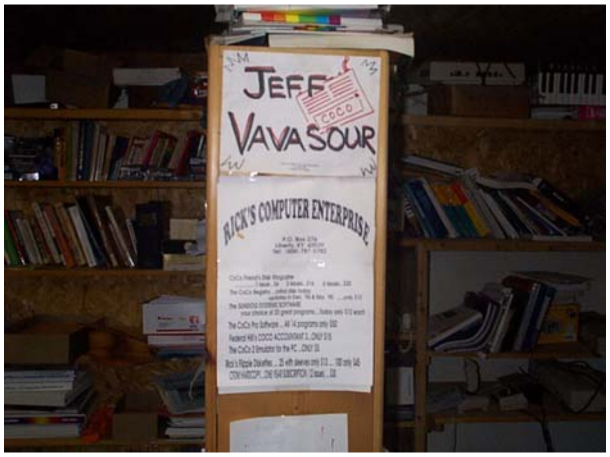
CoCoNutz! E-Zine

March / April 2008

An Interview with Rick Cooper from CoCo Friends Disk Magazine



Rick Cooper's magazine was an inspiration that sparked an idea I wanted to continue. I knew in the begging that dad loved CFDM and I wanted to relive it and know why. I read through issues of CFDM where my dad posted, and I come to realize what was so special about it. It wasn't like any other magazine, it was a family event. People here were so close knit and held together; even though they had probably never met each other. They shared much more than just the color computer, they shared a friendship. These were people who didn't want to lose their community and this gave them a way to keep in contact. I wanted to see what it was like to be part of such a great group of people where money didn't stop them from giving to each other. The first time I tried to find anything out about my dad I had contacted Mr. Cooper via the phone and he was very nice and spoke to me a little bit about my dad when he had come to visit him. He was very nice to take a strange phone call that day and to talk with me. For that I will always remember him and his wife for their kindness. So Rick you truly are the reason I got into the coco scene and you truly are the reason I started my E-Zine. So here's to you Mr. Cooper! Viva la coco!

Let me preface this by saying that my answers will be given to the best of my remembrance…this is going back a long way!

ED: So when did you start CoCo Friends Disk Magazine (CFDM)?

RC: I believe the first issue was January of 1993. My mother died at the end of that month after a very short period of illness with cancer. The February issue did come out as scheduled despite this very trying ordeal.

ED: How did you think of the Idea for it?

RC: Rainbow was fading, the computer was no longer being produced, and basically I was so selfishly involved with the computer that I did not want to let it go. There were still a lot of talented people out there, and I thought a significant number that still enjoyed their CoCos. I loved to program. So these factors probably brought on the idea, and it just blossomed in a very short period of time.

ED: Was there anyone else who helped you with it in the beginning?

RC: From the beginning I knew that I could not possibly produce all the programs and articles needed to fill the issues that were to come. I needed help…and quality help…so, I went thru the then recent issues of Rainbow looking for articles and programs that caught my fancy wrote down the author's names and addresses and wrote a form letter with my plan for CFDM. I don't remember exactly how many responded (probably 50% at least). A few of the responders said it sounded like a good idea, but they had moved on. One notable responder, Jim Bennett of New York, was hesitant at first, but later joined us and turned out an amazing amount of contributions to the magazine. Probably 15-20 said "Yes" from the start…and followed that up by contributing lots of quality material.

ED: How did you spread the word about it?

RC: Before Rainbow folded, I took out a few ads and received good response. Word of mouth, attending a few CoCo fests in Atlanta, and affiliating with the few existing newsletters at that time were about the only ways I knew to get the word out.

ED: Who were some of the people you remember from it?

RC: I remember many…but, I can't attempt to list them all here because I would leave some out. Here are a few names though…Norm Barson, George Quellhorst, Jim Gibbons, Jim Bennett, Sock Master, H. Allen Curtis, Stuart Wyss Gallifent, Arthur Hallock, Ray Berney, Jim Davis, Jeff Vavasour, Godfrey Moll, Keiran Kenny, Harold Moenich, Dale Kramer, William Astle, Herb Forger, Nickolas Marentes, Ben Walker…and the list goes on…

ED: Was there a theme that you tried to keep with it?

RC: I guess you could say the theme was 'Friends sharing with Friends'. Because that's exactly what we did. It was a medium where you could share your creations with others who could appreciate your talents and hard work.

ED: What was the idea or concepts for the disk magazine?

RC: The magazine was a disk magazine originally with a hardcopy version (CFDM Hardcopy) added some time after the first year. Arthur Hallock of Texas shared with me how he was transferring the magazine to his PC and using his editing skills to produce a hardcopy of the magazine. I thought it looked good and we made a deal where he received the first copy, made the hard copy, sent it to me for mass production. I believe the Friends had a choice to add Hardcopy to their subscription for an additional fee.
We had sections to the magazine like: Active CoCo, Advertisements, Articles of the Month, Art Gallery, Family Tree, Forum, From the Editor, Letters to the Editor, Potpourri, Programs of the Month, Reviews, and Questions & Answers.
Every issue was on a double sided disk with the magazine on one side and programs, art, etc. on the back. Every third month there was a double sided bonus disk with more programs, etc.

ED: When did you start holding the picnics?

RC: There were two picnics. I can't recall if they were in consecutive years or not. The first was in 1995.

ED: What were those like?

RC: Unbelievable! I cannot describe how wonderful it was to have the Friends visit me in Liberty, Kentucky. We held them at the school where I was principal. We catered a meal. We used the library for the Friends to display items they had for sale. We had demonstrations and presentations or just short talks from almost every visitor. They came from as far as New York, Texas, and Washington State. For the first picnic we flew Jeff Vavasour down from Vancouver, Canada as our special guest. Just great!

ED: How many of them did you have?

RC: Two.

ED: Did you ever meet any of the people that submitted stuff for your disk magazine?

RC: Yes, mostly at the picnics, but also at CoCo fests in Atlanta.

ED: How were submissions made?

RC: I had created a submission process (that some folks didn't follow J), with a special submission disk format and a paper form to explain their submission.

ED: How did the bonus disks evolve?

RC: The bonus disks came about for two reasons. For a long time we just had too much good material for the one side of the usual magazine disk. The second reason was that I felt if the Friends went to the trouble of putting their work together and submitting it, they deserved to be published if at all possible.

ED: What are some of your favorite submissions about from the disk magazine?

RC: There were so many…but, I really enjoyed much of the art from people like our cover artists Jim Gibbons, Norm Barson, and Jim Bennett. Jim Bennett also did some very creative and educational programs. H. Allen Curtis created wonderful Solitaire games! Godfrey Moll used my jigsaw puzzle to make enjoyable puzzles for everyone. George Quellhorst did wonderful articles on programming. Sock Master gave us some unbelievable graphics demos. Stuart Wyss Gallifent gave us stories, fractal programs, and many other types of material. Everyone gave of themselves even if it was just a bio for the Family Tree section. THESE WERE QUALITY PEOPLE!!

ED: Did you have any contest for the disk magazine?

RC: Yes…the one that comes to mind was the CFDM Logo contest. I wrote a little program that the Friends used to draw a logo for the magazine. We probably had 50 entries…many of them very good. Then we had a vote of all the Friends to see which would win. Jim Bennett's won and became our official logo. One note here, Ray Berney had some CFDM logo patches made and sold them probably at cost to all who wanted one. That's just one example of how the Friends took the magazine and extended it's enjoyability!

ED: What are some of your favorite memories from back then?

RC: My favorite memories are the people. How wonderful they were! How hard they worked together to make something not only worthwhile, but 'Oh so enjoyable for us all'!
You didn't ask this question, but I'm going to address it. My least favorite part of being involved in this adventure was when I would receive word of the passing of one of the Friends. It would break my heart! And that kind of news came often….I'm sure that was because so many of the Friends were of advanced years to begin with….
ED: Got any pictures from the picnics?

RC: Actually I hired one of my aides to video the first picnic. I made the tapes available and recently Jeff Vavasour took the time to put it on DVD for me. I really enjoyed looking back at the picnic in this way. Others took their own personal pictures and some were shared with me.

ED: What are some of the favorite moments of the picnics?

RC: Putting real faces with names. Picking up Jeff Vavasour from the airport with Stuart Wyss and Ray Berney with me. We had no idea what Jeff looked like (likewise for him). We made a welcome sign (which is still taped to a shelf about 10 feet from me as I write this…it's never been taken down since 1995!) to get Jeff's attention…..He walked right by it!! So…my favorite moments were the people…no question about this!

ED: Ok and now for personal info:
Where do you live?

RC: I still live at 30 Middleburg Street in Liberty, Kentucky.

ED: Do you have kids and pets?

RC: I have a wife and 3 daughters ages 25, 21, and 19. The oldest, Kristin, was married about 2 years ago and presented me with my first grandchild on July 10, 2007. That's Lauryn Elizabeth Hendrix. Kasey and Kayla are getting ready to start back to college at Eastern Kentucky University. We have two golden retrievers (Kola and Charlotte), and (at the moment) four outside cats!

ED: How old are you?

RC: I'll be 59 in 5 days (August 4th).
ED: What was your first coco?
RC: My first CoCo was the CoCo I. I got it in 1981 and fell in love. It cost over $500 bucks and used a tape player for storage! I believe it ran faster that the PCs I have today !

ED: What was your main coco setup like?

RC: 512k with 2 double-sided drives and a printer.

ED: What are your present day hobbies?

RC: I now use only PCs. I retired from the education system 6 years ago. I have written a program that is used by many of the Family Resource Centers and Youth Service Centers in the Kentucky educational system. (About 400 of the 800 centers in Kentucky have purchased the program.) I'm busy with tech support and improving the program.
My other hobbies would be playing chess on the internet and singing gospel music. I also enjoy reading (when possible), listening to music, watching tv, and spending time with my family.

ED: Do you still remain active with the coco?

RC: No…sad to say but, when I turned CFDM over to Jim Davis I decided it was best to separate myself from the magazine and let him finish the task without looking over his shoulder. I do still have CoCos and CFDM stuff all over my office. And…sometimes I get a call out of no where for help….Just last week Richard Hult of Texas called me. His RGB monitor had quit and the CoCo is still his only computer. Richard said he was 70 years old now and sure could use another monitor. I found two in my basement and made him a reasonable offer. He was really grateful for the chance to continue with his CoCoing!

ED: Did you ever attend any coco fest back then?
RC: I attended 2 or 3 Atlanta Fests and 1 fest in Pennsylvania.

Recursive Programming
by
Robert Gault

A recursive routine is one which calls itself. That seems simple enough but is it? Is a FOR/NEXT loop recursive? It does, after all, call itself. The answer is no. In general, a FOR/NEXT loop is not recursive but iterative; a linear repetition. Hmmm. This could get confusing. Perhaps it will be better to start with some examples and then work back to the definition. Besides, this gives me the opportunity to demonstrate some of my favorite graphics; the patterns in Benoit B. Mandelbrot's "The Fractal Geometry of Nature."

All of the graphics in this article are generated using recursion. In addition, all of the graphics patterns have fractal dimensions. The term Fractal was coined by Mandelbrot and literally means broken into irregular fragments. If a line has one dimension and a square two dimensions, this can be rephrased. Lines have a dimension (D) = 1, squares have D = 2, and fractals have 1<D<=2. Fractals therefore have very unusual properties.

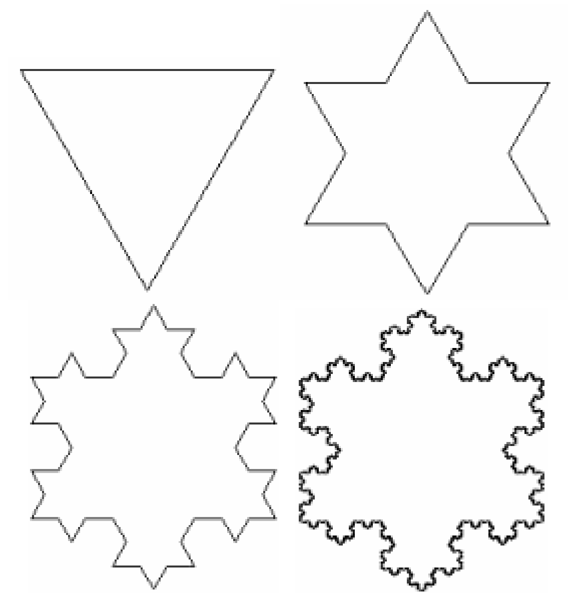


Figure 1

Look at the series of diagrams in figure 1. This is the Koch curve, a famous mathematical "monster." It is generated by continuously pasting smaller and smaller triangles upon the sides of the previous figure. Each new set of triangles being 1/3 the size of the previous triangles. The limiting curve has infinite circumference and finite area.

Readers of Rainbow may remember seeing the patterns in figure 1 previously on the title page of "Fascinating Fractals" by R. Delbourgo, July 1983. The programs presented there were quite complex, using many large arrays (ex. 256x256), and limited to three or four levels of detail because recursion was not used. Despite the art work, the programs did not generate figure 1 at all; perhaps because it is too difficult by non-recursive methods. By using recursive programming it is easy to go to 20 or more levels of detail if needed. This could far exceed the resolution of Coco's screen in many cases. The final image in figure 1 uses 6 levels of recursion and has reached the limits of screen resolution. However, one of the images in figure 5 required about 18 levels of recursion.

Notice that the Koch snowflake can be broken into three equal parts; it is "three" sided. If we can draw one side, we can draw the curve.

This article's main program with its modules are written in BASIC-09. Recursion is easier in this and similar languages. However, listing #1 is written in Extended Color Basic so that all readers can enjoy fractal art. With listing #1 as a guide, readers should be able to convert the remaining data sets from BASIC-09 to Extended Color Basic if desired. Note that with careful planning, the more advanced languages are not needed for recursive programming. Run Listing #1 or

Listing #2 (first pattern). I always find the results amazing!

Listing #1 should be easy to follow. Notice that there is only one decision making step. If the level (LV)= 1, then a line is drawn. If the level is higher, subroutine SIDE calls itself. This occurs in the GO STRAIGHT section. Note that the recursive call decreases the level (LV) which does not increase until level #1 is reached and a line drawn. That is simple enough. Now try to predict in your head the flow of the program for three (3) levels. Give up? Turn on the tracing function (TRON) and run the program. This will work best if you have left out the REM statements. Don't feel badly if you get confused. I can't predict in my head past two levels and I wrote the program!

The key to recursion is the presence of two types of variables, local and global. Global variables allow communication between levels of recursion. Local variables apply to one level only. This can be a handy concept even if you are not using recursion.

In Listing #1, only one variable is local LG(x). Variable LV is effectively local through a programming trick. Since the other variables are global, a change at any level is seen at all other levels. Length (LG) however, has a specific value at each level. The values of LG can therefore be calculated in the SETUP section.

BASIC-09 makes the use of local and global variables very easy. Parameters, which are the contents of the parentheses () in the RUN statements, pass variables back and forth between the modules. When the parameter is a variable, the parameter is global. When the parameter is an equation, the parameter is local to the called module.

In the Basic-09 Listing #2, there are five types of modules SETUP, CURVE, SIDE(s), FORWARD, and MOVE. CURVE is designed to have one set of data statements for each pattern. While this makes the program complex, this newsletter does not have to print many duplicate listings for the large number (21) of patterns. They are written to run in Level II Windows, not VDG screens.

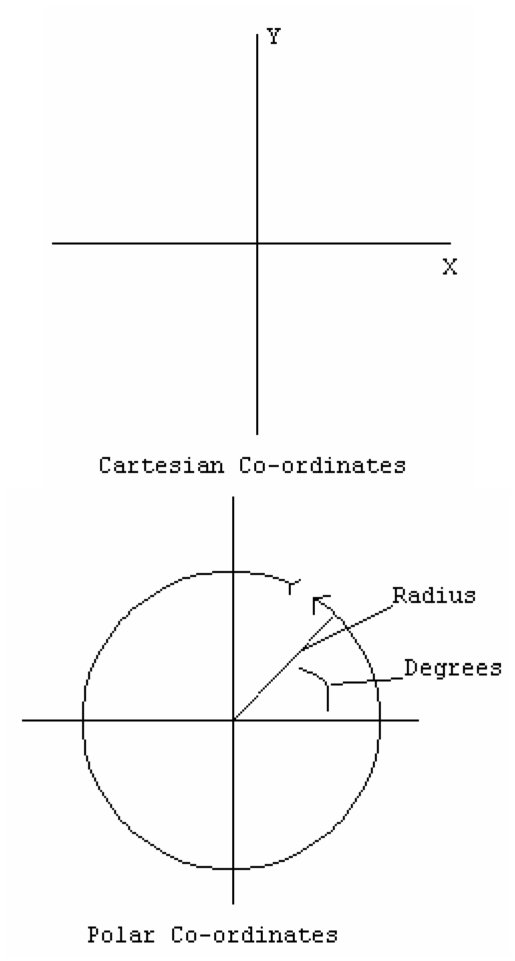


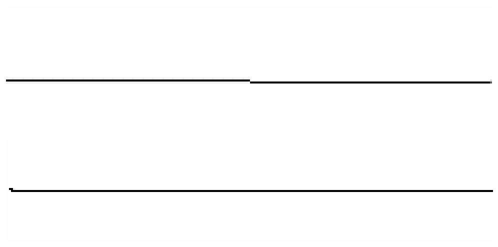
Figure 2

For the purposes of these programs, the graphics screen has been defined using the polar coordinate system as in figure 2. MOVE and FORWARD convert from the polar to Cartesian coordinates; those used by ECB and Basic-09. SETUP, CURVE, and SIDE use the polar system because it's easy to express a move as a direction and length: ex. "Go one mile NE."

To create new patterns, just add new data-restore sets to CURVE. Data sets are structured as follows: original direction, # of pattern sides at level #1, # of angles in data, angle between sides, set of angles used in SIDE, # of data items in a SIDE sequence, sequence length, set of sequence commands, starting x position, y position, length, length decreasing factor.

At the end of the article, figure 5 shows some of the initiating patterns and some of the final patterns. Bet you can't match them up.

It is interesting to compare the quality of the LINE commands in BASIC-09 vs Extended Color Basic. With five levels in the Koch snowflake, it is easy to see that ECB is the more pleasing of the two.



TOP: Basic ROMs
Bottom: Basic-09 OS-9
Figure 3

BASIC-09 (actually OS-9) draws a slightly non-symmetrical line. When line lengths approach screen resolution, noticeable mismatches are apparent. If you draw the following line (0,96 to 255,97) in ECB and Basic-09, you will see how Basic-09 is unsymmetrical; figure 3. (By the way, you can use DISPLAY directly from OS-9 to draw your line.)

The second half of this article will go into detail about this line drawing problem. These programs were written before OS-9 Level II was released and I had created patches for CCIO rev 1.01.00. Now that the source code for NitrOS-9 is available I might see about patching it to get symmetrical diagonal lines.

Listing #2 is a large program and needs about 8000 bytes of workspace to function. Be sure to request this room from Basic-09. Once entered and saved, PACK the modules to get maximum work area. If this does not give the necessary room, you can switch to RUNB, unlink OS-9 modules or reduce the number of CURVES data statements. None of the above problems should exit if OS-9 Level II is being used.

Be sure to always run the SETUP module first PACKed or unPACKed. If you save or pack all modules under one name, be sure your ordering puts SETUP as the first module to run.

Listing 1 for the Basic ROMs

```
1 PCLEAR4:'NOT NEEDED FOR COCO3
HI-RES SCREENS
10 'KOCH TRIADIC SNOWFLAKE BY
ROBERT GAULT
12 'WRITTEN SEPTEMBER, 1986 BY
ROBERT GAULT
20 'THIS PROGRAM CAN BE
MODIFIED FOR OTHER PATTERNS.
SEE TEXT
22 'COCO3 USERS; RUN FROM
WIDTH32 SCREEN
23 'OR EDIT INDICATED LINES
30 REM SETUP
40 REM ***********************
50 DIM LG(10)
60 CLS:PRINT"HOW MANY LEVELS OF
DETAIL? 1-10": INPUT"MAXIMUM
PRACTICAL (6) #=";LV
70 IFLV<1 OR LV>10 THEN 60
80 PMODE4,1:PCLS1:
SCREEN1,1:'FOR COCO1 PMODE
GRAPHICS
81 'PALETTE0,63:PALETTE1,0:
HSCREEN4:HCLS0:HCOLOR0,1:' FOR
COCO3
90 PI=3.14159265:RD=PI/180
```

```
99 REM ***********************
100 REM LG STANDS FOR LENGTH,
AG FOR ANGLE, LV FOR LEVEL
110 REM POKE IS OPTIONAL
SPEEDUP
111 REM **********************
120  POKE&HFFD7,0:' COCO3 USE
POKE&HFFD9,0
121 REM **********************
130 REM AS THE LEVEL OF DETAIL
INCREASES, LG=LG/3
131 REM **********************
140 FOR I=LV TO 1 STEP-
1:LG(I)=162/3^(LV-I):NEXT I
150 AG=0
160 X=50:Y=50:' FOR PMODE
SCREEN
161 X=120:Y=50:' FOR COCO3 HI-
RES SCREEN
170 REM **********************
180 REM POLYGON
190 REM **********************
200 FOR I=1 TO 3
210 GOSUB330:   REM SIDE
220 GOSUB630:   REM TURN RIGHT
230 NEXT I
240 POKE 65494,0
250 REM***********************
260 REM UNDOCUMENTED INKEY$
270 REM REPEAT PROGRAM
280 REM***********************
290 EXEC &HA1B1
291 'HSCREEN0:' FOR COCO3
292 GOTO60
300 REM***********************
310 REM SIDE
320 REM**********************
330 IF LV=1 THENGOSUB500:RETURN
340 GOSUB450:   REM GO STRAIGHT
350 GOSUB580:   REM TURN LEFT
360 GOSUB450:   REM GO STRAIGHT
370 GOSUB630:   REM TURN RIGHT
380 GOSUB450:   REM GO STRAIGHT
390 GOSUB580:   REM TURN LEFT
400 GOSUB450:   REM GO STRAIGHT
410 RETURN
420 REM***********************
430 REM GO STRAIGHT
440 REM***********************
450 GOSUB680:GOSUB330:GOSUB720
460 RETURN
470 REM***********************
480 REM DRAW AND RETURN NEW X,Y
490 REM***********************
500 X2=X+COS(AG*RD)*LG(LV):'
FOR PMODE4 GRAPHICS
501 'X2=X+2*COS(AG*RD)*LG(LV):'
FOR COCO3 HI-RES GRAPHICS
510 Y2=Y+SIN(AG*RD)*LG(LV)
520 LINE(INT(X+.5), INT(Y+.5))-
(INT(X2+.5), INT(Y2+.5)),PRESET
521 'COCO3 USERS MAKE 520 HLINE
530 X=X2:Y=Y2
540 RETURN
550 REM **********************
560 REM TURN LEFT
570 REM **********************
580 AG=AG-60
590 RETURN
600 REM **********************
610 REM TURN RIGHT
620 REM **********************
630 AG=AG+120
640 RETURN
650 REM **********************
660 REM DESCEND ONE LEVEL
670 REM **********************
680 LV=LV-1:RETURN
690 REM **********************
700 REM ASCEND ONE LEVEL
710 REM **********************
720 LV=LV+1:RETURN
```

Listing2 for Basic-09

```
PROCEDURE move
PARAM x,y,length:REAL
PARAM angle:INTEGER
DIM x2,y2:REAL
DEG
y2:=y+SIN(angle)*length
x2:=x+2.*COS(angle)*length
x:=x2 \y:=y2
```

```
PROCEDURE forward
PARAM x,y,length:REAL
PARAM angle:INTEGER
DIM x2,y2:REAL
DEG
y2:=y+SIN(angle)*length
x2:=x+2.*COS(angle)*length
RUN gfx2("line",FIX(x),
FIX(y),FIX(x2),FIX(y2))
x:=x2 \y:=y2

PROCEDURE side
TYPE figure=direction,sides,
angle(5),sequence(60):INTEGER;
fraction:REAL
PARAM level:INTEGER;
polygon:figure;
x_pos,y_pos,length:REAL
DIM i:INTEGER
IF level=1 THEN
RUN forward(x_pos,y_pos,
length,polygon.direction)
ELSE
FOR i=1 TO polygon.sequence(1)
ON polygon.sequence(i+1) GOSUB
1,2,3,4,5,6,7,8,9,10
NEXT i
ENDIF
END
1 RUN side(level-
1,polygon,x_pos,y_pos,
length*polygon.fraction)
RETURN
2 RUN side2(level-
1,polygon,x_pos,y_pos,
length*polygon.fraction)
RETURN
3 RUN forward(x_pos,y_pos,
length,polygon.direction)
RETURN
4 polygon.direction:=
polygon.direction+polygon.angle
(2)
RETURN
5 polygon.direction:=
polygon.direction+polygon.angle
(3)
RETURN
6 polygon.direction:=
polygon.direction+polygon.angle
(4)
RETURN
7 polygon.direction:=
polygon.direction+polygon.angle
(5)
RETURN
8 length:=length*1.555
RETURN
9 RUN move(x_pos,y_pos,length*
polygon.fraction,polygon.direct
ion)
RETURN
10 RUN side(level-
1,polygon,x_pos+.0,y_pos+.0,len
gth*polygon.fraction)
RETURN

PROCEDURE side2
TYPE
figure=direction,sides,angle(5)
,sequence(60):INTEGER;
fraction:REAL
PARAM level:INTEGER;
polygon:figure;
x_pos,y_pos,length:REAL
DIM i:INTEGER
IF level=1 THEN
RUN
forward(x_pos,y_pos,length,poly
gon.direction)
ELSE
FOR i=1 TO polygon.sequence(1)
ON
polygon.sequence(i+1+polygon.se
quence(1)) GOSUB
1,2,3,4,5,6,7,8
NEXT i
ENDIF
END
1 RUN side(level-
1,polygon,x_pos,y_pos,length*po
lygon.fraction)
RETURN
```

```
2 RUN side2(level-
1,polygon,x_pos,y_pos,length*po
lygon.fraction)
RETURN
3 RUN
forward2(x_pos,y_pos,length,pol
ygon.direction)
4
polygon.direction:=polygon.dire
ction+polygon.angle(2)
RETURN
5
polygon.direction:=polygon.dire
ction+polygon.angle(3)
RETURN
6
polygon.direction:=polygon.dire
ction+polygon.angle(4)
RETURN
7
polygon.direction:=polygon.dire
ction+polygon.angle(5)
RETURN
8
polygon.direction:=polygon.dire
ction+polygon.angle(6)
RETURN

PROCEDURE curve
DIM i,j:INTEGER
TYPE
figure=direction,sides,angle(5)
,sequence(60):INTEGER;
fraction:REAL
DIM x_pos,y_pos:REAL;
polygon:figure
PARAM level,line:INTEGER
ON line GOSUB
101,102,103,104,105,106,107,108
,109,110,111,112,113,114,115,11
6,117,118,119,120,121
READ
polygon.direction,polygon.sides
,j
FOR i=1 TO j
READ polygon.angle(i)
NEXT i
READ j
FOR i=1 TO j
READ polygon.sequence(i)
NEXT i
READ
x_pos,y_pos,length,polygon.frac
tion
x_pos:=x_pos+x_pos
y_pos:=191.-y_pos
FOR i=1 TO polygon.sides
RUN
side(level,polygon,x_pos,y_pos,
length)
polygon.direction:=polygon.dire
ction+polygon.angle(1)
NEXT i
END
1 DATA 60,3,3,120,-60,120,8,7
DATA 1,4,1,5,1,4,1
DATA 126.,191.,162.,1./3.
2 DATA 30,6,3,60,60,-120,8,7
DATA 1,4,1,5,1,4,1
DATA 126.,191.,94.,1./3.
3 DATA 0,1,3,0,-90,90,15,14
DATA
1,4,1,5,1,5,1,1,4,1,4,1,5,1
DATA .0,96.,255.,.25
4 DATA 0,1,3,0,-90,90,31,30
DATA
1,4,1,1,5,1,1,5,1,5,1,4,1,4,1,1
,5,1,5,1,4,1,4,1,1,4,1,1,5,1
DATA 10.,96.,235.,1./6.
5 DATA 0,1,3,0,-90,90,59,58
DATA
4,1,5,1,4,1,4,1,5,1,5,1,1,4,1,5
,1,5,1,1,5,1,4,1,4,1,1
DATA
5,1,1,4,1,1,5,1,5,1,4,1,1,4,1,4
,1,5,1,1,4,1,4,1,5,1,5,1,4,1,5
DATA 50.,96.,162.,.125
6 DATA 60,3,5,120,-120,120,-
30,30,8,7
DATA 6,1,5,1,4,1,7
DATA 128.,170.,140.,1./SQRT(3)
7 DATA 45,4,5,90,-90,90,-
27,27,8,7
DATA 6,1,5,1,4,1,7
DATA 128.,181.,120.,1./SQRT(5)
8 DATA 36,5,5,72,-72,72,-
22,22,8,7
DATA 6,1,5,1,4,1,7
DATA 128.,181.,92.,1./SQRT(6)
9 DATA 30,6,5,60,-60,60,-
19,19,8,7
DATA 6,1,5,1,4,1,7
DATA 128.,188.,92.,1/SQRT(7)
10 DATA 0,1,3,0,-90,90,18,17
DATA
1,4,1,5,1,5,1,5,1,4,1,4,1,4,1,5
,1
DATA 30.,96.,190.,1./3.
11 DATA 0,1,3,0,-45,45,7,6
DATA 4,1,5,5,1,4
DATA 70.,46.,100.,1./SQRT(2)
12 DATA 0,1,3,0,-45,45,13,6
DATA 4,2,5,5,2,4,5,1,4,4,1,5
DATA 40.,46.,160.,1./SQRT(2)
13 DATA 0,1,3,0,-45,45,13,6
DATA 4,1,5,5,2,4,5,1,4,4,2,5
DATA 60.,86.,140.,1./SQRT(2)
14 DATA 30,6,3,60,-120,60,14,13
DATA 1,4,1,5,1,5,1,5,1,5,1,4,1
DATA 128.,158.,62.,1./3.
15 DATA 0,3,3,-120,-
120,120,10,9
DATA 1,4,1,5,1,5,1,4,1
DATA 40.,10.,180.,1./2.
16 DATA 0,1,3,0,-60,60,15,7
DATA 4,2,5,1,5,2,4
DATA 5,1,4,2,4,1,5
DATA 18.,.0,220.,.5
17 DATA 0,4,3,-90,-90,90,16,15
DATA
1,4,9,1,5,1,5,1,5,1,4,9,4,1,1
DATA 33.,.0,190.,1./3.
18 DATA 60,1,5,0,60,-60,-
19,19,35,17
DATA
6,1,4,2,4,4,2,5,1,5,5,1,1,5,2,4
,7
DATA
6,5,1,4,2,2,4,4,2,4,1,5,5,1,5,2
,7
DATA 128.,188.,160.,1./SQRT(7)
19 DATA -90,1,3,0,-44,44,8,7
DATA 3,4,10,5,5,10,4
DATA 128.,.0,86.,.59
20 DATA 0,1,3,0,-90,90,17,8
DATA 4,2,5,1,1,5,2,4
DATA 5,1,4,2,2,4,1,5
DATA 32.,.0,191.,.5
21 DATA 0,1,3,0,-87,174,10,9
DATA 1,4,8,1,5,1,4,8,1
DATA 10.,.0,230.,.28
101 RESTORE 1 \ RETURN
102 RESTORE 2 \ RETURN
103 RESTORE 3 \ RETURN
104 RESTORE 4 \ RETURN
105 RESTORE 5 \ RETURN
106 RESTORE 6 \ RETURN
107 RESTORE 7 \ RETURN
108 RESTORE 8 \ RETURN
109 RESTORE 9 \ RETURN
110 RESTORE 10 \ RETURN
111 RESTORE 11 \ RETURN
112 RESTORE 12 \ RETURN
113 RESTORE 13 \ RETURN
114 RESTORE 14 \ RETURN
115 RESTORE 15 \ RETURN
116 RESTORE 16 \ RETURN
117 RESTORE 17 \ RETURN
118 RESTORE 18 \ RETURN
119 RESTORE 19 \ RETURN
120 RESTORE 20 \ RETURN
121 RESTORE 21 \ RETURN

PROCEDURE setup
ON ERROR GOTO 2
DIM ans:STRING[1]
DIM i,line,level:INTEGER
DIM length:REAL
DIM window:BOOLEAN
REPEAT
PRINT CHR$(12);
PRINT "Input the figure to draw
(1-21)."
INPUT "figure #=",line
UNTIL line>0 AND line<22
1 PRINT
REPEAT
```

```
PRINT "Input level to run ( # >
0 )"
PRINT "or sequence (0 yields 1-
5):"
PRINT
PRINT "-1 will restart program
for new figure selection"
INPUT "# ?",level
UNTIL level>=-1
IF level=-1 THEN RUN setup
ENDIF
PRINT CHR$(12)
window:=FALSE
RUN
gfx2("owset",0,0,0,80,24,1,0)
window:=TRUE
RUN gfx2("curoff")
IF level=0 THEN
FOR i=1 TO 5
PRINT CHR$(12)
RUN curve(i,line)
NEXT i
ELSE
RUN curve(level,line)
ENDIF
GET #0,ans
PRINT CHR$(12);
RUN gfx2("owend")
RUN gfx2("curon")
PRINT "Figure selected is ";
line
PRINT "Last level was "; level
GOTO 1
2 errnum:=ERR
ON ERROR
RUN gfx2("curon")
PRINT "Error number "; errnum
IF errnum=183 THEN
PRINT "Illegal window type;
must be ";
PRINT "graphics."
ENDIF
IF errnum=189 THEN
PRINT "Illegal coordinates."
IF window THEN
PRINT "Check data table."
ELSE
PRINT "Must be 640x192 window."
ENDIF
ENDIF
IF errnum=32 THEN
PRINT "Memory full; check mem
#."
ENDIF
PRINT
IF window THEN
RUN gfx2("owend")
ENDIF
END
```

Editor's note: any line in the previous Basic09 program which is **NOT** in **bold** type, is a continuation of the previous line, and should be typed as part of that line.
P.S. this applies to all source listings in this issue except "Numbersquare".

~~~oooOOOooo~~~

One-Liner

```
0 A=43350:B=140:FORS=1TO240: POKE
B,S:EXECA:NEXT:
FORS=240TO10STEP-15:
POKEB,S:EXEC:NEXT:FORS=50
TO200STEP.5:
POKEB,S:EXEC:NEXT:FOR
S=200TO10STEP-15:POKEB,S:
EXEC:NEXT:
FORS=1TO100STEP.25:POKEB,S:
EXEC:NEXT:S=145:
FORX=0TO1:POKEB,S:EXEC:
X=0:NEXT
```

Keep on Coco-in'
~~~

Derek's Pokes from the Past

Did you know you can output your disk directory to your printer on your Color Computer ? Well you can by entering the following Poke Command.

Poke111,254:DIR

Did you also know you can use the same command in some of today's emulators to output a disk image file directory to a text file on your PC?

Here is how using the MESS Emulator. (The Following assumes your using the MESSGUI Front end that now comes in each build of MESS)

1: Create a blank .txt file. In Windows right click in the directory you want the file to be saved in and select the New option then under the New Option select Text Document. You can name it whatever you want. I used coco.txt

2: Load MESS and single click to select which ever Color Computer Model you want to emulate. For my example I will use Color Computer 3. Now on the right side pane select the "Device View" tab.

3: The very 1st line is "Printer" and on the far right side of that line is a button with three dots. Click on that button and select "Mount" now browse to the file you created in step 1. The coco.txt file

4: Next mount your floppy disk image. In the same device view tab find the line labeled Floppy#0 and select the button with three dots and again use the mount command and select the disk image you want to use.

5: In the left pane double click on Color Computer 3 to start the emulator.

6: Once the emulator starts type Poke111,254:DIR and it will output the disk directory to the text file you created in step number 1.

How to read a disk directory in DISK EXTENDED COLOR BASIC – Part 2

by Bob Devries

In this part of my tutorial, I am going to add a few lines into the code I presented in the first part. These lines will add to the programme the ability to show which granules were used by each file, and also the size of each file in bytes.
The GAT (granule allocation table) is an array of 68 bytes. As we saw in part 1, the entry point to the chain of granules is provided in the directory entry for each file. The value that is in the cell of the GAT array is a pointer to the next cell, or a value to show that this cell is the last granule used by the file. This last special value is a value between 193 (&HC1) and 201 (&HC9). Sometimes, a value of 192 (&HC0) is used to show a granule is allocated but unused. This does not happen in the normal usage of DISK EXTENDED COLOR BASIC.

To enable the sample programme to show the granules used and the length of the file, I have added three lines into the code, and added a subroutine of 13 lines to the end.

The first line is added right at the beginning of the programme, and is there to allocate a numeric array to accept the granule numbers for each file as it is extracted.

```
15 DIM GA(68):REM ALLOCATE AN
ARRAY FOR THE GRANULE NUMBERS
```

The next line is the command to read the GAT sector into a string variable. The

GAT is on track 17 and sector 2, and is stored into string variable GT$.

```
105 DSKI$DN,TK,2,GT$,C$:REM GET
GAT SECTOR
```

The next line concatenates the two halves of the sector into one string variable, omitting only the last byte, which has no value. The next line concatenates the two halves of the sector into one string variable

```
106 GT$=GT$+LEFT$(C$,127)
```

The last insertion is line 245, which calls the subroutine to calculate the additional values which we need.

```
245 GOSUB 300
```

Lines 300 to 420 are the subroutine which do the actual work of finding the granules that are used by the file, and its actual length in bytes. Line 300, the start of the subroutine sets some variables to 0, because they will be re-used each time the subroutine is run.

```
300 GT=0:LF=0
```

Line 310 gets the value of the next cell in the GAT array in GT$, which is pointed to by the variable GR which is set in line 200. This is placed into variable NG.

```
310
NG=ASC(MID$(GT$,GR+1,1)):REM
GET NEXT GRANULE
```

The next line tests the value of NG to see if this granule is the last in the chain. There will be more granules in the chain if the value of NG is less than 192 (&HC0). A value of 192 is actually a special case which rarely shows up. The granule total counter (GT) is bumped up by 1, and the length of file counter has 2304 added to it. Remember from part 1, 2304 is the size in bytes of one granule. If the value of NG is 192 or more, then the code does a jump to line 350, to do the special handling of the end of granule chain situation.

```
320 IF NG<192 THEN
GA(GT)=GR:GT=GT+1:LF=LF+2304
ELSE 350
```

Line 330 puts the value of NG into the GR variable, so that it can be re-used.

```
330 GR=NG
```

Line 340 loops around to continue working down the chain of granules.

```
340 GOTO 310
```

Line 350. The code jumps to here when it detects an end of granule chain condition. Here it subtracts 193 from the value found in the GAT to get a value between 0 and 8; multiplies that by 256, which is the number of bytes in each sector, and adds the value from the LS variable which was calculated in line 210.

```
350 LF=LF+((NG-193)*256)+LS
```

Line 360 stores the granule number into the GA array which we created at the beginning of the programme. The offset into that array is given by variable GT, which is the granule total counter. GT is the index into the array.

```
360 GA(GT)=GR
```

Line 370 prints a title for the line which shows the granules.

```
370 PRINT#PR,”GRANS:”;
```

The next three lines, 380-400 are a FOR-NEXT loop to print the values of the GA array one after the other on the screen or printer, decided by the PR variable.

```
380 FOR X=0 TO GT
390 PRINT#PR,GA(X);
400 NEXT X
```

Line 410 appends a linefeed to the end of that, ready to print more information.

```
410 PRINT#PR
```

Line 420 prints the length of the file in bytes.

```
420 PRINT#PR,"LENGTH:";LF
```

And the last line returns us from the subroutine back to the main code.

```
430 RETURN
```

Here's a sample of what the programme produces:

```
FILENAME EXT TYP ASC 1ST LAST
                 FLG GRN SECT
EXTEND0 .BIN  2  B  0   193
GRANS: 0  1  2
LENGTH= 6337
EXTEND1 .BIN  2  B  3   193
GRANS: 3  4  5
LENGTH= 6337
EXTEND2 .BIN  2  B  6   193
GRANS: 6  7  8
LENGTH= 6337
MENUCUST.BAS  0  B  9   153
GRANS: 9
LENGTH= 1689
MENU    .BIN  2  B  10   215
GRANS: 10
LENGTH= 1751
```

I hope this two part tutorial has been useful to you. It was fun and educational for me to write, because although I knew how to do what was needed, I had never written the code before. They say you're never too old to learn. Maybe next time I'll add the code to find the START, END and EXEC addresses of a binary file as well.

Happy Coco-ing.

~~~oooOOOooo~~~

The CoCo/OS-9 Documentation Archive

What is it?

The purpose of the archive is to put as much CoCo, OS-9 and OSK information together before it is completely lost. This includes magazines, newsletters, ads, program manuals, etc. While the archive does contain some files (i.e. User Group Archives) it is mostly limited to documentation at this point. Eventually I expect this information to be 'folded in' with other projects or archives so that the CoCo/OS-9 community will have a definitive reference source.

How did it all start?

Well many years ago, in the mid '90s, I started a project to scan in the Bellingham OS-9 User Group newsletter, the "OS-9 Newsletter." Unfortunately, due to equipment and time restraints at the time I abandoned the project. So, fast forward to 2004 and the CoColist, where there was some discussion about the loss of CoCo and OS-9 Programs with the closing of the old on-line forums. I mentioned that I had a copy of the archive of the Bellingham OS-9 User Group library, but to my surprise, hardly anyone even knew of the existence of the Bellingham UG or its newsletter. So, as a service to the CoCo community, the old project was reincarnated and I started to scan the old "OS-9 Newsletter." As with a lot of projects, it started innocently enough, but soon took on a
~~~

life of its own. After I finished the "OS-9 Newsletter" I figured I might as well scan the manuals I could load into my sheet fed scanner. Then I started in on some books, and well, the rest is history.

The Contributors

Many people have contributed to the project, by either scanning documents I didn't have, or graciously allowing me to borrow their manuals and magazines. I have also included any manuals, magazines, and books I have found hunting on the internet. There are too many people to acknowledge here, but special thanks go to Boisy Pitre, Tim Lindner, Stephen H Fisher, Steve Ostrom, and Ken Carlin for their help and encouragement. A special thanks to Dennis Bathory-Kitsz for the FTP site and the new CoCoList!

So what's in the archive?

Well that would be a very big list, so I'll just list a few examples. The magazines include CFDM, CoCo Clipboard, Color Micro Journal, The World of 68 Micros, Micro-80, 80 microcomputing, TRS-80 MicroComputer News, Ninetimes, OSKer, Under Color, Uptime, and Mary Kramer's famous e-zine CoCoNutz!. There are newsletters from the Australian, European and US OS-9 UGs, as well as various Delphi, CompuServe, CoCoList and FIDO messages. The books include some of Bill Barden's works, "Inside OS-9", "The BASIC09 Tour Guide" and many others. There are more product manuals than I can begin to name, but some would include DISTO, Tandy's OS-9 Manuals, Various Game Manuals, Microware OSK Manuals, MM/1, Sub-Etha, Sundog, and the FHL TC/70. Many categories are incomplete, but hopefully in time the missing pieces will show up.

And last but not least, there is a separate section for the Dragon Computer, which includes Dragon User Magazine, Dragon World, Stop Press, and numerous manuals and pictures.

So where is the project now?

Well, the last official DVD release was V3.1, which was a special release strictly for the CoCoFEST! I donated a number of copies to Glenside to help with the FEST!, however I want to stress that the archive is strictly a non-profit venture for the benefit of the community. I make nothing off of this, and it actually costs me money to do the DVD releases. Most of the information in the archive has been uploaded to the maltedmedia FTP site, but due to my being on dial-up I just can't upload all the magazines. Release 4.0 is scheduled for a December or January release.

Currently the DVD set (5 DVDs) is only distributed to major contributors. If you want a DVD set, you'll have to warm up your scanner or allow me to borrow documentation to scan (all documents are returned.) This is my way of rewarding people since scanning documents is not exactly a fun task. I don't sell the set because I promised the contributors this would be a community project so it would be inappropriate for me to be involved in any kind of distribution (even for just the cost of the DVD media.) Perhaps Glenside would be willing to make copies available at the next CoCoFEST!, but that is something I'll have to discuss with them. Ideally, I would like a separate on-line home for the archive, but finances do not allow that at the moment. So, if anyone has a lot of server space and bandwidth free I would be highly appreciative.

How do I Contribute?

Anybody who wishes to contribute can contact me at 'os9project@1stconnect.com' and put 'archive' in the subject line to make sure I see it. Any kind of documentation is welcome, from flyers to CoCo related magazine articles! Also, anyone who is working on a similar project, please contact me so we don't duplicate our efforts.

Dean Leiber

Drawing Diagonal Lines with Computers

If you have tried both of the fractal generating programs, you will be aware of the differences in the line drawing routines of Extended Color Basic and OS-9/Basic09. I will be using the programs to demonstrate those differences, but you can just look at the figures.

There are several ways to draw lines with a computer. An excellent discussion of these methods is available in "Microcomputer Displays, Graphics, and Animation" by Bruce A. Artwick, 1985, Prentice-Hall. Other references are available in stores and libraries.

There are two techniques used by Coco to draw lines. The first, used in Extended Color Basic (ECB), is the Octantal Digital Differential Analyzer (OctDDA). The second, used in OS-9/Basic09, is the Quadrantal DDA (QuadDDA).

These tongue twister names mean that the programs do not use fractions or floating point math. Instead, the slope of a diagonal line is represented by a delta-x and delta-y value. An error term is maintained that measures the difference between the desired line and the actual line. As each point making up a line is generated, the error term is adjusted using the delta-x and delta-y values. The delta terms are the difference between the start and finish points of the line in the horizontal and vertical directions. By using the error and delta terms, the DDA line drawing procedures (there are several variants) are able to produce a stair step line that is a good approximation of the desired diagonal line.

The resolution of the graphics screen is the prime factor in line quality, but other factors are also important. Four additional criteria are fit, symmetry, reversal, and overlap. Reversal means the line is identical regardless of which end you start. Symmetry (or lack of) can be seen in figure 3. Fit and overlap are shown in figure 4.

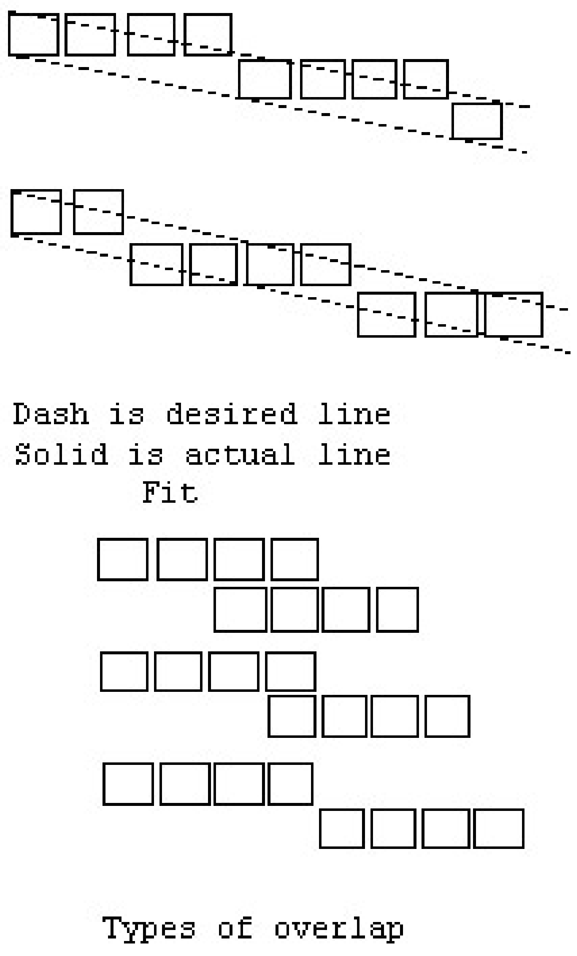
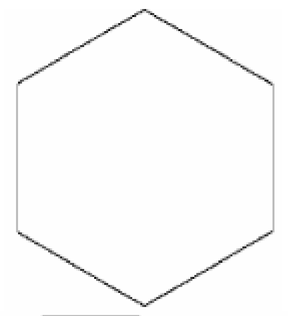
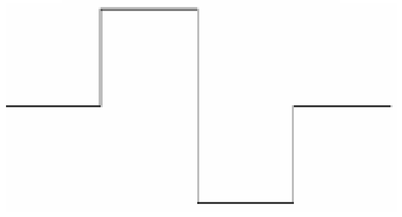
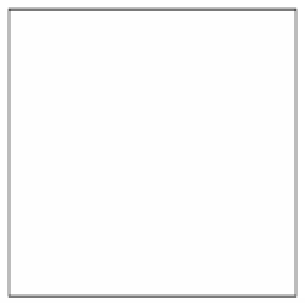
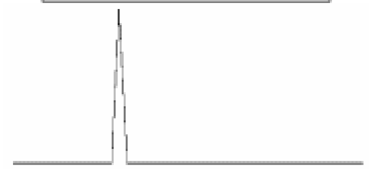
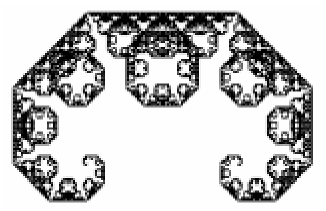
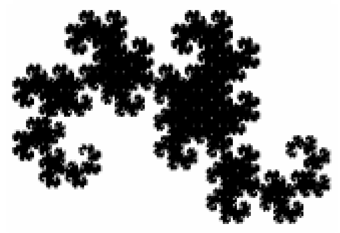
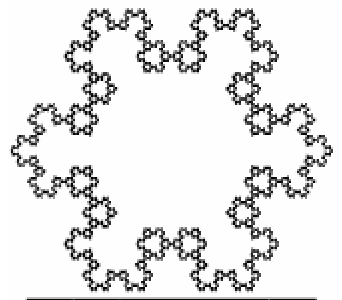
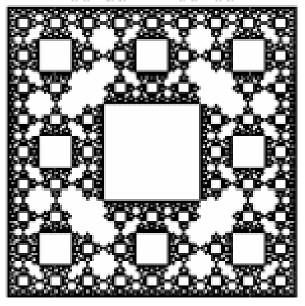


Figure 4

QuaDDA produces lines with one pixel overlap. The overlap increases apparent smoothness, but also makes the lines appear thicker than horizontal or vertical lines. As implemented in OS-9, the error term is started at zero which can cause a non-symmetrical line (fig. 3.) OctDDA as used in ECB is started with the error term half the value of the appropriate delta term. This gives correct symmetry. Since OctDDA has no stair step overlap, apparent ECB diagonal line thickness is closer to horizontal or vertical line thickness. The line, however, may not appear as smooth as with QuadDDA, a slight deficiency.

The best approach to solving this problem would be to alter the NitrOS-9 source code to incorporate system wide changes to OS-9 that would cover both VDG and Windows drawing routines. While selection of QuaDDA vs OctDDA is a matter of personal taste, the error term should be initialized at 1/2 delta.

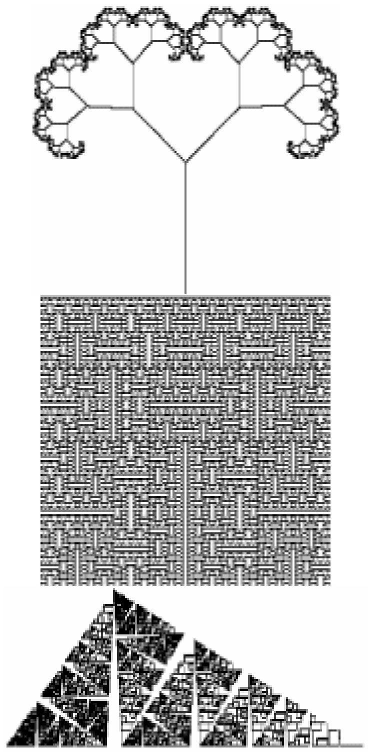


Figure 5

Wizards Den:

Game information & Download : http://nitros9.lcurtisboyle.com/wizardsden.html
Published by: Tom Mix Software
Author: Matt Harper
System Requirements: Color Computer 1, 2 or 3 Disk Drive, 64K RAM,
Joystick Support: Yes
Genre: Arcade, RPG

Ratings:

Graphics: 5/5
Sound: 3/5
Ease of Play: 5/5
Originality: 5/5
Fun Factor: 5/5

Wizards Den is an arcade and rpg type game produced by Tom Mix Software. In the game you play the role of a stoic adventurer out to recover the Gem of Damocles. The graphics are excellent. When you are outdoors there are trees and bushes to dodge around. Your character wields a mean looking sword and the graphics of the monsters are very well done. Once you enter the temple the graphics give you the real feeling of being in a classic RPG dungeon setting.

The sounds when you shoot your fireballs was a nice touch and when you were taking damage from the monsters the sounds were timed perfectly. One bothersome thing I found was as your health ticked down each second a sound would play and repeat every second and after a few minutes of play the repetitive sound was unwanted. There was no music in the game which I think could have added to the mood.

The game was very easy to start. The ability to play using a joystick was a real joy (pun intended). I found myself on the 1st level which is an outdoor area filled with trees, piles of bones and dens of bad guys. There were plenty of monsters to hunt and some goodies to find like magic rings and potions. I also found myself challenged when I would find areas where I could see monsters and treasures on the other side of a wall that was not easily accessible. I found myself needing to find my way in to slay the beasts and grab the loot. Once inside the temple and subsequent dungeon levels the game became more of a challenge as there were more monsters and suddenly I as the hunter became the hunted.

Overall this is a great game! If you enjoy games like Gauntlet, this game is for you. I highly recommend taking some time to enjoy this game. Now I am off to find that elusive Gem of Damocles.

* Game was reviewed using the MESS Emulator under the TRS-80 Color Computer 3 selection. Please send any questions or comments to dml_68@yahoo.com.

"Rat Attack"

Game authors: Wayne Wood and Gerry Casey

Publisher: T&D Software, Issue #28, July 1984

Runs in PMODE 4 in all 32K+ CoCos

Review by Richard Kelly

Introduction: There are **lots** of Nibbler wannabes out there, and not just in the CoCo market. This is my favorite of the bunch, though. Both the graphics and the sound are a step above Nibbler, as is the gameplay.

For those who haven't played Nibbler, the game involves eating all the gems before your bonus runs out. The more you eat, the bigger your tail gets. If you bite your tail, you die.

In "Rat Attack", you eat rats on the board instead of gems; some are alive and running around, while others are quite dead, displayed upside-down and not moving at all. There's also a "thru" exit where you go through one side of the maze, and come out the other. One more thing is added, too: A computer player that can be a real nuisance if you're not careful. He's a snake whose tail is just as deadly as yours.

Graphics: A. What can I say? The graphics top the Nibbler arcade game, and even that of many CoCo 3 games out there. The game isn't sparse on detail one bit; yet it's not overcrowded with decoration, either. The wire-thin font doesn't look very good, but that's about all there is to complain about. No one could ever be convinced that this game was home-grown; it has all the qualities of a real professional here. You'd swear this game was published by Tom Mix Software! No joke.

Sound: C+. No melodies of any sort, no music, and nothing's heard during the actual gameplay except snake-like drumbeats. Until I actually took the time to review this game though, I never really noticed, and I've played it for over ten years. So I guess the problem isn't **too** bad.

Animation: A-. Since the game was color-artifacted, the authors couldn't get away with animating the game the same way east and west as they did north and south. It just didn't show up on the CoCo this way. Maybe they flat ran out of time to change the east and west animation so it moved the same as the north and south animation. In any case, it's not the same, and the inconsistency did bug me sometimes. Thankfully, I only have to play a level or two before I'm too excited about the game to let the animation bother me. And you're hearing this from a **big** animation fan.

Game engine: A-. The game is color-artifacted, yet there's no Reset until the Screen Is This Color feature. For CoCo 3s, this isn't a problem, I suppose, but what about the rest? You'll be stuck with the game being the wrong color half the time, and to be honest, there's no real way to know what the game's "right color" is supposed to be.

Also, the game is often very picky about the player not getting too close to the computer snake's head. It gets very annoying at times.

Add to this, the graphics coding. Even though the game doesn't lag at any point, the code of the game does show its age a little, in that the game was written for a cassette-based system, and not a disk-based one. Like Grabber and a number of other ML-based games, this one has the "flashing colored line" problem – a problem on disk systems that shows up whenever an area near the northwest corner of the screen is drawn over. In Rat Attack, that would be every time the maze is redrawn. After a while, the game just stops working altogether.

Lastly, here are some features in the game that are going to give you a pause: "No Sound" and "Sound On". What's up with that? Doesn't every TV the CoCo is hooked up to have a volume control? I thought so. It seems someone has been writing for a different machine for so long, he/she forgot that CoCos don't need a "Sound On/off" feature.

Gameplay: A. The game was more interesting to me than Nibbler, despite how fond I am of the arcade game. First, there are three types of rats – The dead, the white-bellied, and the black-bellied. The white-bellied ones will often reverse direction and run right into your mouth. The black-bellied rats, on the other hand, are much smarter, and you have to trap them in order to eat them up. You can do this pushing them into a corner, trapping them where they'll get blocked by a dead rat, trap them in the Thru exit where you can eat the rat and go through to the other side of the maze, or you can trap the rat with your own tail (and die in the process if your bonus isn't high enough to have your tail bonus tallied up).

Another improvement is the number of mazes. It has a far greater number of levels than the Nibbler arcade game. In fact, I remember once going over Level 11 and the mazes still weren't repeating! Maybe the game gives you new mazes forever? I doubt it, but it sure has enough to keep me entertained for a while.

Another great thing about the gameplay is the computer-controlled snake. With him often getting in the way (and you having to find a way to maneuver around him or away from him), the same game never plays twice, even though the layout of all the mazes are the same. Thus, more replay value!

Overall: A. Not much to say that I haven't said here already. So I'll simply conclude the review by saying: It's high-quality programs like this that made me so fond of T&D Software in the first place. Cheers!

Purchase your Coco Mug from Coco3.com

The 17th Annual "Last" Chicago CoCoFEST!

April 12 & April 13, 2008
Saturday 10am-5pm; Sunday 10am-4:30pm

HOLIDAY INN & SUITES ELGIN rates are $80.00 plus 10% tax. YOU MUST REGISTER UNDER "COCOFEST!" to get this rate, so be sure to ask for the Glenside "CoCoFEST!" rate.

RESERVE YOUR ROOM EARLY as there is a limited block of rooms held until April 4th 2008
These rooms will be released to the general public and will not be available for the fest attendees.
Call the Elgin HOLIDAY INN & SUITES at 1-847-488-9000 for reservations.

General admission-all attendees:
Saturday & Sunday: $15.00 1st - $10 2nd & more.
Sunday Only: $10.00 1st - $5 2nd & more.
********* Children 10 and under - FREE *********

For further information, general or exhibitor, contact:
Tony Podraza, GCCCI Brian Goers, GCCCI
847-428-3576, VOICE 708-754-4921, VOICE
tonypodraza@juno.com
briang0671@sbcglobal.net

Double Back:

Published by: Tandy, Radio Shack
Author: Dale Lear
System Requirements: Color Computer 1, 2 or 3, Disk Drive, 4K RAM, 1 or 2 Joysticks
Joystick Support: Yes
Genre: Arcade, Puzzle

Game information:
http://nitros9.lcurtisboyle.com/doubleback.html
Play online here:
http://members.cox.net/javacoco/ (Doubleback is under the BIN tab)
Download here: http://discover-net.net/~dmkeil/coco/software/files/26-3091.ace

Ratings:

Graphics: 4/5
Sound: 4/5
Ease of Play: 5/5
Originality: 5/5
Fun Factor: 5/5

Doubleback is a game of speed and skill. The object of the game is to encircle different objects that appear using your joystick without touching them. The objects include roller skates, cherries, apples and magnets. This is not as easy as it sounds because the lines you make to encircle objects shrink as you try to encircle them, some of the objects move and if you’re not quick enough you will get too many items on the screen at one time which makes avoiding contact tough. Having multiple objects to encircle at one time can be a good thing as well, because you get a higher score if you encircle multiple objects. Graphics are nice (considering that the game can run on a 4K Coco), the game sounds and music are enjoyable and are used in such a way as to not become overly repetitive. The concept of the game is excellent; it takes the best of a game like Qix and adds to it with extra features and graphics. Most important it is a fun game that can be played solo or in 2 player mode. My 3 children and I spent a good 45 minutes playing this game with each other and had a great time! A very family friendly fun game.

* Game was reviewed by Derek Loughrey using a 128K TRS-80 Color Computer 3 with 2 Deluxe Radio Shack Joystick. Please send any questions or comments to: dml_68@yahoo.com.

Licensed Pac-man Developed For the CoCo?

by Retro Rick

Pac-man is an arcade game ported (and/or ripped off) more times on more computers than any other game I can think of. The only games that even come close to having this many wannabes are Space Invaders, Nibbler, Pong, and Galaxian.

For those who don't know exactly what the original Pac-man is, you're this circular, moving mouth that's put in a maze to eat up dots. There are these Ghost Monsters after you that are various colors, except blue. As long as this is the case, their touch kills you. You change this situation by eating a "Power-pellet", and all the ghosts turn blue temporarily, allowing you to eat them almost as casually as you do the dots.

A bonus item appears in the middle of the screen from time to time. If you eat it before it goes away again, you get a small bonus.

The goal of the arcade game is to pass the mazes by eating up all the dots in each level. You should try to make as many points as possible while doing this.

"Pacman" for the CoCo is the ultimate trivia question of all time, in that it seems there really **was** a licensed version of the game developed for the Tandy Color Computer. If I remember the name on the label correctly, I found it on a disk called "Games 17F". This disk series appears to have several sources from where it receives its games, and the software doesn't necessarily have a company name coming with them. So I have no idea where the CoCo game "Pacman" originally came from.

At any rate, this game has the words "unfinished prototype" written all over it. It has loads of silly bugs, very limited sound effects, and lots of missing animation sequences (although the animation to Pacman himself is great). Added to this, Pacman is also **very** hard to control; I remember several times pointing the joystick towards an upcoming turn in the maze, and Pacman stopped moving abruptly, thinking that I was trying to move through a wall. There aren't any Power-pellets or bonus items in this one, either.

So what happened with Pacman? Why wasn't it finished? My theory is that the designers were told to make this for a 16K system, and ran out of memory (and money) in the middle of development. This would explain why there's such a long list of features missing that you'd expect in a licensed version, including things as simple as a title screen.

Whatever the case, even though this game isn't fully functional, at least it's playable. And CoCo folks now know that there actually **was** a genuine, licensed Pac-man clone being developed for the CoCo, even if it never **was** complete.

Making assembler files from machine code programmes, using Disk Basic.
By Bob Devries.

Recently, in my quest to preserve colour computer software, I had a need to re-assemble a file into a different format. The file in question was quite short, thankfully, but rather than typing all the code in (which does require a certain amount of knowledge about the programme you're trying to modify).

Although the file I originally worked on was a single-origin file; that is, it had one load address, and a number of sequential blocks, I knew that I would eventually come across a file with multiple origins, such as those produced by Tandy's EDTASM. These files are split into blocks of 128 bytes, each with a pre-amble and post-amble.

I heard somebody ask: What's a pre-amble
and post-amble?

To explain: For the DISK BASIC to know where an ML file should be copied into memory, and how many bytes (characters) should be copied, there is some extra information saved along with the file. At the beginning of the file, one byte is used to identify that the file is indeed, an ML file. This byte must be 0. ($FF or 255 is used for a BASIC file). After that, there are two bytes that give the length of the file (0 – 65535). Then there are two bytes to show where in memory the file is to be loaded.

At the end of the actual data in the file, there is a similar 5 byte packet which is the post-amble. It consists of one byte ($FF or 255) if the file is now at an end, or 0 if there are more chunks, then for the former, two bytes of 0, and then the execute (or run) address. If there are more chunks, then the post-amble becomes the pre-amble of the next chunk.

So, my programme needed to deal with the
pre- and post-amble data.

The aim of my programme is to produce
assembler code such as this:

```
        FCB    $86, $01, $B6, $FF, $22, $C6,
$FE, $20
```

I remembered, however, that there are a number of different assemblers in use (even by me!), and some, like EDTASM do not like multiple FCB data on one line. For that, I needed a different output.

```
        FCB    $86
        FCB    $01
        FCB    $B6    etc.
```

Also, EDTASM likes to have line numbers,
which makes for further complications:

```
000010         FCB    $86
000020         FCB    $01    etc
```

So, four different variations are available, and the programme asks which you want. The first few lines are REM lines with introductions and usage explanations.

The programme sends it's output to a file on the disk in the drive which you specify, using the same name as the original ML file, but with an extension of /ASM.

Here's the BASIC programme:

```
10 'CONVASM+/BAS (C) 2007 BY
BOB DEVRIES
20 'PLACED IN THE PUBLIC DOMAIN
30 'FOR NON-PROFIT USE ONLY
40 '
50 'USAGE:
60 'ALL PROMPTS ARE SELF
EXPLANATORY
70 'EXCEPT:
80 'FILENAME, WHICH DEFAULTS TO
THE
90 'EXTENSION "/BIN" IF NOT
GIVEN
100 'USING EDTASM (Y/N)
110 'WHICH DETERMINES IF
MULTIPLE FCB
120 'STATEMENTS PER LINE WILL
BE USED
130 'EDTASM DOES NOT ALLOW THIS
140 'LINE NUMBERS (Y/N)
150 'INSERTS LINE NUMBERS AT
THE FRONT
160 'OF EACH LINE. EDTASM
REQUIRES THIS
170 'OTHER ASEMBLERS MAY ALSO
180 '
190 CLEAR 1000:
PT=1:OF=0:WIDTH40:PRINT
200 INPUT"SOURCE DRIVE";SD
210 IF SD<0 OR SD>3 THEN 200
220 INPUT"SOURCE FILENAME";SI$
230 INPUT"TARGET DRIVE";TD
240 INPUT"USING EDTASM
(Y/N)";YN$
250 IF YN$="Y" OR YN$="y" THEN
ED=1 ELSE ED=0
260 INPUT"USE LINE NUMBERS
(Y/N)";YN$
270 IF YN$="Y" OR YN$="y" THEN
LN=10 ELSE LN=0
280 IF TD<0 OR TD>3 THEN 230
290 IN=INSTR(1,SI$,"/"):IF IN=0
THEN IN=INSTR(1,SI$,"."):IF
IN=0 THEN SI$=SI$+"/BIN"
300 IF IN=0 THEN
IN=INSTR(1,SI$,"."):IF IN=0
THEN IN=INSTR(1,SI$,"/")
310 SO$=LEFT$(SI$,IN-
1)+"/ASM:"+CHR$(48+TD)
320 SI$=SI$+":"+CHR$(48+SD)
330 PRINT"CONVERTING
";SI$:PRINT"TO ASSEMBLER
SOURCE":PRINT"USING
FILENAME:";SO$
340 OPEN"D",#1,SI$,1:FIELD #1,1
AS A$
350 OPEN"O",#2,SO$
360 IF LN>0 THEN
GOSUB660:LN=LN+10
370 IF ED=0 THEN PRINT#2,";";
380 PRINT#2,STRING$(10,"*");
390 PRINT#2,SO$;STRING$(10,"*")
400 GET#1,PT:IF A$<>CHR$(0)
THEN CLOSE #2:CLOSE
#1:PRINTSI$;" IS NOT A BINARY
FILE":KILLSO$:END
410 PT=PT+1:
GET#1,PT:LE=ASC(A$)*256
420 PT=PT+1:
GET#1,PT:LE=LE+ASC(A$)
430 PT=PT+1:
GET#1,PT:SA=ASC(A$)*256
440 PT=PT+1:
GET#1,PT:SA=SA+ASC(A$)
450 IF LN>0 THEN
GOSUB660:LN=LN+10
460 PRINT#2,CHR$(9)
;"ORG";CHR$(9);"$";RIGHT$("000"
+HEX$(SA),4)
470 PT=PT+1:LC=1
480 GET#1,PT
490 IF LC=1 THEN IF LN>0 THEN
GOSUB660:LN=LN+10
500 IF LC=1 THEN
PRINT#2,CHR$(9);"FCB";CHR$(9);:
ELSE PRINT#2,",";
510 PRINT#2,"$";
RIGHT$("0"+HEX$(ASC(A$)),2);
520 PT=PT+1:IF PT>LE+5+OF THEN
560
```

```
530 IF ED=0 THEN LC=LC+1 ELSE
PRINT#2
540 IF LC<=8 THEN 480
550 LC=1:PRINT#2:GOTO 480
560 GET#1,PT:IF ASC(A$)=0 THEN
OF=PT-1:GOTO400
570 IF ASC(A$)=255 THEN PT=PT+3
580 GET#1,PT:EX=ASC(A$)*256
590 PT=PT+1:
GET#1,PT:EX=EX+ASC(A$):PRINT#2
600 IF LN>0 THEN
GOSUB660:LN=LN+10
610 PRINT#2,CHR$
(9);"END";CHR$(9);"$";RIGHT$("0
00"+HEX$(EX),4)
620 IF LN>0 THEN GOSUB660
630 IF ED=0 THEN PRINT#2,";";
640 PRINT#2,STRING$
(10,"*");"END OF
FILE";STRING$(10,"*")
650 CLOSE#1:CLOSE#2:END
660 PRINT#2,RIGHT$
("0000"+RIGHT$(STR$(LN),LEN(STR
$(LN))-1),6);
670 RETURN
```

A Basic09 Tutorial
by Bob Devries

Here is the BASIC code for the numbersquare programme from '**Microcomputing**', June 1981. It is written in vanilla BASIC, but is suitable for Extended Colour Basic with minor modifications, in lines 140,490, 760, 920, 960 and 1450. Can you work out what to change? (Note please that a colon “:” is used instead of a REM)

```
0010 : Number square game
0020 : ver 4.0 - 12 nov 79
0030 : Marc I. Leavey, M.D.
0040 LINE= 0
0050 DIGITS= 0
0060 PRINT "N U M B E R   S Q U A R E S"
0070 PRINT "---------------------------"
0080 PRINT
0090 PRINT "WELCOME TO THE WORLD
OF"
0100 PRINT "CONFUSION.  THERE ARE
TWO"
0110 PRINT "VERSIONS OF NUMBER
SQUARES:"
0120 PRINT " 1 - SEQUENTIAL"
0130 PRINT " 2 - MAGIC SQUARE"
0140 INPUT "WHICH IS YOUR
PLEASURE",T
0150 IF T=1 GOTO 310
0160 IF T<>2 GOTO 140
0170 :
0180 : SET UP MAGIC
0190 : SQUARE BOARD
0200 :
0210 FOR I=1 TO 4
0220 FOR J=1 TO 4
0230 READ M(I,J)
0240 LET B(I,J)=M(I,J)
0250 NEXT J
0260 NEXT I
0270 DATA
1,6,15,8,12,11,2,5,10,13,4,3,7,16,9,14
0280 LET I1=4
0290 LET J1=2
0300 GOTO 440
0310 :
0320 : SET UP SEQUENTIAL
0330 : BOARD
0340 :
0350 DIM B(4,4)
0360 FOR I=1 TO 4
0370 FOR J=1 TO 4
0380 LET B(I,J)=(I-1)*4+J
0390 NEXT J
0400 NEXT I
0410 LET I1=4
0420 LET J1=4
0430 :
0440 : NOW SCRAMBLE THE BOARD
0450 : TWO HUNDRED TIMES
0460 :
0470 PRINT "I AM NOW SCRAMBLING
THE BOARD..."
0480 FOR Q=1 TO 200
```

```
0490 LET M=INT(1+RND*4)
0500 ON M GOTO 510,560,610,660
0510 IF I1=1 GOTO 490
0520 LET B(I1,J1)=B(I1-1,J1)
0530 LET B(I1-1,J1)=16
0540 LET I1=I1-1
0550 GOTO 700
0560 IF I1=4 GOTO 490
0570 LET B(I1,J1)=B(I1+1,J1)
0580 LET B(I1+1,J1)=16
0590 LET I1=I1+1
0600 GOTO 700
0610 IF J1=1 GOTO 490
0620 LET B(I1,J1)=B(I1,J1-1)
0630 LET B(I1,J1-1)=16
0640 LET J1=J1-1
0650 GOTO 700
0660 IF J1=4 GOTO 490
0670 LET B(I1,J1)=B(I1,J1+1)
0680 LET B(I1,J1+1)=16
0690 LET J1=J1+1
0700 NEXT Q
0710 :
0720 : PRINT BOARD
0730 :
0740 LET M9=0
0750 : OUTPUT A "HOME UP"
0760 PRINT CHR$(16);
0770 PRINT "---------------------"
0780 FOR I=1 TO 4
0790 FOR J=1 TO 4
0800 PRINT": ";
0810 IF B(I,J)=16 PRINT "   ";:GOTO 840
0820 IF B(I,J)<10 PRINT " ";
0830 PRINT B(I,J);
0840 NEXT J
0850 PRINT ":"
0860 PRINT "---------------------"
0870 NEXT I
0880 :
0890 : ERASE REST OF SCREEN AND
0900 : BEEP FOR INPUT
0910 :
0920 PRINT CHR$(22);CHR$(7);CHR$(7);
0930 :
0940 : INPUT MOVE
0950 :
0960 INPUT "MOVE WHICH PIECE",M
0970 LET I1=0:J1=0
0980 FOR I=1 TO 4
0990 FOR J=1 TO 4
1000 IF B(I,J)=M THEN I1=I:J1=J
1010 NEXT J
1020 NEXT I
1030 IF I1=0 THEN PRINT "I CAN'T FIND
THAT NUMBER":GOTO 940
1040 LET I2=0:J2=0
1050 FOR I=I1-1 TO I1+1
1060 IF I>4 GOTO 1090
1070 IF I<1 GOTO 1090
1080 IF B(I,J1)=16 THEN I2=I:J2=J1:GOTO
1170
1090 NEXT I
1100 FOR J=J1-1 TO J1+1
1110 IF J>4 GOTO 1140
1120 IF J<1 GOTO 1140
1130 IF B(I1,J)=16 THEN I2=I1:J2=J:GOTO
1170
1140 NEXT J
1150 LET M9=M9+1
1160 PRINT "NOT A VALID
MOVE":GOTO 940
1170 LET B(I2,J2)=M
1180 LET B(I1,J1)=16
1190 ON T GOTO 1210,1320
1200 :
1210 : SEQUENTIAL SOLUTION
1220 :
1230 LET C=0
1240 FOR I=1 TO 4
1250 FOR J=1 TO 4
1260 IF B(I,J)<C GOTO 720
1270 LET C=B(I,J)
1280 NEXT J
1290 NEXT I
1300 PRINT "YOU GOT IT!"
1310 GOTO 1450
1320 :
1330 : MAGIC SQUARE SOLUTION
1340 : CHECK
1350 :
1360 FOR I=1 TO 4
```

```
1370 FOR J=1 TO 4
1380 IF B(I,J)<>M(I,J) GOTO 720
1390 NEXT J
1400 NEXT I
1410 :
1420 : A WIN IS DECLARED!
1430 :
1440 PRINT "THAT IS THE CORRECT
SOLUTION!"
1450 INPUT "LIKE TO PLAY ANOTHER
GAME",I$
1460 IF LEFT$(I$,1)="Y" THEN RUN
1470 END
```

The game is a fairly simple one based on the use of multi-dimensioned arrays. Note the use of colons at the beginning of REM lines, which is also possible in DECB. Now comes the tricky part, the conversion to Basic09. Firstly I'll show you the code as I rewrote it, then I will explain it.

```
PROCEDURE numbersquare
BASE 1
(* version 4.0 - 12 NOV 79
(* Marc I. Leavey, MD
(* Basic09 version By Bob Devries April 91
DIM t:INTEGER
DIM i,j:INTEGER
DIM b(4,4):INTEGER
DIM i1,j1:INTEGER
DIM q,m,m9,c:INTEGER
DIM mm(4,4):INTEGER
DIM lop:BOOLEAN
DIM valid:BOOLEAN
DIM solution:BOOLEAN
DIM newgame:BOOLEAN
SHELL "tmode -pause"
newgame=TRUE
WHILE newgame=TRUE DO
  PRINT CHR$(12);
  PRINT "N U M B E R   S Q U A R E S"
  PRINT "---------------------------"
  PRINT
  PRINT "Welcome to the world of"
  PRINT "confusion.  There are two"
PRINT "versions of number squares:"
PRINT " 1 - sequential"
PRINT " 2 - Magic Square"
INPUT "Which is your pleasure ? ",t
IF t=1 THEN
  RUN setupssb(b,i1,j1)
ELSE
  RUN setupmsb(b,mm,i1,j1)
ENDIF
PRINT "I am now scrambling the board..."
FOR q=1 TO 200
  lop=FALSE
  WHILE lop=FALSE DO
    m=1+RND(3)
    IF m=1 THEN
      IF i1<>1 THEN
        b(i1,j1)=b(i1-1,j1)
        b(i1-1,j1)=16
        i1=i1-1
        lop=TRUE
      ENDIF
     ENDIF
  IF m=2 THEN
    IF i1<>4 THEN
      b(i1,j1)=b(i1+1,j1)
      b(i1+1,j1)=16
      i1=i1+1
      lop=TRUE
    ENDIF
  ENDIF
  IF m=3 THEN
    IF j1<>1 THEN
      b(i1,j1)=b(i1,j1-1)
      b(i1,j1-1)=16
      j1=j1-1
      lop=TRUE
    ENDIF
  ENDIF
  IF m=4 THEN
    IF j1<>4 THEN
      b(i1,j1)=b(i1,j1+1)
      b(i1,j1+1)=16
      j1=j1+1
      lop=TRUE
    ENDIF
  ENDIF
```

```
   ENDWHILE
NEXT q
(* print board line 720
solution=FALSE
REPEAT
   m9=0
   RUN gfx2("cwarea",29,12,22,12)
   PRINT "-----------------"
   FOR i=1 TO 4
      FOR j=1 TO 4
         PRINT ": ";
         IF b(i,j)=16 THEN
            PRINT "  ";
         ELSE
            IF b(i,j)<10 THEN
               PRINT " ";
            ENDIF
            PRINT b(i,j);
        ENDIF
      NEXT j
      PRINT ":"
      PRINT "-----------------"
      NEXT i
(* input move
valid=FALSE
WHILE valid=FALSE DO
   i1=0
   j1=0
   WHILE i1=0 DO
      RUN gfx2("bell")
      INPUT "Move which piece ? ",m
      IF m=0 THEN
         RUN gfx2("cwarea",0,0,80,24)
         PRINT CHR$(12)
         SHELL "tmode pause"
         END
      ENDIF
      FOR i=1 TO 4
         FOR j=1 TO 4
            IF b(i,j)=m THEN
               i1=i
               j1=j
            ENDIF
         NEXT j
      NEXT i
      IF i1=0 THEN
         PRINT "I can't find that number"
      ENDIF
   ENDWHILE
   i2=0
   j2=0
   FOR i=i1-1 TO i1+1
      IF i>=1 AND i<=4 THEN
         EXITIF b(i,j1)=16 THEN
            i2=i
            j2=j1
            valid=TRUE
        ENDEXIT
     ENDIF
   NEXT i
   IF valid=FALSE THEN
     FOR j=j1-1 TO j1+1
        IF j>=1 AND j<=4 THEN
           EXITIF b(i1,j)=16 THEN
              i2=i1
              j2=j
              valid=TRUE
           ENDEXIT
             ENDIF
           NEXT j
         ENDIF
         IF valid=FALSE THEN
            m9=m9+1
            PRINT "Not a valid move"
         ENDIF
      ENDWHILE
      b(i2,j2)=m
      b(i1,j1)=16
      IF t=1 THEN
         c=0
         FOR i=1 TO 4
            FOR j=1 TO 4
               IF b(i,j)<c THEN (* reprint board
                  solution=FALSE
               ENDIF
               c=b(i,j)
            NEXT j
         NEXT i
         IF solution=TRUE THEN
            PRINT "You got it!"
         ENDIF
      ENDIF
      IF t=2 THEN
         FOR i=1 TO 4
            FOR j=1 TO 4
               IF b(i,j)<>mm(i,j) THEN
                  (* reprint board
                  solution=FALSE
               ENDIF
            NEXT j
         NEXT i
         IF solution=TRUE THEN
           PRINT "That is the correct solution!"
         ENDIF
      ENDIF
   UNTIL solution=TRUE
   INPUT "Like to play another game ? ",i$
   IF LEFT$(i$,1)="n" THEN (* rerun game
      newgame=FALSE
   ENDIF
ENDWHILE
RUN gfx2("cwarea",0,0,80,24)
SHELL "tmode pause"
END

PROCEDURE setupssb
BASE 1
PARAM b(4,4):INTEGER
PARAM i1,j1:INTEGER
DIM i,j:INTEGER
FOR i=1 TO 4
         FOR j=1 TO 4
                  b(i,j)=(i-1)*4+j
         NEXT j
NEXT i
i1=4
j1=4

PROCEDURE setupmsb
BASE 1
PARAM b(4,4),mm(4,4):INTEGER
PARAM i1,j1:INTEGER
DIM i,j:INTEGER
FOR i=1 TO 4
         FOR j=1 TO 4
                  READ mm(i,j)
                  b(i,j)=mm(i,j)
         NEXT j
NEXT i
i1=4
j1=2
DATA 1,6,15,8,12,11,2,5,10,13,4,3,7,16,9,14
```

OK, so here we go. First, I used the command BASE 1. This is because all the arrays in the BASIC programme start at 1, and Basic09 usually starts at zero (actually, BASIC does too, but I see no reason to waste valuable memory). Next, I dimensioned all the variables and arrays, including a variable type which you may not have seen before, the BOOLEAN type. This variable may only contain either TRUE or FALSE! I used the SHELL command to turn off the OS9 screen pause, so that the programme won't sit there waiting at what it thinks is the end of a screen. That can get a bit confusing!

The next thing you must realise is that I have used NO LINE NUMBERS! This is really the best way to programme. Sure, Basic09 will allow their use, but the code is much more elegant without them, if a little more difficult to write. I set up a loop to allow the choice of whether to play another game which is done in line 1450 in the BASIC version. I used a WHILE loop here so that all I need to do is make a variable FALSE to exit the loop.

Next, after clearing the screen (printing a formfeed character), I print the opening message and ask the player for his choice of game. This follows through to line 160 of the BASIC programme. On the basis of the player's answer I RUN either the procedure 'setupssb' or 'setupmsb'. You may notice a slight variation here, I only tested for a '1' to select sequential, and any other key would run the magic square option. Then the next 38 lines do the same as lines 480 to 700 in the BASIC programme. You will notice that the BASIC programme uses quite a large number of GOTOs in this piece of code to break out of various parts of the code to continue the FOR-

NEXT loop. I simply set a variable (lop) to TRUE and again used a WHILE loop. All the other variables have the same names, although you can of course use any name, and are not limited to two significant characters as in BASIC.

To make the screen easier to display, I used in this case a gfx2 command 'cwarea'. This command limits the area of the screen to be printed to, and means I don't need to use cursor manoeuvering code at every print line. I simply re-sized the screen to a 22 by 12 character box in approximately the middle of the (80 by 24) screen. So next I print the scrambled array to the screen with one space so that pieces may be moved around. The game is played by entering the number of the square which you want to move into the space. The programme checks to see if the number is one of the ones adjoining the space. A tone is sounded, and you are prompted for input. Pretty standard here, 'though I could just have used 'PRINT CHR$(7)', but the gfx2 'bell' command is nicer. If the player enters zero, the programme resets the screen back to its 80 by 24 size, and clears the screen and quits.

The rest of the code is fairly straight forward, again the original uses many GOTOs to quit out of FOR-NEXT loops (not a good practice in my opinion, even in BASIC), and I have used boolean tests for this purpose. For every move made, the programme checks the array against the solution, and if the last move solves the puzzle, it prints the necessary result.

One of the hardest parts of the conversion is keeping track of the variables, and the various loops. Basic09 is a bit unforgiving about 'UNMATCHED CONTROL STRUCTURES' so you can't stop doing a conversion such as this in the middle, without generating a series of error messages when you quit the editor. One way around this is to use a text editor (like VED or SCRED or SLED) to create the source code first, and then to load it into Basic09. The only thing you MUST do in this case is to make the word 'PROCEDURE' in UPPERCASE the FIRST WORD in the file. The letter P of procedure must be the first character in the file, or Basic09 will not recognize it.

OK, so there you have it. I would love to hear from you regarding your own trials and tribulations with Basic09 programmes, even if you don't really want to start out on conversions of this type, but are having difficulties managing some aspect of Basic09. Please write to me care of the newsletter editor and let 'Professor Bob' help sharpen your programming skills.

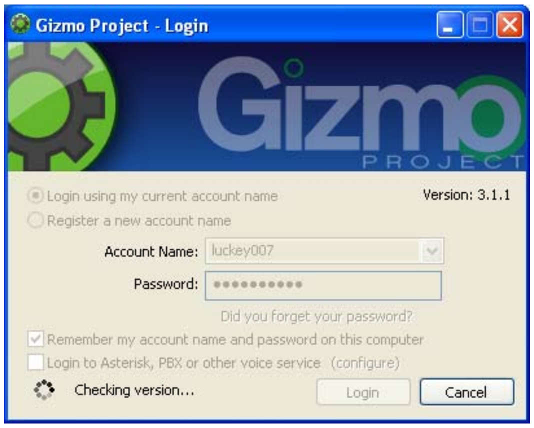
~~~oooOOOooo~~~

Capt's CoCo Hut
The most (legal) fun you can have in 8-bits.
For collectors, gamers, and users of TRS-80 and Tandy Color Computers!
~~~

The Gizmo Project

Gizmo is a program that allows us to make phone calls over the Internet. It can be downloaded FREE at gizmoproject.com. If two computers are running Gizmo then the call is free. A few months ago I did this with Boisy and it worked great, and I have talked to others such as Dave Kelley. If you are calling to somebody's land line or cell phone then there is a small charge. I think is one or two cents a minute. I just went to the Gizmo site and the price of calls to landlines and mobile phones is 1.9 cent per min.

Once you've installed it and set it up with a username and password and get connected on the internet you get this log in window.



Login using my current account name
Register a new account name
Version: 3.1.1
Account Name: luckey007
Password: ●●●●●●●●●●
Did you forget your password?
☑ Remember my account name and password on this computer
☐ Login to Asterisk, PBX or other voice service (configure)
Checking version...

Once the connection is made you will get the window below from which you can make calls.

Another option with Gizmo is the ability to make conference calls. I think this could be a good way to have non-local club members join in on our meetings. We've tried this once or twice and it hasn't worked very well. The number I have proposed is 1-222-222-2626. The first 222 is required by Gizmo. The second 222 is the letter C on the key pad and that can stand for Color Computer Club, and the 2626 can stand for CoCo.

~~~oooOOOooo~~~
~~~

NEW MEMBER
DALE KRAMER

NOW THAT I HAVE A COUPLE OF CFDM ISSUES UNDER MYBELT I BELIEVE I CAN INTRODUCE MYSELF BETTER. I AM GLAD TO JOIN THE REST OF YOU IN CFDM-LAND.

I AM DALE KRAMER AND I JUST MOVED FROM FLORIDA TO ALABAMA. I AM SINGLE AND ENJOY THE GUITAR, CHESS (NO ONE WILL PLAY ME), THE MIAMI HURRICANES, THE MIAMI DOLPHINS, ELECTRONICS, THE PARKS, AND COCO.

I WAS BORN IN MIAMI BEACH, FLORIDA BUT RAISED IN ILLINOIS AND HAVE LIVED IN MISSOURI AND GEORGIA. I MAKE ALL THE BEACHCOMBERS MAD BECAUSE I DON'T USE ANY CHEMICALS TO TAN WITH AND DON'T LAY ON THE BEACH TO GET A TAN. INSTEAD, I SWIM ALMOST THE WHOLE TIME I AM THERE AS THE SALTWATER GIVES THE BEST TAN (ONLY THE NATIVES KNOW THIS! HA!).

I AM INTERESTED IN A CAREER IN POLLUTION CONTROL AND HAVE A WASTEWATER TREATMENT PLANT OPERATORS LICENSE FROM INDIANA AND ALABAMA (CURRENTLY SEEKING A GEORGIA LICENSE). I INTEND TO RELOCATE IF NEEDED TO PURSUE THIS FIELD.

MY 512K COCO3 WITH A DUAL FD-501 DISK SYSTEM AND CM-8 MONITOR ACTS AS MY PRIMARY MACHINE. I'M WAITING FOR A PAL CHIP FOR THE MULTI-PAK (#26-3124). A DWP-230 DOES MY RESUMES AND A DMP-105 DOES THE GRAPHICS PRINTING. A COCO 1, A COCO 2, JOINS THE JOYSTICKS AND THE MOUSE ON MY EQUIPMENT LIST.

MY MAIN SOFTWARE IS COCO MAX III, COLOR GRAPHICS DESIGNER PLUS, AND VARIOUS GAMES. NO ONE EVER HAS AS MUCH SOFTWARE AS THEY WANT.

I LOOK FORWARD TO THE NEXT CFDM AND HOPE TO CONTRIBUTE SOME (SIMPLE) PROGRAMS.