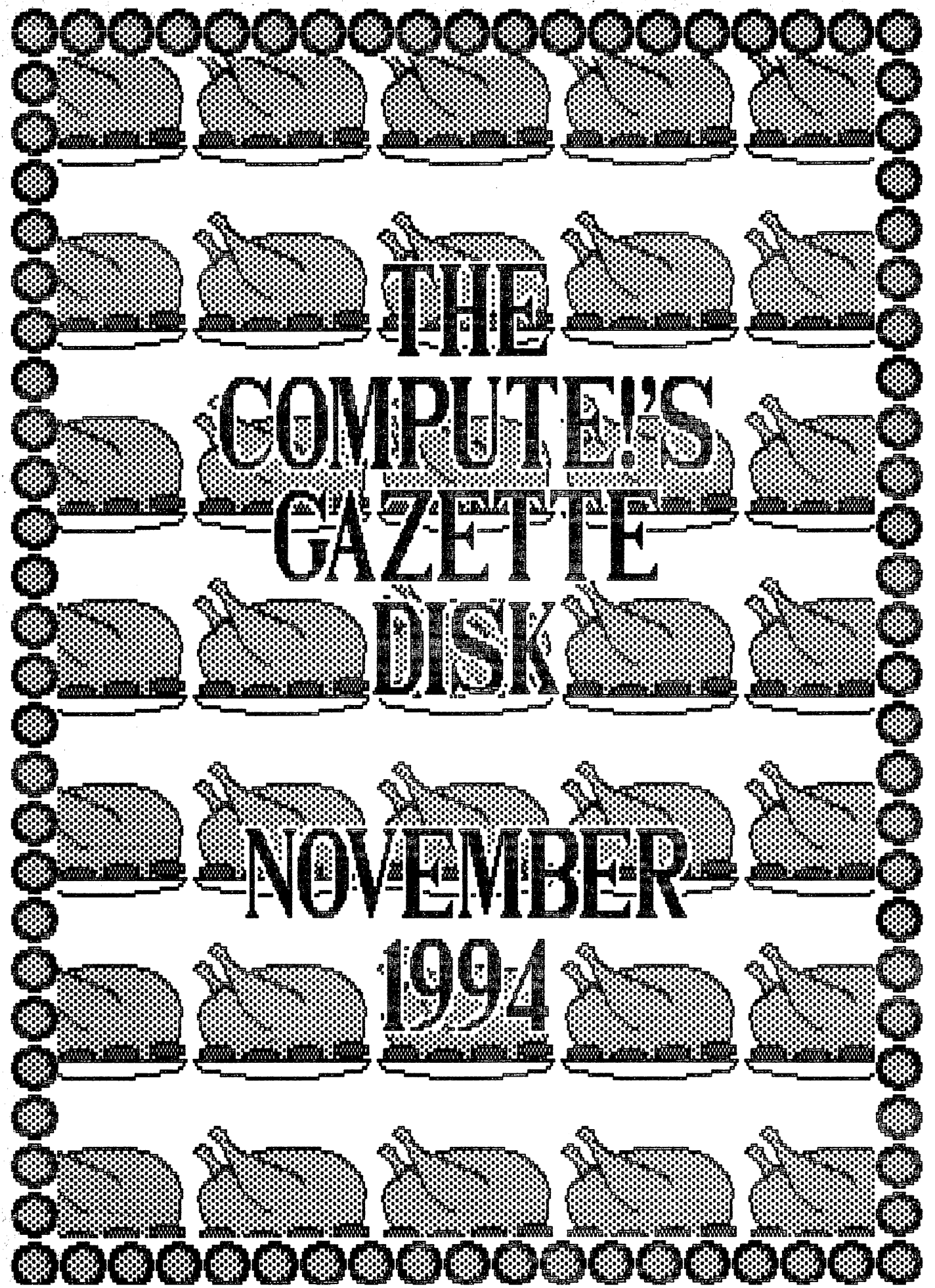
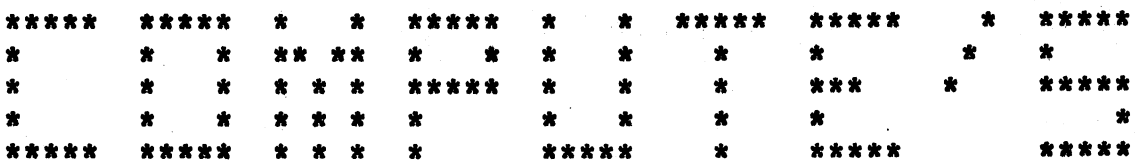

THE
COMPUTE!'S
GAZETTE
DISK
NOVEMBER
1994



G A Z E T T E O N D I S K
==========================

N O V E M B E R 1 9 9 4
========================

(2) = Table Of Contents

Disk Magazine

PROGRAMS:
----

(4) = Muzik Master

(12) = Chute Shootout

(14) = Find-A-Word

(17) = Speed Drive-80

(19) = Catacombs (PD)
Face-Off (PD)
U-Boat (PD)

(20) = Supermon+64 (ML Column)

COLUMNS:
----

(21) = 64/128 View

(23) = D'Iversions

(28) = Beginner BASIC

(32) = Machine Language

(36) = Programmer's Page

(41) = G.E.O.S.

(44) = PD Picks
** Catacombs **
** Face-Off **
** U-Boat **

Compute!'s Gazette Disk
November 1994
Received Disk: November 14, 1994

****

FEATURES:
----

(47) = Making the Most of SpeedScript (by Don Radler)

(54) = REVIEW: 3-In-One Football (reviewed by Don Radler)

Loading Gazette Disk:

C-64 Users, Type: Load "Menu",8,1 and press <Return>.

C-128 Users ----: Enter 128 Mode; then type RUN "128 Menu" and <Return>

****

Feature:

Making the Most of SpeedScript
By Don Radler

Gazette's own word processor has a lot of features built into its small size, but you can customize SpeedScript to make it even more powerful.

Review:

3-in-1 Football
Reviewed by Don Radler

Who cares about a baseball strike when it's football season?

Columns:

64/128 VIEW by Tom Netsel
A couple of old friends close down.

D'IVERSIONS by Fred D'Ignazio
Distance Is a State of Mind

MACHINE LANGUAGE by Jim Butterfield
The MLM and Testing

BEGINNER BASIC by Larry Cotton
Real-Life Algebra

PROGRAMMER'S PAGE by David Pankhurst
Floppy Disks and Drives

GEOS by Steve Vander Ark
Progamming with geoBASIC and Becker BASIC

PD PICKS by Steve Vander Ark
Catacombs, Face Off, U-Boat

128 Programs:

Tonetrainer 128 by Donald Klich
A music utility for the 128 in 40-column mode.

64 Programs:

Muzik Master by Andew Martin
Create music modules for your BASIC programs.

Chute Shootout by Ligia Latina
Drop your parachutists on target before the enemy invades your territory.

Speed Drive-80 by Frank Gordon
This SpeedScript modification gives you two-drive capability plus an 80-column preview mode. SpeedScript required.

Find-A-Word by R. Markland
This word search puzzle offers a number of features, including skill levels, playing against the clock, and playing on paper.

Catacombs (PD)
Search a maze but watch out for the monsters in this arcade-style game.

Face Off (PD)
Play table hockey against the computer or a human opponent. Set skill and speed levels, too.

U-Boat (PD)
Your submarine is hunted in a mine field, but your torpedoes are powerful weapons in this arcade-style game.

Gazette, November 1994

MUZIC MASTER

By Andrew Martin

Create music modules for use in your own BASIC programs.

Muzic Master was designed to help BASIC programmers use music in their programs. The modules created by Muzic Master are complete and need no parameter set up.

Music is created through simple SYS commands. These interrupt-driven programs will allow you to create complex graphical and sound displays. Instead of halting the action while a BASIC routine chirps out some one-voice music, you can combine animation and multivoice music simultaneously! From a text adventure to a shoot-'em-up, any game is improved by adding background music.

GETTING STARTED

After Muzic Master has been booted and is running, the title screen will be displayed. It will take about 45 seconds for the variables to be set. After that, the screen will blank, and the main screen will be displayed.

The main screen consists of various parameter controls, a menu display, and a keyboard at the bottom of the screen. Plug a joystick into port 2 and try moving it from left to right, up and down. What you are doing is adjusting the parameters: (A)ttack, (D)ecay, (S)ustain, (R)elease, (L)ow pulse, (H)igh pulse.

Now try pressing the fire button. This changes the waveform. You don't even have to touch the keyboard to make most parameter changes! Other keys that affect parameters are listed below.

```
KEY          Description
1-3........ Select voice
ADSRLH..... Attack, Decay Sustain, etc.
f2......... Ring Modulation
f4......... Synchronization
@.......... Input Attack/Decay, etc.
*.......... Adjust/Exit filters
(Move joystick to select filter parameters)
 V.. Volume (up & down)
 R.. Resonance (up & down)
 1F-3F.. Filter on/off (fire)
 HP.. High pass (fire)
 LP.. Low pass (fire)
 BP..Band pass (fire)
 FILT..Filter cutoff (up & down)

V.......... Voice options (on/off, starting note number)
$.......... Directory
Run/Stop... Stop song from playing
```

↑..... Input notes
Shift-Q.... Quit program

Note: The following sections of this article require that you have some knowledge of music: sharps, flats, note durations, repeats, key signatures, and so on. If you need information concerning musical theory, I suggest that you look for a book on the subject at your local music store, book store, or library.

INPUTTING NOTES
Before entering notes, find some sheet music. Select something familiar and simple for your first assignment. When you have some sheet music, press the up-arrow key (↑) key, and the input screen will be displayed.

You should see a complete note staff with both treble and bass clefs. You should also see a note on the treble clef. You can move the note by pressing up and down on the joystick. Moving the stick from left to right changes its duration. When you've matched the note with that of your sheet music, press the fire button. You'll then be ready to enter the next note. You have up to 255 notes (0-254) for each voice at your disposal.

TYING NOTES
If you need to enter a note that's tied to another, select the first note and press the fire button. Select the second note, but don't press the fire button. Instead, hit the T key, and a one-note equivalent or an approximation (a note with a rest) will be displayed.

KEY SIGNATURES
Selecting the key of a song is easy. Just press the K key and move the joystick from left to right. Various key signatures will flash by on the treble clef. Find the key signature you need and press the fire button. When you've selected a key signature, the notes will change to sharp or flat automatically. You only have to move the note to the correct line. This can be very handy for key signatures with lots of sharps or flats.

REPEATS
Repeats are used in music to save space. Muzic Master has a repeat function, but it only saves you time, not computer memory. To repeat a phrase, simply go to the beginning of that phrase and press the R key. The note cursor will automatically be set on the very next note after the repeat.

KEY DEFINITIONS
N..... Select note page starting number
V..... Select voice
1-8.... Select note on the note page
£..... Rest key
0..... Natural or Note key
+..... Sharp

-..... Flat
K..... Key signature
R..... Repeat
T..... Tie two notes together
/..... Cut duration in half
L..... One octave lower
H..... One octave higher
X..... Duration total
Z..... Song end
↑..... Exit to main screen

CORRECTIONS
If you enter a wrong note and the note is on the current note page, simply press the correct corresponding key (1-8) and change the note. If the note is not on the current note page, then hit the N key, type the note number (#), and hit Return. A new note page will be displayed starting with the note number you selected. Remember, a note isn't changed until you hit the fire button. You may want to have the song playing while you're putting in the notes. This can become annoying, but it makes catching errors a lot easier.

DISK OPERATIONS
FILENAME
A filename can be from 1-12 characters long. If you have selected the wrong SAVE or LOAD option, inputting a back arrow will return you to the correct menu.

LOAD
Access to the LOAD menu is made from the main menu. There are four types of loading functions, and they are described below.

SONG FILE (SEQ)... Loads a song with parameters and notes. If you're loading a song file with one already in memory, the files will be loaded back to back. Obtain the starting note values from the voice option (V key). If you don't want the two files in memory, then clear all voices before loading.

PROGRAM FILE (PRG)... Loads a self-standing module with parameters, frequencies, and durations (no notes).

PARAMETERS (SEQ)... Loads parameters only.

SONG ONLY FILE (SEQ)... Same as a Song File except no parameters are loaded (parameters remain unchanged).

SAVE... Access to the SAVE menu is made from the main menu. There are three types of save functions, and they are listed below.

SONG FILE (SEQ)... Saves a song with parameters and notes. The save will begin from the starting note values and end at the song end.

PROGRAM FILE (PRG)... Saves a self-standing module with parameters, frequencies, and durations (no notes).

PARAMETERS (SEQ)... Saves parameters only.

LOADING EXAMPLE FILES
A few examples are included on this disk. Below is a list of each file with a description.

DANCE SUITE.... A musical piece written for the harpsichord. Henry Purcell/composer.
File Type: PROGRAM

PRELUDE... 1st Movement (DANCE SUITE)
File Type: SONG

ALMAND... 2nd Movement (DANCE SUITE)
File Type: SONG

CORANT... 3rd Movement (DANCE SUITE)
File Type: SONG

TRIO SONATA... A violin concerto
Arcangelo Corelli/composer.
File Type: PROGRAM

PRELUDIO... 1st Movement (TRIO SONATA)
File Type: SONG

CORRENTE... 2nd Movement (TRIO SONATA)
File Type: SONG

TOREADOR... Georges Bizet/composer. (Continuous play only.)
File Type: SONG

TOREADOR2... Another version. (Continuous play only.)
File Type: SONG

WACHET AUF... Johann Sebastian Bach/composer.
File Type: SONG

PACHELBEL S... Johann Pachelbel/composer.
File Type: SONG

OPTIONS A MENU
Access to the Options A menu is made from the main menu. COPY, CLEAR, SWAP, and OPTIONS B all access other menus. Use the indicated function keys to access most of these options. Each of these menus is described below.

COPY MENU
Copy V1 to V2... Copies all notes from voice 1 to voice 2.
Copy V2 to V3... Copies all notes from voice 2 to voice 3.
Copy V1 to V3... Copies all notes from voice 1 to voice 3.
Copy V3 to V1... Copies all notes from voice 3 to voice 1.

CLEAR MENU
Voice 1..... Erases notes in voice 1 from the starting note value to the last possible note.
Voice 2..... Same as above for voice 2.
Voice 3..... Same as above for voice 3.
All Voices.. Erases all notes for each voice from the starting note values to the last possible note. If you want all notes erased, then use the V key to reset all starting note values to 0.

SWAP MENU
SWAP... This feature allows you to interchange song parts. If this doesn't sound very useful, consider the following example.

```
 song 1                    song 2
V1 000-100,0,102-200,0   53 notes left
V2 000-070,0,072-140,0  103 notes left
V3 000-025,0,027-050,0  203 notes left

SWAP V1&V3

V1 000-100,0,102-125,0  118 notes left
V2 000-070,0,072-140,0  103 notes left
V3 000-025,0,027-135,0  108 notes left
```

After the swap, you have over 100 notes remaining for each voice! You could now load a song with 100 notes in voice 1, where you couldn't before. However, the envelopes of voices 1 and 3 would most likely have to be the same. As with the clear function, the swap is made from the starting note values.

OPTIONS B MENU
TEMPO... Because of the timing of the interrupt and problems with divisions among notes (you can't poke a fraction), there are limitations on the notes available for each tempo.

```
TEMPO........ NOTES NOT AVAILABLE
150/min...... --------------------
113/min...... All Triplets
100/min...... 32nds and dotted 16ths
 90/min...... All Triplets
 75/min...... --------------------
 60/min...... 32nds and dotted 16ths.
```

The tempo denoted at the bottom of the screen reflects the tempo of the next input and may or may not reflect the tempo of the current song.

NOTE-OFF TIME... Each note in music is held for a specific length of time. But there is a length of time in which a note is being played in order to make a group of notes more distinguishable from one another. This length of time is referred to as the note-off time. There are three different settings for the note-off time. The default is 1/30 second. Although this setting will have a profound effect on the articulation of your music, it can't be set for each song in memory.

You can, however, poke the note-off time directly to location BA+153, where BA = 52992.

ECHO... This effect can be very valuable if you have a piece of music with only one or two parts. Copy one part into a voice you're not using. Then select the echo feature for the correct voice. Each time echo is selected, the two voices become more and more discernible. If they become too far apart for your taste, just use the reset function on the same menu. Also, adjusting the tempo will lose the echo effect.

ERROR TRAP
The error trap function will help you find errors of pitch in songs. The prompt will ask you to hit any key when you hear a wrong note. Then you should exit to the input screens to correct your wrong note.

KEYBOARD MENU
This menu will allow you to use your computer keyboard as a synthesizer.

Play Along... Use this function to play along with a song already in memory or just to play the keyboard using one voice.

Polyphonic... This uses all three voices in a rotating type fashion. Set each voice envelope to the same setting for a unison effect.

END GAPS
When song files are loaded back to back, an end gap (one for each voice) is placed between them. This tells the music interrupt where to stop or restart each song. If you want any number of songs to be played one after the other, then you must delete the end gaps. The following is the simplest way to do this.

1. Go to the input screen.
2. Hit the N (note) key, type 000, and press Return.
3. Press the Z (end of song) key.
4. When the end of the song is found, hit the D (delete) key.
5. Repeat 3 and 4 until all end gaps are eliminated for that voice.
6. Repeat 2 through 5 for the remaining voices.

Important: Songs varying in the number of voices must be loaded from the most number of voices to the least.

SETTING UP A PROGRAM FILE
Load each song file you want on your program file. After each load, adjust the tempo (if necessary) to that desired. Then obtain the starting note values from the voice option (V key). Write these values down, because you'll need them later. When all song files have been loaded, check each song by changing the starting note values (V key) and playing them. It's important to mention here that the note-off time and all parameters must be set for all songs to play. In other words, you can't set these effects for each song. If one of the song

files contains the parameters you would like to use, then load these parameters. Otherwise, compromises may have to be made. Just be sure to check each song before saving a program file.

USING SONGS IN YOUR PROGRAMS

First, you must save a program file to the same disk your program is on. You then have two options. You could either load the song file first (LOAD"FILENAME.MZC",8,1), type NEW, and then load your program; or, use the following line as the first line in your program.

```
1 IFA=0 THEN A=1: LOAD"FILENAME.MZC",8,1
```

Use the following statements to play and stop your song(s).

```
BA=52992: REM BASE ADDRESS SYS

SYS BA+202: REM CONTINUOUS PLAY

POKE 781,46: SYS BA+204: REM PLAY SONG ONCE

SYS BA+196: REM STOP SONG
```

When changing songs, use the following lines.

```
POKE BA+32,(Note start value for V1):
POKE BA+34,(Note start value for V2):
POKE BA+36,(Note start value for V3):

SYS BA+224: REM IF INTERRUPT IS ACTIVE OTHERWISE, USE STATEMENTS GIVEN
ABOVE TO PLAY SONGS
```

If some songs use less than three voices and you want to use the remaining voice(s) for sound effects, then use one or more of the following groups of statements.

```
POKE BA+20,18: POKE BA+1,0: POKE 54276, PEEK(BA+235): REM TURN V1 OFF

POKE BA+23,18: POKE BA+3,0: POKE 54283, PEEK(BA+242): REM TURN V2 OFF

POKE BA+26,18: POKE BA+5,0: POKE 54290, PEEK(BA+249): REM TURN V3 OFF


POKE BA+20,1: REM TURN V1 ON
POKE BA+23,3: REM TURN V2 ON
POKE BA+26,5: REM TURN V3 ON
```

You may find PEEKs to following locations useful.

```
(BA+56): VOICE 1 NOTE COUNTER
(BA+65): VOICE 1 DURATION COUNTER

(BA+89): VOICE 2 NOTE COUNTER
```

(BA+98): VOICE 2 DURATION COUNTER

(BA+122): VOICE 3 NOTE COUNTER
(BA+131): VOICE 3 DURATION COUNTER

HELPFUL HINTS
1. If the duration you're looking for is already on the screen, simply press the key (1-8) that corresponds with that note. Then press the number that corresponds with the note you were about to enter.

2. The whole-note and half-note rests that Muzic Master uses are set for common time. If the song you're entering is not in 4/4 time, use the appropriate number of quarter-note and half-note rests.

3. By pressing the / key, the current duration is cut in half. This key is to be used for entering music in cut (2/4) time. It's recommended, however, that you enter the entire song as it appears. When you're sure the song is playing correctly, reenter every note by hitting the / key and pressing the fire button.

4. When entering complex musical works, it's advised that you enter no more than three measures at a time before going on to the next voice. This way, after every three measures, you can check duration totals by pressing the X key.

5. A note that is changed to sharp, flat, or natural at the beginning of a measure stays sharp, flat, or natural to the end of that measure. The reason I mention this is some simple music theory books leave out this important fact.

6. Most classical music pieces are sectioned off in movements. These movements are often played in different tempos. It's recommended that you save each movement as a song file. Then, load them back to back (adjusting the tempo as needed), delete the end gaps, set the starting note values to 0, and resave the song as a program file.

FILE TRANSFER PROGRAM
Also on this disk are two short BASIC and machine language programs that will expand the amount of available song memory. You will still be limited to 255 notes per song. But with this utility, you could have up to four songs in memory with that many notes. To use it, load and run MM.FILE XFER. It will automatically load the machine language routine.

Basically, what this program does is move music program data and the interrupt routine to a protected area of BASIC. In addition to this, the entire 4K section of memory at 49152 is free for a wedge program or any other utility you might have. If you decide to use another utility program, keep in mind that locations 251 and 252 must not be disturbed. Incidentally, the new base location is BA=40704. Each SYS required to access a particular file will be given by the program.

CHUTE SHOOTOUT

[illegible]ember 1984

By Ligia Latina

Chute Shootout is a two-player game for the 64. The object of the game is to fill your enemy's three landing spaces with your parachuting soldiers. The first player to do so manages to invade completely his opponent's territory and wins.

Although it is written in machine language, Chute Shootout loads and runs like a BASIC program. The game begins with each aircraft at the start of its runway. Click on the fire button once to start your engine and accelerate down the runway. When the craft has almost reached the end, move the stick up to lift the nose of the plane to take off. But be careful not to lift the nose too much, or you may go into a stall and crash. If you lift the nose too much, push the stick down to lower it a bit. You maneuver your craft by moving the stick up and down.

Once you have gained enough altitude, you may start releasing your soldiers. (Actually, you can release soldiers at any time after takeoff, but if you do it before your craft has reached sufficient altitude, they will not have enough time to open their parachutes and will crash and die when they hit the ground.)

To release a soldier, move the stick to one of the sides. If you move it to the left, the soldier will jump to the left of the craft, and if you move it to the right, he will jump to the right. Soldiers have to land on their enemy spaces, or they will die when they hit the ground. (If your craft is green, you have to land them on the purple spaces, and vice versa.)

After a soldier has opened his parachute, you can maneuver it slightly by pushing the joystick left or right. This will help you center on target if a parachuter is a little bit off to one side.

SCORING

Besides landing your soldiers, you also have to make sure that your opponent does not land his. To prevent him from doing so, use the fire button to try to shoot down his aircraft and eliminate his parachuters. You get points each time you shoot down his plane and his chutes. You get more points if you manage to hit one of his soldiers before he has opened his chute, but this is much harder to do because they are so small. You also get points for each soldier you manage to land successfully.

There is an infinite supply of bullets, planes, and soldiers, but you give your opponent time to do mischief whenever you crash your aircraft or drop a soldier that misses.

The game ends when one of the two players fills up all three of his opponent's spaces. Press Run/Stop to exit to BASIC or Return if you want to play again.

TRICKS AND TIPS
Immediately after takeoff, your craft speed is so slow that it is very easy to go into a stall. Doing loops makes your craft go faster.

When your opponent drops a soldier, check first to see if he has any chance of reaching the targets. If it is a hopeless miss, do not shoot the soldier down! Your opponent will have to wait until the parachuter hits the ground before he can release another. If you shoot the parachuter down, you are actually helping your opponent.

MISCELLANEOUS
You cannot kill your own soldiers. If you drop a soldier too far from the landing spots, you cannot kill him by firing on him. You have to wait until he hits the ground before you can drop another.

Gazette, November 1994

FIND-A-WORD

By R. Markland

Word-search puzzles are a familiar feature in most variety puzzle magazines. They're known by a number of different names, but the principle remains the same: Find the prescribed words concealed in a matrix of randomly placed letters. Given the ability to select the word list, this puzzle can be more than just a pastime since it gives you the ability to select the word list yourself. A list of spelling words or Civil War battles should also help make learning fun for students.

Find-A-Word retains the traditional aspects of this puzzle genre, but it adds one new feature. Construct a puzzle with 10 to 20 words of your choosing, and then you can print hard copies using virtually any printer. But, then, Find-A-Word adds an entirely new dimension to solving the puzzle.

Here's a feature you won't find in most puzzles of this type. Select any of four skill levels and then solve the puzzle onscreen, playing against the clock.

USING FIND-A-WORD
The program is entirely menu driven and will guide you through the process of constructing, printing, and solving the puzzles.

Begin by deciding how many words you would like to include in your puzzle. A minimum of 10 and a maximum of 20 are allowed.

Next, you'll enter your word list. Normally, all the words are related in some manner, but any word having 5 to 15 characters is acceptable. The program checks for and rejects any duplicate word. If you'd like some variety and an added challenge, try entering numbers instead of words.

It isn't necessary to have someone else enter the words. Prior knowledge of the word list is of no particular benefit. Select one of the four skill levels: Novice allows two minutes per word while Genius allows 30 seconds per word.

The program will sort the word list, create the playing area, and fit your words into a matrix. All words are placed in straight lines, reading forward, backward, up, down, or on either diagonal. They may also cross or overlap each other. On average, it takes about 45 seconds to handle these housekeeping chores.

The puzzle screen consists of the puzzle matrix, with rows and columns numbered from 1 to 20. The maximum time allowed to solve the puzzle is displayed at the lower left, the timer at the lower right, and a cursor at the upper right.

When you locate a word in the matrix, enter the row and column of the first letter. For example: If a word begins in the lower left corner, answer the ROW prompt with 20 and press Return. Answer the COLUMN prompt with 1 and press Return. If you have correctly identified the first letter in any of the words, that word will be highlighted in blue on both the matrix and word list. If you are wrong, you are again given a ROW prompt and must enter a new row and column. To quit the puzzle before finding all the words, enter 99 at the ROW prompt.

You must find all the words in the allotted time to win. Any unfound words will be highlighted in red, at which time you will be prompted to press a key when you are ready to proceed.

At the end of each game you can play again or exit to BASIC. If you elect to play another game, you may use the same words with different placement in the matrix or enter a new word list.

Although its primary appeal lies in competing against the clock, Find-A-Word also allows you to print hard copies of a puzzle with maximum printer/interface capability. Select Generic Printer to use standard PET/ASCII characters. If your printer is capable of elite and expanded print, select either of these options. You can also achieve varied effects by selecting different fonts at print time, if your printer has this capability. If your printer is Epson compatible, the printer driver already on disk will send the proper codes. If not, you can modify the printer driver to customize it for your particular printer.

A printer driver is nothing more than a table defining the specific numeric codes required for a printer to perform various functions. By selecting Modify Printer Driver, you have control over the codes Find-A-Word sends to your printer.

Begin by entering the code your interface requires to place it in Transparent mode with linefeeds. Most interfaces use the secondary address of 4. If you use no interface, try 0.

Consult the chart in your printer manual and enter the codes required to produce the graphic shapes illustrated onscreen.

Find the tables in your printer manual which describe its various functions. Many printers use the Epson codes shown. If this is the case, simply press Return. If not, enter the decimal numbers specific to your printer, making certain to separate two or three number codes with a comma (,) as shown. If the code for your printer is longer or shorter than the one displayed, don't worry, the program automatically takes this into account.

Some printers do not support all of the listed functions. If yours does not, enter a benign code instead. Simply put, this can be any code the program doesn't use and which will not effect printer output. For example, you can use the code to turn underline off (for Epson, Star, and others this is 27,45,0). Because the program doesn't use

underline, a command to turn it off is ignored.

When you are satisfied that your entries are correct, you are asked to enter the name of your printer. If the driver on disk already has that name, you are given the choice of overwriting the existing file or aborting the function.

After your puzzle is printed, you may print the same puzzle again, play the current puzzle onscreen, build a new matrix with the same words, build a new matrix with new words, or quit the program.

If you want a printed answer key, you can easily make one. Print an extra hardcopy and then select Screen Play Current Game. Enter 99 at the ROW prompt and then transfer the puzzle words highlighted on the screen to your printout.

Given enough time, nearly everyone can solve a word-search puzzle. Can you beat the clock?

Gazette, November 1994

SPEED DRIVE-80

By Frank Gordon

A SpeedScript modification that adds two-drive capability, a word counter, and an 80-column preview

SpeedScript has long been a popular word processor with 64 and 128 owners, and it's a program that has undergone a number of changes and revisions. Here's another one that many SpeedScript users should find handy.

With Speed Drive-80 and your copy of SpeedScript, you can create a version of the word processor that lets you load from or save to two drives, count the number of words in a document, and provide an 80-column preview of your document.

This is done by combining a couple of utilities that I wrote a few years ago. Speedram-64 (December 1992 Gazette) enabled SpeedScript to be used with a 1764 RAM expansion unit (REU) or two drives, 8 and 9.

I was able to combine that program with WordCount, which comes on the SpeedScript disk. This revision was called Speedram Count (January 1994). In addition to the two-drive capability, you could count the words in a document by pressing Ctrl-W. I was able to find enough space to combine these two programs in Speedram Count by raising the text area to 9728 ($2600).

This made SpeedScript more useful for many people, but it still lacked some desirable features. I thought it'd be nice to be able to preview my documents before I sent them to the printer. It's annoying to find that a document has a bad break or prints with only one line on the final page. If I could see these problems before printing, I could save time and paper.

SpeedScript does have a preview capability called Instant-80 (December 1987), but it wasn't compatible with my current changes.

That's where this program comes in. With a little tinkering, I managed to work it in. I had to raise the text area further to 10240 ($2800). This now provides room to include the 80-column screen preview, along with the two-drive capability, and the word count.

Drives 8, 9, and the word count can be accessed with Ctrl-Y, Ctrl-N, and Ctrl-W, respectively. To see the Instant-80 preview, press three keys simultaneously, Ctrl-Shift-P.

Unfortunately, Instant-80 is incompatible with the 1764 REU because of a memory conflict above $CF00. You can't use this Speed Drive-80 with an REU, but you can use it with two drives.

CREATING SPEED DRIVE-80

If you already have Speedram Count, you can modify and expand it.

In order to modify your version of SpeedScript, follow these steps.

1. Load and run SpeedScript.
2. Select your favorite background and text colors with Ctrl+B and Ctrl+L.
3. Exit SpeedScript by tapping the Restore key and responding to the prompt by pressing Y.
4. Type POKE44,40: POKE10240,0: NEW and press Return. This will protect SpeedScript's BASIC area.
5. Load but don't run SPDRIVE.CVT with a ,8 extension. It is on the flip side of this disk.
6. Insert a work disk onto which you want to store Speed Drive-80. Now, type RUN and press Return.

Spdrive.cvt will run and save the modified version of SpeedScript to disk with the filename. When the save is finished, turn the computer off and back on before using Speed Drive-80.

USING SPEED DRIVE-80

Load and run Speed Drive-80 like any BASIC program. Your familiar SpeedScript screen should appear in the colors that you selected.

You can test Speed Drive-80 by pressing Ctrl+N (for drive 9) and then Ctrl-4 for a directory. You should get a rapid listing of any files on the disk in drive 9. Press Ctrl-Y (for drive 8), and Ctrl-4 will list programs from that drive. Also, when you press Ctrl-N or Ctrl-Y, the current drive number will appear on the command line. Otherwise it will be the last device accessed. Shift from one drive to the other in this rapid manner to load or save SpeedScript files.

You can then use Ctrl-W for the word count in any of these files.

For an 80-column preview of any document that you have loaded, press Control-Shift-P and press S for the screen option. Your document will appear on the screen just as it would look if it had been printed on paper. Any line spacing commands, centering, headers, or page numbers that you may have embedded will appear. The text will be smaller than normal and somewhat difficult to read, but it should be clear enough for most purposes. To slow the scrolling, press and hold the Control key. At the end of the document, press Return to return to your normal screen.

For more information on how Speed Drive-80 works, see WordCount, Speedram-64, Speedram Count, and Instant-80 in their respective issues. Speed Drive-80 is a combination of all of these programs.

Gazette, November 1994

CATACOMBS, AIR HOCKEY, U-BOAT

The public domain programs on this disk are discussed in Steve Vander Ark's "PD Picks" column.

CATACOMBS
Catacombs is an arcade-style game that will have you scurrying around a maze of brick walls, searching for valuables while trying to avoid some very nasty monsters. It plays with a joystick.

Catacombs comes with instructions that are presented when you select the Program option from the Gazette menu. This program, CATACOMBS INST, then loads the game itself. Should you wish to bypass the instructions later, simply load and run CATACOMBS.

FACE OFF
Face Off is a one- or two-player simulation of the table hockey game in which you use a paddle to sink the puck in you opponent's goal. You can adjust both the game's speed and difficulty.

If the progam appears to lock up while loading data, hit Run/Stop and Restore keys to break out and then type RUN again.

U-BOAT
In this arcade game, you command a submarine that tries to sink as many aircraft carriers an destroyers as possible. Of course, the enemy shoots back and there is also a dangerous mine field through which you have to maneuver. Played with a joystick.

Gazette, November 1994

SUPERMON+64

By Jim Butterfield

In this month's "Machine Language" column, author Jim Butterfield discusses machine language monitors, their commands and their use.

As a convenience to 64 owners who may no already have an MLM (128-owners have one built into their computers), we are presenting a revised version of Jim Butterfield's popular Supermon+64. He included it as a bonus with this month's column.

You'll find the program on the flip side of this disk as SUPERMON+64. Just load it and run it as you would any BASIC program. Of particular interest is the documentation for the program. It also loads and runs like BASIC program. It is this documenation program that loads and runs from the Gazette menu.

Gazette, November 1994

64/128 VIEW

By Tom Netsel

Say goodbye to two old friends.

Just as the October Gazette Disk was going in the mail, we had some disturbing news here at COMPUTE Publications. All the staff members were called into a meeting and told that COMPUTE magazine had been sold to Ziff-Davis Consumer Media Group.

Ziff-Davis bought the assets to our 15-year-old computer magazine, which included its list of 248,000 subscribers. That list of names is what Ziff really wanted. Ziff doesn't plan to continue publishing COMPUTE magazine. The September 1994 issue was he last to be sent to subscribers and newsstands.

Of course, Ziff has an obligation to all those subscribers. Since COMPUTE is defunct, Ziff will offer them a couple of its new publications aimed at the home computer market, Computer Life and FamilyPC. Naturally, Ziff hopes subscribers and advertisers will like what they see and support these new ventures.

Here at COMPUTE, we stopped work on the November and December issues of the magazine. The October issue went to Ziff's trash can, and within a week most of COMPUTE's staff members were looking for new jobs. Such is life in the magazine business.

So how is this going to affect Gazette?
Good question. The joke is that Ziff's pockets weren't deep enough to buy Gazette, so it left us alone.

Gazette was not part of the deal. As editor, I still have a job, which for the past year has been pretty much a one-man show. So I'm now putting the finishing touches on the November issue and pulling together material for December.

Just as Commodore's demise had little affect on us, COMPUTE's expiration shouldn't make many waves in the 8-bit community. If Gazette had still been a 40-page section of COMPUTE, I think itmight have been a different story. Like the concept or not, being on disk is what saved Gazette this time. Like that pink bunny with his drum, Gazette just keeps on going.

Ready for some more news?

I suppose those of you are still have accounts on QuantumLink already know that it's closing its doors as of November first. After a decade of serving the 64 and 128, Q-Link is finally unplugging its modems. America On-Line, Q-Link's parent service, is inviting its Commodore members to transfer their subscriptions to its PC and Mac service.

Unfortunately, you can't access AOL with a Commodore; you need a PC or a Macintosh.

For more than a year COMPUTE has maintained an area on AOL, so I am familiar with that service. In fact, I handle most of the online chores relating to COMPUTE's area. COMPUTE's area on AOL will continue operation although it'll be getting a new look and certainly a new name. Naturally, I hope many of you who have PCs or Macs will consider AOL. I think it's a great service. For more information about AOL, call (800) 827-6364.

While I like AOL, I'll have to admit that it has little to offer 64 and 128 users. If you're going to stay online with your Commodore, you might want to consider GEnie. It has a pretty active Commodore Roundtable and gives you access to 14,000 Commodore files and programs.

Parsec in Salem, Massachusetts, manages the Commodore Roundtable on GEnie. Commodore users who subscribe right away can take advantage of a special offer that will waive the first month's subscription fee and provide ten hours of additional free time online. The usual fee is $8.95 a month, which includes four hours of standard connect time. Each additional hour is $3.00.

If you want to give it a try, here are the setup steps. You don't need special software to access GEnie. You can use any terminal program, but set it for half duplex (local echo), with communication parameters set for 8 bits, no parity, and 1 stop bit (8-N-1).It accepts 300, 1200, or 2400 bps.

Dial 1 (800) 638-8369. In Canada, dial 1 (800) 387-8330.

At the CONNECT prompt, enter HHH.

At the U# prompt, type JOINGENIE and press Return.

At the offer code prompt, type DHE524 to get the special offer. You'll see information about the service, it's rates, and more. You'll also need to have a credit card handy to complete the sign up process. U.S. callers can also pay by check.

Once you are in the system, type COMMODORE at any GEnie menu. That twill take you to the Commodore Roundtable.

So, for those of you who may have heard that Gazette is dead, well, it ain't so, Joe. We're still kicking, thanks for all of you and your support. Now'll I'll get busy putting together another disk next month.

Gazette, November 1994

D'IVERSIONS: Distance Is a State of Mind

By Fred D'Ignazio

When we talk with someone we care about on the telephone, we quickly enter a virtual space of intimacy created from the familiar sound of the person's voice, his or her response to our voice, and the power of our imagination. Our imagination is so powerful that it can weave an imagined real-world "meeting" from the slender thread of two human voices.

We may be having our phone conversation in the kitchen looking out a window at a play set in the backyard, but we no longer see the backyard. Our mind's eye takes over, and we see a constructed image of the person we're talking to. And we see the events, stories, and people who appear like actors on a stage as they're introduced in our conversation. What we are constructing is a virtual play, a radio drama of the 1930s and 1940s, purely from the sounds coming out the ear piece of a small nondescript slab of plastic.

The telephone is popular because it is so easy to enter this virtual space. We can substitute the "shadow" senses of our imagination for our real-world eyes, ears, and bodies during the course of a conversation. During a good conversation, physical distance between you and your friend, your parent, or your child disappears and becomes merely a state of mind. Conversely, if the conversation turns sour, then the physical distance between the two of you may seem trivial compared to the emotional distance which now separates you.

Thinking about telephone conversations helps us form two or three interesting hypotheses about how to create virtual space, virtual meetings, and virtual conversations between two or more human beings:

1. Virtual space does not have to be a complex multimedia experience. Instead, it can be created simply, just by linking up two persons' voices.

2. In virtual space, emotional (or imagination-centered) distance becomes more important than real geographical distance. The illusion of closeness and intimacy with another person can be intellectual, emotional, and even sexual.

3. In virtual space one's normal sense organs (including eyes, ears, sense of taste, smell, touch, and so on) temporarily become disabled if they aren't needed to participate in the virtual experience. This disabling of the external senses is done deliberately by the mind to remove distractions so the mind can focus on believing in the intimacy and reality of the virtual experience.

4. At the same time that the external senses are disabled, internal conceptual senses spring to life. In a phone conversation, for example, we may not be able to see the other person, but we substitute

a remembered or invented image in front of our mind's eye while we are talking with that person. Similarly, if the person talks about an experience involving the senses, we participate vicariously by imagining how a meal tasted, how it felt to climb a steep hill, suffer a headache, or smell the fragrance of a flower. In a good conversation over a telephone, these vicarious experiences can be so powerful that they may be remembered afterwards almost as vividly as if we had lived them ourselves!

5. The experience of virtual intimacy (or tele-intimacy) is experienced by almost everyone who uses a telephone. This indicates that all humans are born with an imagination and (imagination-based) shadow senses to construct a virtual experience out of extremely limited sensory cues.

SNAIL MAIL, DISTANCE, AND SIMULTANEITY

The lowly telephone demonstrates that virtual space and virtual intimacy between two human beings can be created simply and inexpensively. Are there other low-cost forms of telecommunications which also create virtual space for people?

Let's take the case of S-mail, otherwise known as snail mail or regular paper mail. S-mail does not use one of our original, nonsymbolic power senses such as sight, sound, smell, or touch. It is primarily conducted in secondary level symbols--or words--in which humans embed meaning. Unlike the primary senses which sense reality directly (the heat of a fire, for example, or the impact of a hammer on our thumb, the distressing odor of decaying garbage, the horrific sight and sounds of a saber-toothed tiger!), words must first be decoded in order to have meaning.

On the other hand, once we learn to read fluently, the words tend to morph almost instantly into the sights, sounds, and events which they describe. And powerful words can map themselves into vivid real-world experiences through the mental alchemy of our imagination.

WHICH CAME FIRST, VIRTUAL MEETINGS OR WORDS?

In all the eons of our biological past, we communicated face to face--or not at all. That is, two people had to be in the same place at exactly the same time, or communication could not take place. In our recent past, when written symbols and language were invented, we were able to break the lock of simultaneity upon human communications. People separated by gulfs of distance and time could communicate with each other and share intimate experiences, ideas, and emotions.

Letter writing, even with a quill pen or papyrus, has always been a virtual act. In the act of correspondence two people enter an imagined virtual space--a room all their own. In this room they have a conversation. With primitive technology it might have taken months or years for this conversation to unfold, but if the conversation were authentic and meaningful to the participants, then the gaps in time didn't matter. The other person could be called up at will through the correspondent's imagination. Every time he or she read the letter the

other person would return to the room. The person's image would become clear in the mind's eye. His or her voice would come alive in the mind's ear. A strong physical sense of closeness--the other person's presence--would be the reward for using the imagination to reconstruct the virtual room.

AM I HERE WITH YOU NOW?
What I've just described above is the same imaginative illusion that your mind conjures up when you are reading this article. As you read these words where are you? Right this instant! You may be at your computer in any room or office, or you may have printed this article and are reading it on a plane, in a boat, lying in bed, or on the john, but that's only physical reality. Physical reality recedes when your shadow senses go to work and reconstruct a new meeting place inside your mind. What kind of virtual room have you built? What kind of voice do I have when you read these words? Do you imagine what I sound like or look like, or am I disembodied, tumbling in a formless free fall?

In fact, I'm on an airplane, a Beechcraft 1900C (the world's tiniest commercial passenger plane), flying over a farmer's field in south central Illinois. I'm on my way to Quincy, Illinois, on the banks of the mighty Mississippi. I'm wearing my blue shirt with my red Koala tie, and I'm a little punchy after the mad dash I made to the Detroit airport from my hometown in Lansing, Michigan.

Did you guess it right? Do you care? Does it matter what I look like? Of course not. What really matters is that when you read my words (perhaps months after I typed them into my little notebook computer on this tiny plane) you feel like we are having a conversation and we are "someplace" together, talking personally, one on one, about things that matter.

FROM S-MAIL TO E-MAIL
Now, let's return to the present and think about what is already possible. Now that Gazette has set up a virtual "store" in cyberspace, you are able to read these columns regularly just by scanning them on your computer screen.

You can still create a virtual conversation, a virtual room, and a virtual meeting between you and me, all assembled from the glowing text scrolling down your monitor.

What makes our virtual conversation even more realistic is you can now write to me if you have access to the Internet or America Online. (I'm ExplorerOO.) We can begin a real correspondence, not just an imagined one created when you read the words on this page (or screen).

E-mail is actually a pretty powerful generator of virtual reality. All that counts is that both parties write back within a certain time period--say one or two days. If this is done regularly, pretty soon the illusion builds up that you are in a room having a conversation.

This illusion with printed text on a printed (or electronic) page is more like letter writing than carrying on a phone conversation. But it is even more virtual than a phone conversation because the two parties seem to be in the same virtual room no matter where they are in space or time. The "appointment" the two make with each other is not based on a place or hour. The only requirement is a regular conversational "return" (figuratively batting the tennis ball back to the other side of the net) to keep the conversation active and the illusion of a virtual meeting intact.

FROM E-MAIL TO I-MAIL

Now imagine that with the emergence of the information highway, magazine articles like Gazette become instant, realtime, and interactive (hence the term "I-Mail"). Imagine that the moment I write the words "From E-Mail to I-Mail" you see them appearing on your computer screen, no matter where I am and no matter where you are. Imagine too that with I-Mail you have the opportunity to read a paragraph or two of my article and then instantly respond--even while I am still up here flying in my little plane over the Mississippi!

I had an experience similar to I-Mail yesterday when I appeared in a telepanel with fellow multimedia gurus mediated by an editor from Scholastic, Incorporated, on America Online. The moderator and the panelists entered "Scholastic Hall" together, said hello, and immediately began talking about the future of multimedia in education. We watched each other type answers to the moderator's questions, and I became more and more "wired" because I truly felt as if I were in the same room with all these people and in front of a large audience. My heart was beating, and I was short of breath, just like before a real speech or panel. My mouth grew so dry that I downed two soda pops and a bottle of fruit juice in the first ten minutes of the panel.

WELCOME TO VIRTUAL REALITY, FRED!

The cyberroom we entered--purely through text, remember--seemed very real. What made it also virtual (plastic, fantastic, elastic) was that while I was sitting with my fellow panelists at the table responding politely and taking turns, I was also able to clone myself from my panelist body (persona). I was soon running around the room, crawling under the table, passing notes, throwing spitballs, and whispering in the ears of almost everyone in the room--my panelists, the moderator, even the audience.

I was able to do this through simple keystrokes which sent instant private messages to anyone who was in Scholastic Hall. As soon as I realized I could do this, I started windows for everyone in the hall and kept them active around the main window of our panel's conversation. The panel window now became "foreground" mode, and a new "background" mode (the instant messages) was born. I was simultaneously able to participate in conversations in both modes, and I had a ball!

Even while I was making dignified statements about the future of multimedia in the foreground mode, I was slipping around in the

background mode with gossipy little comments, puns, jokes, and prankish remarks. It felt as if I were in a new kind of room--a hive or honeycomb--with several compartments, each with an ongoing conversation, and I could flit like a bumblebee, taking part in all the conversations at once. (For an extreme extrovert like me, it was a moment of pure joy!)

VR ON A SHOESTRING

This column has been a rambling exploration into various low-tech and low-cost forms of virtual reality, including the telephone, normal mail, E-mail, and a new kind of emerging instant, interactive mail which is now taking shape in online conversations, forums, and chat sessions.

The lesson here (if there is a lesson) is that VR, or virtual reality, is not just the stuff of power gloves, VR helmets with stereoscopic 3-D goggles, and multimedia computers. What makes an experience virtual arises out of our most important "cyberorgan" of all: our imagination. If a medium puts us in touch with another human being in a meaningful situation, our imagination seems to kick in automatically and persuasively so that even simple printed or spoken words can act as catapults that launch us into a virtual world.

Gazette, November 1994

BEGINNER BASIC: Real-Life Algebra

By Larry Cotton

You need to borrow a little money, so you call around to a number of financial institutions for various rate quotes. You whip out your 64 money manager program, and the fun begins.

You insert the disk, wait two minutes for the program to load, peruse its very extensive menu until you find a close match to the situation you have, enter the data, wait for some disk activity, correct a few mistakes, answer questions you don't need to answer, wait for some more disk activity, wait again for the computer to do its calculations, and then see the result in a format that's too detailed or too long. All this for just one little calculation?

Perhaps, your son wants you to help him calculate how far a cannonball will land from a cannon when fired at an X-degree angle to the ground and kicked out with Y velocity.

Or you want to lend a bank some of your money (after all, that's what we're really doing when we open an interest-bearing checking or savings account) and would like to see what it'll be worth in a year.

Wouldn't it be nice if you could just sit down and write a few lines of BASIC code to solve these problems?

Some surprisingly sophisticated formulas that are custom-tailored to your needs can be loaded quickly into the computer. You can run these programs repeatedly, changing the data for different calculations.

If you decide that your little BASIC program works well and you think you'll use it again, you can put it on a disk for future use. Pretty soon, you'll have a collection of quick and dirty programs which will do just what you want to do. Best of all, you're the author, so you can modify them to suit any situation.

This month and next we'll see how to convert a few ordinary algebraic formulas into ones the computer can understand.

AREA

Let's start with something simple, like calculating the area of a rectangle, knowing the lengths of its two sides. As you probably know, the formula is Area = Side 1 x Side 2. First ask yourself what information needs to be entered into the formula and what information will be calculated by the computer. In this case, the program should ask you for the length of the two sides. It then should multiply those two figures and print out, in an acceptable format, the rectangle's area.

Let's start writing a program to do this. Use INPUT to get the user's data.

```
10 INPUT "FIRST SIDE";S1
```

Use a valid numeric variable name, such as S1 or X. Get the next side, using a different variable name.

```
20 INPUT "SECOND SIDE";S2
```

All that's left is to multiply the two variables together; that's the part the computer does. No input is required--just multiplication. Use another variable name such as A for the area.

```
30 A = S1 * S2
```

The equal sign is self-explanatory; in computer programs the asterisk (*) replaces the x (times sign) used in the algebra books. All that's left is to print the result:

```
40 PRINT A
```

If the answer contains too many decimals for your liking, you might want to check back to last July's "Beginner BASIC" column on rounding. You could enter the rounding routine on a line between 30 and 40.

```
35 A=INT(A*100)/100
```

If you find it disconcerting to use the same variable name more than once, you can use another, such as B, for the rounded value of A.

```
35 B=INT(A*100)/100
```

Then print B, not A, for your answer.

FUTURE VALUE

Now to tackle something a little more challenging: how to find the future value of a one-time deposit after a certain number of compounding periods. Here's the classic formula.

```
FV = PV(1+i)n
```

FV is what you want to know--the future value of your money. PV is the present value--the amount of your one-time deposit, i is the annual interest rate, and n is the number of compounding periods per year.

The n in this formula is actually a power. It should appear above and slightly to the right of, (1+i). (Some mathematical symbols are tough to write and display with a conventional word processor.) We say that 1+i will be taken to the nth power. (On a com
puter, we usually indicate a number to a power with the up-arrow [↑] symbol.) While it may be somewhat difficult to write, taking numbers or expressions to powers is very easy on the 64 and 128, as we'll see in a minute.

In the past we've talked about My Dear Aunt Sally, who said that when math is performed on a computer (or calculator) it will be in the order of multiplication, division, addition and subtraction (M,D,A,S)--just as you should do on paper. What she didn't say is that expressions in parentheses will be tackled first and then powers taken.

Let's write a short program based on the above formula.

```
10 INPUT"[CLR][DN]ANNUAL INTEREST RATE";I:I=I/100
20 INPUT"[DN]AMT. OF ONE-TIME DEPOSIT";PV
30 PRINT"[DN]COMPOUNDING PERIODS PER YEAR:"
40 INPUT"[DN]1, 2, 4, 12, 365";N
50 T=PV*(1+I/N)↑N
60 FV=INT(T*100)/100
70 PRINT"[DN]FUTURE VALUE = $";FV
```

Lines 10, 20, and 40 get the variables we need to do the calculation. Capital letters should be used for variables, even though the classic formula uses lowercase letters.

In line 10 we reuse the variable I for interest, which should be entered as 6, for example, not .06. It's divided by 100 for the calculation in line 50.

Line 50 contains the main formula. We introduce a new variable T to represent an intermediate step in the calculations--an exact future value. It appears again in line 60.

The asterisk (*) replaces the tacit multiplication sign in the classic formula; the slash (/) is the divide sign. Parentheses are used to force the order in which the calculations are performed; operations within parentheses are always done first. If we took the parentheses out of line 50, for instance, N would have first been taken to the nth power, then 1 and I would have been added, and then that result divided by the result of the first calculation. Instead, 1 and I are added and then divided by N. The results of that calculation are then taken to the nth power.

Use the up-arrow to take numbers or expressions to powers. The expression 4↑2 means 4 taken to the second power (or 4 squared). If you see 32.5↑6, that means 32.5 is taken to the 6th power. Finally, (I+1)↑N means 1 will be added to I, then the result taken to the nth power.

Run the program and enter 6 percent annual interest, $1,000 for the deposit, and compound the money daily (enter 365). Those three variables (I, PV, and N) are used in line 50 to find T. Line 60 rounds T off to two places and line 70 prints the future value of the deposit--$1061.83. Try entering 1 (annually), 2 (semiannually), 4 (quarterly) and 12 (monthly) for the compounding period; you may be surprised what the various periods yield.

You'll find this program, Compounding, ready to run on the flip side of this disk.

Gazette, November 1994

MACHINE LANGUAGE: The MLM and Testing.

By Jim Butterfield

Your computer may have a machine language monitor (MLM) built into it. There's one in every 128 and Plus-4. On the other hand, you may need to load an MLM program into memory and hook it up to your system if you use a 64. There are several public domain MLM programs around; I happen to use SuperMon when I'm working with the 64 or earlier PET computers. Either way, it's important to know how to use this capability.

REGISTER DISPLAY

You might meet the MLM on your 128 by accident. A malfunctioning program might trigger it. One of the first things that appears is a Register Display. It looks something like this.

```
   PC SR AC XR YR SP
;   1002 31 02 03 04 F6
```

PC stands for Program Counter. It tells you where the program was running before the MLM was triggered. You can usually go back from the supplied address by a couple of locations to find the address of the BRK (BReaK, code 00) instruction that zapped you into the MLM.

SR is the Status Register. It shows you the condition of the various flags used by the program. There are eight bits in the value given; all except one match flags used by the program. The matching pattern is NV-BDIZC. While we won't deal with flags in detail in this article, the value given above, hex 31, can be written in binary as 00110001; checking this against the pattern tells us that the B and C flags are on (set) and the others are off (clear).

AC, XR, and YR give the contents of the Accumulator (A Register), the X register, and the Y register.

The SP value shows where the Stack Pointer is set. In the example above, it's at F6, which means that the stack is occupied from hex address 1FF down to 1F7, with 1F6 being the first free stack location.

It's interesting to look at the registers, once you understand their meaning. It's even more interesting to know that you can move the cursor back on the screen and type over the displayed values; pressing Return will set the new values in place. They will take effect when you type G (Go) to run more code.

For example, if you want to turn on the Z flag, just move the cursor over the value 31, type 33, and press Return. You would follow a similar method to change the contents of A, X, or Y. Be careful with the Stack Pointer, though; that's a tricky one to play with. By the way, don't mess with the semicolon character at the beginning of the

data line; the MLM uses it to identify the line type.

MLM COMMANDS
The most important command for the beginner to learn is X (eXit). Normally, that will take you back to the BASIC interpreter, which will respond with READY.

Various MLM packages may have different commands, but the following are fairly universal.

M - display memory. After memory has been displayed, you may go back and type over the data if you wish to change it.

D - disassemble memory. You may supply an address or perhaps two addresses. Disassembly translates the binary data in the computer's memory to mnemonic instruction codes, where possible.

A - assemble to memory. You'd better supply an address with this one. Type in the mnemonic instruction and address, if applicable, and the binary code will be stored into memory. You'll also be prompted for the next A line, but you can ignore that if you wish.

R - display registers (again). Seldom used, since you see the registers when you enter the MLM.

L - load a program into memory. You will need to supply at least a program name and a device number; you may also supply a load address if you wish. For example, L "CAT",08 will bring the program CAT into memory from disk. No BASIC pointers are affected.

S - save a program from specified addresses within memory. As an example, S "DOG",08,2000,2078 will save the contents of hex addresses 2000 to 2077 to file DOG.

G - Go, start running the program. You may supply an address; if you don't, the address in the PC register will be used.

X - As we mentioned, this exits the MLM. You will usually return to BASIC.

SPECIAL COMMANDS
Here are some commands that are common, but not universal. I am naming the handiest ones here.

Disk commands often start with the @ symbol. Used alone, the symbol will give you the disk status; followed by a number, it will reference whatever physical unit you specify. Thus, @9 would give you the status of unit 9.

With or without the unit number, other disk commands can be sent following a comma. Thus, @,$0 will give you a directory of drive 0 on unit 8.

To convert numbers between various bases, type in the number preceded by $ for hexadecimal, + for decimal, or % for binary. Thus, to find the decimal equivalent of hex 64, type $64.

You may use the same prefixes within other commands. For example, you might command type M +8192 +8250 to display memory at decimal addresses 8192 to 8250. Or A 2000 LDA #+100 will assemble a command to Load A with the decimal value 100; the code will be placed at address hexadecimal 2000.

THE BRK COMMAND

When a program encounters a BRK (BReaK) command, whose code is an easy-to-remember 00, it stops running and zaps to a preset address. That address is usually set to be the entry point of the machine language monitor. The MLM will immediately perform a Register Display, and you'll be able to see what the program has been doing up to that point (the breakpoint). You can, of course, investigate further by looking at the contents of memory.

When you are testing a program, it's useful to put one or more BRK instructions at strategic points. That way, you can examine how the program has performed up to that point.

Keep in mind that most MLM packages advance the PC pointer two locations beyond the break. So if you're going to resume operation with command G, you may need to ensure that you're going to the right address.

Similarly, if you have a program that is misbehaving, you might want to replace some instructions with BRK at key points by overwriting op code bytes with 0. When the program stops at that point, replace the original value of the byte, and when ready, go (G) to that location to resume.

Some programs are not easy to stop. Interrupt programs shouldn't have BRK commands inserted, of course. And if your program has switched the input or output streams, try not to do a breakpoint until the normal I/O is restored.

Remember that BRK and the G (Go) command work together. If a program is stopped with BRK, it can be resumed with G. Similarly, if you write some test code and start it with G, the code should end with a BRK instruction. In contrast, a SYS command from BASIC is usually matched with an RTS (ReTurn from Subroutine) instruction at the end of the machine language code.

QUICK TESTS

If you're unsure how a certain instruction or op code works, it's easy to test it using the MLM. Put the code into memory, follow it with a BRK, set up any registers that are needed, and G (Go)! Let's try a simple example of this kind of test.

Chances are, you know that adding values $28 to $28 in decimal mode

will yield $56. But what might happen if we added non-BCD values? Let's try adding $30 to $0F and see what happens.

Crank up your monitor and assemble the following two instructions.

```
A 2000 ADC #$0F
A 2002 BRK
```

Now we'll set up things in the Register Display. Type command R if you don't see this display on your screen. Type over the value shown under the title AC so that it reads 30. Now type over the value under SR so that it reads 08, and then press Return. You have just set the A register to contain hex 30; you have also turned decimal mode on and the carry flag off (hex 08 is 00001000 binary, so flag D is 1 and C is 0).

Command: G 2000. Read the new value under AC as 45. Note that the value under SR is even, so the carry flag is off. It might take more testing for you to figure out the reason for these results, but the test has certainly given you a quick answer.

A BIGGER TEST

If you can figure out what happened on the previous test--or even if you can't--you might be interested in trying out a longer piece of code. Its job is to convert a 4-bit binary value (a nybble) into a printable hex/ASCII digit. Thus, binary 0 will become $30, the ASCII code for 0, and binary 1010 will be transformed to $41, the code for letter A. Using the monitor again, type the following.

```
A 2000 CLC
A 2001 SED
A 2002 ADC #$90
A 2004 ADC #$40
A 2006 CLD
A 2007 BRK
```

This time, the code sets up the Decimal and Carry flags explicitly, so we won't need to adjust SR values. Just pop a value into AC, command G 2000, and examine the resulting value in AC. Clever, huh?

CONCLUSION

It's worthwhile developing skills with a machine language monitor. You'll be able to gain in-depth knowledge of the workings of your machine.

As I said, the 128 has an MLM built in, but we've included a version of SuperMon for 64 users. If your machine doesn't happen to have a built-in MLM and you don't have a loadable program on hand, this disk includes a few versions of SuperMon.

Gazette, November 1994

PROGRAMMER'S PAGE: Floppy Disks and Drives

By David Pankhurst

Perhaps you can remember back to when floppy disks were 8 inches instead of 5-1/4 inches or when single sided, single density was a great deal because nobody ever needed more than that. If you were a Commodore user back then, the VIC-20 was the new computer on the horizon.

It was in those days, a decade or so ago, that disk drives first became popular. One indication of the newness of it all was that when the 64 was released, the default device was tape (which has been a thorn in our side ever since). But with increasing demand, disk drives became more popular and thus cheaper.

Eventually, the tide turned, until now practically no one uses tape files. In fact, by staying with 5-1/4-inch, double density disks, we've slipped from the leading edge of data storage technology to the rear guard. Now, the latest 3-1/2-inch disks are pushing densities of three or four megabytes, compared to the 1541's 1/6th meg!.

Despite not being at the forefront of technology, the 1541 still commands respect. Any programmer using GEOS should be amazed at what can be done with our disk drives; and the add-ons, speed-ups and round-abouts created to communicate with the 64 have formed a small industry, all devoted to one device.

Here's a four-pack of tips for working with that all-important peripheral, the 1541 disk drive.

GOING FOR A SPIN

Cleaning your disk drive is relatively simple. Any cleaning kit you pick up has a fiber disk and cleaning fluid. You put a few drops of liquid on the disk pad, insert it, and run the drive. The problem is getting the disk drive to spin long enough to do the job properly. The following program runs the disk drive's motor for a specific amount of time (variable S on line 120 is the delay in seconds). The file is on the flip side of this disk as SPINDISK.

```
100 REM SPIN THE DISK
110 REM
120 T=TI:S=30:REM S IS SECONDS OF SPIN
130 OPEN 15,8,15
140 REM START DRIVE MOTOR
150 PRINT#15,"M-E"CHR$(126)CHR$(249)
160 IF(TI-T)/60<STHEN160
170 REM STOP DRIVE MOTOR
180 PRINT#15,"M-E"CHR$(232)CHR$(249)
190 CLOSE 15
```

Lines 150 and 180 introduce a rare disk command, M-E, or memory execute. This is the 1541 version of the SYS command. The numbers following are the low and high bytes of the SYS address. Line 150 starts and 180 stops the drive motor; line 160 delays 5 seconds.

SAFE SHIPPING

Ever sent a floppy disk in the mail? If you have, likely you found out what the word "floppy" stands for. I've had disks arrive with the envelope completely folded in half, despite Do Not Fold emblazoned across the package. Since folding tends to make disks difficult to read (not to mention fitting them into the drive), something else is needed to protect them in transit.

Cardboard mailers are a help, as are plastic disk boxes. The disadvantage is cost, either to buy or to mail. One tip I've used for years is to cut two pieces of corrugated cardboard a little larger than the disk's size. I then sandwich the disk between the two pieces, making sure the corrugations of each piece run in different directions. It's this alternating of the corrugations that provides the strength. After taping the corners, I have a lightweight mailer that is surprisingly strong. Now if the postman bends it, it isn't an accident!

SEQUENTIAL BASIC?

Probably everyone at one time or another has listed a program to the printer. But do you ever list a program to disk? Howard Melton, of Saffell, Arizona, fills us in on how to do it. First, load the program and then type the following line in immediate mode.

```
OPEN 2,8,2,"MYFILE,S,W":CMD2:LIST
```

The ,S indicates that a sequential (SEQ) type file is to be created, and the ,W indicates it is to be written to (as opposed to read from). After typing this line, close the file with the following to prevent any errors.

```
PRINT#2,:CLOSE 2
```

Your program is now on disk as a SEQ file. Once the SEQ file is created, it can then be converted back into a runnable BASIC program (PRG), with the following program.

```
10 REM READ SEQ FILE AS INPUT
20 REM END WITH LINE:
30 REM POKE812,47:CLOSE2:SYS65511:END:
40 REM
50 OPEN2,8,2,"MYFILE"
60 POKE 812,73:REM BYPASS KERNEL ROUTINE CALL ($FFE7/65511)
70 POKE 781,2:SYS 65478:END:REM MAKE FILE # 2 THE DEFAULT INPUT & END
```

The program (called SEQ.TO.BASIC on disk) is there basically to illustrate the commands. In fact, you would usually type lines 50-70 as one line in direct mode.

```
OPEN2,8,2,"MYFILE":POKE 812,73:POKE 781,2:SYS 65478
```

What this line does is turn the sequential file into the main input file. Now, instead of looking to the keyboard for text, the computer reads this file one character at a time. This results in the SEQ file being read in exactly as if you had typed it in.

If you do this with a listed BASIC program, the word "READY" at the end of the listing will be entered also, causing an error and returning control to the keyboard. At this point, type the following to return to normal.

```
POKE812,47:CLOSE2:SYS65511
```

This tip doesn't work only with BASIC listings. You can create a file of BASIC commands without line numbers for immediate execution, much as with the batch files of the MS-DOS world. Write your sequential file to disk with this line as the last line of the file to close down the file and return to normal.

```
POKE812,47:CLOSE2:SYS65511:END:
```

INTERROGATION TIME

Programming requires anticipating so much. How many times have you wanted to run a program, only to have to hang up because the disk was write-protected or the drive was turned off? The following routines provide tips on finding out what the drive's status is and can be incorporated into your own programs. They are found in the program called DISK.INQUIRY on the flip side of this disk. It runs from the "Programmer's Page" menu.

```
100 REM  IS DEVICE PRESENT ?
110 REM  P=0 - YES; P<>0 - NO
120 REM
130 OPEN 15,8,15:CLOSE15:P=128ANDST
140 PRINT"DEVICE IS ";:IF P THEN PRINT"NOT ";
150 PRINT"PRESENT"
```

These lines check to determine if the device is present, as a flag is set in the ST variable after opening a file. The variable P is the flag. If the value is 0, then the device is there.

```
200 REM  IS WRITE PROTECT ON ?
210 REM  W=0 - YES; W<>0 - NO
220 REM
```

```
230 OPEN 15,8,15
240 PRINT#15,"M-R"CHR$(0)CHR$(28)
250 GET#15,X$:W=ASC(X$+CHR$(0))AND16
260 CLOSE 15
270 PRINT"WRITE PROTECT IS ";:IF W THEN PRINT"NOT ";
280 PRINT"PRESENT"
```

In the disk drive, a sensor passes a light beam through the spot where the disk is notched. It notes whether a write-protect sticker is blocking its path and stores the result in the disk drive's memory. Using the disk command M-R (for Memory Read), this value is read by the 64.

Because the sensor can't tell what is interrupting its beam, it will assume that a write-protect tag is on the disk anytime that the path is blocked. Since the Gazette Disk comes without this notch (for protection against accidental writing or erasing), the program will report that the tag is on. And if there is no disk at all in the drive, it will dutifully tell you that there is no write-protect label on the disk.

```
300 REM RETURN DISK BLOCKS AVAILABLE
310 REM
320 OPEN 15,8,15
330 PRINT#15,"M-E"CHR$(61)CHR$(198):REM CHECK IF NEW DISK - IF SO,
READ IN BAM
340 PRINT#15,"M-R"CHR$(250)CHR$(2): GET#15,X$:Y=ASC(X$+CHR$(0)):REM
GET LO BYTE
350 PRINT#15,"M-R"CHR$(252)CHR$(2): GET#15,X$:X=ASC(X$+CHR$(0)):REM
GET HI BYTE
360 CLOSE15
370 B=X*256+Y:PRINT B"BLOCKS AVAILABLE"
```

Lines 300-370 check how many blocks are available on the disk. Line 330 executes a disk drive program that determines the blocks available. Only if the information isn't already in the disk drive's memory is the disk read; since this is rarely the case, the program executes rapidly.

```
400 REM PRINT OPEN DEVICE #S
410 REM
420 N=PEEK(152):REM # OPEN DEVICES
430 PRINT N"FILES OPEN":IF N=0 THEN480
440 FOR I=601 TO 600+N
450 PRINT"FILE NUMBER:"PEEK(I);
460 PRINT"DEVICE NUMBER:"PEEK(I+10)
470 NEXT
```

The 64's memory also keeps track of files and devices in use. When executed, lines 400-470 check for currently open files (the total number is stored in location 152) and then lists the open file numbers and their device numbers. The screen isn't listed since it is always open, but every other open file is noted in a table starting at

location 601. Using this table, you could write routines to avoid duplicating file numbers or to check to see if the file limit is about to be reached.

Gazette, November 1994

GEOS: Programming with geoBASIC & Becker BASIC

By Steve Vander Ark

When it comes to GEOS, I am not a programmer. Oh, I read all the manuals, the Official GEOS Programmer's Reference Guide, The Hitchhiker's Guide to GEOS, and the like, and I even understood parts of them. But in order to really do the programming, I'd have to learn how to work with machine language, and that's something I've just never been able to do. I suppose if I'd put enough hours in I could manage to figure it all out--I even bought a book about it once--but frankly, I don't have the time.

I do like to program, though. Trouble is, I only really know how to program in BASIC. I have actually written a few BASIC games here and there. There was a time when I could even program disk operations, believe it or not. But that won't help me with geoProgrammer.

There are a couple of BASIC packages available for GEOS, though. One is called geoBASIC, logically enough; the other is called Becker BASIC. I bought them both once upon a time and have always planned to use them to dig into GEOS programming someday. Well, I decided that someday is now. As I mentioned last month, I'm going to explore what I can do with geoBASIC.

Before I get into the nitty gritty, though, let me describe both geoBASIC and Becker BASIC. I'll also explain how I selected one over the other.

geoBASIC
This package was originally designed by Berkeley Softworks toward the end of the 1980's. It planned to release geoBASIC as part of the GEOS family of products, but as the market for Commodore GEOS products leveled off and Berkeley concentrated more and more on the IBM-compatible version of GEOS, work on geoBASIC halted. Eventually the rights were sold to the folks at RUN Magazine. They gave it a once-over, determined that it was pretty close to ready to go, and created a manual for it. It was released a couple of years ago and the GEOS community cheered. Rumors of geoBASIC had abounded for years and expectations were high.

GeoBASIC hadn't really been completely finished, however. The product RUN was selling was still a little buggy in spots and the documentation was pretty thin. RUN was very up front about the whole thing, since it knew perfectly well that geoBASIC wasn't actually a finished product. GEOS users owe RUN a great deal for just getting the product on the market at all. But there are some problems we'll need to work around if we chose to write our program with geoBASIC.

The strength of geoBASIC is its interface. We'd be writing our program using text mode, which is the mode Commodore BASIC uses. The screen editor, the way the cursor and editing keys work, is the same as

Commodore BASIC. When we want to see the results, we'll just select Run from a drop down menu (or just type it and press return) and watch it go in GEOS's normal high-resolution mode. Other drop down menus offer helpful utilities to create things like dialog boxes and icons. Everything is very interactive, kind of like the BASIC programming we're used to outside of GEOS.

Becker BASIC
Originally written in Germany, Becker BASIC was imported and distributed by Abacus. Becker BASIC, unlike geoBASIC, is a fully functional product. It has an exhaustive manual with plenty of examples and definitions. Becker BASIC is pretty much bug-free.

Like geoBASIC, you write your Becker BASIC programs in text mode. Unlike geoBASIC, however, this package is far from interactive. There are separate modules for writing and for running programs, which mean that every time we want to test our code we have to wait while a new program loads. It's something like the way I used to write programs back in the Dark Ages (the late 1970s) by punching paper tape and feeding it into a terminal. When something went wrong we'd have to go back, debug and rewrite, then punch a whole new tape before we could run it and see if we had it right.

Ok, so it's not quite that slow, but it does make it hard to tweak things a little here and there if every tweak requires a wait while the Testing System loads and runs. It may just be worth it, however. Becker BASIC offers a huge selection of commands with plenty of variations and parameters to create just about anything on the screen.

So which one do we use? The answer to that question is tied up in one simple command that is found in geoBASIC but not in Becker BASIC, the command is MAINLOOP. That simple command makes it possible for the BASIC programmer to create event-driven GEOS programs, just like the machine language ones. That's quite an important difference, let me tell you; it's an entirely different way of writing a program from the way BASIC is usually written. It also happens to be the way that real GEOS programs are written. What that means is that with geoBASIC you are actually writing a GEOS program; with Becker BASIC you're writing a BASIC program that just looks a lot like GEOS.

Unlike a BASIC program which carries out instructions one line at a time, event-driven programs like GEOS don't just run from one line to the next until the last instruction is carried out. Instead, GEOS sets up a "mainloop" which is where the program spends most of its time, waiting and keeping things like the mouse and the screen active.

While in the mainloop, GEOS is also watching for some kind of communication from the user, whether a keypress, a mouseclick, or what have you. When GEOS discovers some kind of message coming its way, it skips out of the mainloop just long enough to make the appropriate response, then goes back into its loop to wait for another message.

Each such message from the user is called an "event," and a program that is designed around events is called "event driven."

A GEOS program written in either machine language or in geoBASIC is event driven. It is arranged around the command MAINLOOP which is where the program will wait for an event. The programmer's job is to tell GEOS what to do when the various possible events take place, such as a mouse click on an icon. Here's the way an event-driven GEOS program is laid out.

```
initial screen and icons drawn
mainloop, waiting for events
list of routines that will happen when each event happens
```

Since we're going to be using geoBASIC to write a real GEOS program, our program will follow that same pattern. Next month I'll show you how it looks.

Gazette, November 1994

PD PICKS: Catacombs, Face Off, U-Boat

By Steve Vander Ark

You know, I've been finding so many great programs lately that I'm going to start putting at least three of them in each of my PD Picks columns. That means a little less introductory jabbering on my part, of course, since I'll have to spend more of my space on program descriptions, but I don't suppose that will lose anyone any sleep.

This month, though, I think it's worth a paragraph or two to talk about the death of an old and cherished friend. November 1st marks the day that QuantumLink, the original and best Commodore telecommunications service, will switch out the lights and lock up the door for the last time. For many of you reading this column, this will bring a pang of sadness; it certainly does for me. QLink was incredible. I doubt that there ever will be an online service like it again.

There certainly won't be one devoted to Commodore, true enough, but also not one which becomes the home of such a close-knit family of users. The computer-using population in this country has changed a lot over the past ten years. Once, the personal computer and the wonderful world of telecommunications were the domain of a select few who dared to brave the complexities and frustrations of an infant technology. Now computers have grown up, becoming more user-friendly, more powerful, and much more plentiful. Along the way, they lost some of the mystique and a lot of the "We're all in this together and isn't it neat" mentality.

GEnie is a good example. I happen to be a great fan of GEnie; I log on every night, without fail. I have a lot of great friends there, people whom I've never met but who I love running into in a chat room. The diversity of the services offered makes my time on the computer very rich and exciting. I honestly never know what I'll find next. But it's a bit of a sterile world. Many users never hang around long enough to get to know anyone else. I suppose that's the nature of computing today; after all, we're all cruising an information superhighway, remember?

QuantumLink was more like a nice, meandering country road with a lot of really friendly people living along it who didn't mind if you dropped by, kicked your shoes off, and set a spell. I'll miss it.

Now for this month's programs:

CATACOMBS
Dungeon exploration is a favorite subject for computer games. There are a number of commercial games that take the concept to its limits. Trouble is, most of them require quite an investment in time before you feel like you're winning the game. One of the best of these programs is Bard's Tale (really three of the best, since there are

three games in the series). I have spent many hours in this exciting world of spells and monsters, building characters, and learning more and more about intriguing world in which they live.

Sometimes, though, I just don't have the time to invest in a game like that, but I still love a good romp around a dungeon, dodging monsters, and accumulating gold. That's where Catacombs fits into things for me. It's quick and easy and keeps me interested without making me think too much. Catacombs is a fast-food dungeon game.

For one thing, it's not really a role playing game at all, like Bard's Tale or Alternate Reality. You don't spend time creating or nurturing a character or trying to figure out some bit of arcane lore. Catacombs is an arcade game, one which sends you scurrying around in a maze of brick walls, avoiding a variety of nasty looking beasties. Even the monsters aren't full-fledged threats. All they do is touch you and you're toasted.

Ok, so it would be nice to be able to fight back once in a while instead of just running away and trying to find the key and then the door to the next level. Sure, I'd like to be able to blast away with a spell or two now and then. This isn't exactly an intricate game. But even so, it's a lot of fun to just run with, trying to see how many levels you can scoot through without seeing the word "CHOMP" flash on the screen as you meet your doom. And it isn't as easy as it looks to avoid the little traps that turn off all the lights and leave you with nothing but a feeble lantern and quick joystick reflexes between you and some weird enemy.

FACE OFF

I don't suppose this game needs much introduction. Face Off is a paddle game which pits you against either a friend or the computer as you try to ram a disk into each other's goal. That's pretty much all there is to it.

What makes this computer version of air hockey stand out is how well it simulates a real game, down to the sizzling speed of the disk on the table. As in a real game, you can corner the disk and then slam it with ferocious abandon toward the other end of the board or try to smack it off a wall or corner to send it right past your opponent's paddle.

You can even adjust the speed of the game, from a low setting that simulates a game with my 6-year-old up to a high setting that reminds me a the last game I played with my friend Steve who sends the disk into warp speed and gleefully watches me flap around at empty air. The computer is actually a very competent player, too, which makes this game a winner on all counts.

U-BOAT

I received this program on a disk from a reader named Frank Gordon who wrote a note telling me that this was one of his favorite PD games. I booted it up and gave it a try.

At first, I was only marginally impressed. U-Boat is a sub-versus-ship game, where each tries to blast the other with missiles. Of course, the sub's missiles are called torpedoes and those of the carriers and destroyers are called depth charges, but essentially they're all just missiles. Littered across the underwater playing field are mines that will blow up your sub if it touches one. That's really all there is to it.

Or so I thought. There are a lot of little touches that make this game just a little more fun, a little harder to beat, and a little more addicting than I first thought. For one thing, all those mines make quite an interesting playing field; they can serve as a nice hiding place, since they'll knock out enemy depth charges just as easily as they will your own sub. They also make navigation a bit trickier. Also, when the depth charges fall, they automatically explode when they reach your depth, which can be a dangerous thing if you're too close. Each explosion actually extends outward in all directions, which means a near miss is as good as a hit from those nasty things.

Of course, as the game proceeds the torpedoes and depth charges clear the playing field of mines, which takes a lot of the fun out of the game. There doesn't seem to be any way to replenish the supply, and once they're gone the game becomes a simple one of timing your shots to hit the ships as they pass overhead.

Even so, U-Boat is a lot of fun. It won't keep you riveted, maybe, but it will give you one more interesting way to spend time on your Commodore. I keep loading it up and giving it one more try, and I think you will too.

Gazette, November 1994

MAKING THE MOST OF SPEEDSCRIPT

By Don Radler

I've been using SpeedScript just about every day for quite a few years now, writing articles, business correspondence and personal letters. I also use it on a work disk to make notes to myself and for documentation on numerous programs. I've experimented with commercial word processing programs, including the Swift word processor from Celery Software, WordPro3+, and the ever popular Easy Script, but SpeedScript remains my favorite. By now, the sequential files created with some other word processors I once used have all been converted to the SpeedScript screen-code program format and cleaned up to perform as they should.

It's possible that part of what I've learned about SpeedScript could be of value to some of the many Gazette readers who also use it, so here's a quick rundown that includes a few tips based on experience; feel free to adopt any of this that might be helpful to you.

SPEEDSCRIPT 3.2

I started with SpeedScript 3.0, which was published in Compute! magazine in early 1985. Later, I upgraded to SpeedScript 3.2 by eliminating some minor bugs with the POKEs suggested by Charles Brannon in the May 1987 Gazette. His detailed and elegant article in that issue remains the definitive documentation for his deservedly most popular Gazette program.

INSTANT 80

When Glen Mackinnon's Instant 80 appeared in the December 1987 Gazette, I latched on to that. It permitted me to see, onscreen, an 80-column, almost WYSIWYG version of my finished document. This feature was well worth typing in just one column of machine language. I changed the preview screen's colors to suit me with a single POKE and then saved the modified version of SpeedScript with the filename SS/80.

EASY CURSOR

There were still things about the program that didn't completely please me, including the rambunctious up-down cursor that didn't seem willing to go where I wanted it to. Then, in November 1989, Gazette published Larry Smith's SpeedScript Easy Cursor. It promised to make the up-down cursor key more docile and also to get around the 1541 disk drive's infamous SAVE-WITH-REPLACE bug by scratching an old file first and then saving its same-name replacement.

In SpeedScript, the up-down cursor key moves the cursor to the next or the previous sentence, respectively, duplicating what the f3 and f4 function keys do. With Easy Cursor, the cursor key moves the cursor up or down one screen line, pretty much the way it operates within the BASIC screen editor. It's not completely predictable, but it's much better behaved than before.

Using it in conjunction with the right-left cursor key yields quite satisfactory control. And, if you want to move forward or back one sentence, f3 and f4 still accomplish this for you. Easy Cursor was compatible with SS/80, and it only required me to type in half a column of machine language. The result was a file I call SS/80/EC, the modified version of SpeedScript that I use and can recommend to anyone.

TAPE OR DISK?

I had to wait until May 1990 to get rid of SpeedScript's TAPE OR DISK? question and have the storage device default be to the disk. That's when a Vermont reader sent the five POKEs that solved that minor but constant annoyance. With this modification, SpeedScript became not only one of the most useful tools I have but also one of the most pleasant to work with. (Editor's note: Here's that tip from Carl. E. Snyder of Cavendish, Vermont. Load but don't run your copy of SpeedScript 3.2 and then type POKE 4904,162: POKE 4905,8: POKE 4906,76 :POKE 4907,61: POKE 4908, 19. Now resave the modified version of SpeedScript with a new name.)

If you want to modify your version of SpeedScript as I did, I think you'll like the result. But if you use any version of SpeedScript, the suggestions below will make its use more efficient and more pleasant; none are specific to the version that I use.

FIRST ON DISK

SpeedScript is fairly small, so I keep a copy of it at the head of each disk that stores my word processor files, including one disk each for letters (LTRS), articles (ARTC), work in progress (WORK), and miscellaneous quotes and such that might be useful in the future (MISC). With the Epyx FastLoad cartridge that I use, I simply hold down the Commodore and Run/Stop keys, and SpeedScript is up and running in a couple of seconds. (If you don't use such a cartridge, you can type the command, LOAD"*",8, and you're ready to go once you type RUN and hit Return.)

INSERTING & DELETING

As a SpeedScript user, you probably use the Inst/Del key just as you do in BASIC. And you surely know that to delete more than one letter, you only have to hold the Inst/Del key down and wipe out letters from right to left. To insert a single letter, hit Shift-Inst/Del. For two letters, press this combination twice.

To insert more than one or two letters, most users use the insert mode. Press Ctrl-I and type as many letters as you need. Then hit Ctrl-I again to get out of insert mode and get ready to continue typing or to type over existing characters. Of course, many people prefer to work in insert mode. That's a matter of personal preference.

Inst/Del and Ctrl-I work well in a short document or near the end of a longer one, but they're slow and awkward when there's a lot of text

after the cursor position. The process becomes slow because insert mode must push all that text forward to make room for the insertion. It must drag it back again to close up the space following a deletion.

To quickly insert a short word, you can hit Run/Stop; this instantly opens up five spaces. (Hit the same key twice, and you open up ten spaces.) The trouble here is that the cursor moves to the end of the newly inserted spaces and must be brought back five or ten spaces before you can insert letters.

A FASTER WAY

But, to make room for a sentence or so, hit Shift-Run/Stop and get 255 spaces in the blink of an eye, all of them ahead of the cursor. Then enter your text, making sure that you're not in insert mode. To close up any remaining space after you've added or changed text within this 255 spaces, press Shift-Ctrl-back arrow, and the text that was pushed ahead will be instantly returned to the cursor position.

I've found that the Shift-Run/Stop combination works perfectly well for any inserts, even short ones, and the Shift-Ctrl-back arrow combination lines everything up again quite well. Most importantly, these commands insert multiple characters or remove multiple spaces as quickly as any other command can deal with a single space.

If you use these combinations all the time, your fingers fall on those keys virtually automatically. Try it for a while; I think you'll like it. You may decide to use the Ctrl-I command less and to use the shifted Inst/Del key only for inserting one or two spaces.

ERASING BEATS DELETING

Another command you might seldom use is Ctrl-D, which deletes text from the bottom up. It's useful if you just wrote a sentence and decide to delete the whole thing, but it only allows the deletion of one word, sentence, or paragraph at a time. It does this because multiple deletions would store text in reverse order, garbling the meaning if you ever attempted to restore it. Ctrl-E, on the other hand, works from the top down, and deletes words, sentences, or paragraphs up to the 12K maximum size of the buffer, much longer than the average letter or short document.

CUT & PASTE

The ERASE command allows you to use Ctrl-E as a cut-and-paste utility: Just erase the words, sentences, or paragraphs you want to move and then restore them at another location with Ctrl-R. (If you want to copy a block of text rather than just move it, erase it with Ctrl-E, restore it immediately with Ctrl-R, and then move your cursor to where you want the copy to appear and hit Ctrl-R again. Actually, you can "stamp" what's in the buffer as many times as you like.)

A VIRTUAL TAB

Such repeatability gives you the potential for creating charts and tables even though SpeedScript is devoid of any tab function. Try

this: Use Run/Stop twice to indent ten spaces and then type "@." Repeat a half-dozen times. This gives you a line with six open areas separated by "@" signs. Use Ctrl-E to erase the line and then Ctrl-R to restore it as many times as you need to create your table. When you go to fill in the blanks, hit Shift-Ctrl-H to get into "hunt" mode and answer the prompt with "@", followed by Ctrl-H to activate the search. The cursor will scroll to the first "@" in your table. Enter your letters or numbers and hit f1 (or Ctrl-H again) to move to the next "@", and so on. In effect, you have created a custom-made tab function just for this purpose. Then go back to the top of your document and invoke global SEARCH AND REPLACE with Ctrl-G and replace the "@" with any character, including a space. Your table is done! (Actually, you could use any character other than "@", but you should use something that is otherwise not used in your document so that it can be replaced by another character, or a space, without making unplanned replacements within your text.)

CTRL-G SHORTHAND

Speaking of the global SEARCH AND REPLACE function, I put the command to good use writing this article. Knowing that I'd be using the word "SpeedScript" many times, I typed "*" instead of "SpeedScript" in my first draft. Then I went to the head of the document and used global SEARCH AND REPLACE (Ctrl-G) to switch each "*" to the word "SpeedScript." That saved considerable typing for this less-than-perfect typist. You might find a similar operation valuable in some of your own work.

Be careful with Ctrl-G, however. You can back out of most commands merely by hitting Return, but if you try this with Ctrl-G after you've identified the copy to be replaced, it will race through your document eliminating that copy. Ctrl-G reads your lack of response as a null character and ruthlessly zaps anything that was stipulated as the copy to be hunted for and replaced.

SETTINGS & HEADERS

Right after SS/80/EC on my word processor disks come 1CUSTOM SETTINGS and 2HEADER, which I can load into SpeedScript with wildcards by simply typing "1*" or "2*". The settings file controls type size and style. I won't go into printer commands in this article, since they differ so much from one printer to another. But if you create a file with your own frequently used printer commands, they'll be loaded and ready to use if you save them in such a file. The header file is actually just a sample, set up with the embedded "#" which tells SpeedScript to print page numbers as part of the header.

Starting a new document, I'll usually load 1* and 2* at the beginning, but if I forget, I can always append them to the end of my document. Header and printer commands aren't of any use at the end of a document, so I erase them (Ctrl-E) and restore (Ctrl-R) them at the head of the document. (Of course, if I'm doing only a short letter that won't need a header or special printer commands, I'll simply start typing.)

TRANSFERRING TEXT
In a like manner, if I want to insert a paragraph or so from my MISC disk, I just place my cursor at the end of the current document and load the file. I then erase it and restore it in the desired location. Alternatively, I can load a document from which I want to use a small part and erase that part with Ctrl-E, which places it in the buffer. I then use Shift-Clr/Home to wipe out the rest of the document. Then I can load the draft on which I'm working, and with Ctrl-R place the data from the buffer in any location.

Both of these techniques work so well that I prefer them to using Larry Hagney's SpeedSwap program that was published in the September 1991 Gazette. SpeedSwap is a short but powerful BASIC program that lets you have two SpeedScript documents open at the same time. Unfortunately, it's not compatible with SS/80/EC, which has me so spoiled that I almost can't work without the modified cursor or an 80-column preview.

NAMING DRAFTS
I've read articles about word processing suggesting that you name various versions of a document as DRAFT1, DRAFT2, and so on. If you do that, you'll have to type out the full name in order to load the one you want. Reversing the standard advice, I name current work 3DRAFT, which lets me load it simply as "3*", using the asterisk as a wildcard. Since the SAVE WITH REPLACE command in SS/80/EC is bug-free, I use it all the time and don't keep multiple versions of any document on disk.

This file for instance, was saved and replaced repeatedly as 3WORD, while another article I was working on sat on disk as 4HINT.

When an article is completed, I store it on my ARTC disk with its actual title, or as much of the title as will fit in 16 letters. Once it appears in print, I scratch it from the disk; that way, all of my word processor directories are short enough to appear onscreen in their entirety when I hit Ctrl-4.

LETTER FORMATS
I've also read articles that suggest setting up fairly elaborate formats for letters, using graphics characters to start and end paragraphs and leaving blanks to be filled in with new text as needed. But I've found that I write to the same people again and again, and the letters tend to be about the same length each time. So I just store on disk my most recent letter to, say, Gazette editor Tom Netsel, and then load that when I'm about to write him again. Leaving my return address and his address as they are, I strike over the old date with a new one and then type over the old paragraphs with new ones. When the letter is finished, I use SAVE:@:NETSEL to replace the old one with the new one. In this way, the letter on the disk is always the most recent one.

IN BRIEF
Here are some brief comments on SpeedScript's keyboard commands:

Ctrl-A Changes case; can be held down to affect many characters

Ctrl-B Changes background color; changed version can be saved

Ctrl-D Deletes (sentence, word, paragraph); Ctrl-E works better

Ctrl-E Erases (S,W,P) and overwrites buffer; works for cut and paste

Shift-Ctrl-E Erases but doesn't overwrite buffer

Ctrl-G Global SEARCH AND REPLACE; BE CAREFUL with null character!

Ctrl-H Searches (hunts) for phrase

Shift-Ctrl-H Selects phrase to be searched for

Ctrl-I Toggles insert mode; can be too slow with longer documents

Ctrl-J Replaces phrase one at a time as it appears in text

Ctrl-K Kills buffer; required if erasures reach 12,000 characters

Ctrl-L Changes letter (text) color; changed version can be saved

Ctrl-P Prints; if this fails, use Shift-Ctrl-P and answer prompts

Ctrl-R Restores buffer; copies erased copy anywhere you like

Ctrl-V Verifies (for tape storage only)

Ctrl-X Transposes characters; strikeover is easier

Ctrl-Z Moves cursor to end of document

Ctrl-= Displays, in command line, amount of free memory

Clr/Home Hit once for top of screen; hold down for top of document

Shift-Clr/Home Erases document; BE CAREFUL!

Run/Stop Indents cursor five spaces; meant as paragraph indent

Shift-Run/Stop Clears 255 spaces ahead of cursor position for insertion

Ctrl-Shift-Back Arrow Deletes all spaces after cursor position

Inst/Del Deletes character; repeats from right to left

Shift-Inst/Del Inserts a space; also repeats if held down

PRINTING

SpeedScript was designed as a generic word processor that could be used with many different printers. I work with an Okimate 20 and have programmed my printkeys to send signals to that machine. Since printer commands are probably different on the printer you use, this article and the above list address only keyboard commands.

In issues of Gazette since the May 1990 one from which I learned how to get rid of the TAPE OR DISK? question, there have been fully half a dozen auxiliary programs published for SpeedScript users. I've added some to my USSX disk (Extra SpeedScript Stuff). You might find one or more of them exactly right for a word processing application of your own. Here's a short look at what they do.

Right/Side by Robert B. Cook, November 1990, is a sideways file printer for sequential files up to 255 columns wide. It prints such files sideways with any odd-numbered Commodore or compatible dot-matrix printer.

Text Fitter by Keith M. Groce, December 1990, prints text in columns or makes it flow around artwork. It, too, uses sequential files such as those that SpeedScript prints to disk when you press Shift-Ctrl-P and then D for "disk."

Speedway by Daniel Lightner, October 1991, is a file reader and disk management program for SpeedScript files. It lets you read files, enter disk commands, and view a directory without interfering with any program you have running.

Another Lightner program, SpeedPurge, July 1992, searches SpeedCheck dictionary files for duplicated words and deletes them, making the spelling checker faster and more efficient.

In October 1992, there was another Lightner program called SpeedSpell, a faster machine language replacement for the earlier spelling program known as SpeedCheck, but one which can incorporate the older dictionary entries.

Then, in December 1992, there was a program by Frank Gordon called Speedram-64 which allows you to use SpeedScript with the 1764 RAM expansion unit or with two disk drives.

Given the vast body of auxiliary programs that have been written for SpeedScript, its popularity should continue indefinitely. And, as I said in the beginning, it certainly remains my personal favorite.

7@7@7Z7a7@7@7@7a7@7@777e7@7@O@7@7e7Z7@r@7@777b7@7@777@7@7@7@7@777@777@

3-in-1 FOOTBALL

Reviewed by Don Radler

Many sports fans argue that football is more exciting than baseball because the action is faster and the hits are harder. Certainly football players take punishment as if they were latter-day gladiators; they frequently get hurt, and often quite severely.

If you couldn't see the action and hear the hits and grunts, would football be as exciting to watch? (For that matter, without sound and motion, would "Spartacus" have been much of a movie?)

Back in February, I reviewed Full Count Baseball by Lance Haffner Games, describing it as "a manager's program" and deploring the lack of sight and sound effects that help make sports simulations gripping. Now I've been asked to review Haffner's football simulation, called 3-in-1 Football. You don't really play this game, either; you direct it. I have to call it a coach's program, because that's the role you adopt when you run this simulation.

Again, there's a bare-bones text screen describing the gameplay as it occurs. Again, there are none of the sights and sounds of the game and no call for frantic manipulation of a joystick. And, again, there are statistics--lots of statistics.

College teams and pro teams are covered (and each group plays by its own rules). On the 1992 Team Disk, there are 28 pro teams and 48 college teams; on a second team disk, there are 70 great past pro teams and 68 great past college teams.

But, somehow, things come together enough in 3-in-1 to make this a more absorbing program, one that keeps you playing through four whole quarters of such flat statements as "Long pass from J. Baker to Hopkins complete for 45 yards."

Once again, statistically, it's an impressive piece of programming. The main game program is just 921 lines of BASIC, not compiled, but tightly written and crunched. It moves swiftly through its paces as the coaches' play calls are made, and the screen is refreshed with refreshing speed throughout.

You can play 3-in-1 against a friend, with the computer acting as scorekeeper. You can play against the computer. Or, you can choose to have two computer teams play each other, watching the onscreen text change as the game develops.

In my earlier review, I complained that the setup before a game for Full Count Baseball just took too long; that problem doesn't arise in 3-in-1. You load the game from the game disk, flip it over to side 2, and choose two teams by number. Then you pick the location for the game: either team's stadium or a neutral location.

The defensive coach chooses his strategy for each play from among six possibilities, based on whether he anticipates an inside run, a wide run or screen, rollout or sideline pass, a short or medium pass, or a long pass. Haffner calls the sixth strategy the "honest" defense, used when you honestly don't have a clue what the offensive coach is going to call.

Here's what a defensive coach's options look like.

```
92 AIR FORCE     0  0  0  0 -  0
92 ARIZ STATE    0  0  0  0 -  0

 QTR 1           TIME   15 : 0
92 ARIZ STATE    YL:  OWN  20
DOWN  1          TOGO:  10

 SELECT DEF 92 AIR FORCE
 1 -HONEST
 2 -INSIDE RUN
 3 -SPREAD
 4 -BLITZ
 5 -SHORT PASS
 6 -LONG PASS
```

The offensive coach can call for an inside run, off tackle play, end run, option run, draw play, short pass, medium pass, long pass, short screen pass, play action pass, sideline pass (only in the last two minutes of the half), rollout pass, punt or field goal attempt. Haffner's descriptions of each of these options in the documentation for the game is a veritable short course in strategy for offensive coaches.

To help either side get some clue about what the other side might do, a scouting report can be called up on your screen before any offensive play. It shows how the two teams stack up against each other, based on individual ratings of rushers, passers, receivers, kickers, and kick returners and on team ratings. All of this is derived from actual past performances and can be expected to serve as a tipoff to current performance. Statistically, the law of compound probability is invoked; it's what handicappers use to calculate the odds for any sporting event. (If that seems somewhat tenuous, remember that it's all Jimmy Johnson and Jerry Jones had to work with when they were building the latest Dallas Cowboy dynasty. This reviewer remains silent as to the ego needs that brought their strange synergy to a screeching halt.)

According to Lance Haffner, 3-in-1 has been thoroughly researched and play-tested; he claims that it's THE most statistically accurate game on the market. It would take more numbers than are available to this reviewer to challenge that statement.

You can add more numbers of your own after any game that you play with

3-in-1 Football. A complete statistical breakdown is displayed on the screen, showing a scoring summary plus team and individual statistics. (This breakdown can also be saved to a printer.) You're then asked, SAVE STATS (Y/N). If you answer Y, you're prompted to insert a formatted statistics disk and then hit Return twice. Make a note of the name you save the stats under--you'll need it to review them using another program that Haffner calls LOOKIE. Here, simply by typing in the name of any team, you can view its past schedule and both individual and team statistics. (This, too, can be saved in hardcopy form.)

As with Full Count Baseball, you can form your own teams, drafting or trading players from any of the teams on disk. And you can create your own league, competing with others who have the same software. Finally, there are additional multiteam disks available at $19.95 each, with frequent season updates. If coaching a college or pro team vicariously sounds like fun and if the strategy means more to you than the glitz, 3-in-1 Football could be just your cup of tea--or whatever football coaches drink....

LANCE HAFFNER GAMES
P.O. Box 100594
Nashville, TN 37210
(615) 242-2617
$39.95; S&H $3.00

Gazette, November 1994