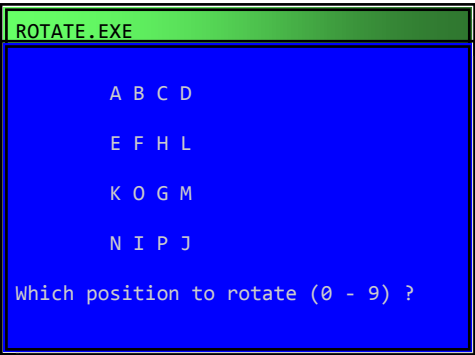


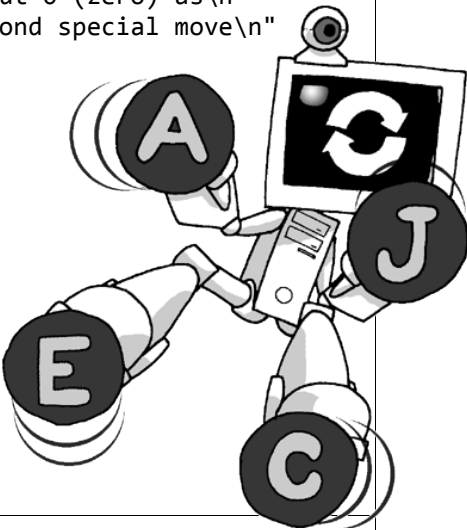
# ROTATE.C

Rotate is a game for one, a solitaire puzzle. A board is presented to you scrambled and you have to put it back in order from A to P, left to right, up and down. It's similar to the sliding puzzles you sometimes see as party favors except instead of moving one piece, you move 4 at a time, rotating their position clockwise.

Control is mapped so that you can use the number pad. If you find all but two blocks are in place you can swap them by pressing 0 (zero) answering that you do not want to exit the game, and choosing which two to swap, however this move is available to you only once, so it's often the last move.

Rotate is written by Joseph Larson, inspired by a BASIC program of the same name by David H. Ahl as found in 'More BASIC Programs' © 1978.



ROTATE.C	You will need: a C/C++ compiler .
<pre>#include &lt;stdio.h&gt; #include &lt;stdlib.h&gt; #include &lt;time.h&gt; #include &lt;ctype.h&gt;  #define DRAW for (c = 0; c &lt; 16; c++) printf((c % 4) ? "%c " : "\n\n\t%c ", b[c])  int main (void) {     int c, d, temp, turn, special, done, b[16];     char input;     int x[4] = {0, 4, 5, 1};     int t[9] = {8, 9, 10, 4, 5, 6, 0, 1, 2};     long mask;      srand (time (NULL));     printf ("Rotate\n-----\n" "Order the puzzle board from top to botom, left to right, in alphabetical\n" "order by rotating four blocks at a time clockwise:\n" "\n\tA B C D\n\t7 8 9\n\tE F G H\n\t4 5 6\n\tI J K L\n\t1 2 3\n\tM N O P\n" "\nIn the above illustration, the numbers betweenet the letters are what you\n" "will input. So if you want to rotate the A, D, E, and F you would input 7\n" "as your move.\n\n" "You also have one special move that you may not need where you can switch\n" "the location of two adjacent blocks with each other. Input 0 (zero) as\n" "your move if you want to do this. If you try to do a second special move\n" "you will be asked if you want to quit.\n" "\nGood luck!\n\nPress ENTER to begin...");     getchar ();     do {         mask = 0;         for (c = 0; c &lt; 16; c++) {             while (1 &lt;&lt; (b[c]=rand () % 16) &amp; mask);             mask  = 1 &lt;&lt; b[c];             b[c] += 'A';         }         done = 0;         for (turn = special = 0; !done; turn++) {             done = 0;             putchar ('\t');             DRAW;         }     } while (1); }</pre>	
	
Listing continued on page 2...	

TITLE.C	Listing Continued from page 1....
<pre> printf ("\n\nWhich position to rotate (0 - 9) ? "); while (!isdigit(input = getchar ()))     if (isalnum(input))         printf("\nPlease choose a number between 0 and 9 ? "); input -= '0'; if (input) {     temp = b[t[--input]];     for (c = 0; c &lt; 3; c++)         b[t[input] + x[c]] = b[t[input] + x[c + 1]];     b[t[input] + x[3]] = temp; } else {     printf ("\nDo you want to quit? (y/n) ");     while (!isalpha(input = getchar ()))         if (isalpha (input) &amp;&amp; toupper (input) != 'Y'             &amp;&amp; toupper (input) != 'N') printf("\nY or N please ? ");     if (toupper (input) == 'Y') done = 2;     else if (!special) {         printf ("Which two letters do you want to switch? ");         do input = getchar ();         while (toupper (input) &lt; 'A'    toupper (input) &gt; 'P');         for (c = 0; b[c] != toupper(input); c++);         do input = getchar ();         while (toupper (input) &lt; 'A'    toupper (input) &gt; 'P');         for (d = 0; b[d] != toupper(input); d++);         if (d - c != 1 &amp;&amp; d - c != -1 &amp;&amp; d - c != 4 &amp;&amp; d - c != -4) {             puts ("Those two letters are not adjacent."); turn --;         } else {temp = b[c]; b[c] = b[d]; b[d] = temp; special = 1; turn++;}     } } if (!done) for (c = done = 1; c &lt; 16; c++)     if (b[c - 1] &gt; b[c]) done = 0; } if (!--done) {     DRAW;     printf ("\n\nYou ordered the board in %d moves!", turn);     printf ("\nDo you want to play again? "); } else printf ("\nDo you want to try another? "); do input = getchar (); while (toupper (input) != 'Y' &amp;&amp; toupper (input) != 'N') ; done = (toupper (input) == 'N'); } while (!done); puts ("Goodbye!"); exit (0); } </pre>	

#### Author's Notes:

This program is encompassed entirely in the *main()*, no sub functions, and one macro. Although it's technically bad form to do this it did cut the program down in size. In this case breaking the program into additional functions could have made the flow of the code more readable, tho not necessarily. Functions serve to make frequently repeated processes possible without copying code blocks but since this game was pretty linear in its flow it lent itself well to a single *main()* routine.