

Tiny BASIC Shortcuts

Tom Pittman's Tiny BASICs (6502, 1802, etc.) are somewhat limited in capabilities. This is the first of several articles discussing methods to expand those capabilities.

Charles R. Carpenter
2228 Montclair Place
Carrollton TX 75006

Writing small but useful programs in Tiny BASIC (to paraphrase Tom Pittman) is a practical reality. Getting the most out of your programs is easier if you work with the inter-

preter's limitations. The utility program in Fig. 1 shows how to work with some of these limitations. This program is titled "Loans," but it could be any comparison of WHAT-IF alternatives. Here's what we'll be working with (and without):

- Decimal numbers not allowed.
- Number range limited from -32768 to +32767.

- 72 characters maximum on input lines.
- Implied statements and abbreviations to save bytes of memory.

(Note: Tom Pittman now has an experimenter's manual available that explains many of these features and how to work with them. They are not as simple as my approach. The manual is available from Itty

Bitty Computers, PO Box 23189, San Jose CA 95153.)

These are not significant handicaps if you're estimating the effect of several alternatives. Round numbers are usually acceptable if you only want to get on base in some specific ball park (cliches are fun once in a while).

Byte-saving Tips

Saving bytes of memory is a practical approach if your computer has limited memory (I have 1250 bytes of free space now). Let's talk about the memory-saving part first.

Fig. 1 is an example of a program with no statement shortcuts; Fig. 2 uses all the implied and abbreviated statements possible in this Tiny BASIC interpreter. Memory in Fig. 1 is 492 bytes, an average of 17 bytes per line, while Fig. 2 uses 410 bytes for an average of 14 bytes per line. REM comments were added later and used 470 bytes.

Using implied statements causes the program to run

```

:
:LIST
10 REM TINY BASIC FOR KIM-1
11 REM 6502 V.1K BY T. PITTMAN.
12 REM
13 REM PROGRAMMED BY:
14 REM C.R. (CHUCK) CARPENTER WSUSJ
15 REM 2228 MONTCLAIR PL.
16 REM CARROLLTON TX 75006
17 REM
18 REM THESE PROGRAMS ILLUSTRATE BYTE SAVING
19 REM TECHNIQUES IN LIMITED MEMORY SYSTEMS.
20 REM THE FIRST PROGRAM USED 492 BYTES. THE
21 REM OTHER USED 410 BYTES. AN INCREASE
22 REM (OR SAVING) OF 82 BYTES. IMPLIED
23 REM STATEMENTS AND ABBREVIATIONS ARE
24 REM THE REASON.
25 PR
26 PR
100 PRINT"LOANS : HOW MANY -"
110 INPUT N
115 PRINT
120 LET A=0
130 PRINT"INPUT: PRINCIPAL IN HUNDREDS (P)"
140 PRINT"    RATE IN PERCENT (R)"
150 PRINT"    TIME IN YEARS (T)"
160 PRINT"    PAYMENTS IN MONTHS (X)"
170 INPUT P,R,T,X
190 LET I=P*T*R
200 LET O=100*P+I
210 LET M=O/X

220 LET A=A+1
230 PRINT
240 PRINT"LOAN NUMBER -";A;"
250 PRINT"INTEREST IS $";I
260 PRINT
270 PRINT"MONEY OWED IS $";O
280 PRINT
290 PRINT"PAYMENTS ARE $";M
300 PRINT
310 LET N=N-1
320 IF N>0 THEN GOTO 170
360 PRINT
370 PRINT"DONE"
380 PRINT
390 END
:
:
:
:
:I=0
:I I=I+2
:2 GOSUB 1
:RUN

1226 AT 1
:END

:PRINT"THERE ARE ";I;" BYTES LEFT"
THERE ARE 288 BYTES LEFT
:

```

Fig. 1. First program version using no shortcuts to write the program or save bytes. This program uses 492 bytes, exclusive of the REM statements. REM statements use 470 bytes. The short routine above illustrates how Tiny BASIC finds the number of bytes of free space remaining. The user's manual tells how to do it.

slower, but the increase in program lines is worth the loss of speed (if speed is your concern then Tiny BASIC may not be for you, anyway). Memory saving wasn't really necessary for this short program; but in a 100-line program over 200 bytes could be saved (12 to 15 lines' worth). Such significant savings allow you to write longer programs. The programs are still small, but even a few more lines make them more useful. And that's what we're trying to do. Bytes could be saved in a few more places, such as the spaces in the print input, lines 130 through 160, but in the interest of clarity, I left them alone.

Decimal Values

Calculations involving decimal numbers can be handled several ways. Anytime a percentage or a calculation resulting in a fraction occurs, a decimal number results. Dollars and cents are decimal numbers, too. Tiny BASIC truncates decimal numbers down to the next lower whole number. If the number is less than one, the result is zero. (For this

reason, accountants would probably not want to use Tiny BASIC.)

Lines 130 through 180 are the input lines for this program. I used principal in hundreds and rate in percent to avoid decimal percentage entry and to prevent dividing percent by 100 (to get back to a decimal percentage). The math comes out right when it's printed out in line 250. I then multiplied the total loan value by 100 in line 200 to make the right amount print in lines 270 and 290.

Principal input in hundreds also helps avoid the number-limitation problem. Keeping the numbers to be operated on small limits precision but keeps the multiplication results in range. Adding a statement in a print line to multiply (or divide, etc.) by some factor will put the answer back in the right magnitude. This is sort of like using engineering notation with a slide rule. The difference is the lack of decimal numbers.

An input-line limitation of 72 characters restricts the amount of data you can input. Two character spaces are used

by the prompting question mark and following space. This reduces actual data input to 70 characters, including the required commas between the data entries. With the loan amount in hundreds, I was able to input values for six loans instead of five. To overcome the limited data-input situation, write programs that will perform calculations, hold the results and return for more

data. I've done this on some data-processing routines with good results.

There's another way to accommodate more data than the line will hold. Simply input as many loan numbers (or WHAT-IFs) as needed in line 100. When the program has used the data entered, it will ask for more until the number of N entries is reached in line 320. Question marks will show up each time

```
:LIST
100 PR"LOANS : HOW MANY -"
110 INPUT N
115 PR
120 A = 0
130 PR"INPUT: PRINCIPAL IN HUNDREDS (P)"
140 PR"      RATE IN PERCENT (R)"
150 PR"      TIME IN YEARS (T)"
160 PR"      PAYMENTS IN MONTHS (X)"
165 PR
170 INPUT P,R,T,X
190 I = P*T*R
200 O = 100*P + I
210 M = O/X
220 A = A + 1
230 PR
240 PR"LOAN NUMBER - ";A;" "
250 PR"INTEREST IS $";I
260 PR
270 PR"MONEY OWED IS $";O
280 PR
290 PR"PAYMENTS ARE $";M
300 PR
310 N = N - 1
320 IF N > 0 GOTO 170
360 PR
370 PR"DONE"
380 PR
390 END
```

Fig. 2. Second program version using implied statements and abbreviations to save bytes. This version uses 410 bytes.

```
LOANS : HOW MANY -
?6

INPUT: PRINCIPAL IN HUNDREDS (P)
      RATE IN PERCENT (R)
      TIME IN YEARS (T)
      PAYMENTS IN MONTHS (X)

?40,10,3,36,40,12,4,48,40,18,5,60,50,10,3,36,50,1
2,4,48,50,18,5,60

LOAN NUMBER - 1
INTEREST IS $1200

MONEY OWED IS $5200

PAYMENTS ARE $144

LOAN NUMBER - 2
INTEREST IS $1920

MONEY OWED IS $5920

PAYMENTS ARE $123

LOAN NUMBER - 3
INTEREST IS $3600

MONEY OWED IS $7600

PAYMENTS ARE $126

LOAN NUMBER - 4
INTEREST IS $1500

MONEY OWED IS $6500

PAYMENTS ARE $180

LOAN NUMBER - 5
INTEREST IS $2400

MONEY OWED IS $7400

PAYMENTS ARE $154

LOAN NUMBER - 6
INTEREST IS $4500

MONEY OWED IS $9500

PAYMENTS ARE $158

DONE
```

Fig. 3. Sample run. Simple interest calculations of two different loan values at three rates.

From Fig. 3 Simple Int			From Fig. 5 Compound Int		
Interest%	Years	Amount	Equiv-Int%	Years	Amount
1. 10	3	\$5200.00	11	3	\$5320.00
2. 12	4	5920.00	15	4	6400.00
3. 18	5	7600.00	26	5	9200.00

Mult	Actual Loan Value	Difference
1. 1.331	\$5324.00	+\$ 4.00
2. 1.574	\$6296.00	- 104.00
3. 2.288	\$9152.00	+ 48.00

Fig. 4. For a loan of \$4000.

line 170 runs out of data and line 320 is still greater than zero.

This program only calculates simple interest loans. Compound-interest calculations require decimal numbers and raising numbers to some power. The multiplier for compounding over n periods is $(1 + I)^n$, where I is the interest expressed as a decimal and n is the number of years (or periods).

You can use this multiplier to calculate the approximate equivalent while percentage over the term of the loan. Your calculated answer will result in a much more realistic loan evaluation. I made some of these calculations, and Fig. 4 has some examples.

In the program itself, there are no unusual or unique programming techniques. There are two counting loops—one starting at line 110 and the other at line 120. Whatever

value is input for N is decremented in line 310 until the data sets, input in line 170, are used up. The counter that starts in line 120 numbers the printed output each time a pass through the program is completed.

I tried to use N to do both, but could not without using more program lines. Otherwise, this is simply a fundamental program with input between lines 100 and 170, calculations between lines 190 and 220 and output between lines 240 and 290.

Summary

It is easy to save bytes of memory if you remember to use implied statements and statement abbreviations. The user's manual for Tiny BASIC shows what is, and is not, allowed. Both the decimal number and number range limitation can be handled by using software math techniques (multipliers, dividers, engineering notation,

```

LOANS : HOW MANY -
?3

INPUT: PRINCIPAL IN HUNDREDS (P)
      RATE IN PERCENT (R)
      TIME IN YEARS (T)
      PAYMENTS IN MONTHS (X)

?40,11,3,36,40,15,4,48,40,26,5,60

LOAN NUMBER - 1
INTEREST IS $1320

MONEY OWED IS $5320

PAYMENTS ARE $147

LOAN NUMBER - 2
INTEREST IS $2400

MONEY OWED IS $6400

PAYMENTS ARE $133

LOAN NUMBER - 3
INTEREST IS $5200

MONEY OWED IS $9200

PAYMENTS ARE $153

DONE
  
```

Fig. 5. Loan value two, rerun to show the effect of compound interest on the total loan value. Compare the results with the simple interest calculation.

etc.). Line input characters limited to 70 (72 with prompting question mark and space) can also be handled by programming techniques.

Remember, if you input more than a total of 72 characters in a single line, the program will stop. Nothing more will happen

until you reset your system. If you have to reset and want to save the program already in memory, then reenter the interpreter at the soft entry point. The Tiny BASIC user's manual explains how to do this, too. A program does not have to be big to be useful. ■

North Star Software

Mailist

Mailist is a general purpose mailing label program capable of producing formatted lists for tractor-fed or Xerox type labels. Mailist will also sort lists for any field.

Price \$39.95 on diskette with manual/stock to 14 day delivery.

In-out driver

Dos in-out driver is designed to set up mapped memory video boards in conjunction with hard copy device. The user may switch output under software control. Any file directory may be listed while in BASIC without jumping to dos. Spacebar will stop output for line by line listings. Designed for use with 3P+S and any tv board.

Price \$12.95 on diskette with manual/stock to 14 day delivery.

Register

Register is a cash register and inventory control program. The software will control a point of sale terminal and printer. It will search inventory for an item, price and ticket #. Register has provisions for min-max, automatic reorder, and critical list.

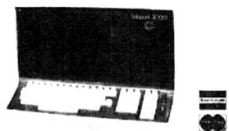
Price \$299.95 on diskette with manual.

All prices are FOB Santa Barbara, California. Terms CDD Residents add 6% sales tax and \$1.00 shipping.

Alpha Data Systems A48

Box 267, Santa Barbara, Ca. 93102 • 805/682-5693

Datapoint CRT Terminals



Fully-Assembled — Guaranteed

#3360 **\$649.50**

- Add \$15 packing
- Guaranteed
- Add \$45.50 for scrolling mod, or do it yourself
- NOW — Power your KIM-1 or other small processor from these terminals. Up to 2 Amps at 5, 14, 25 volts.
- Model 3360 speeds from 300-4800 Baud, numeric keypad, cursor controls, Edit, Block-Transmit, search modes, ASCII keyboard with codeable options.
- Green phosphor, 24.82 Ch lines, addressable cursor, RS-232C serial interface, other speeds available.
- Manual \$10, Cable kit \$9.95, Datashare/IBM compatible version \$1,100 • Model 3000 \$675
- M-33 KSR Teletypes \$595, ASR \$3735
- Call us for service on CRTs, micros, main-frames. Leasing quantity discounts available.

TELECOMMUNICATIONS SERVICES CO.

Box 4117, Alexandria, Va. 22303

703-683-4019 / TLX 89-623

T26

RAINBOW COMPUTING INC.

Supplier of

Apple

Wave Mate

The Digital Group

Southwest Technical Products

Digital Equipment Corporation

Computer Products

Peripherals and Supplies from

PerSci Computer Devices

Contronix Lear Siegler

Diablo Multi-Tech

Maxell Texas Instruments

'Scotch' Brand Magnetic Media

Specialists in Design, Implementation and

Support of Custom Hardware/Software

for Business, Educational, and

Personal Use

Consulting/Contract/Programming

Operating Systems/Applications Software

Experts in most major computer

software including

CDC, IBM, PDP

BASIC, COBOL, FORTRAN, PL1

Lisp, Simula, Snobol, SPSS, BMD's

COMPASS, MACRO, 6800, & Z80 assembly languages

10723 White Oak Ave., Granada Hills, Ca. 91344

(213) 360-2171

R10

**JOHNSON
COMPUTER**

P. O. BOX 523, MEDINA, OHIO 44256
(216) 725-4560

FINALLY!

HDE inc. FLEXIBLE DISK SYSTEM FOR KIM

Features:

- *Top Quality Industrial Grade Controller
- *Proven High Reliability Disc Drive
- *Line-Numbered Text Entry and Editing
- *A Powerful Command Structure
- *Adaptation to any 6500 Based System
- *Capability for User Defined Commands
- *Complete Compatibility with KIM
- *Multiple Resident Files
- *Indexed and Non-Indexed Disk Storage

Complete 90 Day Parts and Labor Warranty

HDE FILE ORIENTATED DISK SYSTEM

"FODS"

Includes:

- *Full Size Sykes Drive
- *6502 Based Controller
- *Power Supply
- *FODS Software
- *Cables, Interface Card
- *User Manual

HIGH RESOLUTION GRAPHICS

In response to your requests, we now offer the K-1008, a Dot Matrix display board (320H x 200V) for the KIM-1.

But — we didn't stop there. We also call it an 8K memory board, directly connected to your KIM-1. Full read/write with no wait states or snow ever.

And — we made it low power to reduce system costs. In fact our 18 watt K-1000 power supply can typically power your KIM-1 plus 32K of K-1008 memory.

How to use it — The K-1008 visible memory only needs a power supply and a KIM-1 to function as memory. Add a standard monitor and you have high resolution graphics for diagrams, graphs, even variable font text up to 22 lines of 53 characters.

K-1008 Assembled/Tested

\$289.00 board \$40.00

Graphic Software Listing

\$20.00

K-1000 power supply \$40.00

K-1005 5 slot card file \$69.00

Micro Technology Unlimited

P.O. Box 4596

Manchester, N.H. 03103

M44