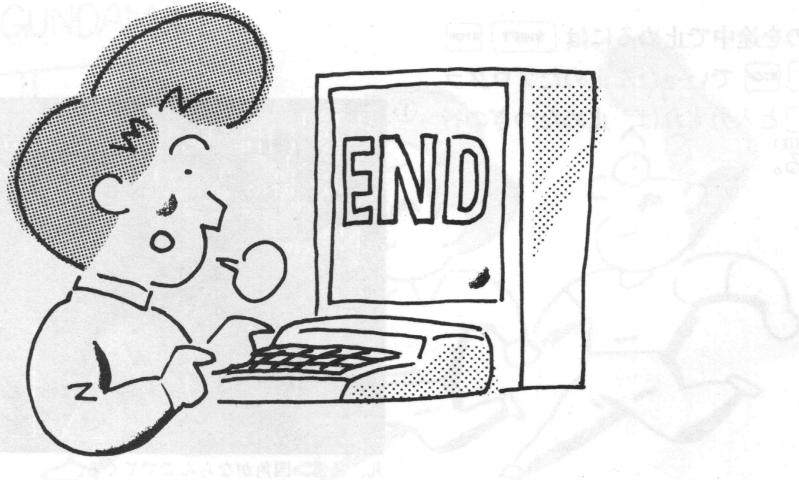
END RX-78

プログラムの終わりを示す。

RX-78というプログラムであることを示す。これがおがる。

一般型式 END

- 説明**
- END命令にきたところで、RX-78はプログラムを動かすことをやめ、画面にReadyが出る。
 - プログラムでENDを書かなかったときは、RX-78はプログラムを最後まで動かす。^(注)ただし、プログラムをストップするか、エラーがあつた時を除く。



①

```

LIST N$="GUNDAM"
20 M$="-----"
30 PRINT N$
35 END
40 PRINT M$
Ready
RUN
GUNDAM
Ready
■

```

行番号35にENDがあるので、行番号40のPRINT M\$をやらずにプログラムが終わった。

②プログラムの終わりに、必ずENDをつける習慣をつけましょう。複雑なプログラムにENDがないと混乱してしまいます。

RX-78よ走れ

ラン

RUN

使用例

① **RUN**

RX-78 が記憶しているプログラムを行番号の小さい順に動かす。

② **RUN 100**

行番号 100 からプログラムを動かす。

一般型式 **RUN 行番号**

説明

●プログラムを動かす命令で、行番号の小さなものから順番に動かしていく。

●RUN100というように、行番号を指定すると、その行から順番に動かしていく。

●プログラムを動かすのを途中で止めるには **SHIFT STOP** を押せばよい。**SHIFT STOP** でいったん止めたプログラムは、**CONT** と入力すれば、止めたつぎの行(注)から再び動きはじめる。

```
① LIST
    10 CIRCLE 48,150,30
    20 LINE 95,125,75,180,115,180
    ,95,125
    30 BOX 125,120,165,180
    40 END
Ready
RUN
Ready
```



丸、三角、四角がならんででてくる。

```
② LIST
    10 CIRCLE 48,150,30
    20 LINE 95,125,75,180,115,180
    ,95,125
    30 BOX 125,120,165,180
    40 END
Ready
RUN 20
Ready
```



丸はでないで、三角、四角だけがでてくる。

①CONTの詳しい説明は68ページを見てください。

プログラムを見やすくする

レム
REM

使用例

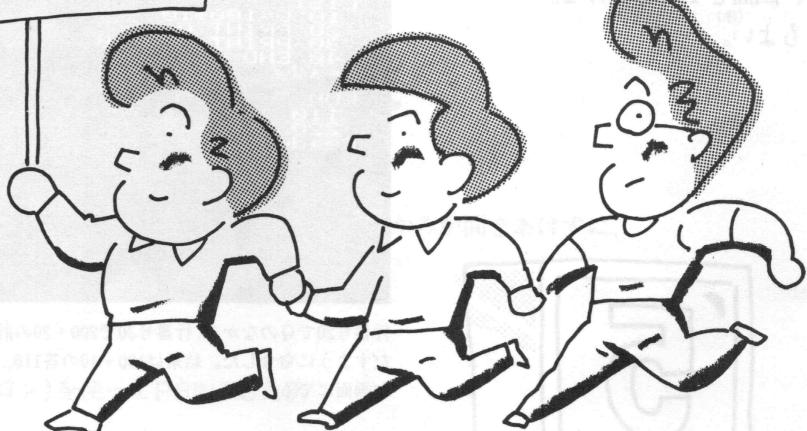
REM *RX-78*

プログラムに見出しをつける。
RX-78というプログラムであることがわかる。

一般型式 **REM** 注釈文

- 説明**
- プログラムをわかりやすくするため、プログラムの先頭や途中に注釈をいれるときに使う。REM文は、プログラムの実行には関係しない。
 - 注釈文には、プログラムの内容や作った人の名前、日付などを書いたりする。

REM*GUNDAM*



①

```

LIST
10 REM ****
20 REM * ダイケイ ノ メンセキ ヲ モタル *
30 REM * 1983.6 BANDAI *
40 REM ****
50 A=20
60 B=50
70 H=25
80 S=(A+B)*H/2
90 PRINT S
100 END
Ready
■

```

行番号10から40までに、注釈として、プログラムのタイトル、作った日付、名前をいれている。

②REM文の中に：(コロン)を使用するとエラーになります。

画面に文字や記号を出す

プリント

PRINT**使用例**

- ① **PRINT A** 変数Aに入っているなかみを画面にだす。
- ② **PRINT 2+3** $2 + 3$ の答、5を画面にだす。
- ③ **PRINT "2+3=?"** " "でかこまれた、 $2 + 3 =$ そのまま画面にだす。
- ④ **PRINT** 画面を1行分あける。

一般型式 **PRINT** 変数あるいは式

- 説明**
- 変数（数値変数、文字変数）のなかみを画面にだす。
 - PRINTの後に計算式を書いた場合は、計算結果がでる。
 - " "でくくった文字（ストリング）は、そのまま画面にでる。
 - PRINTとだけ書くと、画面を1行分あける。
 - PRINTを?と書いててもよい。



④PRINTを?と書いた時、LISTを出すと、?と画面にでないでPRINTとでてくる。

①

```

LIST
10 Q=100+10
20 PRINT Q
30 PRINT 200+20
40 END
Ready
RUN
110
220
Ready
■

```

行番号20でQのなかみ、行番号30で200+20の計算結果を画面にだすように命令した。結果は100+10の答110、200+20の答220が画面にでる。

②

```

10 ? "12+20="
20 ? "12+20"
30 END
RUN
12+20=
32
Ready
■

```

行番号10～30でPRINTのかわりに?を使って、12+20の結果を1行あけて画面にだすように命令した。

PRINT文のあれこれ - , (コンマ) ; (セミコロン)

PRINT文で文字や数字を画面にだすと、1行に1つしか書きだせないが、それではどうにも不便。, (コンマ)や ; (セミコロン)を使うとこの悩みが解決される。

, (コンマ)で間をあけて複数PRINT

PRINT A, B, C

このようにPRINT文を書くと、A, B, Cのなみが画面の同じ行に、この順番に書きだされる。文字や数字を書きだす位置は、前の文字や数字の先頭から6文字あけたところである。^(注1)

, (コンマ)を使ってPRINTした例

① PRINT 300, 25, -123456, -82 RETURN

3 0 0	2 5	- 1 2 3 4 5 6	- 8 2
1 2 3 4 5 6	1 2 3 4 5 6	1 2 3 4 5 6 1	2 3 4 5 6 1 2 3

↑
符号の1文字分 (注2)

② PRINT "ハ・ンタ・イ", "カ・ンタ・ム" RETURN

ハ・ンタ・イ	カ・ンタ・ム
1 2 3 4 5 6	1 2 3 4 5 6

; (セミコロン)で間をあけずに複数PRINT

PRINT A; B; C

このようにPRINT文を書くと、A, B, Cのなみが間をあけずに、画面の同じ行に、この順番に書きだされる。^(注3)

; (セミコロン)を使ってPRINTした例

① PRINT 300; 25; -123456; -82

3 0 0	2 5	- 1 2 3 4 5 6	- 8 2
1 2 3 4 5 6	1 2 3 4 5 6	1 2 3 4 5 6	1 2 3

↑
符号の1文字分

② PRINT "5*6="; 5*6

5 * 6 =	3 0
↓ 符号の1文字分	

注1. 6文字あけただけでは前の文字や数字と重なる場合は、さらに6文字あける。

注2. 数字をPRINTする場合、必ず先頭に符号の1文字分がとられる。

注3. 数字の場合、符号の1文字分だけ間があく。

桁をそろえてプリント—PRINT USING

プリント

ユージング

使用例

① PRINT USING "##.##"; 12.3

12.3を#にあわせて画面にだし、
12.30となる。

② PRINT USING "-###.##"; -13

-13を#にあわせて、前に符号を
つけて画面にだす。-13.0とな
る。

③ PRINT USING "#.##"; 0.56

0.56を#にあわせて、後ろに符号
をつけて画面にだす。
-0.6+となる。(小数点以下第2
位を四捨五入した。)

一般型式 PRINT USING "######";
; 数値変数または数字

説明 ●#記号の個数と.(ピリオド)の位置によって、画面に
だす数字の桁をそろえる。#記号は8個まで使える。

●符号をだす場合は、#記号の前か後ろに-をつける。

●画面にだす数字と#記号の型があわない場合は、#記
号が画面にそのまま出てくる。

```

① LIST
100 A=27
200 B=115.57
300 C=0.565
400 D=25.65
500 E=2562.5
600 F=-31
70 PRINT USING "###.##";A,B,C
80 PRINT USING "-##.##";D,E,F
90 END
Ready
RUN
27.00      115.57      0.57
+ 25.5      -##.##      - 31.0
Ready

```

A～Fの6つの変数が行番号70, 80のPRINT USING文でつぎ
のようにPRINTされる。

A…小数点以下に0が2桁はいる。

B…そのままである。

C…小数点以下第3位が四捨五入される。

D…前に+の符号がつく。

E…数字と#記号の型があわないので、#記号がそのままである。

F…前に-の符号がつき、小数点以下に0が1桁はいる。

PRINTで特殊命令

使用例

PRINT "C"

画面の内容がすべて消えて、カーソルが画面の左上すみに移る。

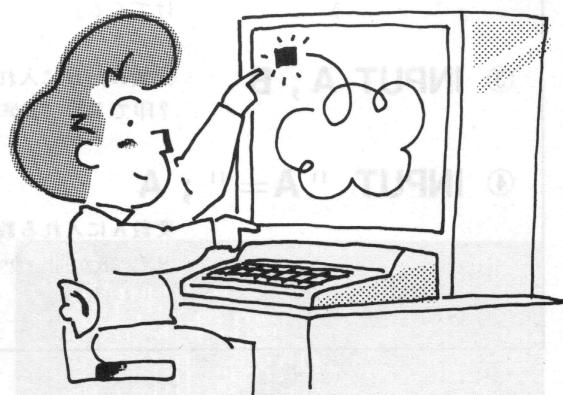
一般型式 **PRINT "特殊文字"**

説明 ●特殊文字に対応する命令を実行する。" " の中は特殊文字をいくつ書いてもよい。

●"特殊文字"のかわりに、CHR\$(数字)を使うことができる。(例) PRINT CHR\$(6)

●特殊文字一覧表

特殊文字	同じ意味のCHR\$	命令内容
↓	CHR\$(1)	カーソルを下に1文字分移動
↑	CHR\$(2)	カーソルを上に1文字分移動
⇒	CHR\$(3)	カーソルを右に1文字分移動
←	CHR\$(4)	カーソルを左に1文字分移動
□	CHR\$(5)	カーソルを画面左上すみに移動
C	CHR\$(6)	画面の内容をすべて消し、カーソルを画面の左上すみに移動
D	CHR\$(7)	カーソルの左にある文字を1文字分消し、右に続く文字をつめる。
I	CHR\$(8)	カーソルの左に文字を割りこませていく。
	CHR\$(9)	何も実行しない。
	CHR\$(10)	SHIFT LOCKの状態にする。
	CHR\$(11)	何も実行しない。
	CHR\$(12)	何も実行しない。
	CHR\$(13)	改行を行なう。
	CHR\$(14)	何も実行しない。
	CHR\$(15)	SHIFT LOCKを解除する。



```
① Ready
LIST
10 PRINT "B"
20 CIRCLE 100, 120, 50
30 CIRCLE 100, 120, 35
40 CIRCLE 100, 120, 20
50 PRINT "B"
60 END
Ready

```

行番号10で画面の内容をすべて消してから、三重丸を描いた。
行番号10を 10 PRINT CHR\$(6) と書くこともできる。

```
② Ready
10 s$="*****"
20 PRINT "0.0.0";s$
30 PRINT "<><>";"GUNDAM"
40 PRINT "0.0.0"
50 END
Ready
RUN
***GUNDAM***
Ready
```

行番号20~40で特殊文字を使った命令を行なっている。" " の中に特殊文字を複数指定している。

注1. 特殊文字のいれ方は14ページの「**CTRL** キーと組めば特殊文字」を、CHR\$()のくわしい説明は85ページの「アスキーコード表を使え」を見てください。

注2. 特殊文字コード表は160ページにあります。

RX-78が問い合わせてくる

INPUT

使用例

① INPUT A

変数Aに入れる数字を、?印でRX-78が問い合わせてくる。

② INPUT A\$

変数A\$に入れる文字(ストリング)を、?印でRX-78が問い合わせてくる。

③ INPUT A, B

変数A, Bに入る数字を順番に?印でRX-78が問い合わせてくる。

④ INPUT "A="; A

変数Aに入れる数字を、A=でRX-78が問い合わせてくる。

一般型式 **INPUT 变数, 变数, ……**

INPUT "文字"; 变数

説明

●変数のなかみを?印、またはINPUT文の中の" "でかこまれた文字(ストリング)で問い合わせてくる。答えをキーボードからいれると、変数にその答えがはいる。

●複数の変数を、(コンマ)で区切ってINPUTの後に書くと、変数のなかみを順番に問い合わせてくる。

●数値変数には数字を、文字変数には文字(ストリング)をいれてやらなければならない。

●実行するたびに変数のなかみを変えられるので、同じプログラムでいろいろな場合の結果を求められる。

```

① LIST INPUT N$, A, B, C
10 01=A+B+C
20 02=(A+B+C)/3
30 PRINT N$
40 PRINT "A="; A, "B="; B, "C="; C
50 PRINT "ヨウケイ"; 01
60 PRINT "ハイキン"; 02
70 END
80 END
Ready
RUN
? TEST
? 20
? 55
? 42
TEST
ヨウケイ 127
ハイキン 42.333333
Ready

```

行番号10のINPUT文によって、変数N\$, A, B, Cのなかみを順番に問い合わせてくる。なかみをすべていれ終ると、行番号20からうしろの命令文の実行を行なう。

```

② LIST INPUT "ヨコシ^ク?(20<X<170)=";
10 INPUT "タテシ^ク?(20<Y<160)=";
20 CIRCLE X, Y, 20
30 END
Ready
RUN
ヨコシ^ク?(20<X<170)=50
タテシ^ク?(20<Y<160)=130
Ready

```



行番号10, 20のINPUT文で円の中心を、横の位置、縦の位置の順に問い合わせてくる。答をいれると、円を描く。

RX-78の記憶をたどれ

リスト
LIST

使用例

- ① **LIST** RETURN RX-78 が記憶しているプログラムのすべてを、行番号順に画面にだす。
- ② **LIST 50** RETURN 行番号50の命令文を画面にだす。
- ③ **LIST 40—** RETURN 行番号40からいちばん大きい行番号の命令文までを、行番号順に画面にだす。
- ④ **LIST —100** RETURN いちばん小さい行番号から行番号100までの命令文を、行番号順に画面にだす。
- ⑤ **LIST 30—70** RETURN 行番号30から行番号70までの命令文を、行番号順に画面にだす。

一般型式 **LIST** はじまり行番号—おわり行番号

説明 ● RX-78が記憶しているプログラムを、行番号順に画面にだす。「はじまり行番号」や「おわり行番号」を書くことによって、プログラムの見たいところだけを画面にだすこともできる。

● プログラムを動かして、エラーがおこったとき、プログラムを調べたり、直したりするのに、LISTを使うと便利である。

● プログラムが画面にでている途中でスペースキーを押すとプログラムのでるのが一度止まる。はなすと続きのプログラムがでてくる。

(1) STEP 1
10 REM サンプルプログラム
20 PRINT "RX-78サンプルプログラム"
30 INPUT A
40 INPUT B
50 INPUT C
60 A=A/P*100
70 B=B/P*100
80 C=C/P*100
90 PRINT "A";A1;"
100 PRINT "B";A2;"
110 PRINT "C";A3;"
120 END

```
①
LIST
10 INPUT A,B,C
20 P=A+B+C
30 A1=A/P*100
40 A2=B/P*100
50 A3=C/P*100
60 PRINT "A";A1;""
70 PRINT "B";A2;""
80 PRINT "C";A3;""
90 END
Ready
```

行番号10～90までのプログラムを入れて、LIST RETURN とやるとプログラムがすべて画面にでてくる。

```
②
LIST 50
50 A3=C/P*100
Ready
LIST 60-
60 PRINT "A";A1;""
70 PRINT "B";A2;""
80 PRINT "C";A3;""
90 END
Ready
```

写真①と同じプログラムを使って、行番号50だけ、行番号60から最後までの命令文を画面にだした。

```
③
LIST -30
10 INPUT A,B,C
20 P=A+B+C
30 A1=A/P*100
Ready
LIST 30-50
30 A1=2/P*100
40 A2=B/P*100
50 A3=C/P*100
Ready
```

写真①と同じプログラムを使って、最初から行番号30まで、行番号30から50までの命令文を画面にだした。

STEP 1 サンプルプログラム(まとめ)

```

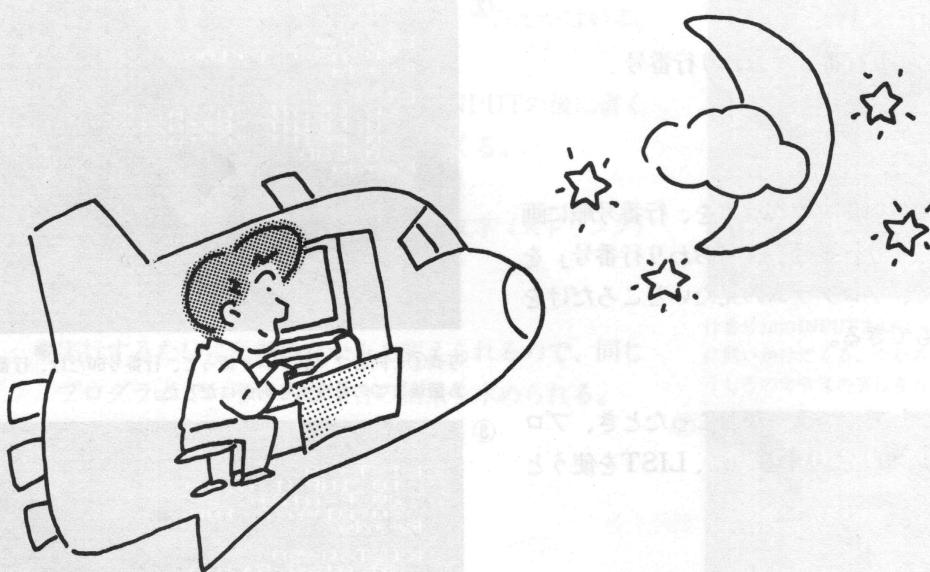
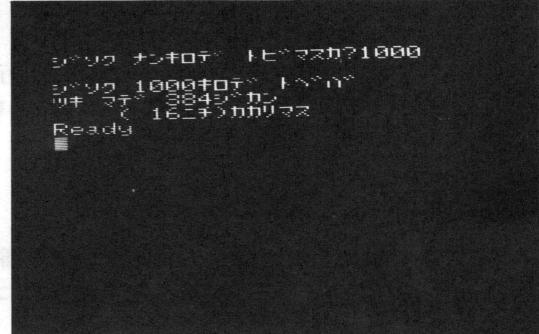
10 REM ** ツキ リヨコウ **
20 PRINT CHR$(6)-----注釈文
30 INPUT "シ"ソク ナンキロテ トヒマスカ?";U-----画面にでている内容をすべて消す
40 S=3.84*10^5-----時速何キロで月まで飛ぶかを入力させる
50 H=S/U-----月までの距離は、 $3.84 \times 10^5$  (km)である。
60 D=H/24-----距離を時速で割り、月まで何時間かかるか計算する
70 PRINT -----月まで何日かかるか計算する
80 PRINT "シ"ソク";U;"キロテ トヒハ"
90 PRINT "ツキ マテ";H;"シ"カン"
100 PRINT " [ ";D;"ニチ]カカリマス"
110 END-----画面を1行分あける
          -----時速何キロで飛べば、月まで何時間(何日)かかる
          -----るかを3行にわたって画面にだす。
          -----プログラムの終りを示す。

```

〔解説〕

RX-78で月旅行にかかる時間、日数をもとめるプログラムを作つてみないか。時速何キロで月まで飛ぶかを入れると、すぐに答えがでてしまう。さあ、科学者になったつもりでどんどん計算してみよう。

-----注釈文
 画面にでている内容をすべて消す
 時速何キロで月まで飛ぶかを入力させる
 月までの距離は、 3.84×10^5 (km)である。
 距離を時速で割り、月まで何時間かかるか計算する
 月まで何日かかるか計算する
 画面を1行分あける
 -----時速何キロで飛べば、月まで何時間(何日)かかる
 -----るかを3行にわたって画面にだす。
 -----プログラムの終りを示す。



* サンプルプログラム中に出てくる〔 〕はキーボード上で〔 〕と同じです。
 * べき乗「^」はキーボード上では↑になります。

STEP 2 RX-78が判断する

STEP1では、いくつかの基本的な命令を覚えた。

STEP2では、さらに一步進んで、条件によって実行の内容をかえたり、同じ処理を何回もくり返したり、先へとんだり、後もどりしたりするための命令を覚えよう。こうした命令を使えば、プログラムは広がりをもってくる。

STEP 2 サンプルプログラム

```
10 REM ** テイシタ ホウコウニ チョッカクサンカクケイ ヲ カワ  
**  
20 PRINT CHR$(6)-----注釈文  
30 INPUT "ウエムキナラ1, シタムキナラ2, オシマイナラ3 ヲ オシナ  
サイ:"; A-----画面の内容をすべて消す  
40 IF A=3 GOTO 240-----三角形の方向またはプログラムの終了をAに指定する  
50 INPUT "タカサハ?" ; N-----Aが3なら、行番号240へとぶ。  
60 IF N<=21 THEN 80-----三角形の高さNを指定する  
70 PRINT "タカサハ 21 マテ" : GOTO 50-----Nが21以下なら行番号80へとぶ。  
80 PRINT CHR$(6)-----「高さは21までですよ」と画面に出す。行番号50へとぶ。  
90 ON A GOTO 100, 170-----画面の内容をすべて消す  
100 FOR I=1 TO N-----Aが1なら行番号100、2なら170、にとぶ。  
110 FOR J=1 TO I-----Iが1からNの間、行番号150までを繰り返す。  
120 PRINT "*";-----Jが1からIの間、行番号130までを繰り返す。  
130 NEXT J-----"*"を画面に出す  
140 PRINT-----行番号110の繰り返し命令の終りを示す。  
150 NEXT I-----画面の行をかえる。  
160 GOTO 30-----行番号100の繰り返し命令の終りを示す。  
170 FOR I=N TO 1 STEP-1-----行番号30へとぶ。  
180 FOR J=1 TO I-----IがNから1の間、行番号220までを繰り返す。  
190 PRINT "*";-----Jが1からIの間、行番号200までを繰り返す。  
200 NEXT J-----"*"を画面に出す。  
210 PRINT-----行番号170の繰り返し命令の終りを示す。  
220 NEXT I-----行番号30へとぶ。  
230 GOTO 30-----プログラムの終りを示す。  
240 END-----
```

【解説】

RX-78はどんな図形でも書くことができる。ここでは一番簡単な図形である三角形を書かせてみよう。

三角形を上向きに書くか、下向きに書くか、書かずに終了するかを指定する。

●上向きなら → 行番号100から行番号150まで上向きの三角形を書く。

●下向きなら → 行番号170から行番号220まで下向きの三角形を書く。

●終了なら → 行番号240にとんでプログラムを終了させる。

行番号70ではマルチステートメントを使っている。くわしくは21ページの「:(コロン)で区切ってマルチステートメント」を参照のこと。

* サンプルプログラム中に出てくる〔 〕はキーボード上で()と同じです。

注釈文

画面の内容をすべて消す

三角形の方向またはプログラムの終了をAに指定する
Aが3なら、行番号240へとぶ。

三角形の高さNを指定する

Nが21以下なら行番号80へとぶ。

「高さは21までですよ」と画面に出す。行番号50へとぶ。
画面の内容をすべて消す

Aが1なら行番号100、2なら170、にとぶ。

Iが1からNの間、行番号150までを繰り返す。
Jが1からIの間、行番号130までを繰り返す。

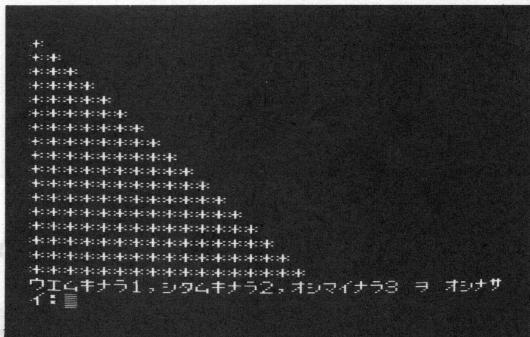
"*"を画面に出す

行番号110の繰り返し命令の終りを示す。
画面の行をかえる。

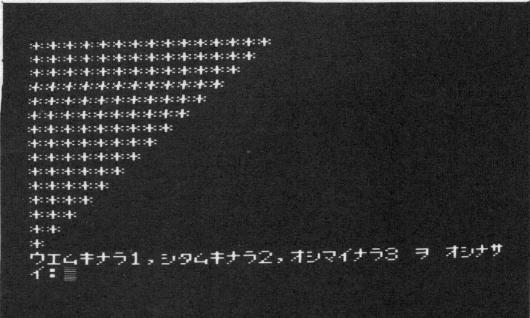
行番号100の繰り返し命令の終りを示す。
行番号30へとぶ。

IがNから1の間、行番号220までを繰り返す。
Jが1からIの間、行番号200までを繰り返す。

"*"を画面に出す。



1を選んだ場合



2を選んだ場合