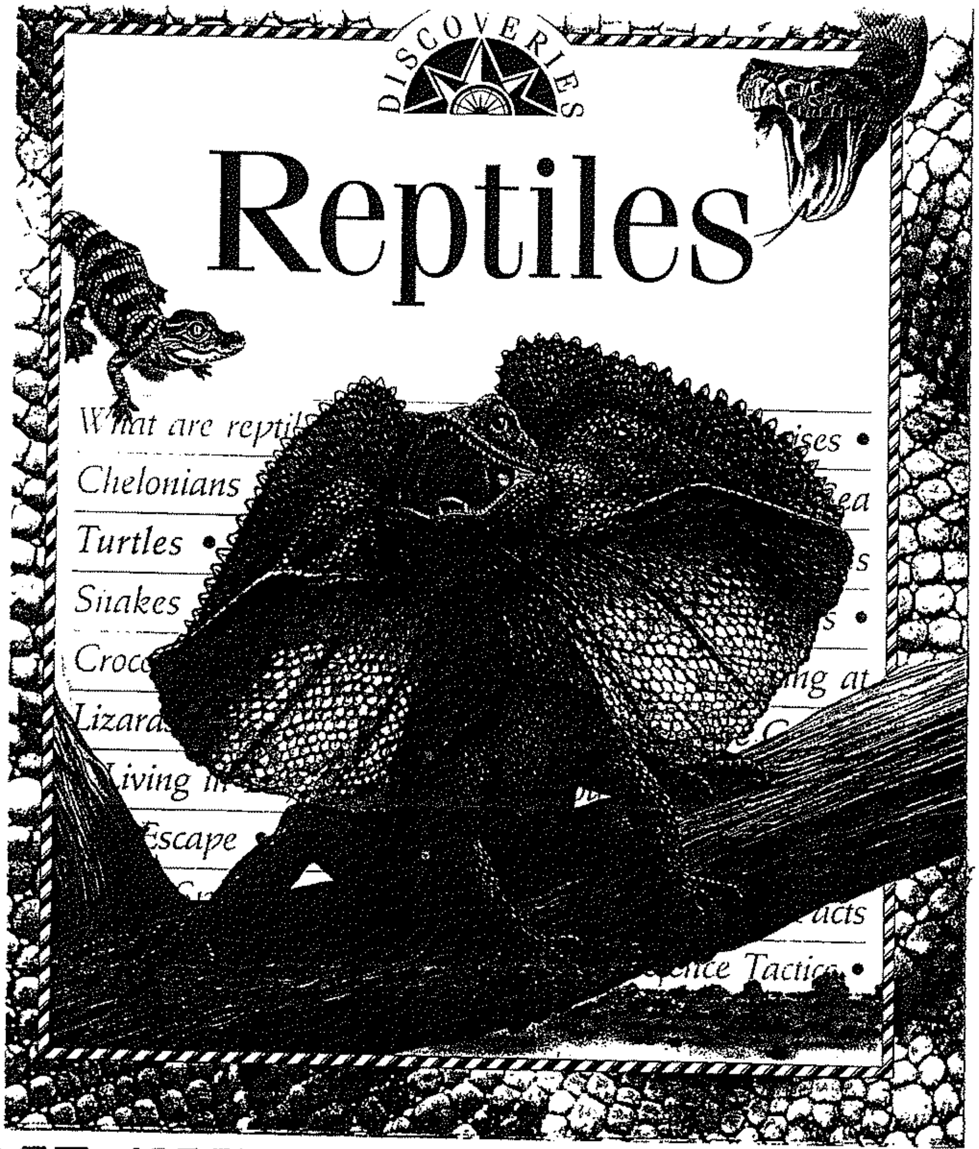
TI-99 USERS OF PERTH

TIUP TIT BITS

VOLUME 15No.1
JUNE 1996.

TIUP Pays the Rent

TIUP meets monthly in the Bayswater - Hillcrest Primary School Art Room. The annual 'Pepper Corn Rental', required for use of this facility, is the donation of a book suitable for the school's library. A reptile book, has been selected for this year's payment, and it should be a winner with the school kids.



PLEASE NOTE, OUR ADDRESS IS

* Index Page 14*

SECRETARY TIUP
20 HUDSON ST.
BAYSWATER 6053
WESTERN AUSTRALIA.

EDITORIAL by F.Graham Secretary/Editor

On the 20th.April, this year's Annual General Meeting, a group of dedicated Texas Instrument 99 Users of Perth gathered to celebrate TIUP's 14 years of operation in Western Australia.

At this meeting, the following committed members were elected as the Office Bearers of TIUP, for the next 12 months: President: Merve Trowbridge, Vice President: Bill La Frentz, Treasurer: Greg Buck, Secretary: Frank Graham.

I wish to express my gratitude, to Bill La Frentz, for his contribution to this month's issue.

Please submit any contributions to the Editor care of:
TIUP (TI-99 Users of Perth Inc).
20 Hudson St.
Bayswater 6053.
WESTERN AUSTRALIA.

SUBSCRIPTIONS

FULL MEMBER - A$20-00 P.A.
NEWSLETTER ONLY - A$15-00 P.A.

ADVERTISING

Members and newsletter subscribers may advertise their PERSONAL computer - related products in TIt BITs free of charge (subject to space limitations and the Editor's discretion). Commercial rates by negotiation.

BILLS BURST No.5

Well here is another burst in the series that I have typed up, from MICROpendiun, this time it is for the disk named TIUP/DOMØ6.

(1) @DATA 5 DIS/VAR 163 (XBASIC).

(3) CON 2 DIS/VAR163 (XBASIC).

(4) D/MERGE 2 DIS/VAR 163 (XBASIC).

(5) D/MERGE2 3 DIS/VAR 163 (XBASIC).

(21) SHELL 3 DIS/VAR 163 (XBASIC).

Source: MICROpendium July 1989. These programs are from "Routine creates instructions for programs" by Jim Peterson. If you load SHELL into memory and then MERGE D/MERGE you can then RUN this program and you will have a blank screen, with the cursor at the top left of the screen, ready for you to enter instructions or any information. It has most of the editor capabilities except that when you press enter for any line, you cannot go back, therefore, plan what you want to enter before you start. You can enter as much information as you like because as soon as that screen is full you are given another blank screen, which is saved to disk as the @DATA program. CON and D/MERGE are only needed to create D/MERGE2, SHELL and D/MERGE2 are the programs you use, @DATA is the result. You then load the program that you wish to add instructions to, making sure the lowest line number is greater than 8 and its highest line number is less than 3Ø721. You can then MERGE the @DATA program, that you created, into memory and then list it to screen.

(2) COMPARE 7 DIS/VAR 163 (XBASIC).

Source: MICROpendium January 1994. This is from "COMPARE - finds differences between programs" by Mike Dodd. This program is useful if you have two copies of the same program and you wish to see the differences between them. Before you RUN it, make sure the two programs you want to compare, are in MERGE format, then run this program and you will be asked for the OLD file name, the NEW file name and also the OUTPUT file name. When completed, you type in NEW and press ENTER then MERGE the OUTPUT file into memory and LIST it to screen, where the differences will be displayed.

(3) REFER TO (1).

(4) REFER TO (1).

(5) REFER TO (1).

(6) DAY/WEEK 3 PROGRAM (XBASIC).

Source: MICROpendium June 1991. This is from "Days of the week" by Larry Tippett. When running this program you are asked to enter the month, day and year and then press ENTER and it will give you the day of the week of the date that you entered. Make sure you enter 'commas' between digits and the year must be in 4 digits,(12,6,1942) for the 6th day of the 12th month 1942. This program will display, the day of the week you were born, or the day of the week some important event in history took place.

(7) EFONT 14 DIS/FIX 8Ø (ASSEMB).

(8) EFONT/M 2 DIS/VAR 163 (XBASIC).

(9) EFONT/XB 12 PROGRAM (XBASIC).

(1Ø) EFONTDEMO 3 PROGRAM (XBASIC).

(11) EFONTLOAD 2 PROGRAM (XBASIC).

Source: MICROpendium September 1987. These are all from "Adapting character sets with EFONT" by Bill Gaskill. When you LOAD and RUN EFONTDEMO it will load EFONT (the assembly version). When installed, you will see almost immediately, how it changes the entire character set, one by one. To stop the program, you will have to press FCTN-4. If you LOAD and RUN the EFONT/XB program you will see similar character set changes, as outlined above. However, the program will crash unless you change Line 1Ø2 to either 'GOTO 1Ø2' or fill in "RUN DSK1.____" so as to

(continued on page 4)

run it with another program. If you want to use the assembly version with another program, use EFONTLOAD, but you will still have to fill in the blank space in Line 2 with the name of the program you want to use it with.

(12) FILE-DUMP 14 PROGRAM (XBASIC).

Source: MICROpendium June 1987. This is from "File dump program in hex and ASCII" by Chuck Reinhart. When you run this program you will be prompted all the way, asking for the the Drive No., if the output is to screen or printer and if you want a HEX output. The disk will be read and the first 34 files, that's all this program can work with read, with the exception of PROGRAM, will be listed. After making a choice, the file will be listed to the screen, or printer, further, you may see blacked out sections. I do not know if this is suppose to happen as the original write up did not say, it only happens if you do not select the HEX output. However, if you do select the HEX output, you get the ASCII output as well, without the blacked out areas. When you check the ASCII output that is given with the HEX output you will notice the areas that were blacked out, have unprintable characters. I have tested and rechecked this program but cannot find any errors in it, I have not checked the printer output, if you want a printout make sure that you set the printer requirements in Line 260, to suit your system.

(13) GERMAN 16 PROGRAM (BASIC).

Source: MICROpendium January 1992. This is from "Learning German" by REGENA. To use this program you will need a Terminal Emulator 2 and TI Speech Synthesizer to see and hear the German words and phrases. I am not learning German, but I thought this might be usefull to someone who is, and as it is by REGENA it must be good. The program comes complete with instructions so you should have no trouble with this one.

(14) HELP 16 PROGRAM (XBASIC).

(16) MAKE 8 PROGRAM (XBASIC).

(19) PIANO 16 PROGRAM (XBASIC).

(24) STAFF 6 PROGRAM (XBASIC).

(28) WORK 8 PROGRAM (XBASIC).

Source: MICROpendium January 1986. These 5 programs are from "Digital Music" by Stephen D. Peacock. There should be a 6th program with these but it is only a one line LOAD program, (100 RUN "DSK1.HELP"), so I left it out. If you check the programs you will find that MAKE and WORK are the same, just saved under different names. The write up for these programs is rather complex and I don't think I could improve on them, so I have included the write up from MICROpendium. The main menu consists of five options:

1. Instructions
2. To Play Music
3. To Compose A New Song
4. To Add to "Work" File
5. Print Numbers On Staff

The HELP program consists of instructions to operate the program. Option 1 loads this program. The second option controls the program called PIANO. This program controls the playingof the music. The MAKE program corresponds to Option 3 and is used to compose music. Option 4 also accesses the MAKE program. Both Option 3 and 4 utilize a fourth program called WORK, which is used to hold data while you are creating and testing a song.

When PIANO is run you will see a picture of a piano with a message to "PLEASE WAIT, I AM STUDING MY MUSIC." This pause is needed to calculate the values of several arrays. When these values are computed, a menu will appear. This menu consists of the following options:

```
AGAIN----------1
NEW------------2
END/COMPOSE----3
INDEX----------4
```

To play a song select NEW. You will be asked for the title of a song

(continued on page 5)

that has been saved. The prompt "DSK1." will appear. Just type the name of a song and press enter. It will run and play. If you enter an incorrect name you will get the message "DEVICE CAN NOT BE ACCESSED" and you may try again. To repeat the same song, select AGAIN, and the song will be repeated.

To compose a song you must enter the total number of notes you will use in a DATA statement, as well as the duration you want the notes to be played and numbers to correspond to the notes. Digital Music requires three numbers for each note (which are played in chord-like fashion). If you want to have a series of six notes or chords assigned to a single DATA statement the first entry in the DATA statement would be the number 6, followed by a comma, of course. Then you would enter the duration of the first note or chord, followed by a comma, and the three numbers corresponding to the three notes that are required by Digital Music. Here is an example: 490 DATA 3,300,20,20,20,300,2,21,25,300,3,22,26

Three notes would be played for 300 milliseconds each, consisting of sounds created by the numbers 20, 20 and 20; 2,21 and 25; 3,22 and 26.

You may string as many of these combinations together as you can per DATA statement, just make sure that the first number is equal to the total number of chord or note sequences. Of course, you may use as many DATA statements as it takes to compose your song. Data is entered using the MAKE program, which can be run from the main menu, at the start of this program or by running DSK1.MAKE from a cold start.

Each note corresponds to a position on a musical staff. The lowest is C, which is equal to 110 hertz. C is given the number 1. The next note, C sharp, is 2, with D being 3. D sharp/E flat is 4. This is continued for a total of 45 half steps, up to the G sharp above the treble staff.

To enter a sharp give the next higher number. For example, if F is equal to 6 then F sharp is equal to 7. A flat uses the next lower number. For example: if B is equal to 12 then B flat is equal to 11. To play a rest enter the number 46. This has been assigned a frequency of 44000 hertz. The duration of the note is entered in milliseconds; 300 is equal to 300 milliseconds.

When the compose option is selected the file DSK1.MAKE will run. You will see the prompt RUN/TEST/BREAK/ SAVE. These options are activated by entering the first letter of the one you want to use "R,T,B or S". It is here that the information about DATA statements, is put to work. You would select "B" when writing a song. This will BREAK the program. You will be prompted to start entering data at program Line 490. Simply enter NUM 490 and you can start entering musical data. After the data has been entered, enter RUN. When the prompt line reappears press "T" for TEST. This will read your data and print it to the screen. You will be able to see if you have entered the data properly. If not, it will inform you of your errors. Hold any key to stop the listing. Release the key to restart.

After the data file has been tested select "S" for SAVE. The program will end and prompt you to save the file as DSK1.WORK. If it's saved in this way it can be run from the main program or from the start of the MAKE program. The DATA statements will be intact and you can add to or change them. The file DSK1.MAKE will still be available to create a different song. Please note that only one WORK file may be run from this system. If you want to work on more than one song at a time, save them under other names. You will then need to run that file from a cold start. After you have tested and saved your data, select "R" for run. The PIANO program will then run, and you may listen to your new composition.

To see all of the songs that you have saved to disk, select 4 for

(continued on page 6)

INDEX. Every song that has been saved with the MAKE program will be listed. Press enter to step through all of the titles. At any time you may press "Q" to quit, and return to the menu.

That should be enough, to get you started on the way, to composing with Digital Music.

(15) KEY-CODE 4 PROGRAM (XBASIC).

Source: MICROpendium April 1987. This program is from "Key-Codes helps identify that'K'", by Ray Kazmer. This program is similar to KPVALUES on TIUP/DOMØ3, the first disk that I type up, but this one allows you to select the different CALL KEYS, (Ø-5), and see what the K would equal if you press any key or combination of keys, but be warned once you run this program the only way out is to turn the computer off.

(16) REFER TO (14).

(17) MATEY 4 PROGRAM (XBASIC).

Source: MICROpendium January 1992. This is from "Cracking the code" by a programmer whose nom de plume is 'Cracker Jack'. I have not used this program as I do not have a program that requires it to be used, but I have checked and rechecked and there were no errors when I assembled it. I thought it might be handy to have and if I ever get a program that causes the computer to lock up when I try to list it I will try this program out. To use this program you load the problem program and then type in CALL INIT :: CALL LOAD("DSK1.MATEY") and press ENTER, when the cursor reappears on the screen you should be able to list it and edit it.

(18) NOTES 18 PROGRAM (BASIC).

Source: MICROpendium September 1991. This is from "Playing musical notes" by REGENA. This program is for those who are interested in learning to play the notes on the piano. It is written in BASIC but will run in XBASIC. When you run this program you will see a treble staff in the upper half of the screen and a keyboard (with Middle C at the left) on the lower half of the screen. Six notes appear on the treble staff and to play them you move the flashing indicator with the arrow keys and then press ENTER. When you press ENTER the note that the indicator is on is shown if you are wrong but if you are correct the question mark is replaced by the note and the question mark moves onto the next one on the treble staff. Once you get all six correct the computer lists another six for you to try but you can quit by pressing "Q" when the indicator is flashing.

(19) REFER TO (14).

(2Ø) ROULETTE 46 PROGRAM (XBASIC).

Source: MICROpendium December 1989. This is from "Gambling on the holidays" by Jerry L. Stern. It is best run with a speech synthesizer attached, so that the computer can announce the winning numbers and colors, it also makes some smart comments during the game. You start off with $5ØØØ, but you must bet $1ØØ (or more), or you invoke the wrath of the manager. For help you bet #Ø and to quit you bet $Ø. If you do not know how to play ROULETTE, a good idea is to copy the list of possible bets, so you don't have to keep going back to the help screen.

(21) REFER TO (1).

(22) SOUNDER 29 PROGRAM (XBASIC).

Source: MICROpendium July 199Ø. This is from "Collecting sound effects" by Jerry L.Stern. When you run this program you will be presented with screen of 2Ø choices, 19 sound effects and quit. All the sound effects are in the form of subprograms so that you can use them in your own programs.

(23) SOUNDSTAGE 51 INT/VAR 254 (XBASIC)

Source: MICROpendium August 199Ø. This is a big program from "Orchestrating noises,sounds and

(continued on page 7)

silences" by Jerry L.Stern. This program will allow you to create, edit, play and save sound effects so that you can use them in other programs. The program is menu driven and helps you all the way, it will even give you a second chance if you choose quit and then change your mind. Play with this program and see what you can create but just remember if you use the SAVE option the file can only be read by the SOUNDSTAGE program so you will have to use WRITING A MERGE FILE option, also, if you want a print out be sure to check the printer settings in Line 9Ø, to suit your system.

(24) REFER TO (14).

(25) TETRIS 26 PROGRAM (XBASIC).

(26) TETRIS/1 17 PROGRAM (XBASIC).

Source: MICROpendium September 1989(1), October 1989(2), March 199Ø(3) and May 199Ø(4). TETRIS/1 is the original program from "A challenging game in XBASIC" by Steve Kurasek (1) with corrections "Tetris character changes are noted" by Henry E. Kochne (2) while TETRIS is from "Bells and whistles for Tetris" by Jack B. Cunningham (3) with corrections from "One more bell to ring" also by Jack B. Cunningham (4). If you are playing TETRIS/1,make sure that the Alpha key is depressed, and you will be asked for a starting level of 1-9, the higher the level the faster the shapes drop and higher points. You use S and J to move left, D and K to rotate the shape 9Ø degrees, F and L to move right A or semicolon to pause the program, you then press any other key to resume play and press the space bar to drop the shape into position quicker. The object of the game for those who have not seen TETRIS, which would be very few, is to build solid lines across the screen. These will dissappear from the screen and any blocks above this will drop down. To change levels during the game press U and R and if you want to quit press Q.

TETRIS is the improved game with sounds added, instructions added and the option to use keyboard or #1 joystick. In this program you are asked to make sure the Alpha Lock key is up. When playing the game I found that you could not quit so I added the option in Line 6ØØ. I also found that if you put one shape in the wrong way and then try to slip another one in sideways, to fill in the space under the main shape, the one you are trying to put in will jump up to the top of that line when you press the space bar. I was unable correst this problem and seek advice and help to do so.

(27) TI-HIDE-IT 18 PROGRAM (XBASIC).

Source: MICROpendium November 1993. This is from "Hidden Secrets of TI-Writer" by Barry A. Traver. This program is suppose to hide lines at the beginning of a D/V8Ø file from TI-Writer (or its equivalent). I have tried this program a number of times but it does not work. I have checked the program that I typed in and it matches what was listed in the MICROpendium, except for obvious errors (e.g. Line 75Ø ends with IF M$="" THEN 89Ø but there is only 82Ø lines in the program). I have tried a number of different things, but could not get this program to work, and conclude that it must be something omitted from the print out. I have been expecting to see a correction in a later issue of MICROpendium, but have not seen it yet. I have left this program on the disk as someone who gets a copy maybe able to get the program to operate correctly and let me know where the error is.

Well that's enough for this disk, but I have to report that I have found an error in ACE which is on TIUP/DOMØ5 disk. Line 85Ø should be: N=N+1 :: NEXT NLINK. After you make this correction, the program should work properly. Good-bye for now and keep on TI'ing.

Bill La Frentz

TiSHUG Software File Review
June 1996
Editor By Larry Saunders

I hope you enjoy this disk collection. Since it was submitted to Micropendium for inclusion in their FAIRWARE section, there have been some additional programs added.

Reverserle- creates an inverse (negative image) of any Fixed 128 RLE picture.
IMPORTANT: The input file must be an RLE picture in df/128, no other type will be accepted by the program. If the picture is in another format such as Graphx or TI-Artist, first using the Max-rle program convert it to 128 format using the SAVE(S) command (paged with space bar). Read the doc's for Max-Rle for a complete description. The output filename of Reverserle may not be the same as the input filename, UNLESS it's sent to a different drive. (the program will issue an error if you attempt to call both input and output files the same name)
I have also included a copy of MAX-RLE, for those who don't have a copy.

Besides the graphic screens, which are in RLE and GRAPHX , there is an additional program included called SHOWGX. SHOWGX loads the assembly program READG, this program unlike REVERSERLE is to run in BASIC with the E/A cartridge.

SHOWGX- Will scan any drive selected and look for GRAPHX pictures. When found it displays the name and loads it.
Press any key for it to go on to the next. When all pictures on that disk are loaded it asks for more?. If you respond yes, you can now load another disk from any drive .
DO NOT ATTEMPT to alter the basic loading program. If you have any questions please feel free to write me:

Stephen J.Tuorto
18-Chimney Lane
Bayshore,NY 11706

CALENDER PROGRAM

By Adrian Robinson

[In the May issue of our magazine we printed an article on "Calender programming", however the printing of this article wasn't the best, lets hope this copy comes out clearer for the members who would like to type the program in. We at the editorial apologise for any inconvenience caused.

I would like to take this opportunity to thank the members of our club for their input and articles towards the production of this great magazine, month after month. (ED)]

```
10 ! SAVE DSK6.CALENDER
11 ! CALENDER PROGRAM
12 ! by Adrian Robinson
13 ! ROM Newsletter, August 1989
14 !
16 ON WARNING NEXT :: CALL CLEAR
18 DATA JANUARY,31,FEBRUARY,28,MARCH,
31,APRIL,30, MAY,31,JUNE,30
20 DATA JULY,31,AUGUST,31,SEPTEMBER,30,
OCTOBER,31,NOVEMBER,30,DECEMBER,31
22 W$=" SU ^MO ^TU ^WE ^TH ^FR ^SA
"&&RPT$("-",28) 24 M$="" :: FOR I=1 TO 31 ::
M$=M$&&RPT$("^",3-LEN(STR$
(I))&&STR$(I)&&" " :: NEXT I
26 INPUT " YEAR: ":Y :: IF Y<1583 THEN
PRINT^"YEAR^MUST^BE LATER THAN 1582": : :
GOTO 26
28 OPEN #1:"PIO",DISPLAY ,VARIABLE 28 :: PRINT
#1:Y: :
30 FOR M=1 TO 12 :: READ MO$,L :: CALL MONTH
((Y),M,B)
32 IF M=2 THEN IF (Y/4=INT(Y/4))-(Y/100=INT
(Y/100)) +(Y/400=INT(Y/400))THEN L=L+1
34 D$=RPT$(" ^^^",B)&&SEG$(M$,1,4*L)
36 PRINT " ";MO$:W$:D$: : :: PRINT #1:"
";MO$:W$:D$: :
37 IF M=6 THEN PRINT #1:CHR$(12): : :
38 NEXT M :: CLOSE #1 :: END
40 SUB MONTH(Y,M,B)
42 F=365*Y+31*(M-1)+1
44 IF M>2 THEN F=F-INT(2.3+.4*M)ELSE Y=Y-1
46 F=F+INT(Y/4)-INT(Y/100)+INT(Y/400)
48 F=F-1 :: B=F-7*INT(F/7)
50 SUBEND


100 ! SAVE DSK6.CALENDER1
110 ! CALENDER PROGRAM
120 ! by Adrian Robinson
```

(continued on page 9)

```
130 ! ROM Newsletter, August 1989
140 ! Printout mods by Ross Mudie
150 DIM P$(12,9)
160 ON WARNING NEXT :: CALL CLEAR
170 DATA JANUARY,31,FEBRUARY,28,MARCH,
31,APRIL,30, MAY,31,JUNE,30
180 DATA JULY,31,AUGUST,31,SEPTEMBER,30,
OCTOBER,31,NOVEMBER,30,DECEMBER,31
190 W$=" SU ^MO ^TU ^WE ^TH ^FR ^SA
"&&RPT$("-",28)
200 M$="" :: FOR I=1 TO 31 ::M$=M$&&RPT$("^",3-
LEN(STR$(I)))&&STR$(I)&&"^" :: NEXT I
210 INPUT " YEAR: ":Y :: IF Y<1583 THEN PRINT
"YEAR^MUST^BE LATER THAN 1582": : :: GOTO
210
220 OPEN #1:"PIO",DISPLAY ,VARIABLE ::
PRINT^#1:TAB(28);CHR$(14);Y:
230 FOR M=1 TO 12 :: READ MO$,L :: CALL
MONTH((Y),M,B)
240 IF M=2 THEN IF^(Y/4=INT(Y/4))-
(Y/100=INT(Y/100))+ (Y/400=INT(Y/400)) THEN
L=L+1
250 D$=RPT$(" ^^^",B)&&SEG$(M$,1,4*L)
260 PRINT " ";MO$:W$:D$: : :
270 P$(M,1)=MO$ :: P$(M,2)=SEG$(W$,1,28)::
P$(M,3)=SEG$(W$,29,28):: P$(M,4)=SEG$(D$,1,28)::
P$(M,5)=SEG$(D$,29,28)
280 P$(M,6)=SEG$(D$,57,28):: P$(M,7)= SEG$(D$,85
,28) ::P$(M,8)=SEG$(D$,113,28)
290 IF LEN(D$)>141 THEN P$(M,9)=SEG$ (D$, 141,28
) ELSE P$(M,9)=""
300 NEXT M
310 FOR M=1 TO 12 STEP 2
320 FOR L=1 TO 9
330 IF L=1 THEN PRINT #1:CHR$(14);TAB
(3);P$(M,L); TAB(22);P$(M+1,L)
340 IF L>1 THEN PRINT #1:TAB(2);P$(M,L);TAB(40);
P$(M+1,L)
350 NEXT L
360 PRINT #1: :
370 NEXT M :: PRINT #1:CHR$(12):: CLOSE #1 ::
END
380 SUB MONTH(Y,M,B)
390 F=365*Y+31*(M-1)+1
400 IF M>2 THEN F=F-INT(2.3+.4*M)ELSE Y=Y-1
410 F=F+INT(Y/4)-INT(Y/100)+INT(Y/400)
420 F=F-1 :: B=F-7*INT(F/7)
430 SUBEND
```

END OF ARTICLE

Transferring Data Base Files From Our Trusty "TI" To the IBM

by Loren West

This task is not new, but it is to me, so I though I would share my problems and success with everybody.

The data base that I am transferring from is the popular PR BASE and to "ACCESS" data base program in Microsoft Office.

Access can receive data base files from various sources, also it can use text files somehow, this will be another story later.

Ok with text files in mind. How can I make text files from my PR Data base. Easy, well almost easy, To help me on the way, I spoke to Dick Warburton as he was doing the same. Dick was using a different method but the results turned out the same.

My method was similar to the method that I am using already, transferring text files from the TI to the IBM, (having a cable hooked up using the serial ports.) and that is to PRINT them to the RS232, then on the IBM side I use the Terminal program that comes with Windows, which receives the information from the serial port and then saves it as a text file.

On the PR base side of things, I had set up the program to print out all of the data that is in the data base as if you were expecting to see it on a printout from the printer. The only difference was in the direction of the printer, normally (PIO) But I changed it to (RS232.BA=9600.DA=7.LF).

Having received this information into the Terminal program, which appeared an the screen exactly as it would have if it were sent to the printer.

Now I have a text file containing all the data base information that I need, the next step is to move this text into the Access data base system, when this happens I will let you know.
See you all at the next meeting.

END OF ARTICLE

TiSHUG Software File Review
June 1996
Editored By Larry Saunders

INSTRUCTIONS FOR MAX-RLE

Public Domain Program
by Travis Watford

RLE stands for Run Length Encoded. It is a program for preparing and viewing digitized pictures, both artwork and photographs, sent between computers over phone lines using a terminal emulator like Fast-Term. Many computers use this technique with the VIDTEX terminal emulator protocol, which permits viewing pictures on-line. For the TI-99/4A at present, pictures can be viewed off line only, but pictures can be exchanged with other brands of computers. The program supports four different formats - both TI-ARTIST and GRAPHX formats, as well as Display/ Fixed 128, the usual format used in other computers, and Display/Variable 80 format.

LOADING MAX-RLE. The program is loaded using the Editor-Assembler module or equivalent, Option 3 - Load and Run. The Filename is MAX-RLE and the Program Name is START. The MAX-RLE title screen will then appear, asking for the name of the picture file you want to load.

RUNNING MAX-RLE. At the title screen, you have two options - you can load a picture or you can catalog a disk.

** To load a picture, just type the filename, for example, DSK1.PICTURE, and press ENTER. Whichever format the picture is in, the program will recognize it and load it. (NOTE: For TI-Artist files, omit the "_P" and "_C" at the end of the filename - the program will provide these automatically.) You will then see a grey screen for a short while as the picture loads. Give it a chance! It does not appear immediately, but there is nothing wrong with the program. The picture will appear all at once on the screen when it is ready. Note: The program supports the Horizon Ram Disk HD command.

** To catalog a disk, just type "DSKn.", where "n" is the drive number. Be sure you include the period. The catalog function does not work with the MYARC RAM Disk.

OPTIONS WITH PICTURE ON-SCREEN. There are three options when the picture appears - you can return to the title screen, print the screen on your printer, or save the picture in a different format to disk.

** To return to the MAX-RLE title screen, press ENTER. Be careful. An accidental pressing removes the picture.

** To print on your printer, press P. The default setting of PIO.CR will appear on screen. If you are using the parallel interface, use this. If you are using a serial interface (RS232), enter your printer's description just as you would, for example, using the TI-Writer Formatter. Your printer must be compatible with the GEMINI/EPSON family in its handling of dot-graphics.

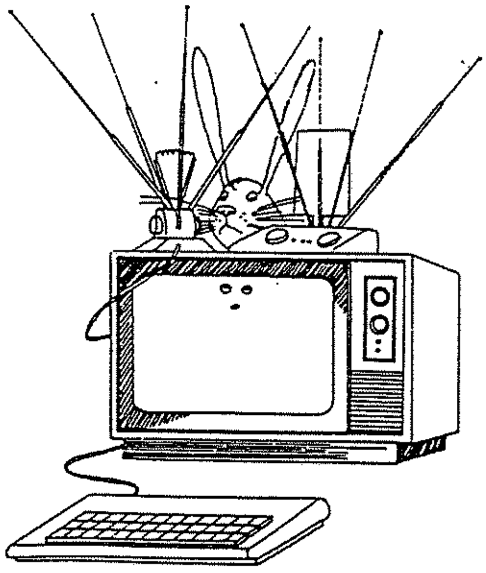
** To save to disk in a different Format, press S. The default setting of GRAPHX format will appear on screen. To save in a different format, press the space bar until the format you want to use appears. Then type the filename you wish to save to, for example, DSK1.MYPIC.

SENDING PICTURE FILES. Generally, pictures to be transmitted should be saved in the D/F128 format and uploaded with XMODEM transfers. This is the format used by other systems. Pictures can be sent in D/F80 format using ASCII (text) transfers, but they lack error checking in transmission and a noisy or weak connection can ruin the resulting picture.

PICTURES ON COMPUSERVE. Pictures readable by MAX-RLE can be found in CompuServe in the TI Forum Data Libraries, the PICSIG, the ARTFORUM, and the CB simulator area. They are also starting to appear on many BBS.

-text adapted by Walt Howe from a version by Paul Grey, the author of the first TI-RLE program. This text is primarily for non-CIS subscribers.

END OF ARTICLE


antenna

Fast Extended BASIC

Program lets user test sprite attributes before programming

By LUCIE DORAIS
©1995 L. Dorais

The following was originally published in the newsletter of the Ottawa 99/4A User Group.—Ed.

Sprite Tester allows you to test all the sprite attributes before you actually write your program line: definition, color, starting position (dotrow and dotcolumn), motion (velocity), and magnification. The variables are kept on the screen all the time, and change before your eyes as you play with them. If you don't want to enter a sprite definition, there is a pre-defined sprite. This program could also be useful for people who just want to learn more about sprites.

We start by defining PD$ as our predefined character, repeated four times to make a sprite; N$ is a 32-character string used in the test screen. In line 140, we initialize the limit checking values for sprite movement: I hope to have made the movement routines a bit faster by using variables. The remainder of the line is the beginning of the pre-scan.

The program proper starts in line 160. You must choose between a pre-defined sprite (CHAR 136=PD$ in line 170) or your own. In this case, you enter four lines of hex codes (for four characters). If you need only one character (magnification 1 or 2), enter nothing on lines 2-4; lines 200-210 will adjust the default magnification and define characters 136-139 accordingly.

Lines 220-230 display the instructions — STEPPING means the number of pixels the sprite will jump if you position or move it. The starting default "J" is set to 8 in line 250, for a full character jump. For fine tuning, press any digit key from 1 to 8. I used "K" for Color because "C" is used to position the sprite.

Other variables initialized here are sprite color, starting dotrow and dotcolumn, and velocities. After you press any key, the test screen is built, with the 24 rows and 32 columns numbered at top and left (the sprite's dotrow and dotcolumn pixel values are eight times bigger). The CALL SPRITE statement and the magnification factor are displayed at the bottom, with a line of explanation (290). Because we refresh that statement each time we change something, we use the subroutine in line 520 to display an IMAGE. But since the character "#" tells Tex to expect a value, how do we display that character for itself? Well, we send it as a constant string value, along with the other variables.

Now you can start to play with your sprite! You can find its starting position or set it in motion, but you cannot do both at the same time. The flag MOT is set to "0" when velocity values CV and RV are 0 (line 300). Otherwise, as soon as you press an Arrow+FCTN key, MOT is set to "1" and the position keys (ESDXWRZC) will not respond anymore (line 330: IF MOT THEN 310; remember, IF MOT means IF MOT<>0). If you want to modify the starting position after you started motion, just play with CV[EL] and RV[EL] until they are both 0: line 300 will then reset MOT to 0 and relocate your sprite at its starting position indicated by DR and DC. To indicate that you are back into "positioning," the little squares reappear in the background. The other functions, Color (K), Magnification (M), Stepping (1-8) and Exit (Q) can be used any time, whether positioning or moving the sprite.

The digit key press is dealt with in line 320 — stepping default J is reset to the key value minus 48. Changing color is done in lines 470-480; it is a cycle, coming back to black (2) at the end. Magnification (lines 480-490) is also a cycle. Upon exiting with Q, you are asked if you want to Rerun or Quit.

I tried to simplify my positioning and moving routines. I still use a user-defined SUB SP to check for limits Y and Z passed as parameters A, B, C, D (initialized in line 130). But by carefully designing the key-pressed position string (line 330), I grouped them by two and found that the only difference for DR or DC is in the stepping: positive or negative, depending on K being odd or even. By setting ST to j or -J accordingly (line 340), I got rid of half the moving statements! J is still used when we move the sprite diagonally (lines 380-390). Those four lines all lead to line 400, where the sprite is re-located and the new values are shown on the screen.

As soon as you press a FCTN+Arrow, i.e. a key value of 8-11, you branch to line 410; flag MOT is set to "1" and the little squares (character 140) disappear. Then again, ST becomes J or -J depending on the key pressed (the odd-even bit did not work here...). Lower limit for velocity is really -128, but I set it to -127 (-D) for simplicity. Again, our sub SP does the job of stepping and checking. The sprite motion is modified in line 450, and the new values are displayed by sub 520. Have fun!

SPRITETEST

```
100 ! SPRITE TESTER / L.Dora
is, Ottawa UG / Aug. 89 !127
110 !!131
120 CALL CLEAR :: CALL SCREE
N(4):: CALL COLOR(14,3,4)!07
8
130 PD$=RPT$("993C7EFFFF7E3C
99",4):: N$=RPT$("1234567890
",3)&"12" !156
140 A=1 :: B=192 :: C=256 ::
 D=127 :: GOTO 160 :: A$,CV,
DC,DR,J,K,MA,MOT,RV,S,SC,ST,
X :: CALL KEY :: CALL CHAR :
: CALL HCHAR !249
150 CALL SPRITE :: CALL DELS
PRITE :: CALL MAGNIFY :: CAL
L LOCATE :: CALL MOTION :: !
@P- !001
160 DISPLAY AT(3,9)ERASE ALL
:"SPRITE TESTER": : : :"SPRI
TE:  <1> pre-defined": :"
      <2> user-defined" !229
170 GOSUB 530 :: MA=1 :: IF
K=49 THEN CALL CHAR(136,PD$)
```

(Continued on page 12)

EXTENDED BASIC —

```
:: GOTO 220 !147
180 DISPLAY AT(16,1):"SPRITE
 DEFINITION:" !133
190 FOR X=18 TO 21 :: DISPLA
Y AT(X,2):">";TAB(19);"<" ::
 ACCEPT AT(X,3)VALIDATE("012
3456789ABCDEF")SIZE(-16):A$
!209
200 IF X>18 AND A$<>"" THEN
MA=3 !045
210 CALL CHAR(118+X,A$):: NE
XT X !250
220 DISPLAY AT(1,1)ERASE ALL
:"In the SPRITE TEST screen,
": :"PRESS:": :" ARROWS/WRZC
 to POSITION it": :"then": :
" FCTN+ARROWS for MOTION." !
176
230 DISPLAY AT(12,1):"any ti
me:": :" 1-8 to change STEPP
ING": :" K to cycle COLOR":
:" M to cycle MAGNIFICATION"
: :" Q to EXIT" !011
240 DISPLAY AT(24,3):"press
ANY KEY when ready" !183
250 CALL KEY(0,K,S):: IF S=0
 THEN 250 ELSE CALL CLEAR ::
 J=8 :: SC=2 :: DR=73 :: DC=
113 :: RV,CV=0 !074
260 FOR X=1 TO 32 :: K=ASC(S
EG$(N$,X,1)):: CALL HCHAR(1,
X,K)!129
270 IF X>1 AND X<25 THEN CAL
L HCHAR(X,1,K):: CALL HCHAR(
X,2,140,31)!066
280 NEXT X :: CALL MAGNIFY(M
A):: CALL SPRITE(#1,136,SC,D
R,DC)!238
290 DISPLAY AT(24,1):"(s#,c#
,co,dro,dco,rvel,cvel)" :: C
ALL HCHAR(24,32,109)!130
300 IF RV=0 AND CV=0 THEN MO
T=0 :: CALL CHAR(140,"FF8181
81818181FF"):: GOTO 400 !149
310 CALL KEY(0,K,S):: IF S=0
 THEN 310 ELSE IF K>7 AND K<
12 THEN 410 !145
320 IF K>48 AND K<57 THEN J=
K-48 :: GOTO 310 !195
330 K=POS("EXSDWRZCKMQ",CHR$
(K),1):: IF K=0 THEN 310 ELS
E IF K>8 THEN 460 ELSE IF MO
T THEN 310 !038
340 IF K/2=INT(K/2)THEN K=K-
```

ROOTS answers a need I had for years: Tex can give the square root of any number with SQR(x), but I sometimes needed the cubic root, and found no function. I got the answer in a math program on TV Ontario....

```
1 :: ST=J ELSE ST=-J !117
350 ON (K+1)/2 GOTO 360,370,
380,390 ! position sprite !0
63
360 CALL SP(DR,ST,A,B):: GOT
O 400 !178
370 CALL SP(DC,ST,A,C):: GOT
O 400 !164
380 CALL SP(DR,-J,A,B):: CAL
L SP(DC,ST,A,C):: GOTO 400 !
219
390 CALL SP(DR,J,A,B):: CALL
 SP(DC,ST,A,C)!183
400 CALL LOCATE(#1,DR,DC)::
GOSUB 520 :: GOTO 310 !238
410 MOT=1 :: CALL CHAR(140,"
"):: IF K=9 OR K=10 THEN ST=
J ELSE ST=-J !072
420 ON INT(K/2)-3 GOTO 430,4
40 ! move sprite !227
430 CALL SP(CV,ST,-D,D):: GO
TO 450 !174
440 CALL SP(RV,ST,-D,D)!041
450 CALL MOTION(#1,RV,CV)::
GOSUB 520 :: GOTO 300 !038
460 ON K-8 GOTO 470,490,510
! color,magnify,quit !121
470 SC=SC+1 :: IF SC=17 THEN
 SC=2 !048
480 CALL COLOR(#1,SC):: GOSU
B 520 :: GOTO 310 !122
490 MA=MA+1 :: IF MA=5 THEN
MA=1 !219
500 CALL MAGNIFY(MA):: GOSUB
 520 :: GOTO 310 !086
510 DISPLAY AT(12,1)BEEP:" P
ress  <1> RERUN  <2> QUIT" :
: GOSUB 530 :: IF K=50 THEN
END ELSE CALL DELSPRITE(#1):
: GOTO 160 !064
520 DISPLAY AT(23,1):USING "
(#n,ch,##,###,###,####,####)
":"#",SC,DR,DC,RV,CV :: CALL
 HCHAR(23,32,MA+48):: RETURN
 !106
530 CALL KEY(0,K,S):: IF S=0
 OR K<48 OR K>50 THEN 530 EL
SE RETURN !079
540 !@P+ !062
550 SUB SP(V,S,Y,Z):: V=V+S
:: IF V<Y OR V>Z THEN V=V-S
!189
560 SUBEND !168
```

ROOTS

ROOTS answers a need I had for years: Tex can give the square root of any number with SQR(x), but I sometimes needed the cubic root, and found no function. I got the answer in a math program on TV Ontario: you cube many carefully chosen numbers until the result matches your input number. This is called the "Averaging" process because at each pass the program decides if your answer lies before or after the average value of HI and LO. And why not add more roots? My program works with roots 1 to 9!

ON WARNING NEXT in line 120 tells Tex to repeat the statement if by mistake you enter a letter (or nothing) where it expects a number. E$ is a 6-line eraser, and S$ just some spaces. I cut the character definition to single characters, but of course you could CALL CHAR(120,"0000....101"), i.e. up to 123, as one long string.

Line 150 displays a nice root symbol on the screen and GOSUB 280 to ask for a root value, default being 3. The ACCEPT routine in the sub should not accept 0, therefore we validate a string instead of digit. The ACCEPT N statement in line 160 validates numeric because it has to include "E+" if you wish to use scientific notation — even if you don't, Tex will for

(Continued on page 13)

EXTENDED BASIC —

very big numbers (see below).

Next we deal with root values of 1 and 2, which don't need the averaging process. Any number powered to 1 is always itself, therefore its one root is also itself; and of course the XB guru gave us the SQR(x) function to find square roots.

The averaging process starts in line 180: we arbitrarily set LO to zero, and HI to the SQR value of our number N; why not? It will always be higher than our final answer. Next, AV becomes the average of HI and LO, and Tex finds what the answer T would be if AV were raised to the R power. In line 200, it is told to stop when the current AV is equal to the previous (OAV) one, otherwise it could go on forever!

If AV and OAV are still different, we go on. If our number N is smaller than the trial answer T, the root we are seeking is between LO and the average value AV. Therefore, AV becomes the new HI. Otherwise, if N is higher than T, its root is between AV and HI, so AV becomes LO, and we return to line 190. Harder to explain than to do. Tex takes only a few seconds, and displays all its intermediate AV values in line 200. The final AN(swer) is displayed in line 220, preceded by a "=" and announced by a beep. You can then print the result if you wish (with a nice root symbol), or change the default root, do another, or quit.

How accurate is it? Well, up to a few decimals, and I think only the really big numbers are a bit off mark, because Tex transforms them into scientific notation. For example, for the cubic root of 57899987899, I got 3868.65045. Tex printed "5.79E+10" as my number. So I reentered 5.79E+10 as my N value, and got 3868.650719. If you stick to smaller values, you will be closer to the truth.

ROOTS

```
100 ! ** ROOTS ** L.Dorais/O
ttawa UG/July 1989 !071
110 !!131
120 ON WARNING NEXT :: CALL
CLEAR :: PR$="PIO" !226
130 L$=RPT$("}",28):: E$=RPT
$(" ",168):: S$=RPT$(" ",8)!
069
140 CALL CHAR(120,"000000000
002050F",121,"1F102020404080
80",122,"018182C2C4646830",1
23,"0B0101",125,"FF")!167
150 DISPLAY AT(5,9):"x3y"&RP
T$("}",10):S$&"{z" :: GOSUB
280 !011
160 ACCEPT AT(6,12)VALIDATE(
NUMERIC)BEEP:N :: IF R>2 THE
N 180 !100
170 IF R=1 THEN AV=N :: GOTO
 220 ELSE AN=SQR(N):: GOTO 2
20 !151
180 AN=N^(1/R)!221
220 DISPLAY AT(12,8)BEEP:"="
;AN !203
230 DISPLAY AT(22,1):L$:" [A
]nother   [C]hange root   [P
]rint     [Q]uit" !181
240 CALL KEY(0,K,S):: IF S=0
 THEN 240 ELSE K=POS("ACPQ",
CHR$(K),1)!226
250 IF K=0 THEN 240 ELSE ON
K GOTO 260,260,270,290 !085
260 DISPLAY AT(7,12):E$:E$:E
$ :: IF K=2 THEN GOSUB 280 :
: GOTO 160 ELSE 160 !183270
OPEN #1:PR$ :: PRINT #1:S$&"
  __________":S$&" "&STR$(R
)&"/" :: PRINT #1:S$&"\/ ";N
;TAB(26);"=";AN:"" :: CLOSE
#1 :: GOTO 240 !210
280 ACCEPT AT(5,10)VALIDATE(
"123456789")SIZE(-1)BEEP:R :
: RETURN !008
290 END !139
```

NEWSBYTES

MUG plans set

The Cleveland area TI99/4A users groups, comprising the TI-CHIPS and the Northcoast 99ers, have announced further details of the Multi-Users Group Conference May 25 at the National Guard Armory in Brookpark, Ohio.

Setup will be 3:30-8 p.m. May 24, with the National Guard providing its regular after-hours security on the premises. At 8 p.m. a social get-together will be held at the Middleburgh Heights Recreation Center, with snacks and nonalcoholic beverages. Food will be available during the conference May 25.

The conference is free to attendees and vendors. However, organizers ask that reservations be made as early as possible.

For information, contact Glenn Bernasek, 13246 Harper Rd., Strongsville, OH 44136; phone: (216) 846-0865 (after 9 p.m. EST); e-mail: dd314@cleveland.freenet.edu.

MLM enhanced

Bill Gaskill provided the following information on an enhancement he has created for his Mailing List Manager (MLM) program. It applies to version 1.2's BROWSE program and adds a GOTO option to the BROWSE program so another record number can be entered when one is already on the screen. This means a user who currently has record 25 on screen, say, can press G for GOTO or N for NUMBER and then enter a record number to jump to for printing or editing.

Users can add the enhancement to their copies of MLM by loading the BROWSE program, then entering:

ELSE IF K=71 THEN 1100 ELSE IF K=78 THEN 1100

to the end of line 280. Because of the length of line 280, users will probably have to bring it up under the FCTN-8 editing technique to get the extra code in, Gaskill says. He says those who are unfamiliar with that technique in Extended BASIC can create a line 282 and add:

IF K=71 THEN 1000 ELSE IF K=78 THEN 1100

Next, create line 1100 at the end of the BROWSE program.

```
1100 DISPLAY AT(24,1):"go-to
-record:";R :: ACCEPT AT(24,
15)SIZE(-5)VALIDATE(DIGIT):R
:: IF R<=B THEN 250 ELSE IF
R>B THEN R=B-1 :: GOTO 250
```

PRINTED MATTER
ONLY

TO:
TIBUG

18 ZAMMIT STREET
DECEPTION BAY 45Ø8, QUEENSLAND.

SENDER:

Secretary - TIUP (Inc.)
2Ø Hudson Street,
Bayswater 6Ø53.
Western Australia.

INDEX