#include <stdio.h>

#include<stdlib.h>

#include<time.h>

#include<windows.h>

#include<math.h>

#include<string.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"

void gamerules( )

{

int i,j,k=1;

system("cls");

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 "3. 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-----------------\n" ANSI\_COLOR\_RESET);

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

{

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

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

{

if(i==3&&j==3)

{

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

break;

}

printf(ANSI\_COLOR\_YELLOW"%2d |" ANSI\_COLOR\_RESET,k);

}

if(i==3&&j==3)

break;

printf("\n");

}

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

}

int readenteredkey()

{

char c=\_getch();// ascii code in char variable

if (c==-32)

c=\_getch();

return c;

}

void creatmatrix( int a[][4])

{

int i,j,num[15];

for(i=0; i<15; i++)

{

num[i]=i+1;

}

srand(time(NULL));/\*srand-Providing the seed for the generation of the random

number on each run

time- is used as seed \*/

int lastindex=14, index;

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

{

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

{

if(lastindex>=0)

{

index=rand()%(lastindex + 1);

a[i][j]=num[index];

num[index]=num[lastindex--];

}

}

}

a[i-1][j-1]=0;

}

void showmatrix(int a[][4])

{

int i,j;

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

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

{

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

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

{

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

{

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

}

else

printf(ANSI\_COLOR\_YELLOW"%2d |" ANSI\_COLOR\_RESET,a[i][j]);

}

printf("\n");

}

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

}

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

{

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

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

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

printf("");

}

int winning(int a[][4])

{

int i,j,k=1;

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

{

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

{

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

{

break;

}

k++;

}

if(j<4)

{

break;

}

}

if(j<4)

return 0;

return 1;

}

int shiftup(int a[][4])

{

int i,j;

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

{

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

{

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

break;

}

if(j<4)

break;

}

if(i==3)

return 0;

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

return 1;

}

int shiftdown(int a[][4])

{

int i,j;

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

{

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

{

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

break;

}

if(j<4)

break;

}

if(i==0)

return 0;

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

return 1;

}

int shiftright(int a[][4])

{

int i,j;

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

{

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

{

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

break;

}

if(j<4)

break;

}

if(j==0)

return 0;

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

return 1;

}

int shiftleft(int a[][4])

{

int i,j;

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

{

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

{

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

break;

}

if(j<4)

break;

}

if(j==3)

return 0;

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

return 1;

}

int main()

{

int a[4][4], maxtry = 10;

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

char name[20];

printf("Player Name: ");

gets(name);

while(1)

{

gamerules();

creatmatrix(a);

while(!winning(a))

{

system("cls");

if(!maxtry)

break;

printf("\n\nPlayer :- %s\t\t\t",name);

printf("Maxtry :- %d\n\n\n\n",maxtry);

showmatrix(a);

int key=readenteredkey();

switch(key)

{

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(a))

maxtry--;

break;

case 80: // arrow down

if (shiftdown(a))

maxtry--;

break;

case 77: // arrow right

if (shiftright(a))

maxtry--;

break;

case 75: // arrow left

if (shiftleft(a))

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 = 10;

}

}