"DOS NT" THE FINAL DOS (SEE PAGE 60)

COMPUTE

MAY 1993

DO-IT-YOURSELF WINDOWS

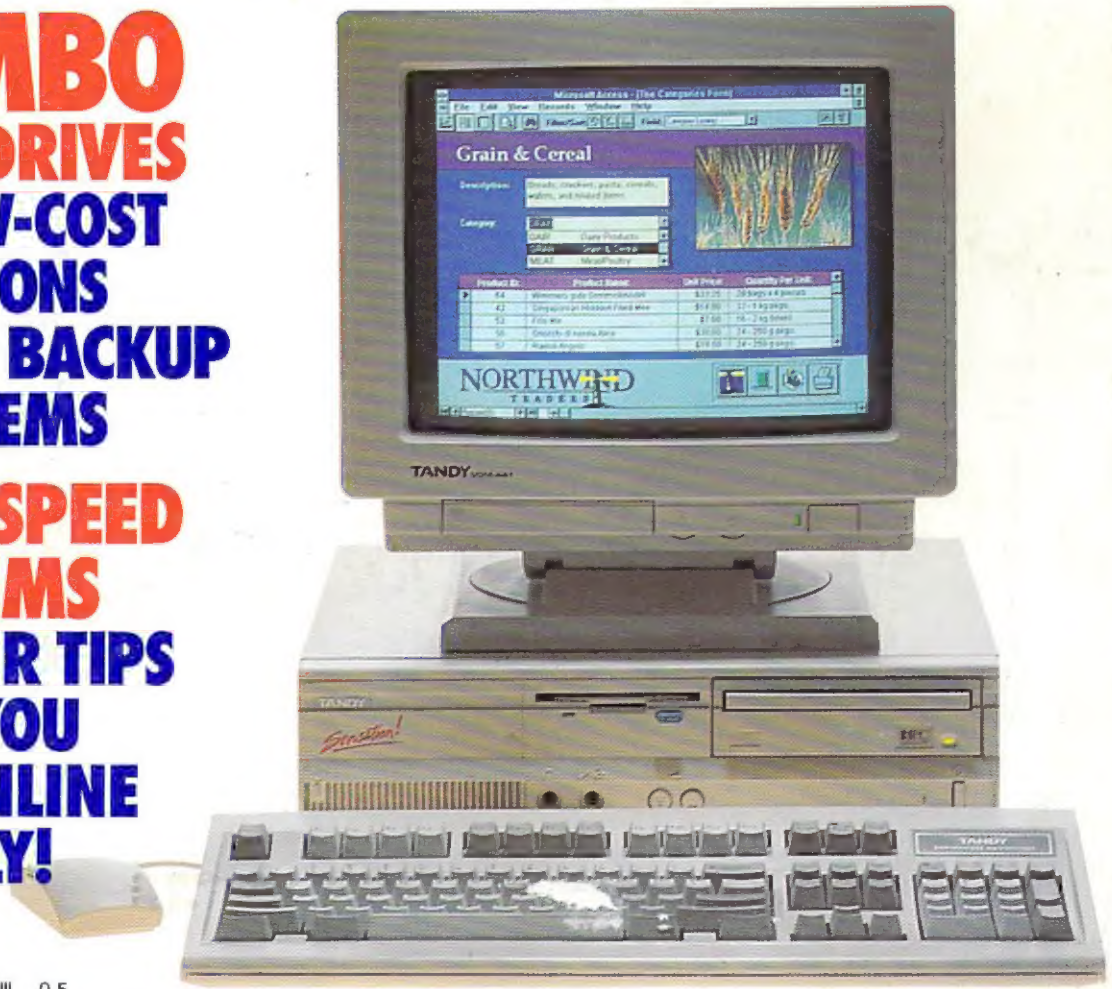
3 GREAT TOOLS LET YOU CREATE YOUR OWN APPS

JUMBO TAPE DRIVES

10 LOW-COST SOLUTIONS TO BIG BACKUP PROBLEMS

HIGH-SPEED MODEMS

6 SUPER TIPS HELP YOU GET ONLINE AND FLY!



U.S. $2.95

0 71486 02193 3 05

64/128 VIEW

Guess what, GEOS fans! You've got your own magazine—again!

Tom Netsel

Computer publications have a way of appearing and disappearing faster than a magician's rabbit. But when one pops up for 64 and 128 users, I have this reaction to wish it well—despite the fact that it's a potential competitor. After all, it's nice to see someone else taking a serious interest in our machines.

Rather than trying to cover the whole Commodore 8-bit spectrum, publisher and managing editor Grady Brown is focusing his attention on GEOS with his new journal, *geoVISION International*. The premier issue has just arrived, and it looks impressive—and the typeface is clear and legible. Too many GEOS publications are tough to read.

A bimonthly publication of 28–32 laser-printed pages, *geoVISION International* plans to print columns once found in other Commodore and GEOS magazines in addition to new articles. Brown states in his editorial that the philosophy behind his new publication "is to bring all GEOS users and programmers together, spreading the knowledge, experiences, and information of GEOS users worldwide, and to give you the best GEOS support possible."

That's quite a goal for any publication, but Brown offers a good selection of articles and information. Articles in the first issue include what's hot and what's not on the big telecommunication networks, a tutorial on geoPaint by Australian artist Jane Voskamp-Jones, a geoWrite tutorial, and a roundup of the four GEOS programming languages.

There are reviews of several GEOS products, including geoCanvas, Perfect Print LQ, Dual Top, and Collete Utilities. (For more information about Collete Utilities, see Steve Vander Ark's "GEOS" column in this issue.) There's a list of bulletin boards, publications, and user groups that offer GEOS support. There are even several Australian newsletters and groups mentioned.

In his editorial, Brown addresses questions that are bound to be asked by people who have subscribed to magazines only to have them cease publication without honoring their obligations. "We cannot guarantee we'll be around forever. No one can. But we do promise to follow through with all our commitments to each and every subscriber."

In addition to the magazine, Brown offers six GEOS disks per year. They include programs and utilities covering fonts, desk accessories, applications, drivers, geoCalc and geoFile templates, and clip art.

To give *geoVISION International* a try, U.S. subscription rates are $21 for six issues or $4 for a single issue. Washington residents must add local sales tax. Canadian and Mexican subscriptions are $27 per year, $5 for a single issue. International subscriptions (via air mail) are $33 per year, $6 for a single issue.

Disk subscriptions are $30 for six issues in the U.S., $39 in Canada and Mexico, and $45 elsewhere. Order by writing to geoVISION International, 816 Southeast Polk Street, Camas, Washington 98607. ❐

GAZETTE

Screen

TRY THESE NINE PROGRAMMING GEMS TO

One of the biggest advantages the 128 has over its little brother, the 64, is its powerful built-in BASIC 7.0 language. This language's rich array of 165 different commands (not counting OFF and QUIT which were planned but never implemented) allows you to do many things more easily than you can on the 64, not the least of which is graphics. Consequently, you can create interesting graphic displays on your 128 with a minimum of effort and without using complex machine language.

That leads me to the subject of this article. Following are nine concise programs that show off some of BASIC 7.0's graphical talents. These hacks, or screen gems, do a variety of fascinating things on both 40- and 80-column screens. I did my best to write them in clear, straightforward code so you can modify or incorporate them easily into your own programs. Experimenting with programs is one of the best ways to learn to harness BASIC 7.0's power for yourself.

Now on to the gems. Let's first look at some colorful sparklers that work on 40-column screens.

Sprite Chaos

Sprite Chaos is a short sprite demonstration program. First, all eight sprites, depicted as colored spheres, are set in motion at random angles and speeds. Here's the twist. Once they're moving, the process is repeated, but with new random angles and different speeds. This occurs continuously, resulting in sprites that fly around the screen in unpredictable and interesting ways. You might want to use the effect as an eye-catching backdrop for a title screen in your own programs.

```
1 REM SPRITE CHAOS
10 COLOR 0,1:COLOR
   4,1: COLOR 5,13:
   GRAPHIC O,1
```

Gems

WRITTEN BY
HENNING
VAHLENKAMP

SEE JUST WHAT BASIC 7.0 CAN DO IN A 128.

```
20 CHAR ,13,10,"SPRITE
   CHAOS!"
30 FOR D=0 TO 63:READ
   V:POKE3584+D,V:
   NEXT
40 FOR I=2 TO 8:SPRSAV
   1,I:NEXT
50 SPRCOLOR 16,16
60 FOR S=1 TO 8:MOVSPR
   S,0,0:NEXT
70 FOR S=1TO8:SPRITES,
   1,S+1,1,1,1,1:NEXT
80 DO:FOR I=1 TO 8
90 A=INT(RND(1)*360+.5)
```

```
100 S=INT(RND(1)*15+1)
    :IF S<5 THEN 100
110 MOVSPR I,A #S
120 NEXT:LOOP
130 DATA 0,170,0,2,90,
    128, 9,106
140 DATA 160,9,170,160,
    37,170,168,38
150 DATA 170,168,38,170,
    168,150,17 0,170
160 DATA 154,170,170,
    170,170,170,170,170
170 DATA170,170,170,170,
    170,170,170,170
180 DATA 170,170,42,170,
    168,42,170,168
190 DATA 42,170,168,10,
    170,160,10,170
200 DATA 160,2,170,128,0,
    170,0,0
```

Curve Explosion

Picture an infinite number of different parabolic curves streaming out like a fountain from a central point on the screen. That's what Curve Explosion does with its short plotting routine. The program works like a circle algorithm, but it draws only half a circle. Each curve has a common starting point. Run this one on your 128 to see the intriguing display.

```
1 REM CURVE EXPLSION
10 COLOR 1,2:COLOR
   0,1:COLOR1,4
20 GRAPHIC 1,1
30 DRAW ,0,199 TO 319,
   199
40 DO:A=INT(RND(1)
   *70+10)
50 B=INT(RND(1)*150+10)
60 C=INT(RND(1)*2+1)
70 IF C=1 THEN BEGIN
80 D=3.14:E=-.1:F=-.2:
   G=A:BEND:ELSE
   BEGIN
90 D=0:E=3.3:F=.2:G=-
   A:BEND
100 LOCATE 160,199
110 FOR R=D TO E STEP F
```

```
120 X=INT(A*COS(R)):
    Y=INT(B*SIN(R))
130 DRAW TO 160+G+X,
    199-Y
140 NEXT R:LOOP
```

String Bounce

Are you looking for a new way to display a message on your 128? If so, try String Bounce. Just put any message into A$, and this program will move it around

the screen. If the message hits a border, it bounces off and continues in the opposite direction. Your message will leave a trail as it moves if you change the color in line 90. Change the 39 in lines 30 and 120 to 79 to make the program work on the 80-column screen. Keep the message reasonably short, or it will move down excessively.

```
1 REM STRING BOUNCE
10 SCNCLR:A$="COMMODORE!"
20 COLOR 5,2:COLOR 0,1:COLOR
   4,1
30 X=INT(RND(1)*39-LEN(A$)+.5)
40 IF X<0 THEN 30
50 Y=INT(RND(1)*23+1)
60 CHAR ,X,Y,A$:DX=1:DY=1
70 V=INT(RND(1)*3-1)
80 IF V=0 THEN 70
90 DO:XO=X:YO=Y:COLOR 5,1
100 CHAR ,XO,YO,A$:X=X+V*
    DX:Y=Y+V*DY
110 IF X<1 THEN DX=-DX:X=XO
120 IF X>39-LEN(A$) THEN DX=-
    DX:X=XO
130 IF Y<1 THEN DY=-DY:Y=YO
140 IF Y>23 THEN DY=-DY:Y=YO
150 COLOR 5,2:CHAR ,X,Y,A$
160 LOOP
```

Star Shapes

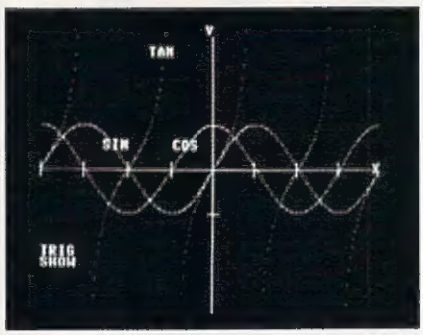
Star Shapes puts simple trigonometry to use so you can create an infinite variety of star-like designs. When you run the program, it asks you for an x and y radius. These are the radii of two circles. Lines are drawn from each of ten points arranged in a circle to each of ten points around your circle. The results produce elaborate symmetrical star shapes. Run it to see what I mean. These programs have few frills, so you'll have to hit Run/Stop–Restore and type RUN again to get this one to repeat.

```
1 REM STAR SHAPES
10 COLOR 1,2:COLOR 0,1:COLOR
   4,1
20 INPUT"[SHFT CLR][CRSR DN]
   ENTER X RADIUS: ";X1
30 INPUT"ENTER Y RADIUS: ";Y1
40 FOR D=1 TO 10:READ A(D),B(D)
   :NEXT
50 GRAPHIC 1,1
60 FOR L=1 TO 10
70 FOR R=0 TO 6.28 STEP .628
80 LOCATE A(L),B(L)
90 X=INT(X1*COS(R)):Y=INT (Y1* SIN
   (R))
100 DRAW TO X+160,Y+100
110 NEXT R,L
120 DATA 249,100,232,147,187,176
130 DATA 132,176,87,146,70,99
140 DATA 87,52,132,23,188,24,233,53
```

Brownian Symmetry

This little hack differs from typical kaleidoscope-type programs by simulating Brownian motion, the random movement of microscopic particles in a fluid. I won't attempt an explanation of this phenomenon here, but it can produce eye-catching designs.

The program works by displaying a pixel influenced by Brownian motion and copying it across four lines of symmetry, resulting in eight reflections. You can choose hi-res or multicolor mode for the display as well as whether to erase the points after they're plotted. I think you'll find that Brownian Symmetry shows that there can be beauty in scientific phenomena.




```
1 REM BROWNIAN SYMMETRY
10 V=1:COLOR 0,1:COLOR 4,1
20 INPUT"[SHFT CLR][CRSR DN](H)I-
   RES OR (M)ULTI ";GR$
30 IF GR$="H" THEN SC=1:ELSE SC
   =3:V=2
40 INPUT"[CRSR DN]ERASE POINTS
   (Y/N)";ER$
50 X=0:Y=0:P=2:GRAPHIC SC,1
60 IF SC=3 THEN P=INT(RND(1)* 15
   +2)
70 COLOR 1,P
80 D=INT(RND(1)*15+1)
90 MX=INT(RND(1)*(D*2+.5)-D)
100 MY=INT(RND(1)*(D*2+.5)-D)
110 X=X+MX:Y=Y+MY
120 IF X<-100 THEN 80
130 IF Y<-100 THEN 80
140 IF X> 100 THEN 80
150 IF Y< 100 THEN 80
160 FOR L=1 TO 2
170 DRAW ,(160-X)/V,100+Y
180 DRAW ,(160-Y)/V,100+X
190 DRAW ,(160-X)/V,100-Y
200 DRAW ,(160-Y)/V,100-X
210 DRAW ,(160+X)/V,100+Y
220 DRAW ,(160+Y)/V,100+X
230 DRAW ,(160+Y)/V,100-Y
240 DRAW ,(160+Y)/V,100-X
230 IF ER$="N" THEN 60
240 COLOR 1,1: NEXT L: GOTO 60
```

Star Bursts

Your monitor screen goes black and then slowly fills with an infinite variety of distinct, colorful explosions or star bursts. That's what the following program will do on your 128. Each star burst consists of 25 randomly selected and colored rays emanating from a central point. Try it; I think you'll find this one is a real eye-catcher!

```
1 REM STAR BURSTS
10 COLOR 0,1:COLOR 4,1
20 GRAPHIC 3,1:DO
30 X1=INT(RND(1)*159)
40 Y1=INT(RND(1)*199)
50 FOR RP=1 TO 25
60 CS=INT(RND(1)*3+1)
70 CO=INT(RND(1)*7+2)
80 X2=INT(RND(1)*24-11.5)
90 Y2=INT(RND(1)*40-19.5)
100 COLOR CS,CO
110 DRAW CS,X1,Y1 TO
    ABS(X1+X2),ABS(Y1+Y2)
120 NEXT:LOOP
```

Trig Show

Beginning math students often have trouble remembering the six basic trigonometric curves: sine, cosine, tangent, cosecant, secant, and cotangent. Trig Show helps by drawing each curve one at a time from -2 times pi to 2 times pi along the x-axis. It also shows each curve's relationship to another curve since all of them appear and overlap on the same screen. When one curve finishes its plot, press Return to see the next one. Try this useful visual aid to increase your understanding of these trigonometric fundamentals. When typing this one in, remember that to enter pi in line 250, hold down the Shift key while simultaneously pressing the up-arrow (↑) key.

```
1 REM TRIG SHOW
10 COLOR 0,1:COLOR 4,1:COLOR
   1,2
20 GRAPHIC 1,1
30 CHAR ,0,19,"TRIG"
40 CHAR ,0,20,"SHOW"
50 DRAW ,0,100 TO 319,100
60 DRAW ,159,0 TO 159,199
70 FOR X=0 TO 319 STEP 39.75
80 DRAW ,X,95 TO X,105:NEXT
90 DRAW ,155,70 TO 165,70
```

SOFTWARE SUPPORT
INTERNATIONAL
C-64/128 Specialists For Over Six Years

Thousands of Commodore Owners Choose Us As Their One Stop Supplier . Here's Why!

* **Great Selection of Front Line Software.**
  We stock most major Commodore titles that are still in Production.
* **Hardware & Peripherals.**
  Although we can't get keyboards and disk drives any longer, we can offer disks, modems, cables, interfaces, and much more.
* **Closeouts & Bargains.**
  We're constantly seeking out those hard to get gems, and offering them at prices that would make Scrooge grin.
* **Used Software Trade In Program.**
  A Software Support exclusive. It takes a bit of extra work, but we don't mind. Another commitment to personalized service.
* **Un-Advertised Specials.**
  Our regular customers enjoy VIP service and great bargains that are hard to pass up.
* **Expert Technical Support.**
  As a regular Software Support customer, you have access to the most knowledgeable Commodore technical departments in the country. Of this, we're darn proud.
* **Friendly & Courteous Order Takers.**
  You'll have a hard time finding order takers as anxious to help you as Sandy, Ann and Connie.
* **Fast Order Processing.**
  Thanks to our automated computer system, huge inventory and effecient shipping staff, most orders are processed the same day.
* **Walk In Sales Department.**
  We welcome visitors to our warehouse outlet. The same great deals are available and best of all, no shipping charges.
* **Worldwide Distributor Of Major Products.**
  Ever hear of Maverick, Super Snapshot, Super 1750 Clone, and the 1581 Toolkit? We've been actively involved in the ground up production and world-wide distribution of these and many other fine Commodore products.
* **Purchase From Us With Confidence.**
  We were very pleased when the Better Business Bureau came to us and told us that our outstanding reputation qualified us as members in their organization. We're here to stay, not a fly by night, here today and gone tomorrow outfit.
* **Extensive Catalog.**
  Our contacts at other mail order companies have told us that producing a catalog such as ours is too time consuming and worst of all - too expensive. We say it's the least we can do for our customers, and wouldn't have it any other way. Simply call or write for your free copy.

Circle Reader Service Number 190

Check Out These Examples...

GAMES

Tony LaRussa Baseball	$9.97
Bard's Tale 3	$12.97
Airborne Ranger	$9.97
Land, Sea, & Air	$19.97
Test Drive 2	$12.97
California Challenge	$8.97
European Challenge	$8.97
Muscle Cars	$8.97
Super Cars	$8.97
Jack Nicklaus Golf	$12.97
J.N. Courses 2, 3 or 4	$6.97
TV Sports Football	$9.97
Batman(Arcade Ver)	$9.97
Dragons of Flame	$9.97
Marvel Comics Trilogy	$12.97
Operation Wolf	$9.97
Shoot'Em Up Const.	$9.97

EDUCATIONAL

Color Me	$9.97
Keyboard Cadet	$6.97
Carmen Time	$26.97
Ideas for C-64 Book	$8.97
Tri Math	$9.97
Donald's Alphabet	$9.97
Sky Travel	$19.97
Perfect Score SAT	$9.97
Songwriter	$9.97

PRODUCTIVITY

Super 1750 Clone	$99.95
Dome Accounting	$37.97
Big Blue Reader 64/128	$29.97
Maverick 64	$39.97
Graphic Label Wizard	$14.97
Home Designer 128	$24.97
Sylvia Porter 128	$19.97
Mastertype/Writer	$12.97
Printshop	$29.97
Word Writer 6	$29.97
Paperclip Publisher	$29.97

Items Listed Above Do Not Include Shipping. Call Or Write For Your Free Catalog Listing Hundreds Of Products And Special Offers For Your Computer. Our Order Takers Are On Duty 6:00 a.m. - 5:00 p.m. M - F and 7:00 a.m. - 3:00 p.m. Sat. - Pacific Time.

SOFTWARE SUPPORT

Software Support Int.
2700 N.E. Andresen Rd.
Suite A-10
Vancouver, Wa 98661
(206) 695-1393

CALL TOLL FREE TODAY! **1-800-356-1179**

Major Credit Cards Accepted.

```
100 DRAW ,155,130 TO 165,130
110 CHAR ,39,12,"X":CHAR ,19,0,"Y"
120 BB=0:DEF FNY(X)=SIN(X)
130 CHAR ,7,10,"SIN":GOSUB250
140 BB=0:DEF FNY(X)=COS(X)
150 CHAR ,15,10,"COS":GOSUB250
160 BB=1:DEF FNY(X)=TAN(X)
170 CHAR ,12,2,"TAN":GOSUB250
180 BB=1:DEF FNY(X)=1/SIN(X)
190 CHAR ,30,22,"CSC":GOSUB250
200 BB=1:DEF FNY(X)=1/COS(X)
210 CHAR ,12,22,"SEC":GOSUB250
220 BB=1:DEF FNY(X)=1/TAN(X)
230 CHAR ,30,2,"COT":GOSUB250
240 END
250 FOR X=.05 TO 4*(π) STEP .05
260 IF BB=1 THEN IF FNY(X)*30>=-
    100 AND FNY(X)*30<=100 THEN
    DRAW ,X*25.5,100-FNY(X)*30
270 IF BB=0: THEN DRAW ,X*25.5,
    100- FNY(X)*30 280 NEXT:WAIT
    212,1:RETURN
```

Now let's shift our attention to some 80-column gems.

Close and Open

Over the years, many routines have been written to clear the 40-column text screen in different ways. The following short routine demonstrates an interesting screen clear for the 128's often neglected 80-column text screen. First, text is displayed, and the program waits for a keypress. Then the left and right sides of the screen come together, squeezing out the text. The screen is cleared, new text is printed, and the screen expands again to reveal it. Try this one in your own programs instead of a boring SCNCLR command.

```
1 REM CLOSE & OPEN
10 SCNCLR:PRINTCHR$(27)CHR$
   (82)
20 CHAR ,26,10,"HERE IS THE
   [CTRL 9] F I R S T[CTRL 0]
   SCREEN"
30 GETKEY K$:IF K$=" " THEN 30
40 R=86:L=6:DO
50 SYS 52684,R,35:SYS 52684,L,34
60 R=R-1:L=L+1
70 LOOP UNTIL R<L:SCNCLR:SLEEP1
80 CHAR ,25,10,"HERE IS THE [CTRL
   9] S E C O N D[CTRL 0] SCREEN"
90 DO:R=R+1:L=L-1
100 SYS 52684,R,35:SYS 52684,L,34
110 LOOP UNTIL L<6
```

Shaker

Shaker does for the 128's 80-column screen what many routines have done for the 40-column screen: It shakes it back and forth. This hack takes advantage of the VDC chip's little-known smooth-scrolling feature. By repeatedly moving the screen eight pixels to the left then eight to the right in increments of one, it produces this smooth effect. Try it and liven up dull text displays.

```
1 REM SHAKER
10 GRAPHIC 5,1:COLOR 5,5
20 FOR I=15 TO 64:CHAR ,I,6,"*":
   CHAR,I,16,"*":NEXT
30 CHAR ,36,8,"SHAKER"
40 CHAR ,31,10,"COMPUTE'S
   GAZETTE"
50 CHAR ,30,12,"324 WEST WEN
   DOVER AVENUE"
60 CHAR ,29,14,"GREENSBORO, NC
   27408"
70 FOR L=7 TO 0 STEP -1
80 SYS 52698,,25:RREG AC
90 SYS 52684,(AC AND 248)+L,25
100 NEXT L
110 FOR R=0 TO 7
120 SYS 52698,,25:RREG AC
130 SYS 52684,(AC AND 248)+R,25
140 NEXT R:GOTO 70
```

I hope you take a few minutes to type in these little gems and see what a 128 can do. Feel free to modify and embellish these programs and use them as a basis for your own programming ideas. ❑

Software Hut

Commodore AMIGA

Folcroft East Business Park 313 Henderson Dr Sharon Hill, PA 19079

Info 215-586-5701
FAX 215-586-5706

Orders 800-932-6442

Hours: Mon-Fri 9 to 6
Sat 9 to 5 - Eastern

We do not publish a catalog. Please call if you don't see what you want.

- *One of the East Coast's largest Authorized Commodore dealers*
- *Full line of Hardware, Software, & Peripherals... IN STOCK*

Games & Closeouts

Bridge 5.0	$22.00
Champions of Krynn	16.00
Curse of the Azure Bonds	14.00
Death Knights of Krynn	14.00
Dr. Doom's Revenge	12.00
Donald's Alphabet Chase	13.00
Duck Tales	22.00
Gateway to the Savage Frontier	14.00
Gettysburg, The Turning Point	42.00
Goofy's Railway Express	13.00
Linkword (Specify)	20.00
Mickey's Runaway Zoo	13.00
Microleague Baseball 2	28.00
Microleague Football	29.00
Neuromancer	17.00
Payday	7.00
Pirates	35.00
Pool of Radiance	14.00
Scrabble or Monopoly or Risk	17.00
Secrets of the Silver Blades	14.00
Sim City	22.00
Simpsons Arcade	29.00
Skate Wars	10.00
Star Control	29.00
Sticky Bear (Specify)	18.00
Strip Poker	22.00
Teenage Turtles Arcade	29.00
Test Drive 2 - The Duel	28.00
Tetris	13.00
Tony LaRussa Baseball	12.00
Typhoon of Steel	19.00
Ultima 6	48.00
Weird Dreams	11.00
Where (Specify) is Carmen S. Diego	27.00
Win, Lose or Draw	10.00
Xenophobe	10.00

128D REFURBS

Just in! These 128Ds have been completely refurbished by Commodore, and come with a full 90 Day Warranty.

$274.95

1541-2 DRIVE

Refurbished $79.00
Cabling & Power Supply Add $20.00
Individually tested by CBM, each drive is of consistent high quality!

MINDSCAPE
POWERPLAYERS JOYSTICK

- **Microswitches**
- **Steel shaft**
- **Ball-bearing pivot**
- **Pistol grip**
- **Large trigger**
- **Originally $34.95**

One for $8.95 - Two for $8.50 each

1802 MONITORS

These ***BRAND NEW*** composite Monitors are Boxed with Complete Cabling & Warranted by Commodore for 1 year.

$145.00

Hardware

5 or 8 pin Composite Cable	$8.95
1541-2 Disk Drive NEW	***164.95***
C128 to Magnavox/1084 (Specify)	14.95
Computer/Disk Drive Cable - 6 ft.	12.95
Disk Notcher	4.95
Ergo Joystick	18.00
G Wiz Printer Interface	40.00
IconTroller	15.95
JVC Disks 3.5" - 10 Pack	8.00
JVC Disks 5.25" - 10 Pack	7.00
Kraft Triple Trackball	40.00
SpeedKing Joystick	15.00
Super Snapshot ver 5	***56.00***
Super Graphix Jr	36.95
Surge Protector w/EMI (6 Outlet)	14.00

We carry a complete line of printer ribbons for Commodore 801, 802, 803, 1525, 1526 and for Panasonic 9 and 24 pin printers.

$7.00

Productivity

1541/1571 Drive Align by Free Spirit	$29.00
1541/1571 Physical Exam by Cardinal	29.00
1581 ToolKit	22.00
Algeblaster	22.00
Big Blue Reader 4	35.00
CAD 3D	29.00
Data Manager 2 (64)	16.00
Dome Home Accounting	44.00
Elementary Grade Builder	22.00
Family Tree 2.x	42.00
Fast Load	28.00
GEOS (64)	39.00
GEOS 128 V2	45.00
GEOS International Fonts	25.00
Jr. High Grade Builder	22.00
Maverick V5	29.00
Math Blaster or Spell It	22.00
Pocket Writer	22.00
Pocket Writer 2	42.00
Pocket Writer 3	49.00
Pocket Writer Dictionary	17.00
Pocket Filer 2	22.00
Pocket Planner 2	22.00
PrintShop	30.00
Print Shop Companion	27.00
Speed Reader	22.00
SwiftCalc (64)	16.00
Super 81 Utilities 64-128 (Specify)	14.00
WordWriter 6	29.95

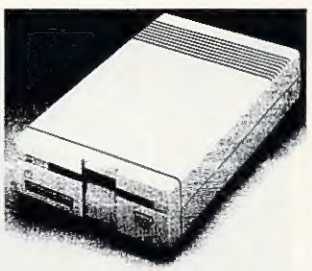
UPGRADING?

AMIGA

As the LARGEST Amiga Dealer in the United States, we carry a complete line of software, hardware, and Amiga computers, including the A600, A1200, A2000, A3000, & A4000.

Software Hut Delivers a Commodore EXCLUSIVE!

A BRAND NEW production run of 1581s, by Commodore, has been made available to Software Hut EXCLUSIVELY! These drives come with Commodore's full 90 Day Warranty. The drives are complete with all Cabling & Software. Call for special quantity pricing for Dealers and Users Groups.



Brand New 3.5" 1581 Floppy Drive

We offer these drives at a SUPER NEW Low Price of

$109.95

1571

1571 Drives have just arrived. These totally NEW units come with all cabling, a comprehensive manual, and a full 90 Day Warranty. Perfect for your 128 or 128D computer. Last year's batch lasted only 2 weeks! Order early as these will not last long at

$169.95

1750 RAM EXP

512K RAM for ALL 64 & 128 CBM unit w/disk, Manual, 90D Wty
$109.95

Chip Level Design Unit
$115.00

PARTS

Many of the following CBM parts are in SHORT SUPPLY. Except as noted, all are NEW & w/90 Day Warranty.

- 1541/1541C Internal Drive - *$85.00*
- 1571 Interanal Drive - *$85.00*
- 64 P. Supply - *$19.*
- C128 Keyboard - *36.00*
- C128D Keyboard - *$22.*
- C128D Int. P. Supply - *$38.*
- C128 Ext. P. Supply *Refurb - $34.*
- P. S. for 1541-2 1571-2 1581 - *$26.*

MICE & MODEMS

1351 Mouse NEW *$32.00*
CBM 1670 - 1200 Baud *$29.00*
Aprotek 2400 w/software *$89.95*
Supra 2400 - *$79.00*
Peak Modem Interface for non-CBM units - *$29.00*

CHIPS CHIPS CHIPS

IC 6526A	$9.00
IC 6567 40 COL IC VIC	14.00
IC 6581 R4	14.00
IC 7700-00 PLA 82S100	14.00
IC 8701 Clock Gen	5.00
IC 8701 FG	5.00
IC LSI 8564 VIC Rev 5CA	22.00
IC PLA 8721 R3	14.00
IC ROM 2332 Character	9.00
IC ROM 2364 Basic or Kernel	9.00

OUR POLICIES

No waiting for your orders to ship. Orders in by 2PM go out the same day. Second Day & Overnight shipping is available. International orders are shipped by Air Parcel Post or DHL. Domestic orders are shipped UPS.

- ***All orders are subject to credit card verification***
- ***All prices subject to change***

VISA MasterCard DISCOVER

Supplies are limited and on a first-come first-serve basis. We accept Visa, Master Card, and Discover. We also ship COD, accepting Cash, Certified Check, or Money Order. Software and accessories shipping is $4.00. Hardware shipping is $6.00 for small items. Call for larger items (ie. Monitors). COD add $5.00. Minimum COD order is $40.00. Canadian, APO, & International orders are welcome. We will bill only for actual shipping charges & insurance at time of order. 15% re-stocking fee on all returns not exchanged.

THE LOST WORLD

Does this sound familiar? You're tired of shoot-'em-up computer games that do nothing for a child's mind, but your child shuns educational games in favor of action and adventure. If that's the case, then Free Spirit Software has the answer for both of you in an adventure program it has just imported from the land down under.

Based on a Sir Arthur Conan Doyle novel published at the turn of the century, The Lost World is an offering from Satchel Software, a company that promotes the use of computers and computer text games in the school systems throughout South Australia. This particular program, geared toward junior high students, ties in nicely with the study of fossils, dinosaurs, and natural history. The game also helps students by encouraging reading, increasing vocabulary, widening thought processes, developing concentration, and developing problem-solving strategies.

With these goals in mind, you would probably expect The Lost World to be boring. It's not! The game is great fun! It combines text commands with colorful graphics in a rollicking adventure that can be played alone or solved as part of a group effort. You'll find yourself facing quite an array of obstacles and many unique elements. In fact, if you don't use your head, you might end up as a tasty tidbit for a hungry dinosaur.

The game's text commands are relatively simple. Directions (north, west, up, down, and so on) can be abbreviated by typing the first letter of the word. Youngsters who aren't familiar with a keyboard won't be put off by having to type in a lot of text. Action commands are kept simple, too, using verbs such as take, cut, drop, make, and so on. This lets children with varying reading levels play and enjoy the game. You can also save your adventure—a nice feature if you've just had an unfortunate encounter with an Allosaur.

Players will have fun finding treasure, discovering fossils, and meeting dinosaurs in The Lost World, but they'll also be learning.

Satchel wants its software to be challenging but not frustrating. The programmers have found that a little help goes a long way, so they've included a 176-page manual that is divided into two parts. The first 67 pages are for the teacher, and they offer suggestions and possible solutions for the game. This section should be used sparingly. Children in the appropriate age range should be able to solve the game with just a few hints to steer them in the right direction. The remaining 109 pages of the manual are a bonus. They are crammed with games, crossword puzzles, and short articles that will pique a child's curiosity and offer hours of related activities. There's lots of good information here for students who want to do some reports for extra credit, too.

The Lost World is far more than a text game and activity book. In fact, this three-disk package outdistances any game program I've ever seen. Satchel actually gives you an incredible resource disk in this program that includes a basic database, a simple-to-use word processor called Easy Word, and several disk utilities—all designed for junior high students.

The database is already set up with five files that pertain to dinosaurs, explorers, and imaginary lands. Users can choose to view, edit, search, or sort the files. They can also add, delete, and print records. While the users can't create their own files, this program helps familiarize them with ways to work within databases and access information.

The people at Satchel Software realize that novices often inadvertently erase portions of their projects while learning to use a new program. To reduce this frustrating possibility, some commands, such as DELETE, have safeguards built into them that prevent a user from erasing the entire database. Speaking from personal experience, there've been many times when I've wished for this feature on my own database.

The word processor included in The Lost World is called Easy Word, and it lives up to its name. It's a practical, 40-column word processor that is so simple to use that kids will enjoy using it for reports and other writing tasks. Satchel has built several safeguards into the word processor program, too. For example, function keys handle most commands within the program (LOAD, CLEAR, SAVE, ERASE, CENTER, PRINT), and these commands stay onscreen throughout the program. That way users won't lose any work trying to remember the right command.

Another safeguard built into the program restricts access to the Erase option. When you press f8 to erase files, you get a message informing you that Erase is a restricted option. You then have to go through several more steps, including entering the full name of the file you want to erase. After all these steps, it's rather unlikely that you'll accidentally delete an important file.

I think you're going to be very surprised when you check out The Lost World. This package delivers everything that its developers promise and more. I for one am going to be watching for more programs bearing the Satchel Software name.

MARTI PAULIN

Commodore 64 or 128—$39.95

SATCHEL SOFTWARE
Distributed by Free Spirit Software
720 Sycamore St.
Columbus, IN 47201
(812) 376-9964

Circle Reader Service Number 414

VIDEO DIGITIZER

There's a new digitizer on the market. This German import, called simply Video Digitizer, is distributed in the U.S. by RIO Computers. Video Digitizer does far more than simply replace ComputerEyes, which has been discontinued—it leaves it in the dust!

The screen image is limited to standard Commodore high-resolution, and you need a decent image and a stable source signal if you hope to accomplish anything. But this little gem manages to find a clean image where ComputerEyes could see only black-and-white streaks. I ran the cable from my new VCR (which I bought only after making sure it could produce a nearly perfect freeze-frame) to my Commodore 128 (in 64 mode) and compared Video Digitizer with ComputerEyes on the same video image. Video Digitizer created a perfectly presentable picture of Captain Kirk, while ComputerEyes covered the poor captain with streaks. I was impressed!

Of course, any digitized image needs at least a little touching up. With ComputerEyes this always meant loading a third-party conversion program such as Icon Factory and porting the image over to Doodle or geoPaint for touch-ups. Video Digitizer, on the other hand, has a drawing program called Eddison that's built right in! As a matter of fact, Eddison is a full-featured, sophisticated graphics tool in its own right. In a sense, Video Digitizer is an add-on accessory to Eddison, not the other way around. After all, the digitizer is controlled from an icon that's a selection on Eddison's menu, just like the pencil or brush choices.

Video Digitizer scans an image in black-and-white or gray scale. The former requires less time to scan and is considerably more forgiving about the picture you choose. It creates a very recognizable scan of a person's face, for example, even if it's not a close-up. The downside is that the image is comprised of areas that are strictly black or white. There are no shades of gray between the two. You can manually set the threshold level (the darkness level at which the software chooses to make a pixel black instead of white), which greatly alters the look of the scan. This high-contrast image is great for many things. If you plan to resize the image later in a program like geoPublish, it's the only way to go.

Gray-scale scans are created with a myriad of dots in patterns to create levels of gray. This sounds great in theory, but when it comes down to practice, there usually isn't sufficient contrast in the source image to give you a

well-defined image. This isn't a flaw in the Video Digitizer, however; ComputerEyes does the same thing. A live video image from a video camera with good lighting to create sharp contrast can be scanned successfully this way. A video tape image, on the other hand, will usually wash out. Fortunately, it's a simple matter in Eddison to scan and rescan using various methods until you come up with an image you like. This is perhaps the single most impressive feature of the Video Digitizer system. You not only have the choice of repeating your scan quickly and easily, but you can also immediately use Eddison to alter each image to see if it fits the bill. This little unit, barely the size of a standard cartridge, is a joy to work with.

This is not to say that everything is perfect. Probably the biggest drawback to Video Digitizer is its price. The competition (now limited to VideoByte and its companion cartridge, VideoMate) retails for less than half Video Digitizer's price. Granted, Video Digitizer runs circles around VideoByte's converted multicolor images if you want a high-resolution scan and can justify the price.

Another major disappointment is Video Digitizer's documentation. Like VideoFox and other products in this line, the manuals for Digitizer and Eddison are poor translations of the original German, with sentences running from comical to unintelligible. A number of features are so poorly described that I was forced to give up when I couldn't figure them out by trial and error. Richard Ollins, president of RIO Computers, assures me that a new comprehensive set of manuals are available. With the new documentation, all the products in this line, including VideoFox, PageFox, and the new genlock that RIO plans to release, will be integrated into an impressive set of graphics and video tools for the 64.

If you're a "Star Trek" and a graphics fan like me and dream of using great pictures on your computer of Spock, the *Enterprise*, and maybe your brother, then Video Digitizer is worth the cost. I know I'll gladly earmark $250 for one when my strained budget allows. I've never been so sad as when I had to pack up this review unit of Video Digitizer and ship it back to the company.

STEVE VANDER ARK

Commodore 64 or 128—$249

RIO COMPUTERS
3430 E. Tropicana Ave., Ste. 65
Las Vegas, NV 89121
(702) 454-0335

Circle Reader Service Number 415 ❐

FEEDBACK

The 64's missing memory, loading the first file in a directory, and more

Why Only 39K?

If the 64 has 64K of memory, how come only 39K is available to BASIC programming?

A. MARE
SAN ANTONIO, TX

At the heart of all personal computers is the microprocessor. In the 64, this processor is the 6510, a variation of the 6502 used in the Apple. One characteristic of this chip is that it can access only 64K of memory at a time.

If the designers of the 64 simply filled the computer with 64K of empty memory, the machine would be helpless. Computers need certain built-in programs to function. Don't confuse these built-in programs with the programs you write yourself or load from disk. The built-in programs are stored in special memory chips called ROM (Read Only Memory). Unlike other programs, they remain safe in their ROM memory chips even when power is turned off. The BASIC language itself is one of these programs. Another program is the computer's operating system, which performs housekeeping duties.

These internal programs need some of the 64K of space that the 6510 microprocessor chip can address. To make room for them, the 64 designers used some of the RAM area. That's why, when programming in BASIC, you have only 38,911 bytes of memory. The 25K of RAM that appear to be missing are taken up by the BASIC interpreter program, the operating system, and other things. The designers did make it possible, however, to switch off the 64's ROM, thus freeing the full 64K of RAM underneath.

Unfortunately, when you switch off BASIC and the operating system, you are left with what amounts to an unconscious computer. You must replace the operating system software with some of your own that takes care of the necessary housekeeping tasks. Since BASIC is gone, this replacement must be machine language. Therefore, the full 64K of RAM is available, but only to advanced programmers fluent in machine language.

Load Me First

I have a question about making a menu program the first one on a directory. I have a disk of files, and no matter how often I save the menu program, it doesn't come first in the listing. I want to be able to load the menu by simply typing *LOAD"*",8* and *RUN*. How do I do that?

T. AIGHT
BOSTON, MA

You can load the first program on a directory listing with LOAD"",8 only if you're loading a program for the first time. After that, that command will load the last program loaded. To load the first program any time, use LOAD "0:*",8. Now let's address the real question.*

To make any program the first program on the disk requires that the program that is currently first be replaced. Here's how to do it.

Place the disk you want to rearrange into the disk drive. Load and list the directory. Note the program at the top of the directory list. (Press Run/Stop if necessary to keep the list from scrolling off the screen.) This first program on the disk is the one you'll have to move to make room for your menu program.

Use the COPY command to duplicate the first program with another name, for example, OPEN 15,8,15, "COPY0: newname=oldname*": CLOSE 15.* Newname *is the name of the copy, and* oldname *is the name of the original.*

Once you've copied the program elsewhere on the disk, you can safely delete the original with the SCRATCH command: OPEN 15,8,15, "S0: oldname*": CLOSE 15. As before,* oldname *is the name of the original. Now you can use the RENAME command to change the new name back to the original. Its format is OPEN 15,8,15, "R0:* oldname =newname*": CLOSE 15. Now you have two possibilities. If your menu program is already on the disk, you can use the COPY command, which will move the menu to the front of the disk. If the menu program isn't already on the disk, insert a disk that contains the program and load it in. Replace that disk with the one you've been working with and save the menu program to it.*

If you've done everything right, your menu program should now be the first program on the disk. Test it by by entering LOAD"0:",8.*

Underlining

I have a 64, Cardco + G interface, Star SG10 printer, and WordPro 3 Plus word processor. I can't get my system to underline anything. All four products claim to support underlining, so please tell me what to do in this regard. I'm not that knowledgeable about BASIC programming or the equipment. My main desire is to use the equipment for word processing.

PENNY CAESAR
FORT WAYNE, IN

One of the most common questions we receive is how to get a certain printer to work with a certain interface and a certain software package. Although the questions are similar, there are literally hundreds of answers, depending on what hardware and soft-

Underlining from within a word processor, and making files impossible to load

ware are being used. Given three interfaces, five printers, and seven word processors, you could find 105 different ways to hook them up and possibly 105 different answers to the question of underlining. Unfortunately, we don't have access to much of the equipment or software mentioned, so it's difficult for us to find the specific answer.

There are three routes you could take: First, write or call the manufacturers of your printer, interface, and software. The worst that could happen is that you'd get no reply. If you bought your printer at a local Commodore dealer (very few of those, these days), someone at the store may be able to help.

Second, try contacting a local user group. You may find someone there who has similar equipment. Or perhaps someone with printer experience can help you find the answer through experimenting.

Third, check your printer manual for the codes that turn underlining on and off. In your case, the way to enable underlining should be ESC – 1, which means send an escape (ESC) character, a minus sign, and a CHR$(1). To disable underlining, you'd send ESC – 0. This is often called an escape sequence because you use the ESC character followed by a sequence of one or more other characters. The ASCII value of ESC is 27, and the ASCII value of the minus sign is 45. To test this, enter and run the following BASIC program.

```
10 OPEN 4,4
20 PRINT#4,CHR$(27); CHR$(45);
   CHR$(1);
30 PRINT$4,"THIS SHOULD BE
   UNDERLINED.";
40 PRINT#4,CHR$(27); CHR$(45);
   CHR$(0);
50 PRINT$4," AND THIS IS NOT."
60 PRINT#4:CLOSE 4
```

If the printer underlined the first sentence, but not the second, you have the proper codes.

If it didn't work, there could be several things wrong. Here's where it becomes a little complicated. You may have mistyped the program; check the spelling and punctuation. The interface may have intercepted the codes before they reached the printer. Escape sequences can sometimes be used to program interfaces as well as to set printer options. If you have such an interface, it may have seen the ESC and thought that it was intended to be an interface code and not a printer code. If that's the case, you'll have to send ESC twice. Usually, when a programmable interface receives two ESC codes, it sends the second one to the printer.

Another potential problem is that the DIP switches on the printer or interface might be in the wrong position. DIP switches control the way your interface or printer acts. These vary widely from brand to brand, so check the printer manual and the interface manual for guidance on the proper settings.

Finally, it may be that your printer needs a different escape sequence for underlining; again, it's necessary to check your manual.

Before you start experimenting with the word processor, try to underline from a BASIC program. When you know the proper escape sequence from BASIC, you'll be prepared to try it from the word processor.

Some word processors use printer files to keep track of various printer settings. When you place a generic underline command in the document, the proper escape sequence is sent when you print the document, providing you've previously loaded the correct printer file.

Other word processors require you to know the commands to enable or disable underlining and other features. You'll have to check the documentation for your word processor. If you need to use the commands, you'll have to define three format keys for CHR$(27), CHR$(0), and CHR$(1). Then, whenever you want to underline, place the defined key for 27, –, and the key for 1 in the text of your document.

Unloadable Files

I remember seeing a programming tip somewhere that protected files from being loaded by unauthorized people. It required a code to load. Do you know of such a way that I can protect my programs?

RAY FISHER
PALATKA, FL

There are a number of ways a programmer can "lock" his or her BASIC programs from prying eyes, but here's a simple way that won't force you to remember any exotic codes. Just remember the character string (CHR$) for a comma, which is CHR$(34).

*When you are ready to save your program, use this format: SAVE "*filename*"+CHR$(34),8. Use your own program name for* filename. *When you list the disk's directory, this program will appear normal. Try to load it, however, and you'll get a FILE NOT FOUND error. To load the program, you'll have to enter* LOAD"filename"+CHR$(34),8.

Send your questions and comments to Gazette Feedback, COMPUTE Publications, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408. ❐

PUMP UP YOUR PRODUCTIVITY!

Harness the potential of your 64 and 128 with these powerful programs.

Get more work out of your 64 and 128 with these two new disk products from COMPUTE's Gazette – the 1992 Best of Gazette Utilities, and the Gazette Graphics Grab Bag!

The 1992 Best of Gazette Utilities

Seize control of your operating system and your world!

Here's what's on it–MetaBASIC 64, MetaBASIC 128, Quick, Sprint II, Ultrafont+, RAMDisk 64, RAMDisk 128, BASSEM, SciCalc 64, List Formatter, MegaSqueeze.

The Gazette Graphics Grab Bag

Do it all with Commodore graphics!

Here's what's on it–Starburst Graphics, Screen Designer 128, 128 Graphics Compactor, 64 Animator, VDC Graphics, Dissolve 128, Super Slideshow, 128 Animator, 1526 PrintScreen, Supratechnic, Medium-Resolution Graphics, Screen Maker, GAS!64–Special Edition, GAS!128–Special Edition.

ORDER THEM TODAY!

Extend Your Computer Power With This Powerful Software!

YES! I want to pump up my productivity! Please send me the disks checked below at $11.95 each.

___The 1992 Best of Gazette Utilities

___ The Gazette Graphics Grab Bag

___ Subtotal

___ Sales Tax (Residents of NC and NY please add appropriate sales tax for your area. Canadian orders, add 7% goods and services tax.)

___ Shipping and Handling ($2.00 U.S. and Canada, $3.00 surface mail, $5.00 airmail per disk.)

___ **Total Enclosed**

MasterCard and VISA accepted on orders with subtotal over $20.

___ Check or Money Order ___ MasterCard ___ VISA

Credit Card No. _____ Exp. Date _____

Signature _____
(Required)

Daytime Telephone No. _____

Name _____

Address _____

City _____

State/Province _____ ZIP/Postal Code _____

Mail this coupon to COMPUTE's 1991 Utilities, 324 West Wendover Ave., Ste. 200, Greensboro, NC 27408.

BEGINNER BASIC

Larry Cotton

NONREPEATING NUMBERS RETURN

Readers respond with their own programs to generate 52 nonrepeating random numbers.

Last November, I challenged you to submit programs that would generate nonrepeating random numbers in either BASIC or machine language. Thank you for your responses. Appropriately, all of the responses were in BASIC. (I'd hoped for a few ML versions, but I suppose mine couldn't be improved. In my dreams!)

The object was to compare BASIC and ML speed by generating 52 nonrepeating numbers, such as you get when shuffling a deck of cards.

And the winner is Howard Monroe. (I'm sorry, Howard. I misplaced your envelope, so I don't know where you live.) Here's Howard's program.

```
5 PRINT"(CLR)(DOWN)PRESS
  ANY KEY TO RANDOMIZE 52
  NUMBERS"
6 PRINT"WITHOUT
  REPEATS.":PRINT
7 GETA$:IFA$="" "THEN7
10 C=52:Q=RND(-TI/101)
20 DIM R(C),X(C)
25 REM INITIALIZE THE DECK
30 FORI=1TOC:X(I)=I:NEXTI
35 REM GENERATE 52 RANDOM
  NUMBERS BETWEEN 1 AND 52
  INCLUSIVE
40 FORI=1TOC:R(I)=INT(C*RND(1))
  +1:NEXTI
45 REM GENERATE THE RANDOM
  PERMUTATION
50 FORI=1TOC:A=R(I):B=X(I):
  X(I)=X(A):X(A)=B:NEXTI
60 FORI=1TOC
70 PRINT X(I),
80 NEXTI
90 PRINT:PRINT:PRINT"AGAIN?
  (Y=YES, N=NO)":PRINT
100 GETA$:IFA$<>"Y"THENIF
  A$<>"N"THEN100
110 IFA$="N"THENEND
120 GOTO40
```

Howard points out that the important lines are 10–50 and that in reshuffling the "deck," it's important to branch back to line 40, not line 30.

His program is unique in that the time it takes to generate the 52 numbers is virtually the same each time the program runs. This is not so of my primitive version or of any of the other submissions that rely on IF-THEN statements to test the random numbers.

Howard's program requires only 118 jiffies (I'll explain this term in a moment) to generate the 52 numbers. To time how long a routine takes, add a line like this where you want the timer to start.

```
1 TI$="000000"
```

A clock, based on the 64's CPU clock, measures time in jiffies (about 1/60 second). The jiffy clock can be set to 0 with a line like that above. From then on, that clock runs frantically until it's reset.

If you'd like to see the jiffy clock in action, just type *PRINT TI* in immediate (nonprogram) mode and press Return. Do this a few times and watch as the jiffies fly by. In program mode, you would add a line number to the command, such as *55 PRINT TI*.

The clock is set to 0 in line 1, and line 55 prints the number of jiffies that have elapsed since then. Everything that happens between line 1 and line 55 is timed in jiffies.

But I digress. My November program often took 4700 or more jiffies to generate the 52 numbers. Howard's speed-enhanced version blew mine away by a factor of almost 40. All other programs submitted beat mine by factors of 2–15. My next challenge was to see if I could squeeze any more speed out of Howard's program. Yes, I could—but not much. First, I changed his random statement in line 40.

```
40 FORI=1TOC:R(I)=INT
  (C*RND(.))+1: NEXT
```

The number inside the parentheses that follows RND can be anything, so I made it 0. Actually, by substituting a period for the numeral 0, it works even faster. This shaved 18 jiffies off Howard's time. Then I combined the last two FOR-NEXT loops into one and dropped the I variable. This eliminated a calculation and shaved another 14 jiffies.

I tried using DEF FN to define the RND function, but that actually slowed down the program. I also tried making the 1 in line 40 a constant; that didn't help either. So here's the speed-enhanced version. It times everything up to the actual printing of the numbers and also rudely strips away the program's user-friendliness.

```
10 TI$="000000"
20 PRINTCHR$(147)
30 C=52:Q=RND(-TI/101)
40 DIMR(C),X(C)
50 FORI=1TOC:X(I)=I:NEXT
60 FORI=1TOC:A=INT(C*
  RND(.))+1:B=X(I):X(I)=X(A);
  X(A)=B:NEXT
70 PRINTTI:PRINT
80 FORI=1TOC
90 PRINTX(I),
100 NEXT
```

An elegant (albeit slower) version by David Zammat of Summit, New Jersey, tests whether each new integer generated has been used previously. If it hasn't, the program prints it and generates another one. I received several variants of this program

```
10 D=52:DIM U(D)
20 FOR C=1 TO D
30 N=INT(D*RND(1))+1
40IFU(N)=0THENU(N)=1:PRINTN;:
  NEXT:END
50 GOTO 30
```

Again, thanks, Howard, David, and all the others for your input. I'll try to offer another challenge before long. ❐

MACHINE LANGUAGE

Jim Butterfield

ARRAYS

Create an array in BASIC and modify it with machine language.

If you create an array in BASIC, that array can be used or modified by machine language programs. The start-of-arrays pointer tells where to find the first array. Using the pointer, the ML program looks for the array it wants and then goes for the data.

The first two bytes of an array give its name. The next two bytes give the array's size, which enables a program to leap ahead to the next array if desired. The fifth byte contains the number of dimensions. Following that, each dimension has a two-byte number that shows its size (including the 0 element). Thus, a one-dimensional array has seven bytes of header material, after which the data is stored. Integer arrays—which we'll use in our sample program—store each value in two bytes.

The start-of-arrays pointer may be found at addresses $2F and $30 on the VIC-20, Plus-4, and 64. The 128 sites the pointer at $31 and $32—but be careful, the values themselves are stored in bank 1.

Let's take a look at a sample program that uses BASIC to set up an integer array. We'll then ask a machine language program to calculate a Fibonacci series and place it in that array. A Fibonacci series starts with values 1 and 1 (or 0 and 1, if you like). Each new value is the sum of the two previous values. The series goes 1, 1, 2, 3, 5, 8, 13, and so on.

Our BASIC program creates only one array. To keep our program small, we won't check the array name, but we will check its size, since the calculated values should not overrun the array area.

The code starts by copying the start-of-arrays pointer into a work pointer at $22 and $23.

```
033C: LDA $2F: STA $22: LDA $30
:STA $23
```

Next, we extract the size of the array and add it to the array address. That gives us the address of the next array, the end of this one. We'll store it at $03C0/1.

```
LDY #$02: CLC: LDA ($22),Y: ADC
$22: STA $03C0
INY: LDA ($22),Y: ADC $23: STA
$03C1
```

Skip seven bytes to get past the header data.

```
LDA $22: ADC #$07: STA $22: BCC
$0360: INC $23
```

The array initially contains all 0s. Reaching beyond element 0, we'll change element 1 to a value of 1.

```
0360: LDY #$03: LDA #$01: STA
($22),Y
```

Now the program proceeds to set up a second pointer, with a value of the first pointer plus 2. These two pointers will reference adjacent values in the array.

```
CLC: LDA $22: LDY $23
ADC #$02: STA $24: BCC $0372:
INY
0372: STY $25
```

Here comes our main loop in the program. We add together the two pointed-at values and put the result temporarily on the stack.

```
0374: LDY #$01: CLC: LDA
($22),Y: ADC ($24),Y: PHA
DEY: LDA ($22),Y: ADC ($24),Y:
PHA
```

Now we bump the pointers to the next set of values.

```
CLC: LDA $24: LDY $25: STA $22:
STY $23
ADC #$02: BCC $0390: INY
0390: STA $24: STY $25
```

If we have moved beyond the array's end, we'll exit.

```
CMP $03C0: TYA: SBC $03C1: BCS
$03A9
```

Otherwise, we bring back the value from the stack and store it in the new array location. Then we loop back.

```
LDY #$00: PLA: STA ($24),Y
INY: PLA: STA ($24),Y
CLC: BCC $0374
```

If it's time to exit, we must remember to remove the two unwanted bytes from the stack.

```
03A9: PLA: PLA: RTS
```

You may change the size of the array as defined by the DIM statement in the BASIC program, but integer arrays can't hold a value greater than 32,767.

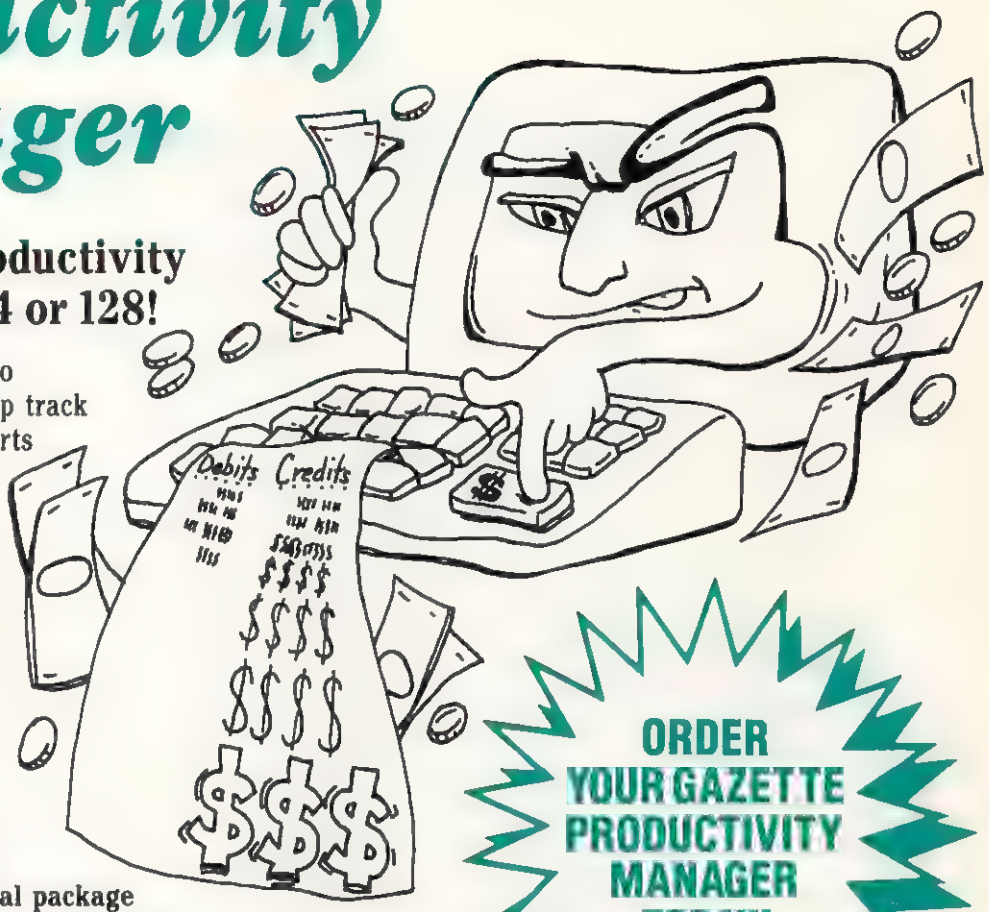
Don't forget that BASIC can move arrays to a new location to make room for new variables. Always work from the start-of-arrays pointer; its value may have changed since you last used it.

```
EG 80 PRINT "(64 ONLY) M/L ARR
      AY ACCESS"
XK 90 DIM A%(20)
EM 100 DATA 165,47,133,34,165,
      48,133,35,160,2,24,177,
      34,101,34
HE 110 DATA 141,192,3,200,177,
      34,101,35,141,193,3,165
      ,34,105,7
AA 120 DATA 133,34,144,2,230,3
      5,160,3,169,1,145,34,24
      ,165,34
XC 130 DATA 164,35,105,2,133,3
      6,144,1,200,132,37,160,
      1,24
CE 140 DATA 177,34,113,36,72,1
      36,177,34,113,36,72,24,
      165,36,164,37
SH 150 DATA 133,34,132,35,105,
      2,144,1,200,133,36,132,
      37
DK 160 DATA 205,192,3,152,237,
      193,3,176,12,160,0,104,
      145,36,200,104
MJ 170 DATA 145,36,24,144,203,
      104,104,96
PH 200 FOR J=828 TO 939
EJ 210 READ X:T=T+X
BP 220 POKE J,X
AE 230 NEXT J
PC 240 IF T<>10638 THEN STOP
GF 300 SYS 828
SB 310 FOR J=1 TO 20
MS 320 PRINT A%(J);
PP 330 NEXT J
```

The Gazette Productivity Manager

Harness the productivity power of your 64 or 128!

Turn your Commodore into a powerful workhorse, keep track of finances, generate reports in a snap, manage your money in minutes—all with the *Gazette Productivity Manager!* Look at all your 64/128 *Productivity Manager* disk contains.



GemCalc 64 & 128—A complete, powerful, user-friendly spreadsheet with **all the features you'd expect in an expensive commercial package** (separate 64 and 128 versions are included). Most commands can be performed with a *single* keypress!

Memo Card—Unleashes the power of a full-blown database without the fuss! **Nothing's easier**—it's a truly simple computerized address file. Just type in your data on any one of the index cards. Need to edit? Just use the standard Commodore editing keys. Finished? Just save the data to floppy. What could be easier?

Financial Planner—Answers all of those questions concerning interest, investments, and money management that financial analysts charge big bucks for! You can plan for your children's education and know exactly how much it will cost and how much you need to save every month to reach your goal. Or, decide whether to buy or lease a new car. Use the compound interest and savings function to arrive at accurate estimates of how your money will work for you. **Compute the answer at the click of a key!**

DON'T MISS OUT ON THIS POWERFUL WORKHORSE!

ORDER YOUR GAZETTE PRODUCTIVITY MANAGER TODAY!

(MasterCard and Visa accepted on orders with subtotal over $20).

☐ **YES!** Please send me ___ *Productivity Manager* disk(s) ($14.95 each).

_____ Subtotal

_____ Sales Tax (Residents of NC and NY please add appropriate sales tax for your area. Canadian orders, add 7% goods and services tax.)

_____ Shipping and Handling ($2.00 U.S. and Canada, $3.00 surface mail, $5.00 airmail per disk.)

_____ Total Enclosed

__ Check or Money Order __ MasterCard __ VISA

Credit Card No. _____

Signature _____ (Required)

Daytime Telephone No. _____

Name _____

Address _____

City _____

State/Province _____ ZIP/Postal Code _____

Send your order to Gazette Productivity Manager, 324 W. Wendover Ave., Ste. 200, Greensboro, NC 27408.

PROGRAMMER'S PAGE

Randy Thompson

TRIVIAL PURSUITS

Believe it or not, the 64 is over ten years old, the 128 a mature eight. That's almost a lifetime in computer years. During this tenure, hackers have found many quirks in these machines. Here are a few.

Hackers have discovered many quirks in the 64—here are a few.

READY or Not

In the old days, it was thought that the *OUT OF DATA* message that spuriously appears when you're editing a program was caused by some bug in the computer's operating system. Not so.

As many alert programmers might observe, this message occurs when you press Return while the cursor is on the same line as the *READY* prompt. Notice that the word *READY* can be interpreted as the BASIC statement *READY*, which is exactly what the computer tries to do: read information from a data statement. If your program doesn't have any data statements, the computer replies with the cryptic but correct *OUT OF DATA ERROR*. If your program does contain data, your computer reads the value into Y and then responds with a more reasonable sounding *SYNTAX ERROR*, because the period (.) that follows *READY* is not a valid BASIC command.

Another common typing mistake is to type the command *RUN* on top of the *READY* prompt. This produces the word *RUNDY*, which your computer rarely accepts. In most cases, your computer complains with an *UNDEF'D STATEMENT* error because it cannot interpret the characters DY as a valid line number. However, whenever the BASIC interpreter is unable to find a line number, it tries to jump to line number 0. So by starting all your programs with line 0, you can enter *RUNDY*—or *RUNIT*, or *RUNNY*, or even *RUNAWAY*—to start your program. For the same reason, you can enter *GOTO* without a line number or even *GOTOJAIL* or *GOTOGO* to start a program from line 0.

How Old Is Your 64?

If you own an early-model 64, then you've probably encountered the infamous lockup bug. To see if you're one of these lucky people (I am!), turn on your computer, move the cursor down to the last line, and hold down the space bar until the cursor passes the right edge of the screen twice. Next, hold down the Del key until the cursor moves back to the far right column. If you own a 64 with the lockup bug, the words *LOAD*, *?SYNTAX ERROR*, *READY*, and *RUN* appear on the screen, and the computer locks up. If you have a program in memory, it runs. The cursor continues to flash, but the computer ignores your keypresses, even Run/Stop–Restore.

If you own a Datasette (remember them?), you can defeat the bug by simultaneously pressing the left Shift key and 3, or X and 5, or V and 7, and so on (every other key from left to right on the top and bottom rows). The screen will display *PRESS PLAY ON TAPE*. Press Play on the Datasette and then Run/Stop. Your computer will return to normal. Disk drive owners can avoid the bug if the first line in the program in memory is OPEN 15,8,15:INPUT#15,S$:CLOSE 15.

Interestingly, the lockup bug will occur only when your cursor color is red, cyan, blue, yellow, light red, dark gray, light blue, or light gray. Safe colors are black, white, purple, green, orange, brown, medium gray, and light green.

Beyond the Call of Duty

Although the widest Commodore screen is 80 characters, the number used in a TAB( ) function can be as large as 255. For example, on a 40-column screen you can use PRINT TAB(240) to move the cursor down six lines. Any number greater than 255 results in an *ILLEGAL QUANTITY* error.

As with the TAB( ) function, the ON *X* GOTO statement can't handle numbers greater than 255. This is contrary to early Commodore documentation, which states that negative numbers and numbers greater than 255 cause the 64 to fall through to the next program line. Negative numbers and numbers greater than 255 both produce *ILLEGAL QUANTITY* errors.

Keyboard Confusion

Plug a joystick into port 1, turn on your computer, play with the joystick, and watch your computer screen fill with seemingly random characters. This happens because the joystick port shares the same CIA chip and I/O lines as the keyboard.

If you're short a joystick and need to emulate one, try these alternatives. For example, just as pressing the joystick's fire button produces a space, hitting the space bar makes a program think that you've pressed the joystick's fire button. To simulate joystick up, press 1; joystick down, press back arrow (←); joystick left, press Ctrl; and joystick right, press 2.

Send your programming tips (or trivia) to Programmer's Page, COMPUTE's Gazette, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408. We pay $25–$50 for each tip that we publish in this column. ❐

GEOS

Steve Vander Ark

MORE GREAT GEOS GOODIES

Last month I talked about some of the latest GEOS shareware/public domain programs on QuantumLink, but Q-Link isn't the only place to find GEOS applications. You can find some great programs in other GEOS collections.

These are disks that contain a number of GEOS files. You'll find a wide variety of goodies, from utilities and applications to clip art files and fonts. Some of the programs and fonts I use most often came on a collection disk. Here are some sources.

Try these disk collections for some great GEOS finds.

GEOS Powerpack

GEOS Powerpack, Powerpack II, and GEOS Companion are collections released by *RUN* magazine. The disks are still available, and they're probably the best GEOS collections ever released. You'll want to get all three.

Powerpack includes such gems as Write Hand Man, a writing and pattern analyzer, and PaintView II, a geoPaint viewer which includes an option to save part of a geoPaint file as a Doodle file. There's a card file database as well as Thumbnail, a utility that creates miniature versions of geoPaint documents to be used as clip art or printed by the page full for reference copies of your collection. There's also a version of Breakout for GEOS. The fonts, particularly those by Susan Lamb, and the scanned, high-quality clip art images are spectacular.

Powerpack II contains the updated version of geoTerm for the 64 and 128, games, and a simple word processor, which imports and converts text files in ASCII. An interesting utility on the disk creates a stand-alone documentation file from a geoWrite document. Unfortunately, all the documentation for the programs is included in that form on the disk itself, leaving little space for clip art files and fonts.

The GEOS Companion disk boasts a music editor and an animation program, as well as a 1581 boot disk creator and other excellent utilities, including a batch loader. There are more fonts, including one of my personal favorites, Smith Corona (regular and megafont versions), and still more clip art files. Thankfully, this disk has a separate manual instead of on-disk documentation files.

As of this writing, these disks are available from Tech Media, Special Products, P.O. Box 2151, Salisbury, Maryland 21802. You can order by calling (800) 824-5499. The two Powerpack disks are $19.97 each; GEOS Companion is $24.97. Add $3.95 shipping and handling to each order, not each disk.

Collette Utilities

Jim Collette is so well known and admired for his GEOS programming that Creative Micro Designs figures that his name on a disk is enough to get folks to buy this collection of his better utilities. This is CMD's most recent GEOS release. Included are updated versions of some of the finest GEOS programs ever written: geoWizard and the premier font creator, Font Edit. Also included are Mini-Desk, one of the more useful desk accessories; a utility which automatically places the current photo scrap into the first photo album on the disk; a set of PostScript utilities for laser printing; and a DOS wedge. As always, Jim's programs are user-friendly and user-proof. The $34.95 price is ridiculously inexpensive for these quality programs. You can order the collection from Creative Micro Designs, P.O. Box 646, East Longmeadow, Massachusetts 01028. To place an order call (800) 638-3263.

GeoPowerTools

The folks who publish Lodestar, the magazine on a disk for the Commodore, offer a collection of utilities by Scott Resh, a talented GEOS programmer. PowerTools includes several photo albums of excellent clip art and a number of fonts. For sheer number of files for your dollar, you can't beat this disk of 21 utility programs and games. You'll find BASIC 8-to-Amiga format conversion programs, directory and sector editors, a program which prints the contents of the Date Book that your GEOS Calendar uses, a nifty desk accessory that turns any part of the visible screen into a photo scrap, a fast formatter, a SID music player, and a utility that prints multiple copies of a geoPaint document. You can get this collection from Softdisk, P.O. Box 30008, Shreveport, Louisiana 71130. Or order by calling (800) 831-2694. The price is $9.95 plus $4.50 shipping and handling.

COMPUTE's GEOS Collection

COMPUTE also has a collection of GEOS programs, selected from the Gazette section of COMPUTE magazine. Included on this disk are a couple of nifty games—I like Skeet, myself—and utilities such as a word counter for geoWrite, a screen dumper, and a help file creator. The disk also includes a nice multidensity printer driver for Epson-compatible printers. This collection is available for $13.95 (shipping and handling included) from COMPUTE, 324 West Wendover Avenue, Suite 200, Greensboro, North Carolina 27408. ❐

D'IVERSIONS

Fred D'Ignazio

WHEN TELEVISION GOES DIGITAL

I opened the *Wall Street Journal* today, and a story jumped out at me. It described how TCI—Tele-Communications Incorporated—may soon offer 500 TV channels to its 11 million cable viewers. TCI is the nation's largest cable TV operator. It's now allying itself with two other cross-industry heavyweights, AT & T and General Instrument Corporation, to upgrade its analog cable system to a digital network. This new network, according to the *Journal,* will "open the gates for a vast sea of entertainment and information options for cable subscribers."

Imagine the TV-zapping cowboy in your family with a remote that accesses 1500 channels.

The conversion to digital is possible because of advances in video and sound compression. Complex mathematical algorithms hard-wired into special-purpose (digital signal processor) computers in TV programmers' studios will shrink a TV signal to one-tenth of its normal size. This will allow cable to carry ten times today's 50 or 60 channels. All this is possible without converting the standard coaxial cable to the more expensive fiber optic cable. As fiber optic cables replace coax, however, we'll see another threefold jump in cable capacity.

Backyard satellite-dish owners will see 30 to 50 new digital TV channels on their TV sets by next summer. The rest of us will come online sometime in 1994. Once the system is up and running, it's expected to carry 500 cable channels simultaneously. And if your local cable uses fiber optics, you might be seeing up to 1500 channels!

Does this sound like overchoice, or what? Can you imagine the TV-zapping cowboy in your family with a remote that accesses 1500 channels? It boggles the imagination! The average zap lasts about 1.4 seconds, so just completing a circuit around the cable racetrack would take a mind-numbing 30 minutes. That's 30 minutes filled with random, nonsequential video and sound bites, brought to you compliments of your zippy zapper. Most TV shows would be over by then, so if your brain isn't reduced to sludge, you can blast off into another button-pushing blitzkrieg that will last another half an hour.

Fortunately, the move to digital cable will result not only in more channels but also in a fundamental redefinition of the word TV. Fifteen hundred passive couch-potato channels offering video pabulum may sound like torture, but that's only if your resident zapper forces you to sit through an orgy of channel-hopping.

A more sane way to approach this new deluge of programs is to think of TV in a new way—not as disjointed programs, but as a multimedia library or bookstore. The key to this way of thinking will be offered to you at the time your TV is hooked up. It'll be in the form of a tiny black box that according to one cable-industry official "will be the most powerful piece of electronics technology" in your home. This box will feature the most advanced multimedia computer chips and programs on the market. The box won't look like a computer, so no one will panic. But it'll turn your TV into a computer monitor and your zapper into a tiny computer keyboard.

The transformation of TV to computer will happen just in time. Instead of randomly hopping through a confusing flood of 1500 programs, you'll have brightly colored computer menus that will allow you to pluck a single program out of the vast sea of choices. The menus will gently guide you (much like a good librarian) into narrowing your selections. Do you want entertainment or news? Do you want a first-run sitcom or an oldie but goody? Do you want nature, history, mystery, or trash? The choice is up to you.

Don't think of TV any more as TV. Think of it as your own private bookstore that houses thousands of titles. (At 1500 titles an hour, your "bookstore" will offer 6000 titles in a single evening of browsing.) Your bookstore isn't a standard print emporium but an electronic multimedia bookstore with books that spring to life when you open them; books that spill over with real people; and books that talk to you, play music, and captivate you with lifelike scenes of drama, suspense, murder, and intrigue. You can browse through all these myriad titles by casually pointing and clicking on menu buttons that group the books into topics such as fiction, nonfiction, biography, current events, animals, horror, and science fiction.

And, remember, this is not a commercial bookstore that you are visiting. It is *your* bookstore, so you can pick up a book, mark a page, set the book down, and ramble off to a different book or a whole new aisle. Later on, you can continue browsing where you left off. Or you can click on your VCR and make "photocopies" so you can review the books at your leisure. Or, you can place requests to the cable operators, and they will reprogram a channel and ship you just the books that you want to see again.

You'll get all this for only a small fee—or maybe a large fee. It'll be worth it. And it'll be a blissful break from the mad TV zapper who is prowling around the TV room of almost every American family. ❐

MAILING LIST

By Maurice Yanney

If you want to keep track of names, addresses, birthdays, and other information, you'll find Mailing List a flexible and useful program. You can easily store information for up to 500 people pertaining to names, addresses, phone numbers, or whatever else you desire.

Once the data is entered, it can be retrieved based on any of the fields. You can also print labels in any desired order by activating some or all of the fields.

Typing It In

Mailing List is written entirely in machine language, but it loads and runs like a BASIC program. To type it in, use MLX, our machine language entry program. See "Typing Aids" elsewhere in this section. When MLX prompts you, respond with the following values.

Starting address: 0801
Ending address: 2238

Be sure to save a copy of the program before exiting MLX.

Using Mailing List

When you begin Mailing List for the first time or start from a new disk, the program will create files to hold data and information on which records are available and which are occupied. So use a disk that has plenty of room on it. Once the files have been created, the program will look for a setup file and, if one exists, load it.

At the top of the screen are three pull-down menus: System, Printer, and Record. On the right side of the screen are five other options called buttons. The main part of the screen displays the names of all the fields and the information contained in those fields for a particular record.

To select an item, move the arrow-shaped cursor with a joystick, mouse, or cursor keys. When the cursor is over a particular item, it will be highlighted. Press the fire button, mouse button, or space bar to select an item.

To select one of the data fields, position the cursor over either the name of the field or the text area for that field. (Some fields are larger than others and thus have a larger area from which the field may be selected).

When one of the pull-down menus is selected, a menu containing five entries appears underneath. To choose one of the options from the menu, move the cursor to the desired option, highlight it, and press either the button or space bar. Moving the cursor out of the menu area will remove the pull-down menu.

Editing Data

To edit any of the fields, just move the cursor over either the field label or the text area and press the button or space bar. Once the field is selected, the arrow will disappear. The Home key positions to the start of the field, the Insert key will insert a character, and the Delete key removes the character under the cursor. The cursor keys can also move the cursor left and right. While editing a field, the Run/Stop key can be used to cancel the edit. Run/Stop also restores the field to the previous text in addition to exiting the current edit.

Adding Data

You can enter data by moving to a field, pressing the space bar or button, typing the desired text, and pressing Return. This approach is tedious and suited only for editing a few fields of a particular record. When adding several records, switch to the Add mode.

To enter Add mode, move the cursor to the right of the screen and click the Mode button. The mode will switch from Edit to Add, and the First Name field will now be selected. Proceed to enter the data. To leave any field blank, just press Return.

When you press Return on the last field (Ind 2), the record is written to disk, and the program advances to the next available record position. Note that the Rec # indicator is updated and the First Name field is selected.

Continue this process until you no longer wish to add records. To stop entering data, press the Run/Stop key. This will switch the mode back to Edit and return the arrow cursor. If any of the fields have data when the Run/Stop key is pressed, the data will not be stored unless the Write button is selected. When all 500 record positions are filled, you'll get a message telling you that no space is available.

While in Add mode, data in the fields remains preserved. If you notice a mistake after you've moved to a new field, press the Run/Stop key, which sets the mode to Edit and returns the cursor to the first field. Make the changes, set the mode back to Add, and continue entering data.

System Options

To change screen and cursor colors, use the System pull-down menu at the top of the screen. Highlight the desired option and then press either the space bar or fire button. Moving the cursor outside the menu box will remove the pull-down menu.

Once one of these options has been selected, the bottom portion of the screen will display an arrow pointing to the current value. Move the cursor left or right and press either the button or space bar to select the setting.

Printer Options

The options in the Printer menu are selected in the same fashion as those of the System. The printer options let you customize your printed labels. You can save the setup, load the setup, reset the default values, set the printer format, and print records.

The first three options are used to save, get, and reset the format that the labels are printed in as well as to save, get, and reset the system options. When the Save Setup option is selected, the current settings for the printer, screen colors, and cursor speed are stored in a file. If the file exists, it will be overridden. Load Setup will retrieve prior saved settings. Setting the default will reset the values to what they were the first time the program was run.

When the option to set the printer is selected, a new screen appears. Use this screen to set the printer device number, the printer secondary address, and the label format. To select a particular item, move the cursor to the desired item and then press the space bar or fire button.

Once an item has been selected, the cursor will change to either a minus, a vertical bar, or a plus. The new cursor signifies the direction the cursor keys, joystick, or mouse can be

moved to change the selected item. When the item is a number (such as the printer device number), moving up increases the value by one; moving down increases the value by ten.

If the item is one of the label fields, then the cursor will be either a minus or a plus. Moving left or right will go through the various fields, which are abbreviated by three characters: First Name (FST), Middle Name (MID), Last Name (LST); Address Line 1 (LN1), Address Line 2 (LN2), City (CTY), State (STA), Zip Code (ZIP), Phone Number (PHN), Other Information 1 (OT1), Other Information 2 (OT2), Indicator 1 (IN1), and Indicator 2 (IN2).

There is also a Numbers option for sending special ASCII codes and a series of spaces to separate items. (This does not result in any output to the printer. It is used to help improve screen readability.) This option is a number between 0 and 255. Moving the cursor left or right will change the selection. Moving it up will change the value by 1, and down will change the number by 10. (A value of 65, for example, would send a lowercase *a* to the printer.) Some special numbers to keep in mind are 10 (linefeed), 13 (carriage return), 27 (escape), 32 (space), and 44 (comma).

Once the desired option appears, press the fire button or space bar to select the item. When you've finished making changes to the printer, move the cursor to the Done option.

Mailing List is designed to be flexible. This means that carriage returns at the end of each line and spaces between the fields need to be explicitly specified. This is the default setting. Most labels require six lines. So if you will be printing many labels, make sure that there are exactly six carriage returns per label, or else they may not print properly. Care must be taken in setting the fields so as not to exceed the label width, since the program does not restrict the line width.

When the option to print the records is selected from the Printer menu, the bottom of the screen will display two choices. You can either print the current record or multiple records by selecting the Based-On-Get option. This second option works in one of two ways, depending on the Get mode. If the Get mode is Seq (sequential), then all the records starting with the current record will be printed. If the mode is Patt (pattern), then all records matching the selected Get pattern will be printed. The printing can be stopped by pressing the Run/Stop key, letting you abort or continue printing.

Customizing the Setup

Once the printer and screen settings are to your liking, save the setup under the Printer menu. The next time Mailing List is started, the setup file will be loaded, and your favorite colors, cursor speed, and label format will automatically be set.

Record Options

The Record options at the top of the screen are used to go to the next and prior records, set the Get condition, count the number of records, or delete the current record.

The Next and Previous Record option will obtain the next or the prior record. If the Get mode is set to sequential, then the next occupied record or the prior occupied record is retrieved. If the Get mode is set to retrieve based on the pattern, then the next record meeting the Get option or the previous record meeting the requirements of the Get option is retrieved.

To set the Get condition, a new screen is displayed. The screen contains three items which can be modified: Get Option, Get Operation, and Get String. Cursor to the desired option and press either the space bar or the button.

When Get Option is selected, the cursor changes to a minus. Moving left or right changes the current field. Any of the data fields can be used in adjusting how records are retrieved. Once the desired field has been reached, press the button or the space bar to make the selection.

The Get Operation has three alternatives: less, equal, and greater. To change the condition, move the cursor beneath the field, press the space bar or button, and then move left or right to the desired operator. Once again, press the space bar or button to make the selection.

The Get String is the string that the Get Option field is compared to. This option is selected by moving underneath the field and pressing the space bar or button. Once it is selected, enter the string and press Return when finished.

Move the cursor to Done and press the space bar or button to return to the main screen. The Get Option, Get Operation, and Get String will be displayed at the top of the screen.

The Get mode is useful when looking for records meeting a specific condition or for printing records matching a particular requirement. Note that although the Get condition may be set, the mode must be set to pattern (via the Get button) to take affect. If the mode is sequential, then retrieval of the records will be sequential regardless of how the Get condition is set.

If, for example, you want to print out all the names of people in the state of North Carolina, set the Get Option to STA, set the Get Operator to equal (=) and make the Get String NC. Set the Get mode to pattern, go to the Printer menu and select the Print Records entry, and then print on the Based-On-Get option.

Other uses of Get involve using the Indicator 1 and Indicator 2 fields. They can be used to keep track of people on certain lists such as Christmas card lists. Each Christmas just print out the records where IND1 = Y, or however you want to set it.

The Count Records option displays the number of records stored. To delete a record, first go to the proper record number and then select the Delete Record option. The record will still be displayed on the screen as a safeguard, but it will be marked as removed on the disk. If you change your mind after deleting the record, press the Write button while the information is still onscreen.

Buttons

The buttons are options that appear on the right side of the screen. A button is selected by moving the cursor to the item and pressing the space bar or fire button.

There are buttons to select a specific record number, set mode to Add or Edit, set the Get retrieval to sequential or pattern, Write a record, and Clear data from the screen.

Once the Record Number is selected, you must then input a number between 1 and 500. If the selected record number has data, that data will be retrieved and displayed. If no record exists, the record number is updated, and the data fields are cleared on the screen.

The Mode button toggles between Add and Edit. Its use is discussed in adding and editing records.

The Get button toggles the retrieval method to either sequential or pattern mode. Sequential mode is used when records will be retrieved sequentially while pattern mode indicates that retrieval will be based on the settings of the Get condition.

The Write button writes the contents currently displayed on the screen to the current record number. If, for example, you want to update the phone number of a friend, first select the proper record number, make the change, and then press the Write button. If the Write button is not pressed, any changes will be aborted.

The Clear button clears all the data displayed on the screen for a particular record. The data is not erased from the record on disk.

MAILING LIST

```
0801:0B 08 70 17 9E 32 34 30 6E
0809:37 00 00 00 20 20 20 20 96
0811:20 20 20 20 20 A0 C4 B9 06
0819:3C 08 99 F8 00 B9 FD 08 F6
0821:99 33 03 88 D0 F1 A0 09 4C
0829:B9 0C 08 99 FF 03 88 D0 A1
0831:F7 A9 FF 85 2D A9 2A 85 E9
0839:2E 4C 00 01 1B E6 03 FF FD
0841:2A AD 20 B9 6E 09 99 E8 25
0849:07 C8 D0 F7 EE 02 01 EE 19
0851:05 01 C6 F9 D0 ED A2 03 23
0859:20 34 03 F0 33 C9 07 D0 95
0861:16 A2 01 20 34 03 D0 0A A0
0869:A2 04 20 34 03 18 69 07 65
0871:10 05 A2 0A 20 34 03 85 1D
0879:A8 A5 A7 85 A9 A5 FE 85 FB
0881:F7 A5 FF 85 F8 20 6C 03 73
0889:A5 F8 85 FF A5 F7 85 FE 72
0891:E8 20 34 03 D0 1E A2 08 21
0899:20 34 03 A0 02 84 A8 85 2A
08A1:A6 18 A5 FC 65 A6 85 F7 58
08A9:A5 FD 65 A7 85 F8 20 6C EF
08B1:03 4C 13 01 E8 20 34 03 FB
08B9:D0 1C A0 03 84 A8 E8 20 36
08C1:34 03 F0 08 A2 08 20 34 F4
08C9:03 4C 5C 01 A2 0F 20 34 CF
08D1:03 E6 A7 4C 5C 01 E8 20 AF
08D9:34 03 D0 0A E8 20 34 03 B2
08E1:18 69 04 A8 D0 D6 E8 20 37
08E9:34 03 D0 0A A2 02 20 34 21
08F1:03 18 69 06 D0 ED A2 08 A2
08F9:20 34 03 D0 E6 A9 00 85 F7
0901:A7 A4 FB F0 0C 06 FA 2A 37
0909:26 A7 C6 FB CA D0 F2 A8 D8
0911:60 48 B1 FE 85 FA A9 08 FE
0919:85 FB 68 A4 FE D0 02 C6 4A
0921:FF C6 FE C0 E7 D0 DE A4 B5
0929:FF C0 07 D0 D8 A9 37 85 BA
0931:01 58 4C 0E 08 A4 A8 F0 59
0939:22 A5 F7 38 E5 A8 B0 03 7E
0941:C6 F8 38 85 F7 A5 FC E5 8A
0949:A8 B0 02 C6 FD 85 FC B1 3A
0951:F7 88 91 FC 98 D0 F8 C4 42
0959:A9 F0 0A B1 F7 C6 FD C6 76
0961:F8 C6 A9 10 EC 60 78 E6 98
0969:01 4C 16 08 60 00 0C 08 75
0971:0A 00 9E 20 32 30 36 32 4F
0979:00 00 00 E7 FA 0F 82 A5 C8
0981:BA 81 D2 90 07 C9 0C B0 DC
0989:03 8D 79 85 9B 51 08 7A B4
0991:1B 20 76 1C B6 3E 03 60 DE
0999:70 46 11 29 19 70 E1 14 8C
09A1:1F 19 0A D6 CD FD 02 A3 46
09A9:13 40 58 2A AE 13 A8 E4 FA
09B1:01 28 99 98 97 3C C0 A8 E2
09B9:28 0E F9 1A 47 08 86 CA 76
09C1:80 61 25 F2 00 C0 D1 07 ED
09C9:D0 78 0C 3E 30 2E 00 37 39
09D1:0A CA AC A9 15 42 E9 67 B8
09D9:88 EF 0C 1C 5E 15 0C 7B 4A
09E1:F0 F0 05 AE C0 2A D9 38 CE
09E9:99 0B D2 78 B8 2A C6 1E 87
09F1:5A 09 19 A7 12 3E 9C 84 58
09F9:0D AE 0A BB 1F 16 1E 80 49
0A01:C0 67 10 0C 60 D7 20 EE A3
0A09:1E 44 46 04 C3 50 C8 35 6C
0A11:A0 07 A8 A6 05 1C 86 C2 1F
0A19:AD A0 08 41 99 06 E4 30 20
0A21:14 6A 0D A0 0B 8A 21 07 51
0A29:87 A1 30 6B 10 86 00 23 E3
0A31:86 08 41 18 0A B1 06 A0 F7
0A39:0D C5 90 09 C3 50 78 35 6D
0A41:A0 0E 20 0B 48 06 0A 61 AD
0A49:28 B4 1A A0 0F 20 17 25 38
0A51:C9 0B 04 67 28 AC 1A A0 CC
0A59:11 14 43 0C 0E 43 21 D5 B9
0A61:A0 12 A0 18 32 08 0B A2 4A
0A69:8C 53 83 A0 71 EC 15 06 82
0A71:0E 61 8C 68 18 0A 48 60 D6
0A79:0F D0 09 A2 8D A9 31 A0 AA
0A81:68 38 F8 23 C9 11 09 DA A8
0A89:40 F5 09 A8 6E 09 1C 8E 45
0A91:6E 3D 23 96 22 34 16 00 08
0A99:0E 62 44 A3 20 61 15 67 28
0AA1:4C BE 0F A2 56 A9 56 8B 48
0AA9:06 4B A3 46 0E E9 07 AD 40
0AB1:78 2D C9 3F AE F2 80 34 F0
0AB9:0A 02 1C 0C 1B 60 71 0F E3
0AC1:05 70 AA 35 05 38 76 03 16
0AC9:0C 28 3C 79 F0 09 26 20 25
0AD1:02 1C 63 B0 36 C0 F0 DE DA
0AD9:09 0A 55 60 B0 20 30 7C 88
0AE1:8A E7 02 85 47 4B 81 58 90
0AE9:C9 12 D0 13 40 6B 03 D1 3A
0AF1:A4 D0 60 C3 03 8E 3A 2B C6
0AF9:20 80 1D C9 13 60 F1 8E 0B
0B01:60 24 6A 22 63 10 5F D6 B0
0B09:07 1A 2D 0A 20 87 14 BD 74
0B11:84 C9 14 8A A9 05 1B 40 DE
0B19:95 1A 4C 72 08 B8 94 02 7F
0B21:C6 C3 08 0C 16 E2 0A 7D 1B
0B29:2D 40 65 01 30 BE 24 18 7F
0B31:8D 50 58 4C 1B 34 C4 01 26
0B39:04 4C A0 93 1A D1 B8 96 D1
0B41:10 04 1B 39 04 A0 1E 55 8B
0B49:48 81 68 20 F6 C0 EF A1 2F
0B51:AA E3 18 A0 40 2C 18 0C B1
0B59:81 07 06 0B 86 40 4C 0A 3B
0B61:4C 5F E0 21 C2 13 B0 40 A7
0B69:CC 81 B4 20 20 A2 09 A0 1D
0B71:17 CA 64 03 1C 79 0B 20 7F
0B79:CD BD 60 B1 FB FA F8 8A 55
0B81:C2 5C 1B 88 81 06 1D 18 72
0B89:71 64 14 1A C9 0A AC A0 06
0B91:30 88 88 AE C2 83 14 C4 EE
0B99:2C 83 48 03 CC 0A AA E8 AC
0BA1:55 3D 18 60 A0 1C 8A A5 EA
0BA9:22 E6 00 E1 4F 11 05 4E BF
0BB1:23 AE A5 0B 89 D0 EC D7 AB
0BB9:2C 99 94 2E 0A 70 7A A1 6A
0BC1:A4 40 01 68 BC 1E 30 04 A3
0BC9:0F 72 44 1E E8 60 99 A3 0E
0BD1:10 80 46 F1 00 6E 10 81 53
0BD9:38 43 C3 07 E7 3C 02 07 01
0BE1:47 80 10 80 7A 8A 8C C7 A4
0BE9:30 C3 8E 8B 09 31 CC D0 0B
0BF1:60 D9 0F 10 FD 09 82 D9 84
0BF9:01 E9 0F 0E D4 20 08 FA FF
0C01:06 3F 93 15 02 60 A0 DA 5D
0C09:0A 38 00 A2 10 6D 05 CF 6E
0C11:00 C3 23 2B 0B 61 C0 8F 20
0C19:8A 80 0E 0E 50 11 50 78 19
0C21:BC 08 30 3C 20 8A 05 80 19
0C29:D8 78 11 C8 D1 01 0E 4C 75
0C31:DB 0B AD 9B 08 F0 22 90 42
0C39:A7 27 01 4C 1A 0C 01 01 D7
0C41:04 B1 07 94 4B 50 23 3F 13
0C49:C1 D8 CF F7 8A 9A 21 A9 9C
0C51:2C 60 21 99 02 54 0D 53 24
0C59:CD 41 C8 0E 66 83 28 28 5C
0C61:7E 28 99 00 00 98 A2 00 9D
0C69:F4 9E 85 6C C9 33 10 06 5C
0C71:17 25 1C CF 00 86 1B 01 30
0C79:E3 E1 8D 56 42 21 40 F4 1F
0C81:82 C2 89 57 13 0E 58 2B DE
0C89:85 90 EC 66 10 0A 0F C2 16
0C91:D8 A0 51 97 F5 E3 0F 20 5F
0C99:2E 2C 1A 01 10 8D B8 8A D9
0CA1:AF 25 A9 BD 0F 35 19 84 EF
0CA9:B8 81 14 99 22 05 71 83 26
0CB1:A9 0A D6 C7 84 A9 0F 1A 7B
0CB9:95 10 AB 12 62 5A 90 C1 96
0CC1:08 C0 A4 82 00 04 48 05 70
0CC9:A9 04 8D DB 31 A9 07 8D F2
0CD1:DC 31 A0 47 B9 C3 29 99 F5
0CD9:93 31 D7 E5 0C 70 A8 04 D8
0CE1:08 18 A0 2A 6A A8 B9 5D 81
0CE9:2B CA 83 B9 57 2C 3D 23 5F
0CF1:00 22 0D C8 C0 FA D0 EB 40
0CF9:C1 E1 01 AA F6 D8 2B E0 88
0D01:78 3F 10 85 0A CD BD 8A 0F
0D09:0A 46 8C 15 A0 A0 50 99 5E
0D11:70 F0 B2 14 30 6E 0F 18 A8
0D19:56 17 0C FE 03 41 00 3B ED
0D21:0D 02 6F A0 14 B9 C1 31 76
0D29:15 39 0B A2 F6 AD 43 4D E9
0D31:40 CD 01 60 32 E0 AE A1 19
0D39:84 0A 2B A4 00 18 60 7D 66
0D41:8D C2 34 80 D6 03 AC 67 E4
0D49:C8 8C 80 3A 8A 03 18 6D 9C
0D51:B2 AA AC A0 06 BD E9 20 2A
```

```
0D59:D0 04 CA 88 6B 49 8C 25 7D
0D61:04 54 07 C0 8C E8 42 BE CA
0D69:F0 26 C9 BC F0 40 83 87 A1
0D71:C4 D0 17 35 12 0D CF 48 05
0D79:0B CE 48 06 4C 8E DE 5E EE
0D81:81 80 B7 60 74 CC F2 D0 07
0D89:F3 F7 11 EC CF 47 E7 CE C6
0D91:47 DF 4C B0 0D B9 A3 30 A2
0D99:DD C4 30 90 D7 F0 02 B0 1A
0DA1:D0 C8 E8 20 EE 0D F0 CC CF
0DA9:CC C6 E0 27 F0 C4 CC EC 8B
0DB1:A2 C2 4C CE 0D AD 7D 80 DE
0DB9:05 08 07 AD 25 C9 21 C5 6C
0DC1:13 22 A1 C3 19 83 08 20 65
0DC9:98 38 ED BF C1 03 F6 39 39
0DD1:C8 A0 06 2E 08 63 10 0D 17
0DD9:01 22 27 F5 67 21 40 E4 66
0DE1:ED E4 04 4B 30 AA 02 32 C3
0DE9:4A 3E 60 20 10 0E F3 A9 11
0DF1:1E 88 AD 18 0F 09 D8 8A 4D
0DF9:2E 67 AB C2 83 67 AB 8E 46
0E01:A4 80 8F AB 4F 6D 31 0C DA
0E09:12 30 0F 90 13 67 1E 20 B7
0E11:07 39 28 F0 19 50 43 21 C4
0E19:AD C1 51 71 8A E7 28 C0 C2
0E21:38 80 F0 D0 B1 4C DF 0E 31
0E29:55 94 85 1C 64 C2 30 55 6B
0E31:CE 80 08 39 14 D4 E8 14 43
0E39:24 34 42 A9 16 B6 04 11 FB
0E41:18 04 00 8C D0 08 03 10 EF
0E49:94 05 22 86 B9 0D 71 89 0C
0E51:63 01 20 36 F8 0C E6 A9 36
0E59:9E 5C 47 86 43 4C 0B 0F 9D
0E61:AD 26 C9 9E 02 3D 18 88 BE
0E69:04 A4 49 00 4C 4A 0F 40 C3
0E71:B5 03 B0 03 B5 C0 18 82 D2
0E79:01 08 C1 8C DF 28 41 7E B9
0E81:8E C3 0E D4 E0 00 88 86 83
0E89:39 DC 77 0F 09 06 8D C0 95
0E91:1C 40 3D 98 17 A9 0D 39 AF
0E99:B8 C5 99 AE D4 02 EE 06 34
0EA1:7B 20 0E 58 1B 20 34 0D 99
0EA9:F0 16 B6 78 A1 36 0B 8F AD
0EB1:DA 03 4C 16 7C 4B 1F 22 58
0EB9:94 A1 34 8E 8F 0A 4C C9 FE
0EC1:0E F0 13 94 FF 05 09 BD B0
0EC9:94 2E E3 80 4C D8 0F BD E1
0ED1:29 5C 3C 60 36 20 A4 39 DB
0ED9:01 2D 05 30 1D AE BE 5B E1
0EE1:CA 04 48 D0 CC BC 94 22 1F
0EE9:9C 90 C2 DF 20 A3 0B F0 65
0EF1:17 C1 C6 49 C9 FF 28 2A 40
0EF9:06 21 CD 22 1C 80 06 CD F9
0F01:CD 8D D0 A5 06 ED 07 AD 81
0F09:BA 30 F0 27 D8 86 07 06 16
0F11:D0 1F AD BB 0A 16 00 87 01
0F19:82 BC E1 3F 2B 30 86 06 05
0F21:A0 CA B9 72 2F 76 68 20 E4
0F29:F5 4F 5D 46 3B 04 AD BD 29
0F31:30 8F 75 80 E1 5E 10 C5 70
0F39:3A C0 C8 0E C7 8D C2 30 C8
0F41:68 4A 20 33 22 ED 3D 65 06
0F49:2B F8 3C C1 4C F1 3D 90 14
0F51:F1 A0 A1 DA F0 F8 1E D8 F2
0F59:4C BB 16 47 35 E2 11 2A 84
0F61:21 14 15 87 0E 14 4E C6 54
0F69:83 80 91 0B 02 0E 10 DE 93
0F71:08 42 A1 03 20 61 53 1C D1
0F79:68 F5 10 78 29 0E 08 42 A6
0F81:50 1C C8 D5 A3 38 90 CD 32
0F89:4C 10 11 53 69 60 07 51 55
0F91:31 B5 71 11 1D 8F 6A 40 31
0F99:23 6F 42 E1 C1 18 28 0B 55
0FA1:A0 D6 03 05 CC 7A 00 0E D4
0FA9:44 01 1C E3 50 2B A0 55 B1
0FB1:11 CB 50 40 C8 48 E0 00 82
0FB9:41 10 D2 10 8C 46 FC A1 F0
0FC1:01 0D D0 28 17 10 13 85 E4
0FC9:08 82 A4 C5 8C D3 D0 13 E5
0FD1:24 97 43 1D 84 0C 8A 20 AB
0FD9:59 AA 11 2E 40 F0 DF C0 9A
0FE1:07 F0 DB C0 33 D0 0C 4A 86
0FE9:0E 17 B1 63 22 83 10 48 C8
0FF1:E4 0F 06 0C 57 51 20 21 29
0FF9:24 4C 0F 12 C0 3F D0 1F 04
1001:D0 F1 AE E2 2E 16 47 4E B0
1009:60 E2 D1 82 C0 C0 01 F0 70
1011:3B E0 7D 4B B9 C2 EB 70 8C
1019:09 E0 0D 09 83 B9 03 EC 1E
1021:4C F9 11 B9 81 EB E2 CE F3
1029:0A 22 AE C2 AD 04 2B 12 BE
1031:80 20 04 24 A5 C5 38 ED FE
1039:78 2D F0 F8 20 6E 12 4C B9
1041:72 11 D5 DD 21 04 52 80 B5
1049:50 2A 60 15 03 2B CA F0 C4
1051:A9 3E 30 C6 14 44 34 84 E6
1059:F5 AD B9 13 9E 13 05 01 94
1061:51 B1 F1 57 54 88 CC 90 39
1069:90 06 03 41 4C 49 12 60 D3
1071:20 73 0A 14 FC 64 0A C8 57
1079:AE 07 2B 57 2F 90 F4 F0 24
1081:F2 60 C2 4A C4 71 4F 80 3B
1089:83 00 B0 00 85 CC AE 41 7F
1091:2B AC 74 80 83 60 8A FB B7
1099:C9 80 90 68 32 1B 91 FB 74
10A1:B0 48 08 18 74 50 D0 1A 4F
10A9:A5 90 F0 16 29 40 C9 40 5E
10B1:F0 10 A0 1C B9 0A 29 54 C0
10B9:F5 90 C2 11 A0 4C 18 45 0E
10C1:0F F1 90 5C 50 02 30 A0 49
10C9:08 70 60 0A 0E 18 26 02 D5
10D1:06 C0 26 F0 04 C8 4C C6 9B
10D9:12 83 F0 CE 03 71 AD B8 E0
10E1:40 19 20 B1 36 D0 0C 67 FB
10E9:91 32 93 60 01 4C 0F 13 42
10F1:AD 51 A0 15 8A C6 08 35 57
10F9:D0 12 AD 52 2D C9 30 D0 A3
1101:0B C7 59 02 04 C3 4C 38 E5
1109:96 6A 85 C7 8D 17 03 8A 97
1111:05 E8 03 83 1E A0 D7 43 EE
1119:89 D0 A9 51 A0 8B 83 76 2F
1121:80 86 42 80 90 71 1C 68 60
1129:90 03 2C 3C 30 21 D7 80 D3
1131:AA 53 91 0F 53 A8 D1 58 D9
1139:54 CC F8 58 21 BD 98 29 B7
1141:00 AE 79 2D A0 02 0B DB 10
1149:20 6A 1B 0A A2 26 A0 29 32
1151:4C 6C 13 8D 5A BC 03 E4 A0
1159:05 84 4F 50 E3 CF C0 62 50
1161:29 C8 E8 85 10 05 C5 05 E4
1169:58 33 77 70 66 E0 A4 68 E2
1171:6C 10 3D 6C 25 59 21 05 12
1179:2B 84 F5 99 32 10 04 1B 9F
1181:F5 17 1E 23 02 0C 20 00 DA
1189:06 21 80 7B 2D 20 AE 84 8A
1191:80 93 E2 0F 40 E0 AD 7A 81
1199:2D D0 47 AB 7D 88 47 D9 A0
11A1:86 CF 81 78 C9 C8 9E 4C AD
11A9:18 14 0F A4 4C DD 14 A3 AE
11B1:AB F0 C0 0D 69 CE 10 04 79
11B9:67 F1 1D E5 0F 50 91 0A F4
11C1:A9 00 A0 F9 99 5D 0A 7C 3F
11C9:57 2C 85 7A F5 72 7A 3E A7
11D1:3C 6E 05 15 8C 72 25 00 18
11D9:65 01 0B AC 03 2B 5D 1A B4
11E1:3F E4 E7 C9 C9 E0 2F 20 C6
11E9:A0 C9 70 B4 0F 62 9C 80 E3
11F1:F8 53 05 2B 66 CD CA 85 3E
11F9:87 91 00 0E 67 06 61 24 5F
1201:B0 F0 CA C9 13 C0 81 06 54
1209:2F 1B 1E 74 50 52 A8 04 B7
1211:08 25 70 29 34 80 25 04 15
1219:87 A5 60 71 29 06 C1 20 92
1221:3C 0B 20 52 13 91 1D 1F 87
1229:44 E6 81 86 13 A2 01 EC D3
1231:10 10 F5 01 1C 46 44 80 33
1239:41 37 47 01 D0 0B 15 99 3B
1241:19 1D 60 CD D6 13 1E C0 22
1249:F0 50 0B 30 50 02 0B 1D 1C
1251:D2 00 ED 0A 85 45 C0 50 50
1259:A9 A0 99 6F 07 86 2A 10 5B
1261:74 0D C9 FF EA 08 24 8A 86
1269:A4 06 71 F0 20 3E A0 0C E5
1271:8A 17 03 82 24 F5 16 7B C9
1279:2D A0 44 04 84 14 90 BC 77
1281:12 88 3E 01 0B 03 20 BA 08
1289:60 04 79 25 20 8B 25 94 6E
1291:AE AF 25 AC AB AE 10 8D 2E
1299:06 A6 2B 4C 80 15 B9 5C BC
12A1:2C F0 13 AD 7C 00 F9 5B 88
12A9:81 1C DA AD 7D 2D CD 5C 64
12B1:2B 00 27 11 17 15 50 CC DB
12B9:02 0F 35 1F 9D 01 3C 53 F7
12C1:37 54 60 90 05 6B 8C 5A F4
12C9:81 2C 18 F0 A0 36 A1 00 EC
12D1:A2 32 70 60 59 80 0C D0 9D
12D9:67 A9 05 AB 56 28 6C 58 FB
12E1:05 4E 18 85 38 A4 05 18 ED
12E9:B9 B0 31 E3 CF 46 05 F0 0E
12F1:DC DE 4D C6 C6 CA 8E 6C 3D
12F9:80 58 15 68 00 C7 AD 52 6A
1301:C9 23 F0 C0 02 22 06 88 2C
1309:B9 00 61 19 C0 4C FD 92 8F
1311:0D 60 A4 09 34 84 01 AD 5E
1319:46 0A C0 06 4C 86 02 E0 BE
1321:0C 16 24 40 3E 24 64 84 2B
1329:06 A9 13 5A 42 60 C3 18 F7
1331:20 08 60 C0 5E 3E 9E 85 30
1339:15 4C CF D8 96 E2 B0 59 7F
1341:DD 90 85 E2 6B 03 47 AD FC
1349:4C 70 16 A9 AB 40 77 09 65
1351:3C 01 7C AE E2 F8 CD 90 77
1359:8F D0 04 A9 0E D0 03 38 88
1361:E9 01 8A F3 15 57 E0 06 1B
1369:5A 02 D0 15 1D 46 AD 3C 42
1371:18 05 3C 83 38 18 E2 A9 36
1379:00 F0 DE B0 63 07 11 06 22
1381:22 56 1E 88 2D 4C 18 17 7C
1389:38 1B 0B 90 B2 E8 80 04 3B
1391:26 04 1C 4C 4F 18 AD 5C A6
1399:06 B0 B1 61 03 CB 00 62 E4
13A1:34 A1 21 4C 56 5F B1 10 D6
13A9:AD 64 19 80 0A AA 8E A7 AA
13B1:7D 28 A6 0A 4C EE 16 D8 39
13B9:91 0A 9D 93 31 4C 98 16 1A
13C1:0C FE 19 80 43 DB 41 14 F8
13C9:E8 16 AA CD 00 07 AE DC 72
13D1:31 E8 4C 04 17 20 E2 0A 9D
13D9:69 01 4C 12 17 A0 C4 D7 3C
13E1:11 06 9A 88 D3 1C 84 07 0D
```

```
13E9:A6 26 DD 07 A1 08 D1 C8 B2
13F1:EC A9 30 A2 04 04 84 0A 6C
13F9:06 A1 90 FF 17 04 28 41 F7
1401:0D 68 10 37 72 10 0E 20 4F
1409:E7 3A 04 46 41 28 A2 F3 7C
1411:04 50 41 10 11 1A 84 64 D6
1419:1C 84 12 06 A1 A0 78 A9 3D
1421:31 A2 13 20 C5 24 AD 8C 95
1429:E4 08 28 07 AD 8D 31 BB FC
1431:B0 00 8D 50 07 AE 91 61 DF
1439:E1 21 77 20 EA 08 27 2A 7B
1441:A9 08 A2 1F D8 31 00 62 74
1449:2C 59 13 41 5D 33 A9 57 B6
1451:B4 64 E6 C9 91 29 6C 8A FA
1459:8B C2 0F 41 A4 08 34 2F CA
1461:85 4C 1D 12 02 B9 86 99 C1
1469:44 66 C0 66 A9 52 8D 2A A7
1471:D3 17 5D EE C6 FF C9 A5 53
1479:90 D0 11 03 8D 03 20 E4 0E
1481:FF DC 5A 80 A0 98 EB 20 93
1489:A1 25 33 12 3D 18 A0 00 DE
1491:B9 DD 31 99 8E 31 C8 C0 59
1499:4F D0 F5 40 F4 CD 03 D1 17
14A1:51 D4 1C 8D 20 D0 38 A6 5F
14A9:03 21 75 C0 30 00 1C 05 15
14B1:0C 20 35 32 07 60 06 0F 86
14B9:70 00 CE 51 58 E8 0E E0 6C
14C1:B0 CA 28 CB 7C A0 29 EB 5B
14C9:2B 4E 36 1E 04 44 62 C9 83
14D1:23 91 AC 53 20 43 21 60 6B
14D9:17 00 02 06 C9 27 D0 0A C4
14E1:AE 98 09 40 8E 5B 3E 03 0E
14E9:9A 8A AE 66 E8 8E 80 30 F0
14F1:E0 24 D0 D2 20 95 0A 20 66
14F9:A6 0A 8C 0B 01 88 A5 05 B4
1501:66 A9 08 9E 69 8C 96 15 73
1509:C8 E8 C4 43 36 20 20 2D 3E
1511:C1 C9 20 08 9B C2 62 0F CE
1519:C0 1C 0C 5A A0 2A F4 82 EB
1521:0D 18 46 00 0D 3A 00 E1 D3
1529:A9 08 8D DF 29 A9 10 D0 BA
1531:61 20 73 08 63 05 5E F6 E5
1539:20 0D 97 7C 20 F0 02 18 52
1541:36 0B 42 03 80 5C 04 75 B4
1549:5B 4C 1C 30 5C 61 E2 80 69
1551:E1 50 60 0F 0B 18 07 00 44
1559:1D AA 02 0E A9 14 6E 4F A7
1561:A0 05 B9 C1 15 C0 E2 80 62
1569:99 55 04 88 FC 4A A9 9E C1
1571:50 68 17 F0 3B 5E 03 90 B9
1579:ED 54 F0 0C 46 0B E8 E8 A7
1581:E0 0E F0 E0 4C 8F 19 AE 4D
1589:1D 47 CA CA E0 04 F0 D4 E7
1591:8E 00 DE E0 06 41 B9 09 9E
1599:06 06 A2 4C 63 19 E0 0A AC
15A1:F0 EE 12 4C A9 B2 31 13 94
15A9:8D A0 30 54 D8 06 69 20 DF
15B1:B8 19 60 AD 9F 67 17 90 BE
15B9:0D 10 48 C9 0A 55 00 80 3A
15C1:18 4C 4D 24 40 F0 49 F1 41
15C9:E8 12 EE A5 88 0D 78 7C 0B
15D1:0A 42 28 4C 05 1A CE 40 C9
15D9:01 AD 41 C9 90 34 F0 0C F7
15E1:8D A1 31 39 18 4C CC 19 99
15E9:A9 AD D2 79 03 A2 7A F5 D3
15F1:20 A2 21 D7 91 09 C9 03 BD
15F9:B0 EC 48 3D F0 12 CE 8B 8D
1601:88 BB D0 17 A9 BE 37 A2 45
1609:4C 44 1A EE 5C D8 C9 BF 38
1611:D0 05 A9 BC 8D E0 02 AD 89
1619:A2 30 8D 7B 05 4C 0B 1A 95
1621:C4 97 00 44 A9 A3 09 1D E4
1629:0E 30 85 FE A0 13 B9 E8 B6
1631:02 58 14 99 C8 05 70 A1 6D
1639:05 F2 A0 0B A2 10 A9 23 35
1641:20 5B 11 3C 2E 03 B9 30 5B
1649:A0 C2 0B 84 2A A1 4E 23 B7
1651:05 37 87 5C A9 9E 05 A2 F8
1659:BC 20 B0 22 A7 7E 00 0A 65
1661:7A FC 00 44 D3 1A 09 89 F0
1669:14 91 01 47 86 11 40 38 C9
1671:3F E4 38 80 21 00 08 AD 4C
1679:01 DC AA 37 30 30 C0 80 6A
1681:02 30 0E E4 1A 28 C2 80 42
1689:03 4C 90 07 91 C9 0F F0 8F
1691:B6 E9 50 88 83 C3 06 47 A4
1699:F0 A7 20 7F 0A 1B 0F 0E ■D
16A1:4C 82 1A A0 0C A2 18 C0 BD
16A9:BF 03 A9 20 D4 CD 81 ■3 17
16B1:3E 10 29 60 20 2F 25 20 54
16B9:39 17 C0 D1 0B 7B C0 28 6D
16C1:84 C9 20 F0 53 C9 30 90 68
16C9:57 C9 3A B0 53 C0 00 F0 F4
16D1:30 76 7A 64 02 83 02 70 DB
16D9:0A C8 00 51 10 C7 45 AD 2A
16E1:02 5C 08 D0 11 03 2B 6D 8C
16E9:DF 5C 82 28 0E 10 07 2E DC
16F1:89 B9 56 2B 38 E9 30 AA 43
16F9:18 8A 6D 30 2E 00 1A 09 34
1701:0B C8 C0 03 D0 A1 DF 94 ■■
1709:04 00 8E 40 9E A4 99 BF ■■
1711:02 88 D0 FA A2 00 BD CB 88
1719:25 99 DD 02 38 47 08 E0 ■■
1721:E0 74 F2 A9 0B 8D F8 07 5D
1729:B3 FB 41 A9 07 2B 88 40 29
1731:EC 10 08 46 00 1B D0 A9 F6
1739:2F 8D BB A9 23 40 4F D0 FD
1741:71 A0 8F 43 46 47 85 74 45
1749:4F 3A 80 E8 98 AC E7 C0 54
1751:05 80 1C 32 2E C0 21 AC 2C
1759:E7 C0 15 F0 07 88 8C 80 51
1761:35 00 D4 80 E1 35 20 06 F6
1769:20 02 E2 00 C4 01 8D 00 C9
1771:D0 98 8D 10 D0 BD F0 2C 6C
1779:00 18 6D 00 A7 07 D0 60 B6
1781:1A 09 76 3C C4 01 55 0C 72
1789:65 1C 12 0C 98 3E 02 11 47
1791:64 C0 60 01 60 69 02 09 F3
1799:36 1E AA 01 E6 1B 4C 64 70
17A1:12 90 02 C0 40 C8 34 73 4A
17A9:04 04 E6 00 D7 C1 20 C0 7E
17B1:1B 81 D0 03 20 B9 1B E5 1C
17B9:12 85 C7 20 54 20 A9 A0 64
17C1:15 1E 7B 20 DA 06 50 0C 0F
17C9:50 88 80 E8 06 90 B6 80 41
17D1:99 0D 12 35 10 07 5D 04 01
17D9:C8 C0 0E D0 F2 E0 2E 06 E8
17E1:BA 99 F7 04 99 BF 05 99 82
17E9:0F 8A 0B AF 06 99 FF ■6 3B
17F1:98 18 69 28 A8 18 17 31 27
17F9:E6 30 5C 00 62 A1 9B 59 5D
1801:88 08 67 21 A1 67 21 0A 6D
1809:9E 85 0C 93 0D 79 16 AC CB
1811:72 16 E2 0F 20 6E 0A A9 C5
1819:B2 76 88 02 A7 63 A7 1E A9
1821:55 01 03 3C F6 54 F3 D8 2A
1829:CF 01 04 0E DC 3B 0E A2 75
1831:01 A0 0E 9B D8 CF A0 26 F2
1839:7B 24 06 00 1D 8F 82 79 96
1841:A0 14 C3 3C 80 0D 87 79 C3
1849:00 06 4F 90 01 06 C7 E1 7F
1851:80 3F 2B B5 A2 00 AD 65 28
1859:11 70 32 C8 AD D0 45 D1 0E
1861:FB 90 98 56 71 FF 2A 90 9C
1869:24 80 52 FB CD 00 2B 90 2B
1871:1D EC C0 21 D0 01 60 8E 6F
1879:F0 00 34 AD 06 B6 BA 09 0D
1881:1A 22 06 60 1C 92 C8 E8 B3
1889:EC 3E F2 BD 4A 9A 09 1A E2
1891:F8 04 42 A0 40 9C 18 AE E4
1899:01 2B E0 FF F0 43 08 6E 44
18A1:02 2B 07 C8 C8 9D EF 80 28
18A9:B5 1D 2A 28 1C A0 20 63 CA
18B1:C8 B1 FD 8D 01 45 00 8B F3
18B9:AC 04 40 7D 11 E0 78 2A 48
18C1:B1 FB 50 85 03 06 A6 BE 68
18C9:4C EE 1D C0 C5 0E 88 F7 FA
18D1:02 05 2B D0 E9 26 11 4C 0D
18D9:00 D1 05 0E D1 71 0E 51 C1
18E1:1D 0B 7C 0D 51 1D 0B 87 60
18E9:0D 51 1D 0B 92 0D 51 5D 12
18F1:3D 20 9D 42 34 10 0F 51 F1
18F9:10 3C 44 A8 3B 44 75 2C 56
1901:B3 34 44 75 2C BE 34 44 03
1909:75 2C C9 34 44 64 F5 80 9D
1911:D4 08 C5 60 4E 29 80 D4 5B
1919:22 A2 BB 1B 91 57 80 DF F8
1921:CE 42 1C 9C ■■ E8 D0 2C 8F
1929:C4 D1 ■5 0B F1 CD 42 07 D5
1931:3C 0B F9 A0 27 79 F6 00 9A
1939:08 20 69 0A A9 04 EC 5D D9
1941:82 1C 77 F7 1D D5 5D 46 6B
1949:A9 66 05 DD CD A2 73 57 9F
1951:E1 51 00 77 19 02 B0 90 03
1959:41 7F 3B CF 5D 36 3C 09 B5
1961:1B 77 6C 39 09 1B 77 89 4D
1969:39 09 1B 77 A6 39 09 19 9A
1971:F4 CC 3B F7 32 C2 D2 1E 98
1979:A0 05 87 7C 1D 3C 3E 7C C8
1981:1B 15 A9 7A C4 37 B1 0F D8
1989:5F 85 47 01 F0 65 08 C0 B3
1991:42 06 42 9F F0 C5 3C C3 83
1999:17 BD 4F C2 06 5F F9 17 95
19A1:56 42 06 5F 53 4E C2 06 A5
19A9:5F 62 8E AF 1F 1C F1 0B 49
19B1:35 0D 1B 63 0E 4C B4 1F 85
19B9:4C 4A 7C 10 27 77 85 DE 36
19C1:1C 2D 00 20 81 1E 20 63 77
19C9:76 23 A9 8E CB 3D 40 1C 0E
19D1:FC A9 C0 58 9D 00 20 8C 44
19D9:AE 36 28 20 A2 23 8D 40 F4
19E1:01 58 C8 FF F0 3D AD 3F DA
19E9:2B F0 D8 AD 3D 2B C9 00 0E
19F1:E6 09 71 4D C2 01 4E 42 D5
19F9:C1 10 95 90 01 0A 20 E3 21
1A01:18 E0 12 03 72 12 D6 0C 81
1A09:AA 84 C9 04 D0 06 20 16 22
1A11:11 4C 27 20 4C C7 1F A0 28
1A19:1B 20 96 23 20 3C 25 AD D1
1A21:07 2B 62 F0 53 02 82 51 F7
1A29:80 09 93 B7 6C 03 73 47 6B
1A31:41 01 53 1C C0 4C E1 14 81
1A39:91 60 50 86 02 48 D8 40 E3
1A41:04 D9 99 00 DA 99 E8 DA 0B
1A49:C8 D0 F1 8C 73 6C 00 01 6B
1A51:47 1D 05 C0 91 90 C3 20 93
1A59:21 1C 2D F7 08 DD 51 00 A4
1A61:28 8F 39 42 21 22 14 47 D9
1A69:49 43 E3 B0 B1 1F C0 D0 F6
1A71:CC A9 59 B2 D8 28 07 10 52
```

```
1A79:19 8E 39 13 C0 91 8E 0A A9
1A81:A9 85 09 28 D0 20 AC 23 13
1A89:AD 2C 20 7C 1A 8D 92 31 C8
1A91:8D 27 D0 20 C3 40 3E 00 0E
1A99:AF 01 E2 23 23 A8 E0 B9 AB
1AA1:10 9F 82 B9 43 76 0A 0E C7
1AA9:B9 11 99 11 0E F8 A8 B9 A2
1AB1:7E 2D CF A3 B9 46 2E DB C3
1AB9:B9 0E A8 0A 00 B9 D6 2F C7
1AC1:C3 2A E6 E3 60 A0 C7 14 A6
1AC9:99 50 04 99 18 53 01 E0 E9
1AD1:05 99 A8 06 88 8E 07 5F 50
1AD9:00 81 0A D0 80 01 08 01 D5
1AE1:80 1B 10 F2 01 0C 8A E0 7C
1AE9:41 AA A0 00 8E CF 16 8C E9
1AF1:84 BD 41 24 AE 7C 2D E8 EC
1AF9:AC 7D 2D C8 C0 05 D0 E8 BA
1B01:60 AD 31 36 BA 45 AD 9E 41
1B09:30 0A A8 C8 B6 BE 93 31 84
1B11:A6 01 17 CF 02 AD 35 29 15
1B19:8E A0 60 2B 12 E0 BB E0 E9
1B21:0A B0 03 94 60 AE 8D 02 0D
1B29:2E BF 02 9A 84 38 D0 3C 33
1B31:C9 07 D0 06 E0 1D EE 3A 1C
1B39:D0 3E 61 4F 0F C9 0F D0 17
1B41:0F B9 62 24 A2 94 0F AA 2C
1B49:A9 81 C9 1E 8A AA 29 01 22
1B51:F0 1B A0 09 02 F0 1C 8C 04
1B59:08 F0 0B 8A 29 04 D0 1E F2
1B61:40 78 41 C6 27 14 02 E4 DC
1B69:14 03 EC 68 FF CB 04 87 4D
1B71:F0 08 20 49 0A 38 24 00 34
1B79:11 3C 48 00 0A 78 90 00 A3
1B81:03 4C A2 21 70 FA 09 04 38
1B89:AE DB 31 AC DC 31 20 BA A5
1B91:FF F0 B8 61 80 BD 41 C0 6F
1B99:10 04 30 04 39 01 CC FF 86
1BA1:60 AA BC 2C 2A BD 39 2A F1
1BA9:AA 26 E7 88 03 A4 90 D0 E0
1BB1:05 88 CA D0 F5 C8 EE 11 B4
1BB9:2E AD 20 07 38 ED 64 A8 D1
1BC1:8C 40 12 D0 0A 20 AC 14 DB
1BC9:B9 C4 30 20 DA 18 20 D2 60
1BD1:FF AC 41 2B 66 04 C8 30 13
1BD9:53 61 32 C9 20 45 14 C9 FC
1BE1:40 B0 01 71 1E 60 C8 41 E0
1BE9:20 60 C9 80 B0 04 18 ■9 B8
1BF1:40 60 38 E9 80 4C 80 46 82
1BF9:B7 AD 6C 08 0A 18 6D 8E 9F
1C01:31 69 CF AA C8 D0 FD E8 3F
1C09:94 0B 60 98 93 61 04 03 11
1C11:E0 00 F0 11 81 4D 1B CA 2B
1C19:4C BE 22 9E 18 FB F5 78 6A
1C21:A9 20 13 45 F3 3C 14 EB 91
1C29:39 75 02 23 20 31 23 B1 8B
1C31:FD C0 D1 01 C0 0A D0 F7 AA
1C39:20 10 23 D0 EF 60 8C 24 35
1C41:8F 81 35 90 87 9E 2B A6 05
1C49:A7 60 FB 69 28 B0 14 38 E7
1C51:2E 18 FC 04 FD 69 0A D8 09
1C59:48 A5 FE 69 00 50 04 C8 A7
1C61:E8 E0 09 60 8E DB 6E AC DA
1C69:E1 04 C8 60 A9 D3 85 FB 56
1C71:A9 25 85 FC A9 27 C9 3F 05
1C79:A9 26 E9 60 52 B6 00 3C FC
1C81:F0 FA AD 00 AC 41 F3 0E 07
1C89:DC 29 10 F0 EC 93 7A 00 36
1C91:30 58 00 04 4E 41 13 A9 7F
1C99:1D A2 04 20 C8 1B CA D0 A4
1CA1:FA A9 05 8D 3E 2B 20 FA 15
1CA9:71 8D 07 F5 12 C0 F8 05 C8
1CB1:09 48 A3 90 04 14 1C 20 C6
1CB9:F5 1B FD ED 93 0C 4E C9 85
1CC1:FF 10 04 00 60 8D 55 20 82
1CC9:33 40 FB 19 16 17 A0 0C 17
1CD1:66 11 07 46 98 8E 46 10 62
1CD9:D0 F4 9E 16 9E 1C 00 03 55
1CE1:80 20 DE 23 DB A9 A0 91 C8
1CE9:FB AD 91 31 91 FD C8 C0 A7
1CF1:78 D0 F2 60 18 A5 FC 69 B9
1CF9:D4 D0 8E A5 FB 85 FD 80 6F
1D01:ED B0 90 1B E3 01 B0 F0 97
1D09:08 91 00 D0 C1 91 91 5D 8D
1D11:70 6C 50 00 47 85 E8 4F 1A
1D19:44 06 45 EC 03 65 0A 22 52
1D21:09 19 A4 A0 90 20 7D 12 D6
1D29:AE 39 CA 8E 40 2B 80 21 1C
1D31:08 C2 00 90 83 00 42 0A D3
1D39:A1 22 C2 44 42 E6 C7 C9 70
1D41:88 A2 C7 22 52 D4 37 08 DF
1D49:49 84 5A 43 40 08 A9 B8 F6
1D51:EC 20 A9 BD A0 28 20 1E 1F
1D59:AB 0E 2E 3D 23 20 8E 24 61
1D61:AF D8 15 D0 55 59 B7 48 21
1D69:1E 89 41 C9 11 B0 17 AD 01
1D71:CB 00 DE AA 20 80 1D AD 03
1D79:01 2B 8D 3A 2B A5 C5 C9 99
1D81:01 F0 FA 4D EF 20 A9 1D 1D
1D89:20 99 23 AF E9 7F 8D 0D 0F
1D91:DC A9 FF 8D 00 DC 88 87 89
1D99:C4 14 13 12 8D CF 20 B7 62
1DA1:22 83 65 B1 FD 11 80 91 5C
1DA9:FB C8 CC 02 2B D0 F3 28 7A
1DB1:F1 85 0F 78 16 2E 0A 7C A9
1DB9:16 EA 84 FD A9 1E A0 5D 8E
1DC1:D1 24 A0 85 16 9C 85 0B 93
1DC9:11 9F 85 85 FE A9 25 86 EC
1DD1:FD A2 08 20 5B 11 3A 39 83
1DD9:20 A0 00 99 C4 30 C8 C0 1F
1DE1:CA D0 F8 03 72 12 A0 8D AF
1DE9:13 13 14 05 8D 15 05 D0 E0
1DF1:0C A0 A2 06 A0 23 C0 41 63
1DF9:0A E5 72 AE 69 22 20 CD CD
1E01:BD CC 85 C7 87 78 8A A6 56
1E09:F4 9C A6 7B 2D D0 04 20 48
1E11:66 ED 01 20 7D 13 03 CA 27
1E19:18 69 B2 C9 40 69 68 52 79
1E21:32 5C A0 F0 0B C9 02 B0 E4
1E29:0A 84 4B C9 F5 B0 03 CA D4
1E31:09 10 28 1A 01 84 A0 A9 A1
1E39:0F 20 C3 FF 20 98 12 60 65
1E41:A9 00 8D E5 0E A9 01 8D 09
1E49:36 60 AD 5B 60 05 7C 2D 61
1E51:AD 5C 2B 8D 7D 2D 60 0E 29
1E59:08 93 00 FC F8 F8 FC DE D2
1E61:8F D8 B5 18 20 2F 23 44 1C
1E69:20 5F 23 70 20 87 23 08 66
1E71:2C 5F 2F 08 30 4B 33 08 1F
1E79:34 5F 37 08 40 9B 43 08 0E
1E81:44 9B 47 08 48 5F 4B 08 8E
1E89:4C 5F 4F 08 50 4B 53 05 89
1E91:58 73 5B 05 5C 73 5F 05 07
1E99:60 73 63 05 64 27 67 05 33
1EA1:68 27 6B 7B 48 8F 4B 7B 93
1EA9:2C 9B 2F 7B 34 9F 37 7B 8A
1EB1:3C 9B 3F 7B 50 8F 53 03 FC
1EB9:54 F1 30 03 10 07 03 1B 90
1EC1:06 06 C0 0B 07 24 08 42 66
1EC9:10 0B 44 0C 01 25 0D 00 D0
1ED1:05 0E 01 16 0F 01 11 11 45
1ED9:80 0B 12 88 13 00 1C 14 C8
1EE1:00 0A 15 00 09 0D E0 06 87
1EE9:82 14 08 1E 09 0A 1E 08 04
1EF1:0F 1E 05 17 24 3F 27 17 D2
1EF9:28 3F 2B 17 2C 3F 2F 17 C4
1F01:30 3F 33 17 34 3F 37 43 4F
1F09:24 6B 27 43 28 6B 2B 43 D5
1F11:2C 6B 2F 43 30 6B 33 43 33
1F19:34 6B 37 6F 24 97 27 6F 67
1F21:28 97 2B 6F 2C 97 2F 6F 43
1F29:30 97 33 6F 34 97 37 04 35
1F31:30 E2 05 05 44 06 40 04 EF
1F39:07 44 08 05 80 1C 40 05 57
1F41:40 04 06 44 07 40 04 08 EE
1F49:10 0A 04 50 05 40 04 06 CE
1F51:44 07 40 04 08 1B 0A CD 4A
1F59:D0 48 40 F0 09 30 5A 80 67
1F61:00 CE 41 4D 01 21 C6 49 B3
1F69:B2 5A E1 01 10 CD 49 C8 F6
1F71:4C 41 62 CC 72 E2 00 04 62
1F79:4C D1 1B 90 03 A1 0E 98 12
1F81:20 CC CA 18 3C 00 A6 F0 FD
1F89:49 54 59 B2 83 D3 54 EA D6
1F91:45 5D 0C 20 DA 49 CC 2B 0E
1F99:80 D0 48 4F 4E A9 23 27 D0
1FA1:08 75 3C CF 54 48 74 3E B0
1FA9:32 8D 41 31 0D 5C F8 01 6C
1FB1:A2 32 4C 64 D9 D3 68 CD 5A
1FB9:41 38 02 D0 ■2 C9 CE D4 24
1FC1:C5 D2 1A 1C 22 80 D2 C5 1B
1FC9:C3 CF D2 C4 0D 48 22 86 D8
1FD1:00 62 25 0E D1 0A 0F 10 13
1FD9:68 BD 43 21 0E DD 8D 2D 66
1FE1:70 85 C1 39 2A 41 44 0C 70
1FE9:40 28 C4 CC 4F 41 44 A9 69
1FF1:0C 80 43 61 23 46 4C 54 F3
1FF9:C8 2A E0 24 C3 00 0E 20 DF
2001:08 26 0A 54 00 06 C9 CE CF
2009:45 58 9E C6 86 09 34 56 59
2011:49 C7 46 24 80 20 89 00 8A
2019:6D 58 55 64 10 40 2C 53 44
2021:34 D2 30 51 24 40 6A 00 42
2029:03 75 E3 4E 41 2A EE C2 FC
2031:B9 0C 10 57 25 D3 43 5E 26
2039:AC E0 0D 41 2F 1E A0 45 36
2041:45 44 26 91 41 57 22 03 C1
2049:19 01 2E 2E C6 41 53 54 35
2051:8C 71 E1 48 E3 03 7C 40 59
2059:03 15 12 F7 D8 80 43 8A FB
2061:18 C3 55 52 53 62 01 AC 41
2069:4C 4F 52 75 6E 90 11 23 40
2071:60 0A CD 4F 44 45 90 00 6B
2079:66 01 04 D7 8D 74 54 45 57
2081:00 C3 21 07 10 00 D0 41 AA
2089:19 ■0 8C 14 51 ■B 01 D■ ■A
2091:20 00 C5 44 49 54 D7 59 83
2099:46 4C 45 79 59 CA 8D 49 AA
20A1:53 4B 20 C5 52 52 3A 00 0F
20A9:12 8C 71 98 8D 6B C2 55 C2
20B1:54 54 84 04 CF 30 09 B0 04
20B9:01 80 C2 47 20 D4 4F 20 7A
20C1:C3 57 61 C1 1D 55 45 92 5D
20C9:00 C4 C0 CA 56 09 64 B1 51
20D1:50 A0 81 45 A9 98 2F C9 BE
20D9:3B 88 D2 45 41 44 59 00 56
20E1:B1 7E 41 56 15 4C 2C 40 9A
20E9:9B 3C C6 94 90 CD C9 C4 3D
20F1:47 50 48 00 41 46 47 CC 71
20F9:73 92 C3 D4 D9 47 D3 D4 C6
2101:C1 60 04 DA C9 D0 46 D0 59
```

```
2109:C8 CE 60 04 19 09 CF D4 11
2111:55 42 41 31 20 04 C9 CE 3D
2119:32 3E 3C 23 23 23 3E C4 A4
2121:03 20 20 20 12 D0 94 04 F3
2129:40 6C CF C4 45 56 49 78 7B
2131:CE 15 09 00 39 B0 E7 78 16
2139:43 95 65 44 41 52 59 DC 56
2141:44 44 52 B2 4C D8 07 73 73
2149:00 19 01 68 32 02 9A 81 C8
2151:03 26 E4 C2 05 A7 2C 40 C6
2159:06 06 A4 20 0D 20 07 00 AD
2161:13 8A 83 3A 16 32 3C C8 9E
2169:60 20 23 00 02 0E C4 B3 CD
2171:4F 79 50 41 43 45 20 C1 09
2179:56 41 12 CE 41 42 43 39 39
2181:80 3A 4D 4C 2E 53 A0 36 37
2189:00 53 2C 57 00 0E 18 27 2B
2191:45 63 72 81 8B 9F B3 C7 BF
2199:C8 C9 D6 0B 0F 1E 1E 0F 7A
21A1:0F 0A 14 14 14 01 01 12 6A
21A9:C9 4E 49 A0 C7 41 4C 49 BC
21B1:5A E0 54 C6 49 4C 45 53 A9
21B9:EE 13 04 6D 58 55 50 D0 19
21C1:42 3C C4 5C 56 A0 05 11 E2
21C9:19 1E 14 21 CF 50 5E 20 51
21D1:41 02 4F 4E 46 11 0D 0D A1
21D9:12 1D 1D 1D C7 45 54 20 FD
21E1:D3 54 52 CC 9D 2D 72 2A EA
21E9:CE 55 4D 42 45 52 78 46 61
21F1:46 82 EA 20 3D 20 9B C8 C1
21F9:10 D4 C9 CE D5 C5 3E A0 82
2201:3C C1 C2 CF D2 51 9E 91 D3
2209:3C C3 D5 D2 D2 C5 CE C1 51
2211:1D 20 3C C2 C1 D3 C5 C4 4D
2219:20 CF CE 20 C7 C5 D4 3E 7A
2221:00 12 D3 AC 27 54 49 4E 9A
2229:47 20 D2 45 43 4F 52 44 08
2231:53 00 30 00 00 00 00 00 25
```

Maurice Yanney, the author of Balloon Pop (August 1992) and Cats and Mice (January 1993), lives in Lebanon, Pennsylvania.

UTILITY PLUS

By Eric Jevon Bryant

Utility Plus is a group of three useful utility programs for the 64 written in machine language to take advantage of the language's power and speed. The first utility, Word Wrapper, wraps text around the screen when words become cut off by the screen border. Integer Array Search lets you quickly search through an array of integers for a specific value, and String Array Search searches through an array of strings for a specific string of characters.

The utilities in the Utility Plus package can be used separately or together. They were programmed independently. Also, they reside at 52376–53247, which is near the top of free RAM, so it may be possible to run other machine language programs with these in place. If you use all three programs, this leaves you with 3227 bytes of free RAM, a little more than 3K.

Originally, Utility Plus was written to complement a text adventure I had written. The majority of the program was in BASIC, and the utilities were created to speed up the parsing of commands and outputting text to the screen. Utility Plus's usefulness, however, is not restricted to text adventures. You may use the package in just about anything from databases to your own word processor.

Entering Utility Plus

Utility Plus is written entirely in machine language. To enter it, use MLX, our machine language entry program; see "Typing Aids" elsewhere in this section. When MLX prompts, respond with the following starting and ending addresses if you want to have all three utilities.

Starting address: CC98
Ending address: CFFF

Note that Utility Plus is a package of three utilities that are independent of each other. This means that you may wish to type in only the ones that you need. If you want only one or two of them, use the addresses listed below and enter only those lines of the MLX listing.

Word Wrapper
Starting Address: CC98
Ending Address: CDFB

Integer Array Search
Starting Address: CDFC
Ending Address: CEE4

String Array Search
Starting Address: CEE5
Ending Address: CFFF

Be sure to save a copy of the program before exiting MLX. When you're ready to use the program, load it with the ,8,1 extension, type *NEW*, and then start to work on your own BASIC programs. You'll find Utility Plus helpful in many programming situations.

Word Wrapper

If you've ever used the PRINT statement in a program to print out instructions or other information, you know how difficult it can be to make the words wrap properly. It usually takes a good deal of trial and error to make the statement print correctly. Word Wrapper does the work for you.

To use this utility, place the text you wish printed to the screen in the string variable A$. (Failure to do this could result in the computer's locking up). Type *SYS 52376,0* and the text contained in variable A$ will be printed. Any words that might have been cut off by the edge of the screen will be wrapped around. If you type *SYS 52376,1* the words will print, but the program will automatically add a prompt that tells you to press Return to continue. Note that you may print something beforehand with a semicolon and then invoke Word Wrapper, and it will continue from that PRINT statement, wrapping accordingly.

Word Wrapper is quite powerful. After Word Wrapper has printed something onscreen, anything that is printed afterward will be tacked onto the end of the string, as if you used PRINT with a semicolon. To avoid this, simply print after you execute Word Wrapper.

Integer Array Search

An integer is any number that does not contain a fraction. On the 64, an integer must fall within the range of from 32767 to -32767. In Commodore BASIC, an integer is recognized as a variable name following by a percent sign, A% or BC%, for example.

Integer Search can look through any array of integers that contains no more than 255 elements. (It's better if your arrays have no more than 254 elements since a 255 is a null value for Integer Search.)

To invoke the Integer Array Search, type the statement *SYS 52732,XY%, NUM,Z*. In this statement, XY% is the integer array to search through (the percent sign is needed), NUM is the number to search for, and Z is the initial value in memory location 251. Index Array Search will retrieve the index held in 251 first and will start the search at the very next index. Therefore, it's a good idea to initialize your integer search by entering *SYS 52732,XY%,0*. This will place a 0 in 251 and start

your search at an index of 1.

The array index that contains your value NUM is held at memory location 251; just enter *PRINT PEEK(251)* to find its value. A value of 255 means there was no match, while anything less indicates the index where there was a match. If the variable you entered for XY% does not exist, the program will return a *SEARCHING ERROR* message.

String Array Search

In BASIC, a string is any string of characters. String Array Search lets you search through any list (an array) of strings for a specific string or even a part of a string.

Place the string you're searching for in string variable A$. This may even be the first few characters of the string. For example, JOH would locate JOHNS, JOHNSON, or JOHANSON, but not JIM JOHNSON. Failure to have something contained in A$ could result in the computer's crashing.

With this done, enter *SYS 52965,AB$,X*, with AB$ being the string array to search (the $ is necessary) and X being the initial value for 252. Once again, if string AB$ does not exist, you'll receive a *SEARCHING ERROR* message. This time, however, the matched index will be stored in memory location 252, so it will not conflict with the Integer Array Search. A 255 in this location indicates that a match was not found.

Utility Demo

This demonstration program gives you an example of the Utility Plus features and how to use their functions. Programmers may find that dissecting the code is helpful. Since Utility Demo is written entirely in BASIC, enter it with The Automatic Proofreader to help eliminate typing errors. Be sure to save the program before running it.

Simply run the demonstration with Utility Plus already in memory and watch as it demonstrates the integer and string searches, saving the Word Wrapper for the finale.

The program sets arrays T%() and IR%() for integer arrays and S$() for a string array. The demo illustrates the Integer Search by picking ten random numbers between 0–254 and then searching for these values in arrays T%() and IR%(). T%() has a maximum of 100 elements, so some of the random numbers will not be found if they exceed 100. Also, the values in T%() are equal to 100 minus the index, while in IR%() there is a direct relationship. Thus, the value 37 is held in index 73 in T%() and 37 in IR%(). Use this to check the searches.

The second demo deals with the string S$(). This array contains the word *TRASH* in 234 of its 254 elements and the word *TREASURE* is scattered at random in 20 elements throughout the array. The numbers onscreen are the indexes of the elements that contain the word *TREASURE*. The variable LI$ contains this list

Finally, the word-wrap demo prints some text on the screen, prompts you to press Return, and continues with more text.

Some Final Notes

First, the Z in SYS 52732,XY%,Z (Integer Search) and the X in SYS 52965,AB$,X (String Search) are optional; they may be left out. The utility will simply begin the search at the current index held in 251 for Integer Search or 252 for String Search.

Second, when searching for negative values in the Integer Search, you must place this value in a variable and then enter *SYS 52732,XY%,VAR* with VAR being this variable. If you attempt to enter *SYS 52732,XY%,-1* you will run into problems.

Third, if your A$ variable contains a short string and you call Word Wrapper, you may see some garbled text printed after your word. To clear this up, simply add CHR$(0) after your A$ text as in A$=A$+CHR$(0). This will add a null character to your string and stop the garbage.

Finally, there is a programming trick that will allow you to set the left margin for Word Wrapper. Simply enter *POKE 52496, 256 – left margin*. For example, if you wanted a left margin of 5, you would enter *POKE 52496, 255 – 5*, or *POKE 52496, 251*. There is also a way to set the right margin, but it isn't as simple; those able to decode machine language may be able to solve it. Hint: fool around with locations 52460 and 52496 combined.

UTILITY PLUS

```
CC98:A2 00 20 79 00 C9 2C D0 6F
CCA0:03 20 00 E2 86 B1 4C 47 CC
CCA8:CD 00 A5 D3 18 E9 27 B0 82
CCB0:02 69 28 8D A9 CC A2 00 49
CCB8:A0 00 B1 A5 C9 00 D0 08 2B
CCC0:A5 B1 D0 01 60 4C AC CD 1F
CCC8:20 D2 FF C9 20 D0 08 A9 C1
CCD0:01 85 A3 A2 00 86 A4 C9 18
CCD8:0D D0 05 A2 FF 8E A9 CC 52
CCE0:A0 01 20 8E CD EE A9 CC 42
CCE8:AD A9 CC C9 28 F0 0C E6 FD
CCF0:A5 A5 A5 C9 00 D0 BF E6 C1
CCF8:A6 D0 BB C6 A3 A2 00 E8 8E
CD00:A9 14 20 D2 FF E4 A3 D0 52
CD08:F6 CA A9 20 20 D2 FF E0 36
CD10:00 D0 F6 A0 01 20 8E CD 3C
CD18:A5 A5 E5 A3 85 A5 A5 A6 9B
CD20:E5 A4 85 A6 4C 2D CD C8 6E
CD28:84 A3 4C AA CC A2 03 E6 01
CD30:A5 A5 A5 C9 00 F0 08 CA F7
CD38:E0 00 F0 08 4C 2F CD E6 84
CD40:A6 4C 37 CD 4C AA CC A5 52
CD48:2D 85 A3 A5 2E 85 A4 A0 1C
CD50:00 B1 A3 C9 41 F0 06 C8 0C
CD58:20 8E CD F0 F2 C8 B1 A3 32
CD60:C9 80 D0 F4 A0 03 20 8E 4A
CD68:CD A0 00 B1 A3 85 A5 C8 75
CD70:B1 A3 85 A6 C6 A3 A5 A3 9C
CD78:C9 FF D0 02 C6 A4 A0 00 3D
CD80:B1 A3 A8 A9 00 91 A5 A0 BF
CD88:01 85 A4 4C 27 CD E6 A3 41
CD90:A5 A3 C9 00 F0 06 88 C0 92
CD98:00 D0 F3 60 E6 A4 F0 F6 8F
CDA0:12 5B 52 45 54 55 52 4E A5
CDA8:5D 92 A0 00 B9 A0 CD 20 B7
CDB0:D2 FF C8 C0 02 D0 F5 A0 BA
CDB8:00 B9 A2 CD AE 18 D0 E0 4C
CDC0:17 F0 0A 20 D2 FF C8 C0 50
CDC8:06 D0 EE F0 05 18 69 80 64
CDD0:D0 F1 A0 00 B9 A8 CD 20 91
CDD8:D2 FF C8 C0 02 D0 F5 A9 EB
CDE0:00 85 C6 A5 C6 F0 FC A0 A5
CDE8:00 A0 00 84 B1 A9 14 20 71
CDF0:D2 FF C8 C0 08 D0 F6 A9 36
CDF8:00 85 C6 60 A9 80 85 A4 D3
CE00:20 FD AE 09 80 85 A3 20 15
CE08:73 00 C9 25 F0 07 05 A4 3D
CE10:85 A4 20 73 00 A9 25 20 E5
CE18:FF AE 20 83 AE 20 AA B1 9A
CE20:85 A6 84 A7 20 79 00 F0 0D
CE28:05 20 00 E2 86 FB A5 2F 1D
CE30:85 AD A5 30 85 AE A0 00 DB
CE38:B1 AD C5 A3 D0 07 C8 B1 F2
CE40:AD C5 A4 F0 1E A5 AD C5 72
CE48:37 D0 0D A5 AE C5 38 D0 7F
CE50:07 A0 0C 20 2F F1 D0 06 06
CE58:20 DB CE 4C 36 CE 4C 65 86
CE60:A4 EA EA A9 00 85 B0 A0 1B
CE68:00 20 DB CE C8 C0 04 D0 98
CE70:F8 A0 00 B1 AD 85 AA C9 70
CE78:03 F0 D6 AA CA 20 DB CE B6
CE80:20 DB CE B1 AD 95 AF E0 1E
CE88:00 D0 F1 C6 AF A5 B0 F0 6B
CE90:02 C6 B0 20 DB CE A9 00 66
CE98:85 AB 85 AC A0 00 B1 AD 75
CEA0:C5 A6 D0 12 C8 B1 AD C5 34
CEA8:A7 D0 0B 18 A5 AB AA E5 48
CEB0:FB 90 03 86 FB 60 20 DB B6
```

```
CEB8:CE 20 DB CE A5 AB C5 AF 45
CEC0:F0 05 E6 AB 4C 9C CE A5 C7
CEC8:AC C5 B0 F0 09 E6 AC A9 3A
CED0:00 85 AB 4C 9C CE A9 FF 7D
CED8:85 FB 60 E6 AD A5 AD F0 03
CEE0:01 60 E6 AE 60 A9 80 85 0F
CEE8:B1 20 FD AE 85 B0 20 73 B4
CEF0:00 C9 24 F0 07 05 B1 85 C9
CEF8:B1 20 73 00 A9 24 20 FF 04
CF00:AE 20 79 00 F0 05 20 00 0A
CF08:E2 86 FC A9 00 85 AC 85 E9
CF10:AA A5 2E 85 AB A0 00 B1 1E
CF18:AA C9 41 F0 0C E6 AA A5 AD
CF20:AA C9 00 D0 F0 E6 AB F0 FF
CF28:EC C8 B1 AA C9 80 D0 ED 31
CF30:E6 AA 20 EB CF E6 AA 20 40
CF38:EB CF A0 00 A2 00 B1 AA F8
CF40:95 A3 E0 02 F0 04 E8 C8 02
CF48:D0 F4 A5 2F 85 AA A5 30 87
CF50:85 AB A0 00 B1 AA C5 B0 26
CF58:F0 20 A5 AA C5 37 90 0E 12
CF60:A5 AB C5 38 90 08 A0 0C EB
CF68:20 2F E1 4C 65 A4 E6 AA 1D
CF70:A5 AA C9 00 D0 DC E6 AB 3A
CF78:F0 D8 C8 B1 AA C5 B1 D0 9B
CF80:DA A0 00 E6 AA 20 EB CF A1
CF88:C0 03 F0 03 C8 D0 F4 A0 AB
CF90:00 B1 AA C9 01 D0 D3 E6 68
CF98:AA 20 EB CF E6 AA 20 EB 1E
CFA0:CF 38 B1 AA E9 00 85 A9 1B
CFA8:E6 AA A0 00 A2 00 B1 AA 9D
CFB0:95 A6 C0 02 F0 04 C8 E8 0F
CFB8:D0 F4 A6 A3 CA E4 A6 B0 F4
CFC0:0F A0 00 B1 A4 D1 A7 D0 B7
CFC8:07 C8 C4 A3 F0 27 D0 F3 AA
CFD0:E6 AC A5 AC C5 A9 D0 05 0A
CFD8:A9 FF 85 FC 60 A0 00 E6 3A
CFE0:AA 20 EB CF C0 02 F0 C2 0B
CFE8:C8 D0 F4 A5 AA C9 00 F0 87
CFF0:01 60 E6 AB 60 18 A5 AC 1C
CFF8:AA E5 FC 90 D3 86 FC 60 23
```

UTILITY DEMO

```
HB 10 REM UTILITY/DEMO
EX 20 REM BY ERIC J. BRYANT
XB 30 REM COPYRIGHT 1993 - COM
      PUTE PUBLICATIONS - ALL
      {SPACE}RIGHTS RESERVED
DD 40 REM DIMENSIONING ARRAYS
EC 90 DIM T%(100)
GR 100 DIM IR%(254)
MH 110 DIM S$(254)
MA 120 REM
XG 130 REM BUILDING ARRAYS
CC 140 REM
EC 150 FOR X=1TO100: T%(X)=100
      -X :NEXT
DP 160 FOR X=1TO254: IR%(X)=X
      {4 SPACES}:NEXT
PD 170 FOR X=1TO254:S$(X)="TRA
      SH":NEXT
CF 180 REM
SS 190 FOR X=1TO20 :IX$=MID$(S
      TR$(X),2)
KB 200 Y=INT(RND(0)*254):IF S$
      (Y)<>"TRASH" THEN 200
PA 210 S$(Y)="TREASURE-"+IX$:N
      EXT X
HH 220 REM
QF 230 REM DEMO
AH 240 REM INTEGER ARRAYS
DK 250 REM
DA 260 PRINTCHR$(147);CHR$(18)
      ;"INTEGER ARRAY DEMO:"
PF 270 FOR X=1TO10:Y=INT(RND(0
      )*254):NU$=MID$(STR$(Y)
      ,2)
BC 280 PRINTSPC(1);NU$;TAB(5);
      "(T%){2 SPACES}";
PF 290 SYS 52732,T%,Y,0:IF PEE
      K(251)=255 THEN PRINT"N
      OT FOUND":GOTO 310
BQ 300 PRINT"LOCATION";PEEK(25
      1)
DS 310 PRINTTAB(5);"(IR%) ";
KB 320 SYS 52732,IR%,Y,0
ER 330 PRINT"LOCATION";PEEK(25
      1) :NEXT X:GOSUB590
JS 340 REM
PK 350 REM STRING ARRAYS
BA 360 REM
JE 370 PRINTCHR$(147);CHR$(18)
      ;"STRING ARRAY DEMO:"
KS 380 PRINT" THE TEXT ";CHR$(
      34);"TREASURE-(X)";CHR$
      (34);" IS LOCATED:":POK
      E252,0
RP 390 FOR X=1TO20:IX$=MID$(ST
      R$(X),2):A$="TREASURE"
DE 400 SYS 52965,S$,PEEK(252)
HQ 410 VU$=MID$(STR$(PEEK(252)
      ),2):LN=LEN(VU$): ZE$="
      000"
KK 420 IF LN<3 THEN VU$=LEFT$(
      ZE$,3-LN)+VU$
SH 430 PRINT" ";VU$;:LI$=LI$+V
      U$+" ":NEXT X:GOSUB 590
EF 440 REM
KF 450 REM WORD WRAPPER
QG 460 REM
HR 470 PRINTCHR$(147);CHR$(18)
      ;CHR$(14);"WORD WRAPPER
       DEMO:";CHR$(146);" ";
HJ 480 A$="THIS IS A DEMO FOR
      {SPACE}THE WORD WRAPPIN
      G UTILITY! SEE HOW THE
      {SPACE}WORDS WERE "
AD 490 A$=A$+"AUTOMATICALLY WR
      APPED AROUND THE HEADIN
      G? WORD WRAPPER KEEPS T
      RACK "
HF 500 A$=A$+"OF WHAT POSITION
       ON SCREEN YOU ENABLED
      {SPACE}THIS UTILITY, AL
      LOWING FOR "
PM 510 A$=A$+"MUCH VERSITILITY
      ."
JK 520 SYS 52376,1:PRINT:PRINT
FH 530 A$="ALSO NOTICE HOW WOR
      D WRAPPER JUST PROMPTED
       YOU TO PRESS {RETURN},
       "
EF 540 A$=A$+"THIS ALLOWS YOU
      {SPACE}TO PROMPT THE US
      ER BEFORE CONTINUING WI
      TH A RATHER "
FQ 550 A$=A$+"LENGTHY TEXT."
FF 560 SYS 52376:PRINT:PRINT:G
      OSUB 590:
BP 570 PRINTCHR$(147);CHR$(142
      );"UTILITY/DEMO COMPLET
      E."
QM 580 END
HX 590 PRINTCHR$(18);"CONTINUE
      ?":PRINTCHR$(18);"(Y)ES
       OR (N)O"
EJ 600 GET A$ :IF A$="" THEN 6
      00
DR 610 IF A$<>"N" THEN RETURN
XQ 620 END
```

Eric Jevon Bryant lives in the Bronx, New York City, NY.

DIRECTOR-EASE

By Randy J. Clemmons

I have subscribed to Gazette since 1985, and I always look forward to reading Jim Butterfield's "Machine Language" column. In October 1992, Jim wrote that programming to read a disk directory is no trivial task. I thought you might like to look at my solution for reading directories. Director-Ease (pun intended) makes it possible to create a directory reader easily wherever 102 bytes of RAM are available.

Director-Ease has an option to create a BASIC loader, which allows BASIC programmers an easy path to incorporate a directory reader into their work. Also, if you wish, Director-Ease lets you display only specific file types, such as PRG, SEQ, REL, or USR files.

Entering the Program

Director-Ease is written entirely in BASIC. To help avoid typing errors, enter the program with The Automatic Proofreader; see "Typing Aids" elsewhere in this section. Be sure to save a copy of the program before you try to run it.

Using the Program

Load and run Director-Ease. To use the program for the first time, use the default starting address of 828 which appears on screen. Location 828 is in the 64's cassette buffer, a favorite place for small machine language routines. Respond to the make loader question with *N* for no. You'll then see a message onscreen that tells you to use SYS 828 to view files. Enter that SYS to verify that everything is working well.

Relocation

Let's check out Director-Ease's relocatability feature. Enter *RUN* and repeat the steps outlined in the previous paragraph, but this time pick an address in the 64's upper RAM. For example, enter a starting address between 49152 and 57145. Remember, use a place where 102 bytes of RAM are available.

You could choose a location inside BASIC's normal program space, but you'd have to take steps to protect the code from being overwritten eventually by BASIC. You could do it by moving BASIC's top-of-memory pointer down and then entering a CLR statement to realign BASIC's pointers. Then you could safely use an address above the top-of-memory pointer.

Create a Loader

To use the BASIC loader option, enter *Y* when the programs asks if you want to create one. When the program finishes executing, several lines of BASIC code will appear onscreen. This code is ready to run, save, or renumber.

It's easy to add this BASIC loader code to your programs by using an append routine. If you don't already have an append routine, see Randy Thompson's "Programmer's Page" in the September 1992 Gazette.

What's Happening

As Director-Ease executes, the following events take place. The code is placed in 102 bytes of RAM and then a relocation routine makes adjustments to some machine language instructions (JMP and JSR) inside the code. When the make loader option is selected, the dynamic keyboard technique is employed to create the data statements for BASIC and to delete Director-Ease from memory, leaving only the code for a BASIC program.

Being Selective

Reading either specific (PRG, SEQ, REL, USR) file types or all file types is easy with Director-Ease. The default for Director-Ease is to view all file types, but by entering a few POKEs, you can change the program to read specific file types only. Another default option is to view sequential files only. To switch to this default, enter *POKE (starting address + 9),6*. Then enter the SYS and starting address to view the directory. If there are no sequential files on the disk, only the disk header is printed. To switch back to viewing all files, enter *POKE (starting address + 9),1*.

Before continuing, let's review the decimal equivalents for ASCII characters *P, S, R,* and *U*. The ASCII value for *P* (PRG) is 80, the value for *R* (REL) is 82, the value for *S* (SEQ) is 83, and the value for *U* (USR) is 85. Here's how those values can be used to select specific file types for display.

POKE (starting address + 99),80
to view program (PRG) files only.
POKE (starting address + 99),82
to view relative (REL) files only.
POKE (starting address + 99),83
to view sequential (SEQ) files only.
POKE (starting address + 99),85
to view user (USR) files only.

To enable the option to view specific file types, enter one of the above POKEs and then *POKE (starting address + 9),6*. Then use SYS to get to the starting address to see the directory. To switch back to viewing all file types, enter *POKE (starting address + 9),1*.

Device Numbers

Director-Ease also lets you see directories on devices other than device 8. You can customize the program by entering the following three POKEs plus the device number of your drive or REU. (Note: The selective directory option will not work with Commodore 1700 series REUs. When used with these REUs, the option only lists directories of all file types.)

For directories on devices other than device 8, you'll have to enter three POKEs, each ending with the desired device or drive number. Here are those POKEs.

POKE (starting address + 1), device #
POKE (starting address + 21), device #
POKE (starting address + 89), device #.

Now, when you check a directory by typing *SYS starting address*, you'll get a listing of programs and files on whichever device or drive number that you selected in the above POKEs.

DIRECTOR-EASE

```
BM 0 REM COPYRIGHT 1993 - COMP
     UTE PUBLICATIONS INTL LTD
      - ALL RIGHTS RESERVED
JM 1 GOTO10:REM BY RANDY CLEMM
     ONS
SD 2 S=10{2 SPACES}:REM * DELE
     TE ROUTINE 2-8 *
JR 3 PRINT"{CLR}{2 DOWN}":FORI
     =S TO S+60 STEP10
FD 4 IFI>340THEN NEXT:PRINT"GO
     TO7":GOTO6
HS 5 PRINTI:PRINTCHR$(20):NEXT
     I:PRINT"S="S+60"{LEFT}:GO
     TO3"
GH 6 POKE198,10:FORK=1TO10:POK
     E630+K,13:NEXTK:PRINT"
     {HOME}":END
XK 7 PRINT"{CLR}{2 DOWN}":FORM
     =1TO8:PRINTM:NEXTM:PRINT"
     PRINT CHR$(147):LIST"
SH 8 POKE198,9:FORK=1TO9:POKE6
     30+K,13:NEXTK:PRINT"
     {HOME}":END
PJ 10 GOSUB130:INPUT"{CLR}
      {2 RIGHT}{2 DOWN}CREATE
      {SPACE}BASIC LOADER Y/N"
      ;C$:IFC$="Y"THENGOSUB110
      :GOTO 30
KJ 20 PRINT"{2 RIGHT}{2 DOWN}S
      YS";S:PRINT"{2 RIGHT}
      {2 DOWN}TO VIEW "T$" FIL
      ES":END
JH 30 PRINT"{CLR}{2 DOWN}":FOR
      I=STOS+47STEP6
JJ 40 IFI>FTHENNEXT:PRINT"GOTO
      80":GOTO70
PK 50 PRINTI;"DATA ";:FORJ=0TO
      5:R$=STR$(PEEK(I+J)):PRI
      NTRIGHT$(R$,LEN(R$)-1);"
      ,";
PJ 60 NEXTJ:PRINTCHR$(20):NEXT
      I:PRINT"S="S+48"{LEFT}:F
      ="F"{LEFT}:GOTO 30"
BM 70 POKE198,10:FORK=1TO10:PO
      KE630+K,13:NEXTK:PRINT"
      {HOME}":END
GH 80 SA=(PEEK(679)*256)+PEEK(
      680):EA=SA+101
DE 90 PRINT"{CLR}{3 DOWN}"SA-1
      "FORI="SA"TO"EA":READDA:
      POKEI,DA:NEXTI:END"
MF 100 PRINT"RUN 2":POKE198,2:
       POKE631,13:POKE632,13:P
       RINT"{HOME}":END
EK 110 REM **{3 SPACES}SAVE ST
       ARTING ADDRESS
       {3 SPACES}**
MB 120 TV=S:HB=INT(TV/256):LB=
       INT(TV-(HB*256)):POKE67
       9,HB:POKE680,LB:RETURN
SC 130 PRINT"{CLR}{3 RIGHT}
       {3 DOWN}ENTER STARTING
       {SPACE}ADDRESS NO.
       {3 SPACES}828"
DX 140 INPUT"{30 RIGHT}{UP}";S
DK 150 PRINT"{3 RIGHT}{2 DOWN}
```

```
        POKING DATA TO MEMORY .
        .."
GG 160  REM * DIRECTORY ML DATA
         *
XR 170  FOR I= S TO S+101:READD
        A:POKEI,DA:NEXTI:F=S+10
        1
FC 180  DATA 169,8,170,160,0,32
        ,186,255,169
KS 190  DATA 1,162,154,160,3,32
        ,189,255,32
XS 200  DATA 192,255,162,8,32,1
        98,255,32
QJ 210  DATA 228,255,32,228,255
        ,32,225,255
HC 220  DATA 240,49,32,228,255,
        32,228,255
AH 230  DATA 165,144,208,39,32,
        228,255,141
PD 240  DATA 160,3,32,228,255,1
        74,160,3,32
MG 250  DATA 205,189,169,32,32,
        210,255
JE 260  DATA 32,228,255,240,6,3
        2,210,255,76
DJ 270  DATA 126,3,169,13,32,21
        0,255,76,91
KJ 280  DATA 3,32,204,255,169,8
        ,32,195,255
FD 290  DATA 96,36,48,58,42,61,
        83,0,0
DH 300  REM ** RELOCATION DATA
        {SPACE}**
QR 310  DATA 94,11,13,100,50,51
        ,100,56,57,66,75,76,31,
        83,84
RM 320  REM ** RELOCATION ROUTI
        NE **
EB 330  FORI=1 TO 5:READ DA:TV=
        S+DA:HB=INT(TV/256):LB=
        INT(TV-(HB*256))
SG 340  READLO:POKE(S+LO),LB:RE
        ADHI:POKE(S+HI),HB:NEXT
        I:RETURN
```

Randy Clemmons wrote this program when he found himself needing to relocate a directory reader to make room for other machine language code which required the same memory. He lives in San Diego, California

CROSSREF 128

By Donald G. Klich

Have you ever wanted to make modifications to someone else's BASIC program but were afraid to touch it for fear of reusing a variable name or removing an instruction that may be used as an entry point? Then you need CrossRef 128.

CrossRef 128 will process any 64 or 128 BASIC program and send an alphabetical listing of all variables and all the lines that refer to them to your printer. It also prints a list of all entry-point line numbers with their associated branching line numbers.

With this listing, it's then possible to choose unused variable names or replace original coding lines without the fear of accidentally queering the original program.

Typing It In

CrossRef 128 consists of two program segments, both written in BASIC 7.0. To help avoid typing errors, enter the programs with The Automatic Proofreader; see "Typing Aids" elsewhere in this section. Save the first segment with any name you wish, but be sure to save the second segment with the filename SEG.2 since the first program loads the second by that name. Note that abbreviated commands (upper case characters) are used on line 20 of the first segment. This is necessary so that the required code will fit in the key-definition area.

Creating a List

Load and run the first short program, and make sure your printer is turned on. This program loads the f1 function key area, displays some instructions, and then exits. Now DLOAD the program to be cross-referenced. Finally, be sure that the disk containing SEG.2 is in the drive and press the f1 key.

The amount of execution time required depends on the length and complexity of the target program. Large programs may take 15 minutes or more to process. As a comfort, turn up the volume on your monitor, and you should hear the churning activity.

The printout lists all variables used in the program in alphabetical order and the line numbers in which they appear. Following that is a listing in numerical order of lines that branch to other lines and their destinations. Now you can consult this cross-reference list before making changes or alterations to 64 or 128 BASIC programs without worrying about deleting or overwriting important variables or line numbers.

CROSSREF 128

```
PK 5 REM COPYRIGHT 1993 - COMP
       UTE PUBLICATIONS INTL LTD
         - ALL RIGHTS RESERVED
JS 10  REM LABELS AND CONSTANTS
SB 20  A$= "FAST"+CHR$(13)+"OP2
       ,8,2,"+CHR$(34)+"0:W,S,W
       "+CHR$(34)+CHR$(13)+"CM2
       "+CHR$(13)+"LI"+CHR$(13)
       +"PR2"+CHR$(13)+"CLOSE2"
MF 25  KEYI,A$+CHR$(13)+"RUN"+C
       HR$(34)+"SEG.2"+CHR$(34)
       +CHR$(13)
BQ 30  COLOR0,1:COLOR4,1:PRINT"
       {CLR}{4 DOWN}{YEL}
       {5 SPACES}LOAD THE PROGR
       AM TO BE PROCESSED":PRIN
       T"{10 SPACES}THEN PRESS
       {SPACE}THE F1 KEY"
```

SEG.2

```
MS 10  REM COPYRIGHT 1993 - COM
       PUTE PUBLICATIONS INTL L
       TD - ALL RIGHTS RESERVED
AG 20  REM SEGMENT 2
ED 30  FAST:OPEN2,8,2,"@0:W,S,R
       ":DIMRS$(150),PU$(14),LB
       $(700),LN%(700),ZS$(20),
       NA%(200),BA%(200):B=1:Q=
       1:QT$=CHR$(34):OPEN1,4
SG 40  FORI=1TO150:READRS$(I):N
       EXT
QA 50  FORI=1TO12:READPU$(I):NE
       XT
PF 60  REM ***ROUTINE TO EXTRAC
       T A LINE
CX 70  GET#2,A$:SU=ST
PB 80  IFA$=CHR$(13)THEN100
GC 90  B$=B$+A$:GOTO150
ER 100   IFLEFT$(B$,5)="READY"TH
         ENB$="":GOTO150
MA 110   IFB$=""THEN150
SB 120   I=INSTR(LEFT$(B$,9),"RE
         M"):IFI>0THENB$="":GOTO
         150
PA 130   I=INSTR(LEFT$(B$,10),"D
         ATA"):IFI>0THENB$="":GO
         TO150
XP 140   GOSUB170:B$=""
KH 150   IFSU=0THEN70:ELSECLOSE2
         :GOTO600
CH 160   REM ***ROUTINE TO EXTRA
         CT AN INSTRUCTION
CE 170   C$="":I=INSTR(B$," ",1)
         :N=VAL(LEFT$(B$,I-1)):P
         =I+1:ZP=1
QQ 180   IFP>LEN(B$)THEN250
BD 190   IFMID$(B$,P,1)=":"THEN2
         40
HK 200   IFMID$(B$,P,1)<>QT$THEN
         220
EH 210   P=P+1:IFCS=0THENCS=1:GO
         TO180:ELSECS=0:GOTO180
GC 220   IFCS=0THENC$=C$+MID$(B$
         ,P,1)
AB 230   P=P+1:GOTO180
DB 240   CL=LEN(C$):GOSUB270:P=P
         +1:C$="":GOTO180
MB 250   CL=LEN(C$):GOSUB270:RET
         URN
DC 260   REM ***SUBROUTINE TO EX
```

```
        TRACT CONSTANTS
AQ 270  L=1
PQ 280  HT=0
JM 290  GOSUB520:IFL>0THEN290:E
        LSEL=-L
DC 300  IFL=CL+1THEN470
BR 310  FORT=1TO12:IFMID$(C$,L,
        1)<>PU$(T)THENNEXT:GOTO
        330
FD 320  HT=1:L=L+1:IFL=CL+1THEN
        470:ELSE310
MA 330  IFASC(MID$(C$,L,1))<>34
        THEN380
RP 340  HT=1:L=L+1
FK 350  FORT=LTOL+254:IFASC(MID
        $(C$,T,1))=34THENL=T+1:
        GOTO370
MM 360  NEXT:END
HH 370  IFL=CL+1THEN470
PC 380  IFMID$(C$,L,1)<>CHR$(44
        )ANDMID$(C$,L,1)<>CHR$(
        32)THEN400
GQ 390  L=L+1:HT=1:IFL=CL+1THEN
        470:ELSE380
FB 400  IFASC(MID$(C$,L,1))>57O
        RASC(MID$(C$,L,1))<48TH
        EN430
RH 410  IFSN=1THEN430
GR 420  HT=1:L=L+1:IFL=CL+1THEN
        470ELSE400
GG 430  IFHT=1ANDSN=0THEN280
AA 440  IFHT=1ANDSN=1THEN460
HK 450  SN=1:E$=E$+MID$(C$,L,1)
        :L=L+1:IFL=CL+1THEN470:
        ELSE280
BQ 460  SN=0:GOSUB480:GOTO280
AK 470  IFSN=1THENSN=0:GOSUB480
        :RETURN
SA 480  REM ***ROUTINE TO BLOCK
         DUPLICATE REFERENCES
RF 490  FORI=1TOZP:IFE$=ZS$(I)T
        HENE$="":RETURN
EB 500  NEXT:LN%(B)=N:LB$(B)=E$
        :ZP=ZP+1:ZS$(ZP)=E$:B=B
        +1:E$="":RETURN
KC 510  REM ***SUBROUTINE TO ST
        RIP INSTR
XE 520  FORT=30TO150STEP30:IFMI
        D$(C$,L,LEN(RS$(T)))>RS
        $(T)THENNEXT:GOTO580
CQ 530  T=T-29:FORU=TTOT+29:IFM
        ID$(C$,L,LEN(RS$(U)))>R
        S$(U)THENNEXT:GOTO580
DX 540  IFMID$(C$,L,LEN(RS$(U))
        )<RS$(U)THEN580
JF 550  IFRS$(U)="REM"THENB$=""
        :GOTO150
AD 560  IFRS$(U)="THEN"ORRS$(U)
        ="ELSE"ORRS$(U)="GOTO"O
        RRS$(U)="GOSUB"THENGOSU
        B700
CG 570  L=L+LEN(RS$(U)):HT=1:RE
        TURN
GE 580  L=-L:RETURN
HF 590  REM ***ROUTINE TO PRINT
         OUTPUT
GK 600  PRINT#1,"VARIABLE CROSS
         REFERENCE LIST":PRINT#
        1,"LABELS MAY APPEAR MO
        RE THAN ONCE ON A LINE"
        :PRINT#1
DR 610  A$="ZZZZ":FORI=1TOB-1:I
        FLB$(I)<A$THENA$=LB$(I)
CB 620  NEXT:IFA$="ZZZZ"THEN650
FM 630  PRINT#1,A$;SPC(6-LEN(A$
        ));"-";:FORI=1TOB-1:IFL
        B$(I)=A$THENPRINT#1,LN%
        (I);:LB$(I)="ZZZZ"
BQ 640  NEXT:PRINT#1:GOTO610
QG 650  PRINT#1:PRINT#1,"ACCESS
        ED LINES AND WHERE THEY
         ARE REFERENCED":PRINT#
        1
GJ 660  A%=32767:FORI=1TOQ-1:IF
        BA%(I)<A%THENA%=BA%(I)
MM 670  NEXT:IFA%=32767THENPRIN
        T#1:CLOSE1:SCRATCH"W":S
        LOW:END
AC 680  PRINT#1,A%;" - ";:FORI=
        1TOQ-1:IFBA%(I)=A%THENP
        RINT#1,NA%(I);:BA%(I)=3
        2767
JH 690  NEXT:PRINT#1:GOTO660
AD 700  L$="":FORI=L+LEN(RS$(U)
        )TOCL:A$=MID$(C$,I,1):I
        FASC(A$)<58ANDASC(A$)>4
        7THENL$=L$+A$:NEXT:GOTO
        730
SC 710  IFA$=" "THENNEXT:GOTO73
        0
RC 720  IFA$=","THENNA%(Q)=N:BA
        %(Q)=VAL(L$):L$="":Q=Q+
        1:NEXT:GOTO740
SC 730  IFL$<>""THENNA%(Q)=N:BA
        %(Q)=VAL(L$):Q=Q+1
RF 740  RETURN
DS 750  DATA ABS,AND,APPEND,ASC
        ,ATN,BACKUP,BANK,BEGIN,
        BEND,BLOAD,BOOT,BOX,BSA
        VE,BUMP,CATALOG,CHAR,CH
        R$,CIRCLE,CLOSE,CLR,CMD
        ,COLLECT,COLLISION,COLO
        R,CONCAT,COPY,COS
PB 760  DATA DCLEAR,DCLOSE,DEC,
        DEF,DIM,DIRECTORY,DLOAD
        ,DO,DOPEN,DRAW,DSAVE,DV
        ERIFY,ELSE,END,ENVELOPE
        ,ERR$,EXIT,EXP,FAST,FET
        CH,FILTER,FN,FOR,FRE,GE
        T
AJ 770  DATA GOSUB,GO64,GOTO,GR
        APHIC,GSHAPE,HEADER,HEX
        $,IF,INPUT,INSTR,INT,JO
        Y,KEY,LEFT$,LEN,LET,LIS
        T,LOAD,LOCATE,LOG,LOOP,
        MID$,MOVSPR,NEW,NEXT
AM 780  DATA ON,OPEN,OR,PAINT,P
        EEK,PEN,PI,PLAY,POKE,PO
        S,POT,PRINT,PUDEF,RCLR,
        RDOT,READ,RECORD,REM,RE
        NAME,RESTORE,RESUME,RET
        URN,RGR,RIGHT$,RND,RREG
        ,RSPCOLOR
JF 790  DATA RSPPOS,RSPRITE,RUN
        ,RWINDOW,SAVE,SCALE,SCN
        CLR,SCRATCH,SGN,SIN,SLE
        EP,SLOW,SOUND,SPC,SPRCO
        LOR,SPRDEF,SPRITE,SPRSA
        V,SQR,SSHAPE,STASH,STEP
        ,STOP,STR$,SWAP
PM 800  DATA SYS,TAB,TAN,TEMPO,
        THEN,TO,TRAP,TROFF,TRON
        ,UNTIL,USING,USR,VAL,VE
        RIFY,VOL,WAIT,WHILE,WID
        TH,WINDOW,XOR,ZZZZZ,#,(
        ,),+,-,/,*,↑,;,<,>,=
```

Donald Klich is the author of File Index 128 (April 1993). He lives in Mount Prospect, Illinois.

YOUR OWN DATABASE

By Jim Butterfield

You can sit down at your 64 or 128 and write your own customized database program. The program will be in BASIC, and although it will have a few limitations, it will work nicely. Here's how to go about it. A sample program is included that will run on either a 64 or 128.

Types of Databases

The simplest type of database is called a flat file system. You may think of it as a set of cards in a drawer, in no particular order. To find a given record, you'll have to search through the whole set. That's not too hard to do if your database is of modest size. Our program will use flat files. This means that any new items that we add go at the end. We may also delete or modify existing records.

Indexed files are the next type of database and are a step up in complexity. The records are put in some type of order, such as alphabetical order by name. Although it wouldn't be a great deal of work to change our simple program to an indexed type, we'll stick with the simple flat files.

Relational databases are more complex because records are linked to each other. A school database might link students to classes to teachers to classrooms. These databases are too complex for us to tackle here.

Limitations

Some Commodore 8-bit machines have hard disks, but the most common setup is a single floppy disk drive. That configuration limits the size and

style of databases that can be easily fitted into our system.

Our project will stay with a database that can be read completely into the computer's RAM memory. The process consists of reading in the whole file, viewing or modifying the data, and writing the whole file back to floppy disk.

Commodore BASIC is limited by the INPUT statement (and the related INPUT# statement). If the data coming in contains a comma (,) or colon (:) character, these statements will not behave. Ornate coding can get around these problems. But to keep it simple, we'll forbid the use of these characters in our database.

Typing It In

Database is written entirely in BASIC. To help avoid typing errors, enter it with The Automatic Proofreader; see "Typing Aids" elsewhere in this section. Be sure to save a copy of the program before you exit Proofreader.

The Program

Database is a brief demo program to show you how to go about doing the job. You may want to modify it and expand on its basic form. As it stands in this demo, the database keeps track of members of a small club: last name, first name, date (of birth or membership), and high score.

In planning a database, it's important to list the fields: the data elements that go to make up a record. The sample program has four fields, and I've deliberately chosen three types: string (a name), date, and numeric value. A fourth type, dollars and cents, is also common, but isn't included here.

Decide on a maximum number of records for your database. I've chosen 50 records, but this can be easily changed by modifying the value of N9 in line 110.

The four fields are set up as arrays named A$( ), B$( ), C$( ), and D( ). The dollar symbols used with the first three indicate that these are strings. Field 3, C$( ), is the date, and it could be a number rather than a string. But since we don't do arithmetic on this field, it may be conveniently left as a string value. You may want to total or average field 4, which is the high score, so we'll drop the dollar sign and make this field a numeric one.

Program Modules

In lines 100–350, the program does a little initialization and then reads in the data file. If the data file doesn't exist, the program warns you of this but allows you to continue with an empty database. The program will create this file once you have entered some data.

Lines 400–510 ask for some action. You have the following options: add, delete, change (a record), show (the file), or quit. Some actions are not available if the database is empty or full.

Quit

Lines 520–760 handle the Quit option. If no changes have been made to the file or if the database contains no records, the program quits immediately. Otherwise, the program creates an updated file called MYDATA and writes it to disk. This is done carefully so that a backup file will exist. The sequence scratches the previous backup file; renames the former data file so that it becomes the backup; and, finally, writes the new data file.

Add

Lines 1000–1160 let you add records to the database. Data for the new record is requested. Then this record is displayed, allowing you to accept or cancel it.

Delete

Lines 2000–2210 contain the routine to delete records. You can select the record for deletion. That record is shown in detail so that you can confirm or cancel the delete request.

Change

The routine to change a record is found in lines 3000–3190. You are allowed to select the record to be changed. The record is shown in detail, with the four fields numbered; you are asked to select which field to change. After any change, you may continue to make changes on the same record.

It should be noted that a program can be set up to allow only certain changes. For example, the high score field might be available for modification, but the name fields, once entered, might be nonchangeable.

Display

Lines 4000–4290 display the records on the screen or send them to the printer. The coding assumes a 40-column screen, so the data is squeezed to fit. The screen display pauses from time to time to allow the data to be read. Printer output, in contrast, takes advantage of the wider output area and prints all records without pause.

Specify

The DELETE and CHANGE commands call for a specific record to be selected. This is done in the subroutine between lines 5000–5160. If you don't find the desired record, a no-record-selected value of 0 may be returned.

Date

Lines 8000–8120 handle the date routine: You are prompted for a date in year/month/day order. This is changed to a numeric string within the database. Dates held in this way, 19930214, for example, may be easily searched or sorted if necessary. Note that you may enter the month either as a name or as a number.

Comments

Database programs often contain extra features not shown here. For example, the program might search for selected data or produce totals or averages. Again, there are related programs (report generators) that sort and summarize data, but our simple program is kept to a minimum of steps.

Watch for the limitations of the INPUT statement. We've mentioned the need to stay away from commas and colons. As you expand the features of your database, keep in mind a few other factors. The computer won't like empty fields. If you have fields such as middle initial, apartment number, or date of marriage, be sure to have your program fill unused items with a dummy character such as a slash or asterisk.

Some databases never delete or change a record. If you need a change, a correcting entry is added. That's a valid way to do things, but it may cause the data file to grow to an unreasonable size. You might find it useful to reform your data at intervals, say

at the start of each year. The old database may be kept as an archive; the new one will contain only current information.

The program Database is just a start. Plan your own data, and modify the program to meet your own needs.

DATABASE

```
FC 50 REM COPYRIGHT 1993 - COM
      PUTE PUBLICATIONS - ALL
      {SPACE}RIGHTS RESERVED
AS 100 REM SIMPLE FLAT FILE DA
      TA BASE
XE 110 N9=50
QM 120 DIM A$(N9),B$(N9),C$(N9
      ),D(N9)
QH 130 S$="{17 SPACES}"
JF 140 M9$="..JANFEBMARAPRMAYJ
      UNJULAUGSEPOCTNOVDEC"
KH 200 OPEN 15,8,15
PR 210 OPEN 1,8,3,"0:MYDATA,S,
      R"
AD 220 INPUT#15,E,E$
PF 230 IF E=0 GOTO 300
QM 240 CLOSE 1:CLOSE 15
DA 250 PRINT "I CANNOT FIND FI
      LE 'MYDATA'"
DS 260 INPUT "CONTINUE";X$
SR 270 IF X$="Y" OR X$="YES" G
      OTO 400
MH 280 END
JE 300 N=N+1
DQ 310 INPUT#1,A$(N),B$(N),C$(
      N),D(N)
EF 320 IF ST=0 GOTO 300
FX 330 CLOSE 1:CLOSE 15
SJ 340 PRINT N;"RECORDS FOUND
      {SPACE}ON FILE."
EQ 350 IF N+10>N9 THEN PRINT "
      FILE ALMOST FULL!"
GB 400 IF N>0 THEN PRINT "SHOW
      , ";
RJ 410 IF N<N9 THEN PRINT "ADD
      , ";
BP 420 IF N>0 THEN PRINT "DELE
      TE, CHANGE, ";
QG 430 PRINT "QUIT?"
JS 440 INPUT X$:X$=LEFT$(X$,1)
HX 450 IF N=0 GOTO 490
GG 460 IF X$="S" GOTO 4000
DP 470 IF X$="D" GOTO 2000
RK 480 IF X$="C" GOTO 3000
BF 490 IF N=N9 GOTO 510
QM 500 IF X$="A" GOTO 1000
MH 510 IF X$<>"Q" GOTO 400
QS 520 REM QUIT .. BUT PERHAPS
       WRITE FILE
XJ 530 IF F9=0 OR N=0 THEN END
FP 540 PRINT "READY TO WRITE F
      ILE!"
HX 550 PRINT "PRESS ANY KEY."
EE 560 GET X$,X$,X$
EJ 570 GET X$:IF X$="" GOTO 57
      0
CH 580 REM DELETE ARCHIVE IF P
      RESENT
AB 590 OPEN 15,8,15
RA 600 PRINT#15,"S0:MYDATA.OLD
      "
SJ 610 REM RENAME PREVIOUS DAT
      A FILE
XA 620 PRINT#15,"R0:MYDATA.OLD
      =0:MYDATA"
JQ 630 REM WRITE NEW MASTER FI
      LE
HS 640 OPEN 1,8,3,"0:MYDATA,S,
      W"
HQ 650 INPUT#15,E,E$
FE 660 IF E<>0 THEN PRINT E;E$
      :CLOSE 1:CLOSE 15:END
FE 670 FOR J=1 TO N
XA 680 PRINT#1,A$(J)
HG 690 PRINT#1,B$(J)
QG 700 PRINT#1,C$(J)
JS 710 PRINT#1,D(J)
FD 720 NEXT J
JD 730 INPUT#15,E,E$,E1,E2
XB 740 IF E<>0 THEN PRINT E;E$
DM 750 CLOSE 1:CLOSE 15
FF 760 END
RG 1000 REM ADD A RECORD
EQ 1010 INPUT "LAST NAME";A$(N
      +1)
RR 1020 INPUT "FIRST NAME";B$(
      N+1)
GX 1030 V=N+1:GOSUB 8000 : REM
       GET DATE
HD 1040 INPUT "HIGH SCORE";D(N
      +1)
KX 1050 PRINT
KJ 1060 PRINT A$(N+1)
AS 1070 PRINT B$(N+1)
PX 1080 PRINT C$(N+1)
KE 1090 PRINT D(N+1)
DG 1100 INPUT "IS THE ABOVE OK
      ";X$
QM 1110 IF X$="Y" OR X$="YES"
      {SPACE}GOTO 1140
JX 1120 PRINT ">>> RECORD CANC
      ELLED! <<<"
XA 1130 GOTO 400
XF 1140 N=N+1:F9=1
PE 1150 PRINT N;"RECORDS TOTAL
      ."
GF 1160 GOTO 400
GE 2000 REM DELETE A RECORD
AB 2010 GOSUB 5000
MJ 2020 IF X=0 GOTO 400
FD 2030 PRINT "READY TO DELETE
       RECORD:"
AX 2040 PRINT "{2 SPACES}";A$(
      X)
ER 2050 PRINT "{2 SPACES}";B$(
      X)
FH 2060 M=VAL(MID$(C$(X),5,2))
PB 2065 PRINT "{2 SPACES}";LEF
      T$(C$(X),4);MID$(M9$,M
      *3,3);RIGHT$(C$(X),2)
HG 2070 PRINT "{2 SPACES}";D(X
      )
PC 2080 INPUT "OK TO DELETE";X
      $
QR 2090 IF X$="Y" OR X$="YES"
      {SPACE}GOTO 2120
DA 2100 PRINT "RECORD >>>NOT<<
      < DELETED!"
RX 2110 GOTO 400
PH 2120 N=N-1
AF 2130 IF X>N GOTO 2220
XQ 2140 FOR J=X TO N
JM 2150 A$(J)=A$(J+1)
RJ 2160 B$(J)=B$(J+1)
SS 2170 C$(J)=C$(J+1)
CH 2180 D(J)=D(J+1)
GE 2190 NEXT J
FH 2200 F9=1
FK 2210 PRINT ">>> RECORD DELE
      TED! <<<"
FE 2220 GOTO 400
QB 3000 REM CHANGE A RECORD
GR 3010 GOSUB 5000
BM 3020 IF X=0 GOTO 400
KS 3030 PRINT "READY TO CHANGE
       RECORD:"
SJ 3040 PRINT "1:";A$(X)
HR 3050 PRINT "2:";B$(X)
MF 3060 M=VAL(MID$(C$(X),5,2))
DS 3065 PRINT "3:";LEFT$(C$(X)
      ,4);MID$(M9$,M*3,3);RI
      GHT$(C$(X),2)
EH 3070 PRINT "4:";D(X)
PA 3080 INPUT "CHANGE WHICH FI
      ELD (0=NONE)";J
EP 3090 J=INT(J):IF J<1 OR J>4
       THEN J=0
SG 3100 ON J GOTO 3120,3140,31
      60,3180
JQ 3110 GOTO 400
JG 3120 INPUT "LAST NAME";A$(X
      )
BH 3130 F9=1:GOTO 3040
DP 3140 INPUT "FIRST NAME";B$(
      X)
PG 3150 F9=1:GOTO 3040
JK 3160 V=X:GOSUB 8000 : REM G
      ET DATE
FE 3170 F9=1:GOTO 3040
RP 3180 INPUT "HIGH SCORE";D(X
      )
XP 3190 F9=1:GOTO 3040
JS 4000 REM SHOW - DISPLAY OR
      {SPACE}PRINT
CM 4010 INPUT "SCREEN OR PRINT
      ER";X$
BP 4020 X$=LEFT$(X$,1)
SC 4030 IF X$="P" GOTO 4190
SM 4040 IF X$<>"S" GOTO 4010
KS 4050 J=1
RM 4060 K=J+20:IF K=N THEN K=N
      +1
FC 4070 PRINT LEFT$(A$(J)+S$,1
      3);
GH 4080 PRINT LEFT$(B$(J)+S$,1
      0);
DE 4090 M=VAL(MID$(C$(J),5,2))
EP 4100 PRINT LEFT$(C$(J),4);M
      ID$(M9$,M*3,3);RIGHT$(
      C$(J),2);
```

```
DB 4110 PRINT RIGHT$(S$+STR$(D
        (J)),7)
EQ 4120 J=J+1:IF J<>K GOTO 417
        0
BA 4130 PRINT "{2 SPACES}>> MO
        RE - PRESS ANY KEY <<"
        ;CHR$(145)
BR 4140 GET X$,X$,X$,X$
BM 4150 GET X$:IF X$="" GOTO 4
        150
RF 4160 GOTO 4060
RS 4170 IF J<=N GOTO 4070
HD 4180 GOTO 400
DP 4190 REM SEND TO PRINTER
KQ 4200 OPEN 4,4
BF 4210 FOR J=1 TO N
MB 4220 PRINT#4,LEFT$(A$(J)+S$
        ,20);
BJ 4230 PRINT#4,LEFT$(B$(J)+S$
        ,15);
JQ 4240 M=VAL(MID$(C$(J),5,2))
EE 4250 PRINT#4,LEFT$(C$(J),4)
        ;MID$(M9$,M*3,3);RIGHT
        $(C$(J),2);
AA 4260 PRINT#4,RIGHT$(S$+STR$
        (D(J)),10)
QG 4270 NEXT J
MB 4280 CLOSE 4
MM 4290 GOTO 400
XM 5000 PRINT "SELECT WHICH RE
        CORD:"
JM 5010 J=1
XK 5020 K=J+20:IF K>N THEN K=N
HS 5030 FOR S=J TO K
AF 5040 PRINT RIGHT$(S$+STR$(S
        ),3);": ";LEFT$(A$(S)+
        S$,12);
BR 5050 PRINT LEFT$(B$(S)+S$,6
        );
BG 5060 PRINT LEFT$(C$(S)+S$,8
        );
CR 5070 PRINT RIGHT$(S$+STR$(D
        (S)),7)
QQ 5080 NEXT S
DF 5090 PRINT ">> SELECT RECOR
        D NUMBER, OR 0 ";
SD 5100 IF K<>N THEN PRINT "FO
        R MORE <<"
AH 5110 IF K=N THEN PRINT "TO
        {SPACE}QUIT <<"
SE 5120 INPUT X
JB 5130 X=INT(X):IF X>N THEN X
        =0
SS 5140 J=K
KM 5150 IF X=0 AND K<>N GOTO 5
        020
XB 5160 RETURN
XP 8000 INPUT "YEAR";Y
CM 8010 IF Y<1900 OR Y>9999 GO
        TO 8000
CQ 8020 INPUT "MONTH";M$
FD 8030 M=VAL(M$):IF M>0 AND M
        <13 GOTO 8050
JM 8040 M$=LEFT$(M$+S$,3)
SX 8050 FOR J=1 TO 12
HP 8060 IF M$=MID$(M9$,J*3,3)
        {SPACE}THEN M=J
FG 8070 NEXT J
QQ 8080 IF M<1 OR M>12 GOTO 80
        20
SE 8090 INPUT "DAY";D
MQ 8100 IF D<1 OR D>31 GOTO 80
        90
RK 8110 C$(V)=RIGHT$(STR$(Y),4
        )+RIGHT$(STR$(M+100),2
        )+RIGHT$(STR$(D+100),2
        )
RS 8120 RETURN
```

Jim Butterfield writes "Machine Language," a regular Gazette column. He lives in Toronto, Ontario, Canada.

ML MACROS

By Cameron Kaiser

On the side of my computer I have taped a list of SYS codes: SYS 62913; SYS 49152,X,Y,Z; SYS 64738; SYS 57812"filename",D,S; and so on. The only way I can keep them straight is to use them or write them down; otherwise, I'll forget them.

Unfortunately, I still have problems keeping all those SYS codes straight. This is where ML Macros comes in. All you have to do is remember one SYS code, and with a simple symbol that you define, you can call up an infinite number of ML programs easily and quickly. And ML Macros can prove a boon to programmers by providing an easily customized and handy interface to their programs.

Entering the Program

ML Macros is written in machine language. To enter it, you'll need MLX, our machine language entry program; see "Typing Aids" elsewhere in this issue. When the program prompts for starting and ending addresses, enter the following.

Starting address: CE00
Ending address: CFE7

Be sure to save a copy of the program before exiting MLX.

Running the Program

To install ML Macros, simply load the program with the ,8,1 extension. After it loads, type *SYS 53052*, press Return, and then type *NEW* to clear pointers. If you want to load and run ML Macros from within a BASIC program, add these lines to your program.

```
0 IFA=0 THEN A=1:LOAD "ML MACROS",8,1
1 IFA=1 THEN SYS 53052
```

Of course, you can use whatever line numbers are convenient for you.

Using the Program

To converse with ML Macros, you'll use commands that all begin with the # symbol. In ML Macros the first command defines a macro. To use it, type *#,2, ASC("character"), address. Character* is any symbol not reserved by the computer. Most punctuation marks are acceptable. *Address* is the starting address of the machine language subroutine that will be triggered when you enter the character. For example, #,2,ASC("&"),64738 will program the ampersand to reset the computer if it is entered as a command.

The second command lets you save a set of macros to disk. To use it, type *#,3,"filename",dev,1,* where *filename* is any legal filename and *dev* is the device number of your drive, usually 8. This command also saves the ML Macros code to disk with the macros so that one LOAD command will give you easy access to macros and ML code on all your disks.

The final command reloads a set of macros from disk. To use it, type *#,4,"filename",dev,1. Filename* and *dev* are used the same as they are in the SAVE command.

ML Macros isn't fussy about its input. It's possible to define two macros to the same character; however, only the first macro defined will be honored. This means you cannot erase a macro by defining its character to another ML routine. Should you wish to wipe the macro memory, type *POKE 52992, 3.* Should you wish to erase only the last macro defined, type *POKE 52992, PEEK(52992)–3.*

While ML Macros was designed with compatibility in mind—it doesn't modify IRQ, NMI, or Kernal vectors and occupies a rarely used portion of memory—any modifications to locations 52736–53223 ($CE00–$CFE7) will cause erratic behavior when ML Macros or any macro defined under it is invoked. This doesn't apply to BASIC, but it does apply to some ML routines. Good luck with your ML programs, and I hope ML Macros makes them easier.

ML MACROS

```
CE00:23 0C CF 00 00 00 00 00 2C
CE08:00 00 00 00 00 00 00 00 A5
CE10:00 00 00 00 00 00 00 00 AD
CE18:00 00 00 00 00 00 00 00 B5
CE20:00 00 00 00 00 00 00 00 BD
CE28:00 00 00 00 00 00 00 00 C5
CE30:00 00 00 00 00 00 00 00 CD
CE38:00 00 00 00 00 00 00 00 D5
CE40:00 00 00 00 00 00 00 00 DD
CE48:00 00 00 00 00 00 00 00 E5
CE50:00 00 00 00 00 00 00 00 ED
CE58:00 00 00 00 00 00 00 00 F5
CE60:00 00 00 00 00 00 00 00 FD
CE68:00 00 00 00 00 00 00 00 06
CE70:00 00 00 00 00 00 00 00 0E
CE78:00 00 00 00 00 00 00 00 16
CE80:00 00 00 00 00 00 00 00 1E
CE88:00 00 00 00 00 00 00 00 26
CE90:00 00 00 00 00 00 00 00 2E
CE98:00 00 00 00 00 00 00 00 36
CEA0:00 00 00 00 00 00 00 00 3E
CEA8:00 00 00 00 00 00 00 00 46
CEB0:00 00 00 00 00 00 00 00 4E
CEB8:00 00 00 00 00 00 00 00 56
CEC0:00 00 00 00 00 00 00 00 5E
CEC8:00 00 00 00 00 00 00 00 66
CED0:00 00 00 00 00 00 00 00 6E
CED8:00 00 00 00 00 00 00 00 76
CEE0:00 00 00 00 00 00 00 00 7E
CEE8:00 00 00 00 00 00 00 00 86
CEF0:00 00 00 00 00 00 00 00 8E
CEF8:00 00 4C DB CF 4C 98 CF 8E
CF00:03 00 00 20 FD AE 20 9E AC
CF08:AD 4C 01 B8 20 03 CF C0 AA
CF10:01 F0 0F C0 02 F0 E6 C0 BC
CF18:03 F0 DF C0 04 F0 15 4C D7
CF20:E7 A7 4C AE A7 D0 08 A9 4C
CF28:01 8D 01 CF 4C AE A7 A9 DE
CF30:00 4C 29 CF 20 73 00 4C 20
CF38:8F CF 60 0D A9 47 A0 CF EB
CF40:8D 08 03 8C 09 03 60 20 07
CF48:73 00 A2 00 DD 00 CE F0 73
CF50:2C E8 E8 E8 EC 00 CF D0 C3
CF58:F3 20 79 00 4C E7 A7 8E 09
CF60:02 CF 20 73 00 20 6B CF 57
CF68:4C E7 A7 AE 02 CF E8 BD E6
CF70:00 CE E8 BC 00 CE 85 FE F1
CF78:84 FF 6C FE 00 AD 01 CF 60
CF80:F0 DD 20 73 00 F0 D8 DD 9E
CF88:00 CE F0 D3 4C 82 CF 20 63
CF90:D4 E1 20 A7 F4 4C AE A7 6F
CF98:20 03 CF 98 AE 00 CF 9D 3F
CFA0:00 CE EE 00 CF 20 FD AE 7B
CFA8:20 9E AD 20 F7 B7 A5 14 B5
CFB0:AE 00 CF 9D 00 CE EE 00 94
CFB8:CF A5 15 9D 01 CE EE 00 47
CFC0:CF 4C AE A7 20 D4 E1 A9 6D
CFC8:00 85 FD A9 CE 85 FE A9 58
CFD0:FD A2 E8 A0 CF 20 D8 FF EF
CFD8:4C AE A7 20 73 00 4C C4 3A
CFE0:CF 00 00 00 00 00 00 00 68
```

Cameron Kaiser is a senior at Lutheran High in San Diego. His interests include Pink Floyd, basketball, rock music, and his nine-year-old Commodore. He lives in La Mesa, California. ❐

The Automatic Proofreader helps you type in program listings for the 128 and 64 and prevents nearly every kind of typing mistake.

Type in Proofreader exactly as listed. Because the program can't check itself, be sure to enter each line carefully to avoid typographical errors or other mistakes. Don't omit any lines, even if they contain unusual commands. After you've finished, save a copy of the program before running it.

Next, type *RUN* and press Return. After the program displays the message *Proofreader Active*, you're ready to type in a BASIC program.

Every time you finish typing a line and press Return, Proofreader displays a two-letter checksum in the upper left corner of the screen. Compare this result with the two-letter checksum printed to the left of the line in the program listing. If the letters match, the line probably was typed correctly. If not, check for your mistake and correct the line. Also, be sure not to skip any lines.

Proofreader ignores spaces not enclosed in quotation marks, so you can omit or add spaces between keywords and still see a matching checksum. Spaces inside quotes are almost always significant, so the program pays attention to them.

Proofreader does not accept keyword abbreviations (for example, ? instead of PRINT). If you use abbreviations, you can still check the line by listing it, moving the cursor back to the line, and pressing Return.

If you're using Proofreader on the 128, do not perform any GRAPHIC commands while Proofreader is active. When you perform a command like GRAPHIC 1, the computer moves everything at the start of BASIC program space—including the Proofreader—to another memory area, causing Proofreader to crash. The same thing happens if you run any program with a GRAPHIC command while Proofreader is in memory.

Though Proofreader doesn't interfere with other BASIC operations, it's a good idea to disable it before running another program. To disable it, turn the computer off and then on. A gentler method is to SYS to the computer's built-in reset routine (65341 for the 128, 64738 for the 64).

AUTOMATIC PROOFREADER

```
0 CLR
10 VE=PEEK(772)+256*PEEK(773):
   LO=43:HI=44:PRINT"{CLR}
   {WHT}AUTOMATIC PROOFREADER
   {SPACE}FOR ";
20 IF VE=42364 THEN PRINT "64"
30 IF VE=17165 THEN LO=45:HI=4
   6:WAIT CLR:PRINT"128"
40 SA=(PEEK(LO)+256*PEEK(HI))+
   6:FOR J=SA TO SA+166:READ B
   :POKE J,B:CH=CH+B:NEXT
50 IF CH<>20570 THEN PRINT "*E
   RROR* CHECK TYPING IN DATA
   {SPACE}STATEMENTS":END
60 FOR J=1 TO 5:READ RF,LF,HF:
   RS=SA+RF:HB=INT(RS/256):LB=
   RS-(256*HB)
70 CH=CH+RF+LF+HF:POKE SA+LF,L
   B:POKE SA+HF,HB:NEXT
80 IF CH<>22054 THEN PRINT "*E
   RROR* RELOAD PROGRAM AND CH
   ECK FINAL LINE":END
90 IF VE=17165 THEN POKE SA+14
   ,22:POKE SA+18,23:POKESA+29
   ,224:POKESA+139,224
100 POKE SA+149,PEEK(772):POKE
    SA+150,PEEK(773):PRINT"
    {CLR}PROOFREADER ACTIVE"
110 SYS SA:POKE HI,PEEK(HI)+1:
    POKE (PEEK(LO)+256*PEEK(HI
    ))-1,0:NEW
120 DATA120,169,73,141,4,3,169
    ,3,141,5,3,88,96,165,20,13
    3,167
130 DATA165,21,133,168,169,0,1
    41,0,255,162,31,181,199,15
    7,227
140 DATA3,202,16,248,169,19,32
    ,210,255,169,18,32,210,255
    ,160
150 DATA0,132,180,132,176,136,
    230,180,200,185,0,2,240,46
    ,201
160 DATA34,208,8,72,165,176,73
    ,255,133,176,104,72,201,32
    ,208
170 DATA7,165,176,208,3,104,20
    8,226,104,166,180,24,165,1
    67
180 DATA121,0,2,133,167,165,16
    8,105,0,133,168,202,208,23
    9,240
190 DATA202,165,167,69,168,72,
    41,15,168,185,211,3,32,210
    ,255
200 DATA104,74,74,74,74,168,18
    5,211,3,32,210,255,162,31,
    189
210 DATA227,3,149,199,202,16,2
    48,169,146,32,210,255,76,8
    6,137
220 DATA65,66,67,68,69,70,71,7
    2,74,75,77,80,81,82,83,88
230 DATA 13,2,7,167,31,32,151,
    116,117,151,128,129,167,13
    6,137
```