**מיני פרוייקט משחק 2048 בc# - אופיר הופמן י3**

class Trect

{

int x;

int y;

double width;

double height;

ConsoleColor Fcolor;

public Trect(int x, int y, double width, double height, ConsoleColor c)

{

this.x = x;

this.y = y;

this.width = width;

this.height = height;

this.Fcolor = c;

}

public Trect()

{

this.x = 10;

this.y = 10;

this.width = 10;

this.height = 20;

this.Fcolor = ConsoleColor.White;

}

public void SetX(int x)

{

this.x = x;

}

public int GetX()

{

return x;

}

public void SetY(int y)

{

this.y = y;

}

public int GetY()

{

return y;

}

public void SetWidth(double w)

{

this.width = w;

}

public double GetWidth()

{

return this.width;

}

public void SetHeight(double h)

{

this.height = h;

}

public double Getheight()

{

return this.height;

}

public void SetFcolor(ConsoleColor Fcolor)

{

this.Fcolor = Fcolor;

}

public ConsoleColor GetFcolor()

{

return this.Fcolor;

}

public double GetArea()

{

return this.height \* this.width;

}

public double GetPerimeter()

{

return 2 \* this.height + 2 \* this.width;

}

public double GetDiagonal()

{

return Math.Sqrt(this.width \* this.width + this.height \* this.height);

}

public void Draw()

{

this.DrawRectPath(this.Fcolor);

}

public void Undraw()

{

this.DrawRectPath(ConsoleColor.Black);

}

public override string ToString()

{

return "X:" + x + " Y:" + y + " Width:" + width + " Height:" + height + " Color:" + Fcolor;

}

private void DrawRectPath(ConsoleColor color)

{

Console.ForegroundColor = color;

Console.SetCursorPosition(this.x, this.y);

int line = this.y;

int width = (int)this.width;

int height = (int)this.height;

if (width > 0 && height > 0)

{

Console.Write('╔');

for (int i = 1; i < width - 1; i++)

Console.Write('═');

if (this.width >= 2)

Console.Write('╗');

for (int i = 1; i < height - 1; i++)

{

line++;

Console.SetCursorPosition(this.x, line);

Console.Write('║');

Console.SetCursorPosition(this.x + width - 1, line);

Console.Write('║');

}

if (height >= 2)

{

line++;

Console.SetCursorPosition(this.x, line);

Console.Write('╚');

for (int i = 1; i < width - 1; i++)

Console.Write('═');

if (width >= 2)

Console.Write('╝');

}

}

}

}

class My2048

{

private int[,] arr;

private int score;

public enum Direction { Right, Left, Up, Down};

public Direction direction;

Random rnd = new Random();

public My2048(int size)

{

this.arr = new int[size, size];

this.score = 0;

AddNum();

}

public void AddNum()

{

int cnt = 0;

int clmnCnt = 0;

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

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

{

cnt++;

clmnCnt++;

}

}

}

int RowInsertIndex = rnd.Next(1, cnt + 1);

int ClmnInsertIndex = rnd.Next(0, clmnCnt + 1);

int num;

int grill = rnd.Next(101);

if (grill >= 0 && grill <= 85)

num = 2;

else

num = 4;

int zeroCnt = 1;

bool cont = true;

for (int i = 0; i < arr.GetLength(0) && cont; i++)

{

for (int j = 0; j < arr.GetLength(0) && cont; j++)

{

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

{

if ((zeroCnt == RowInsertIndex))

{

arr[i, j] = num;

cont = false;

}

else

zeroCnt++;

}

}

}

}

public void Draw()

{

int ypos = 2;

for (int i = 0; i < arr.GetLength(0); i++)

{

int xpos = 2;

for (int j = 0; j < arr.GetLength(0); j++)

{

Trect rec = new Trect(xpos, ypos - 1, 6, 3, ConsoleColor.Blue);

rec.Draw();

if (arr[i, j] == 2)

{

Console.BackgroundColor = ConsoleColor.Blue;

}

else if (arr[i, j] == 4)

{

Console.BackgroundColor = ConsoleColor.Green;

}

else if (arr[i, j] == 8)

{

Console.BackgroundColor = ConsoleColor.Yellow;

}

else if (arr[i, j] == 16)

{

Console.BackgroundColor = ConsoleColor.Cyan;

}

else if (arr[i, j] == 32)

{

Console.BackgroundColor = ConsoleColor.Red;

}

else if (arr[i, j] == 64)

{

Console.BackgroundColor = ConsoleColor.DarkGreen;

}

else if (arr[i, j] == 128)

{

Console.BackgroundColor = ConsoleColor.DarkCyan;

}

else if (arr[i, j] == 256)

{

Console.BackgroundColor = ConsoleColor.DarkMagenta;

}

else if (arr[i, j] == 512)

{

Console.BackgroundColor = ConsoleColor.DarkYellow;

}

else if (arr[i, j] == 1024)

{

Console.BackgroundColor = ConsoleColor.Cyan;

}

else if (arr[i, j] == 2048)

{

Console.BackgroundColor = ConsoleColor.DarkGray;

}

Console.ForegroundColor = ConsoleColor.Black;

Console.SetCursorPosition(xpos + 1, ypos);

if (arr[i, j] > 0)

Console.Write("{0, 4}", arr[i, j]);

else

Console.Write(" ");

xpos += 8;

Console.BackgroundColor = ConsoleColor.Black;

}

ypos += 3;

}

}

public bool MoveLeft()

{

int rowIndex = 0;

int clmnIndex = 0;

bool changed = false;

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

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

{

changed = true;

int save = arr[i, j];

arr[i, j] = arr[rowIndex, clmnIndex];

arr[rowIndex, clmnIndex] = save;

clmnIndex++;

}

}

clmnIndex = 0;

rowIndex++;

}

return changed;

}

public bool MoveRight()

{

int rowIndex = 0;

int clmnIndex = arr.GetLength(0) - 1; ;

bool changed = false;

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = arr.GetLength(0) - 1; j >= 0; j--)

{

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

{

changed = true;

int save = arr[i, j];

arr[i, j] = arr[rowIndex, clmnIndex];

arr[rowIndex, clmnIndex] = save;

clmnIndex--;

}

}

clmnIndex = arr.GetLength(0) - 1;

rowIndex++;

}

return changed;

}

public bool MoveUp()

{

int rowIndex = 0;

int clmnIndex = 0;

bool changed = false;

for (int clmn = 0; clmn < arr.GetLength(0); clmn++)

{

for (int row = 1; row < arr.GetLength(0); row++)

{

if (arr[row, clmn] != 0)

{

changed = true;

int save = arr[row, clmn];

arr[row, clmn] = arr[rowIndex, clmnIndex];

arr[rowIndex, clmnIndex] = save;

rowIndex++;

}

}

rowIndex = 0;

clmnIndex++;

}

return changed;

}

public bool MoveDown()

{

int rowIndex = 0;

int clmnIndex = 0;

bool changed = false;

for (int clmn = 0; clmn < arr.GetLength(0); clmn++)

{

for (int row = arr.GetLength(0) - 1; row > rowIndex; row--)

{

if (arr[row, clmn] != 0)

{

changed = true;

int save = arr[row, clmn];

arr[row, clmn] = arr[rowIndex, clmnIndex];

arr[rowIndex, clmnIndex] = save;

rowIndex++;

row++;

}

}

rowIndex = 0;

clmnIndex++;

}

return changed;

}

public void RightMerge()

{

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0)-1; j++)

{

if (arr[i, j] == arr[i, j+1] && arr[i, j] != 0)

{

arr[i, j+1] \*= 2;

arr[i, j] = 0;

}

}

}

}

public void LeftMerge()

{

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = arr.GetLength(0)-1; j > 0; j--)

{

if (arr[i, j] == arr[i, j-1] && arr[i, j] != 0)

{

arr[i, j-1] \*= 2;

arr[i, j] = 0;

}

}

}

}

public void UpMerge()

{

for (int clmn = 0; clmn < arr.GetLength(0); clmn++)

{

for (int row = 0; row < arr.GetLength(0)-1; row++)

{

if (arr[row,clmn] == arr[row+1,clmn] && arr[row, clmn] != 0)

{

arr[row + 1, clmn] \*= 2;

arr[row, clmn] = 0;

}

}

}

}

public bool Right2048()

{

int[,] check = new int[arr.GetLength(0), arr.GetLength(0)];

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

{

check[i, j] = arr[i, j];

}

}

}

MoveRight();

RightMerge();

MoveRight();

AddNum();

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

if (arr[i, j] != check[i, j])

return true;

}

}

return false;

}

public bool Left2048()

{

int[,] check = new int[arr.GetLength(0), arr.GetLength(0)];

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

check[i, j] = arr[i, j];

}

}

MoveLeft();

LeftMerge();

MoveLeft();

AddNum();

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

if (arr[i, j] != check[i, j])

return true;

}

}

return false;

}

public bool Up2048()

{

int[,] check = new int[arr.GetLength(0), arr.GetLength(0)];

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

check[i, j] = arr[i, j];

}

}

MoveUp();

UpMerge();

MoveUp();

AddNum();

for (int i = 0; i < arr.GetLength(0); i++)

{

for (int j = 0; j < arr.GetLength(0); j++)

{

if (arr[i, j] != check[i, j])

return true;

}

}

return false;

}

public bool Move2048(Direction direction)

{

if (direction == Direction.Left)

return Left2048();

else if (direction == Direction.Right)

return Right2048();

else

return Up2048();

}

}

class Program

{

static void Main(string[] args)

{

Console.CursorVisible = false;

My2048 arr = new My2048(4);

arr.Draw();

bool cont = true;

while (cont)

{

if (Console.KeyAvailable)

{

ConsoleKeyInfo k = Console.ReadKey();

if (k.Key == ConsoleKey.LeftArrow)

{

cont = arr.Move2048(My2048.Direction.Left);

arr.Draw();

}

else if (k.Key == ConsoleKey.RightArrow)

{

cont = arr.Move2048(My2048.Direction.Right);

arr.Draw();

}

else if (k.Key == ConsoleKey.UpArrow)

{

cont = arr.Move2048(My2048.Direction.Up);

arr.Draw();

}

else if (k.Key == ConsoleKey.DownArrow)

{

cont = arr.Move2048(My2048.Direction.Down);

arr.Draw();

}

}

}

Console.Clear();

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("Game Over");

}

}