190 S. Marengo Ave., Apt. # 8
Pasadena, California 91101

September 7, 1982

Robert Fabris, Editor
Arcadian
3626 Morrie Dr.
San Jose, California 95127

Dear Mr. Fabris:

No originality is claimed for the basic idea of "Rotate", rotating groups of letters on a 4 x 4 board, but I feel that this version includes so very many substantial innovations as to qualify for your contest. I did not receive any assistance from anyone on any aspect of the program. Although I <u>did</u> see a very crude implementation running on a Commodore Pet, as indicated on page 3 of my writeup, I <u>never</u> examined the listing.

I am not a subscriber to the "Arcadian", nor do I even own a Bally. I don't believe that the rules for your contest specified either. I only program from time to time on a machine owned by an acquaintance of mine, Chris Pruitt, who <u>is</u> a subscriber of yours. I hope that you will understand this and perhaps stretch the rules if need be.

I trust that you can see that I have gone to some little trouble and expense in preparing my entry. If there are not enough entries for the contest for this month, and you cannot defer until there are sufficient, I would hope that you would find room to publish it anyway, rather than putting it on a tape with other programs to sell. Of course the decision is yours, but these are my feelings. Finally, please take a few moments to briefly indicate on the enclosed postcard your evaluation and plans for disposition of the program.

I also have a few other programs in which you might be interested.

Sincerely,

Robert Newman

Robert Newman

PS: I know that you don't have a great deal of space to spare in the "Arcadian" for lengthy explanations, so a minimal writeup to ensure that features don't go undiscovered might go as follows:

Rotate a 2x2 square by keying in the letter in the upper left hand corner of the square, and so alphabetize the whole 4x4 board. <u>One</u> special move interchanging two horizontally adjacent letters may be made for each reset by entering "S", then left letter of pair. Reset by entering "R". Correct wrong rotation by "T". Quit with "Q". Keypad entry for each move of answer; reenter game after seeing part of answer by "÷" key, or "GO" key for full speed. "LEVEL" equals approximate difficulty.

R O T A T E

A 4x4 playing board contains the letters A-P in scrambled order. A group of letters in a 2x2 square on the board is rotated one position clockwise by keying in the letter in the upper left corner of the square. The object is to put the board in alphabetic order (ABCD on the top row, etc.) in as few moves as possible. In addition to rotating groups of letters you have one special move which will interchange a pair of horizontally adjacent letters. To make a special move, enter "S", then the left letter of the pair. If you're a fumble fingers like me and accidentally rotate a letter you didn't want to, enter "T" and the square will continue to rotate until it returns to its former position. To restore the board to its original setup before any moves were made, enter "R" (for Reset). With each reset you get one special move. To quit, enter "Q". The difficulty of any given puzzle corresponds roughly to the level D which is input by the player.

Some of you may have seen "Rotate" on other systems, or in a book of computer games. No originality is claimed for the basic idea, rotating groups of letters on a 4x4 board. What does seem to be original in this version is the way the computer randomizes the board by rotating the letters in its memory in a direction opposite to that of a player trying to reorder the board. Once this randomizing scheme was hit upon (a case of serendipity, to be sure), it was obvious that if the computer took D counterclockwise rotations to randomize the board, it should be possible for the player to reorder the board with D clockwise rotations (if he is smart enough, anyway). Since D can be specified by the player, the level of difficulty can be selected according to the player's ability or mood. Since the computer stored each of its moves as it randomized the board, its answer is available for display. After first restoring the board to its original setup, the computer changes it accordingly as it prints each move until the board is in proper alphabetic order. Since the computer can make its moves quite rapidly, I have it wait for a keypad input before each move except the first, so that each move can be studied at one's leisure. To get back into the game, press the ÷ (divide) key, or press the "GO" key to have the computer print out the rest of the answer at full speed.

Regular Moves

Regular moves are specified by keying in the letter occupying the upper left corner of a 2x2 square. Each of the letters in the square then rotates one position clockwise. Nothing will happen if an attempt is made to key in a letter in the bottom row or rightmost column.

Special Moves

A special move interchanging two horizontally adjacent letters is accomplished by entering "S" and then the left letter of the pair to be interchanged. If an attempt is made to enter a letter in the right hand column, the entry of the previous "S" is nullified. If a "T" entry immediately follows an "S" entry, the "T" is acted upon, and the "S" is also nullified. One special move is allowed during the initial game and each reset. After a special move is made, a message appears indicating such. Once this message appears, and provided that there are no intervening resets, any further "S" entries will be ignored and the following entry treated as a regular move.

Resets

If the player thinks of a better strategy, or would otherwise like to start over again, an entry of "R" will restore the board to its original setup. A player may reset as often as desired; the reset itself does not count as a move. After each reset a message will appear indicating how many resets have been made and the total number of moves up to the current reset. "MOVE # XX?" is asking for move number XX of the current reset. The computer also resets the board to display its answer, and this will be reflected in the reset count. Finally, the computer resets the board when "N" is entered in response to "NEW GAME? (Y/N)", but in this case there is no reset message.

"T" (Try Again) Entries

If you should inadvertently rotate a letter you did not mean to rotate, input a "T" and the letters will continue to rotate until they come to their previous position. Note that the move counter is also changed accordingly, so that you are not charged for your mistake. "T" entries are ignored unless a regular move has just been made. Multiple "T" enties are also ignored.

Quitting

You may quit by entering a "Q". One of the messages which will appear after a "Q" entry is "ANSWER? (Y/N)". A "Y" response to this message will cause the computer to reset the board, display and make the first move, and wait for a keypad input. Any keypad input other than "GO" or "÷" (divide) will cause the computer to display and make the next move, and wait for another input. By pressing the divide key you may reenter the game. If you do not reenter the game the program will reset the board and ask "NEW GAME? (Y/N)" after the last stored move is displayed. If the "GO" key is pressed the computer will display its stored answer at full speed without waiting for further input (hence no opportunity to try the current puzzle again except as a new player). A "N" response to "ANSWER? (Y/N)" will cause the program to go immediately to "NEW GAME (Y/N)", with consequent reseting of the board.

Additional Notes

Answering "N" to "NEW GAME? (Y/N)" resets the board and all relevant variables (except as noted below), and allows a new player a fresh start on the same puzzle after someone else has given up. Answering "Y" starts the program over and requests a new difficulty level.

If 10 moves of a level 15 puzzle have been displayed, obviously what remains is no longer a level 15 puzzle, but a level 5. A reverse box to the left of the board shows how many moves the computer has revealed, if any. This will continue to be displayed, with the new colors that appear when the answer is asked for, even if a new player tries to solve the same puzzle, since they may have also seen the solution and it would otherwise be too easy to cheat.

If you reenter the game by pressing the divide key, you may notice that the moves the computer made are charged to you. Nothing is free these days.

As noted before, the level is only an approximate indication of the difficulty of a puzzle. For example, some level 3's may be more difficult than some (very few) level 5's. Also, beyond a certain point which I have never bothered to pin down, but which I think is somewhere in the 20's, the higher the level, the easier it is to get the "WOW! SMARTY!" message which appears when you take fewer moves to solve the puzzle than the computer did to set it up. In any case, it is more challenging to get this message if you do not make a special move.

There are a variety of messages that come up when you solve the puzzle. I think these are fairly self-explanatory. "ME" refers to the computer as it tells you how many more or fewer moves it took you to solve the puzzle than the computer took to set the puzzle up (the difference between the number of your moves and the difficulty level).

The system that I saw "Rotate" on and which gave me the basic idea of rotating groups of letters took approximately 6800 bytes, and did no more than give direction, set up a random board, perform and keep count of regular and special moves, and print messages saying how many moves you made before quitting or winning. There were no solutions, no resets, no error corrections, et cetera. I no longer have access to that system, and never did study the listing to see where all that memory was going, but find myself wishing that Bally had even half as much. Maybe then we could afford to waste a few bytes too.

"Rotation" lends itself to endless variations - let me know of any possible improvements you might hit upon. I would also be very interested to know what is the lowest level you are able to get the "WOW! SMARTY!" message on, both with and without special moves. It would be especially interesting if anyone could beat the same puzzle both ways. How many regular moves (and which?) would be equivalent to one special move?

* *

Running the Program

In order to save memory, all variables are not cleared at the beginning of the program. However, key variables are cleared later at appropriate places. This should cause no problem provided the machine is reset prior to inputting the program, and provided that a "Q" entry is used rather than the halt button when a new puzzle is desired in the midst of a game

* *

Notes on the Listing

Since I anticipate having the listing photographically reduced, and also because the limitations of my typewriter, throughout the listing a slash will be used instead of the divide symbol when a computation is being made; lower case "e" for equals, and "b" where a blank is necessary but may not be evident otherwise. Because there is not even one byte to spare for level 40, it is critical that the program is entered accurately, without extra spaces. To help insure accurate entry, the number in parentheses to the left of the line number is the number that should appear in response to PRINT SZ after that line is entered.

Listing

```
(1790)  1.ROTATE
(1784)  2 GOTO 12
(1763)  3 ZeP/4;HeRM;IF HeOHe4
(1719)  4 CXe30+10xH;CYe45-10x((P-1)/4+1);TVe@(P);RETURN
(1619)  5 NeN+1;Ze@(P);@(P)e@(P+4);@(P+4)e@(P+5);@(P+5)e@(P+1);@(P+1)eZ
          TeP;GOSUB 3;PeP+1;GOSUB 3;PeP+4;GOSUB 3;PeP-1;GOSUB 3;RETURN
(1598)  6 FOR Je1TO 16;IF Me@(J)PeJ
(1591)  7 NEXT J;RETURN
(1577)  8 FOR IeOTO 999;NEXT I
(1543)  9 CXe-29;CYe-24;BOX 0,-22,159,43,2;RETURN
(1468) 10 FOR Ie17TO32;PeI-16;@(P)e@(I);GOSUB 3;NEXT I;LeO;ReR+1;SeO;
          TeO;VeN;BOX -19,14,106,31,2;RETURN
(1428) 11 CXe-65;CYe25;PRINT #1,"RESET # ",R,":",N;RETURN
(1369) 12 CLEAR;BCe223;FCe8;FOR IeOTO 40STEP 10;BOX 54,I,41,1,1;
          BOX I+34,20,1,41,1;NEXT I
(1324) 13 CeO;FOR Ie1TO 16;@(I)eI+64;NEXT I;CYe35;INPUT"bbLEVEL"D
(1307) 14 IF (D<1)+(D>40)RUN
(1298) 15 FOR Ie1TO D
(1274) 16 PeRND(11);IF P-P/4x4eOGOTO 16
(1203) 17 Ze@(P);@(P)e@(P+1);@(P+1)e@(P+5);@(P+5)e@(P+4);
          @(P+4)eZ;@(I+32)e@(P)
(1166) 18 NEXT I;FOR Ie17TO 32;PeI-16;@(I)e@(P);GOSUB 3;NEXT I
(1095) 19 CXe-65;CYe5;PRINT"MOVE # ",#1,N-V+1,"?bb",;MeKP;
          IF M>63IFM<85TVe31;TVeM;GOTO 21
(1089) 20 GOTO 19
(1078) 21 IF Me81GOTO 27
(1059) 22 IF Me82GOSUB 10;GOSUB 11;GOTO 19
(1048) 23 IF Me83GOTO 39
(1008) 24 IF Me84IF TFOR Ie1TO 3;PeT;GOSUB 5;NEXT I;NeN-4;SeO;TeO
(997)  25 IF Me84GOTO 19
(976)  26 GOSUB 6;HeP-P/4x4eO;GOTO 39
(957)  27 GOSUB 9;PRINT"SPOILSPORT!
(912)  28 GOSUB 8;CXe-65;PRINT"YOU QUIT AFTERb",#1,N,"bMOVES!
(891)  29 GOSUB 8;PRINT"ANSWER? (Y/N)
(879)  30 IF KP#89GOTO 51
(848)  31 GOSUB 9;BCe119;FCeO;PRINT"CHEATING???
(765)  32 BeO;KeO;GOSUB 10;GOSUB 11;GOSUB 9;CXe-23;CYe-5;FOR IeD+32TO 33
          STEP -1;Me@(I);XeCX;YeCY;CXe-65;CYe5;IFKGOTO 36
(755)  33IFBeOGOTO 36
(739)  34 ZeKP;IF Ze99GOTO 19
(728)  35 IF Ze13Ke1
(639)  36 BeB+1;PRINT"MOVE # ",#1,B,"b",;TVeM;IF B>CbCeB;BOX 21,35,13,9,2;
          CXe19;CYe35;PRINT #1,C;BOX 21,35,13,9,3
(589)  37 GOSUB 6;GOSUB 5;CXeX;CYeY;TVeM;IF B-B/10x10eOCXe-23;CYeCY-8
(580)  38 NEXT I;GOTO 50
(565)  39 IF Me83Se1;GOTO 19
(557)  40 IF LGOTO 44
(547)  41 IF SeOGOTO 44
(534)  42 IF HbSeO;GOTO 19
(445)  43 Ze@(P);@(P)e@(P+1);@(P+1)eZ;GOSUB 3;PeP+1;GOSUB 3;Le1;NeN+1;
          SeO;TeO;CXe-65;CYe15;PRINT"SP. MOVE";GOTO 46
(430)  44 IF (P>11)+HGOTO 19
(425)  45 GOSUB 5
(399)  46 FOR Ie1TO 15;IF @(I)>@(I+1)GOTO 19
(368)  47 NEXT I;GOSUB 9;IF N<DPRINT"WOW! SMARTY!";GOSUB 8
```

```
(291)   48 CXe-47;PRINT"YOU WON INb",#1,N,"bMOVES!";IF N>DGOSUB 8;CXe-41;
           PRINT #1,N-D,"bMORE THAN ME!
(210)   49 IF D>NGOSUB 8;CXe-41;PRINT #1,D-N,"bFEWER THAN ME!";GOSUB 8;
           CXe-77;PRINT"BET YOU CAN'T DO IT AGAIN!
(205)   50 GOSUB 8
(163)   51 Ne0;GOSUB 10;GOSUB 9;Re0;CXe-35;PRINT"NEW GAME? (Y/N)
(148)   52 IF KP#89GOSUB 9;GOTO 19
(144)   53 RUN
```

Index of Major Variables

B	Move counter for computer during answer display
C	Maximum number of moves displayed so far
D	Difficulty level
H	Used in determining horizontal position of letter input
K	Flag used during answer display to see if "GO" has been pressed
L	Counter for special moves
M	ASCII corresponding to input
N	Move counter (both regular and special moves)
P	Board position. Board positions are numbered 1 - 16, left to right, top to bottom (e.g., the positions of the letters in the top row are 1, 2, 3, and 4).
R	Reset counter
S	Flag indicating previous key pressed was an "S".
T	Zero unless previous move resulted in a letter being rotated, in which case it is set to position the letter occupied before it was rotated.
V	Used in conjunction with reset so that "MOVE # XX?" is asking for move XX of initial game *or* move XX of the current reset.

Line Explanation

2	Skips over subroutines which were placed at the beginning of the program so that they would have single-digit line numbers, saving memory in GOSUB statements. This placement also decreases the time taken to find each subroutine.
3-4	Calculates CX and CY for board position P, and changes the board to correspond to current contents of memory for that position.
5	Subroutine handling regular moves; the move counter is incremented and the letters are rotated first in memory and then on the board. The position rotated is saved in variable T.
6-7	The "search" subroutine locates the board position of the letter input. M was previously set to ASCII of letter input. Strings 1 to 16 contain ASCII of letters in corresponding positions of the board. The loop searches these strings until it finds one equal to M. When it does, it sets position variable P equal to the corresponding position.
8	Time loop (see below).
9	Clears the bottom half of the screen and sets frequently used CX and CY values. In conjunction with line 8, there is a time delay beforehand.
10	The "reset" subroutine restores the board to its original position, increcments the reset counter, gives you another special move, and clears everything to the left of the board below "LEVEL".
11	Prints "RESET" message.
12	Clears the screen, sets colors, and constructs the board.
13	Clears counter for places of answer shown, sets up an "internal board" in alphabetic order which the computer will scramble to produce the puzzle, and asks for a difficulty level to be input.
14	In case some smart aleck like one I know tries to enter a difficulty outside the range of the program.
15-18	Loop in which the computer performs its counterclockwise rotations.
16	Selects a board <u>position</u> P (not letter) randomly and checks to see if it is a valid position to rotate. RND(11) because 12-16 are not valid positions. The if statement sees if position is a multiple of 4, and is thus invalid, being the rightmost column.
17	Performs rotation (in memory only) and stores the <u>letter</u> in upper left corner of rotated group. Storing letters is a little more economical of memory than storing positions. It's a little slower, but still quite rapid. I can afford time more than memory.
18	After the computer has scrambled the board in its memory, this stores the scrambled board in strings 17 to 32 (for use with resets) and displays it on the screen.
19	Asks for the next input, checks to see that input is acceptable (a letter from A to T), and displays move briefly while board is changed.
20	Any invalid keypad inputs are ignored and the computer asks for another by going to the preceding line.
21	If "Q" (quit) was entered, go to portion of program devoted to that response (lines 27 to 38).
22	Calls reset and reset message subroutines, goes to ask for another input.
23	If "S" was entered, go to portion of program devoted to special moves (lines 39 to 43).

24 If key pressed was "T", and variable T (distinct from ASCII of "T") is not 0, meaning that a letter was rotated in the previous move, rotate the <u>position</u> that letter was in before it was rotated 3 more times, adjust move counter appropriately, zero S in case someone started a special move and then decided they wanted to change the last rotation, zero T so that multiple "T" entries will be ignored.
25 If key pressed was "T", go ask for another input. This line could not be combined with line 24 since line 24 had another condition in addition to keypad entry of "T".
26 Any response reaching this point must be a letter from A to P. Call the search subroutine to find the board position corresponding to the letter. Set variable H to truth value of expression used in determining if position is valid for a regular or special move. If the truth value is 1, the position is a multiple of 4, and so is in the rightmost column, an invalid position for both regular and special moves. Go to special move section of program to determine if the previous entry was an "S", and whether a special move has been made. Since the computer doesn't know this until it checks, all regular moves must be filtered through the special move routine along with special moves.
27 Clear lower screen, print message.
28 Time delay, clear lower screen, set horizontal position of cursor, print message.
29 Time delay, clear lower screen, print message asking if one wishes to see the answer.
30 If they do not wish to see the answer, go to 51.
31 Clear lower screen, change colors, print message.
32 Zero K (used to see if "GO" key has been pressed) and B (used to keep count of number of moves displayed). Reset board preparatory to showing answer, clear lower screen of "CHEATING???" message, set cursor for start of answer display. Loop beginning here and extending to 38 fetches moves stored previously (see line 17) in strings. Step -1 because the first move the computer made in scrambling the board is the last move it makes in reordering it, and vice versa. Set M to previously stored ASCII of letter computer rotated internally Save cursor position, reset cursor for "MOVE" message. "IF" statement checks whether or not the "GO" key has been pressed.
33 You already gave a keypad input in response to "ANSWER? (Y/N)". I didn't want another to be necessary before something started happening, so the first time through the loop I skip over the keypad input.
34 Set variable Z to ASCII of key pressed. If the divide key was pressed, reenter game and ask for a move.
35 If "GO" key is pressed set flag K to let computer skip over any further keypad inputs and display answer at full speed.
36 Increment move display counter, print "MOVE" message. Set C equal to the maximum number of moves displayed so far, print C in a reverse box after clearing screen of previous value of C (Try leaving the first BOX instruction out and see what happens).
37 Search for the current position of the letter to be moved, perform rotation, print move in lower portion of screen, 10 to a row.
38 End of loop displaying answer. Go to 50, where computer pauses to let you verify that board is in alphabetic order.
39 If "S" was entered, set variable S to 1, go ask for another input.
40 If there has been a special move made already, go check to see if second part of special move entry is a valid regular move; if so, a regular move will be made.

41 If the previous entry was not an "S", go check to see if current entry is a valid regular move.
42 See line 26. If the entry after an "S" is a letter in the right-most column, nullify the "S" entry.
43 Perform special move, increment special move counter and total move counter, turn off special move flag, zero T so regular move correction cannot be made after a special move, print "SP. MOVE" message, go see if the board is in proper order.
44 If entry is not valid for a regular move, go ask for another.
45 Perform regular move.
46-47 Check board to see if it is in alphabetic order. If it isn't, the ASCII of at least one letter in board positions 1 to 15 will be greater than the ASCII of the letter in the next position, in which case go ask for another input.
47-49 Various messages after the puzzle is solved.
50 Wait, clear lower screen, set cursor.
51 Reset board, clear lower screen, zero major variables not zeroed elsewhere, print message to see if new player wants to try a different puzzle.
52 If they don't want a new puzzle, they must want the old one. Clear lower screen and ask for a move.
53 Go set up a fresh board, ask for difficulty level.

* Modifying the program to produce higher level puzzles - The quickest way to do this is to eliminate the 2 BOX instructions in line 36. Line 14 will also have to be changed. Since strings 1 to 16 are used for the board displayed on the screen, and strings 17 to 32 are used for the original or "reset" board, and each string takes up two bytes of memory, you may then compute the highest level the program will handle by subtracting 64 from SZ and dividing by 2. Since the program was originally designed for 5 rows of 10 moves each in displaying the answer, if you want to display more than 50 moves you must also change the expression B-B/10x10e0 in line 37 to reflect a larger number than 10.

SOLUTION OF A LEVEL-8 PUZZLE IN 8 REGULAR MOVES

B	C	D	H
F	G	E	P
I	J	K	L
A	M	N	O

Rotate E

B	C	D	H
F	G	K	E
I	J	L	P
A	M	N	O

Rotate L

B	C	D	H
F	G	K	E
I	J	N	L
A	M	O	P

Rotate G

B	C	D	H
F	J	G	E
I	N	K	L
A	M	O	P

Rotate D

B	C	G	D
F	J	E	H
I	N	K	L
A	M	O	P

Rotate I

B	C	G	D
F	J	E	H
A	I	K	L
M	N	O	P

Rotate F

B	C	G	D
A	F	E	H
I	J	K	L
M	N	O	P

Rotate C

B	F	C	D
A	E	G	H
I	J	K	L
M	N	O	P

Rotate B

A	B	C	D
E	F	G	H
I	J	K	L
M	N	O	P

Reordered!