#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <time.h>

#include <string.h>

#include <windows.h>

#define ANSI\_COLOR\_RED "\x1b[31m"

#define ANSI\_COLOR\_GREEN "\x1b[32m"

#define ANSI\_COLOR\_YELLOW "\x1b[33m"

#define ANSI\_COLOR\_BLUE "\x1b[34m"

#define ANSI\_COLOR\_MAGENTA "\x1b[35m"

#define ANSI\_COLOR\_CYAN "\x1b[36m"

#define ANSI\_COLOR\_RESET "\x1b[0m"

// create matrix 4\*4

void createMatrix(int arr[][4])

{

int n = 15;

int num[n], i, j;

for (i = 0; i < 15; i++) // These 1-15 number will be in th matrix

num[i] = i + 1;

srand(time(NULL)); // for random number generation

int lastIndex = n - 1, index;

for (i = 0; i < 4; i++)

for (j = 0; j < 4; j++)

{

if (lastIndex >= 0)

{

index = rand() % (lastIndex + 1); // idea : performing % operation by (lastIndex+1)

arr[i][j] = num[index]; // will give index , so put that num[index] number in matrix

num[index] = num[lastIndex--]; // and replace last number with this indexed positioned number

} // and finally lastIndex--

}

arr[i - 1][j - 1] = 0; // last number is zero

}

// showing matrix

void showMatrix(int arr[][4])

{

int i, j;

printf("--------------------\n");

for (i = 0; i < 4; i++)

{

printf("|");

for (j = 0; j < 4; j++)

{

if (arr[i][j] != 0)

printf("%2d | ", arr[i][j]);

else

printf(" | ");

}

printf("\n");

}

printf("--------------------\n");

}

// winning check by this function

int winner(int arr[][4])

{

int i, j, k = 1;

for (i = 0; i < 4; i++)

{

for (j = 0; j < 4; j++, k++)

if (arr[i][j] != k && k != 16)

break;

if (j < 4)

break;

}

if (j < 4)

return 0;

return 1;

}

// swap function to swap two numbers

void swap(int \*x, int \*y)

{

\*x = \*x + \*y;

\*y = \*x - \*y;

\*x = \*x - \*y;

printf("");

}

// Reads the user input character and return ascii value of that

int readEnteredKey()

{

char c;

c = \_getch();

if (c == -32)

c = \_getch();

return c;

}

// shift up function

int shiftUp(int arr[][4])

{

int i, j;

for (i = 0; i < 4; i++)

{

for (j = 0; j < 4; j++)

if (arr[i][j] == 0)

break;

if (j < 4)

break;

}

if (i == 3) // shifting not possible

return 0;

// Successfully swapped two numbers !

swap(&arr[i][j], &arr[i + 1][j]);

return 1; // Success

}

int shiftDown(int arr[][4])

{

int i, j;

for (i = 0; i < 4; i++)

{

for (j = 0; j < 4; j++)

if (arr[i][j] == 0)

break;

if (j < 4)

break;

}

if (i == 0) // shifting not possible

return 0;

swap(&arr[i][j], &arr[i - 1][j]); // swap numbers

return 1; // shift up success

}

int shiftRight(int arr[][4])

{

int i, j;

for (i = 0; i < 4; i++)

{

for (j = 0; j < 4; j++)

if (arr[i][j] == 0)

break;

if (j < 4)

break;

}

if (j == 0) // shifting not possible

return 0;

swap(&arr[i][j], &arr[i][j - 1]);

return 1; // shift up success

}

int shiftLeft(int arr[][4])

{

int i, j;

for (i = 0; i < 4; i++)

{

for (j = 0; j < 4; j++)

if (arr[i][j] == 0)

break;

if (j < 4)

break;

}

if (j == 3) // shifting not possible

return 0;

swap(&arr[i][j], &arr[i][j + 1]);

return 1; // shift up success

}

// Game rules

void gameRule(int arr[][4])

{

system("cls");

int i, j, k = 1;

printf("\t\t MATRIX PUZZLE\n");

printf("\n");

printf(ANSI\_COLOR\_RED " RULE OF THIS GAME \n" ANSI\_COLOR\_RESET);

printf(ANSI\_COLOR\_RED "\n1.You can move only 1 step at a time by arrow key " ANSI\_COLOR\_RESET);

printf("\n\tMove Up : by Up arrow key ");

printf("\n\tMove Down : by Down arrow key");

printf("\n\tMove Left : by Left arrow key");

printf("\n\tMove Right: by Right arrow key");

printf(ANSI\_COLOR\_RED "\n2.You can move number at empty position only " ANSI\_COLOR\_RESET);

printf("\n");

printf(ANSI\_COLOR\_RED "\n3. For each valid move : your total number of move will decreased by 1 \n" ANSI\_COLOR\_RESET);

printf(ANSI\_COLOR\_RED "4. Wining situation : " ANSI\_COLOR\_RESET);

printf(ANSI\_COLOR\_RED " Number in a 4\*4 matrix should be in order from 1 to 15 " ANSI\_COLOR\_RESET);

printf("\n\n winning situation: \n");

printf(ANSI\_COLOR\_YELLOW "---------------------\n" ANSI\_COLOR\_RESET);

for (i = 0; i < 4; i++)

{

printf(ANSI\_COLOR\_YELLOW "| " ANSI\_COLOR\_RESET);

for (j = 0; j < 4; j++)

{

if (arr[i][j] != 0)

printf(ANSI\_COLOR\_YELLOW "%2d | " ANSI\_COLOR\_RESET, 4 \* i + j + 1);

else

printf(ANSI\_COLOR\_YELLOW " |" ANSI\_COLOR\_RESET);

}

printf("\n");

}

printf(ANSI\_COLOR\_YELLOW "---------------------\n" ANSI\_COLOR\_RESET);

printf("\n5.You can exit the game at any time by pressing 'E' or 'e' ");

printf("\nSo Try to win in minimum no of move \n");

printf("\nEnter any key to start..... ");

int x = readEnteredKey();

}

// main function

int main()

{

int arr[4][4]; // matrix

int maxTry = 4; // maximum move

char name[20];

for (int k = 0; k < 3; k++)

printf("\n");

printf("Player Name: ");

scanf("%s", name);

system("cls"); // to clear screen

while (1)

{

createMatrix(arr); // calling function to create matrix

gameRule(arr); // game rule function calling

while (!winner(arr)) // checking for winning situation

{

system("cls");

if (!maxTry) // for remaining zero move

break;

printf("\n\n Player Name: %s , Move remaining : %d\n\n", name, maxTry);

showMatrix(arr); // show matrix

int key = readEnteredKey(); // this will return ascii code of user entered key

switch (key) // maping

{

case 101: // ascii of E

case 69: // ascii of e

printf("\a\a\a\a\a\a\n Thanks for Playing ! \n\a");

printf("\nHit 'Enter' to exit the game \n");

key = readEnteredKey();

return 0;

case 72: // arrow up

if (shiftUp(arr))

maxTry--;

break;

case 80: // arrow down

if (shiftDown(arr))

maxTry--;

break;

case 77: // arrow right

if (shiftRight(arr))

maxTry--;

break;

case 75: // arrow left

if (shiftLeft(arr))

maxTry--;

break;

default:

printf("\n\n \a\a Not Allowed \a");

}

}

if (!maxTry)

printf(ANSI\_COLOR\_RED "\n\a YOU LOSE ! \a\a\n" ANSI\_COLOR\_RESET);

else

printf(ANSI\_COLOR\_GREEN "\n\a!!!!!!!!!!!!!Congratulations %s for winning this game !!!!!!!!!!!!!\n\a" ANSI\_COLOR\_RESET, name);

fflush(stdin); // Will clear the buffer

char check;

printf(ANSI\_COLOR\_GREEN "\nWant to play again ? \n" ANSI\_COLOR\_RESET);

printf("enter 'y' to play again: ");

scanf("%c", &check);

// Leave the game here itself !

if ((check != 'y') && (check != 'Y'))

break;

maxTry = 4;

}

return 0;

}