**שיעורי בית יסודות מערכים 3 – אופיר הופמן י3**

**תרגיל 21**

public static void PrintArr(int[] arr)

{

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

{

Console.Write(arr[i] + "|");

}

}

public static void Ex1()

{

int[] arr = new int[101];

Random rnd = new Random();

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

{

arr[i] = rnd.Next(0, 101);

}

PrintArr(arr);

}

**תרגיל 22**

public static void Ex2()

{

Random rnd = new Random();

int[] arr = new int[15];

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

{

arr[i] = rnd.Next(0, 6);

}

PrintArr(arr);

int index = 0;

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

{

if (arr[i] != 0)

{

int save = arr[i];

arr[i] = arr[index];

arr[index] = save;

index++;

}

}

Console.WriteLine();

PrintArr(arr);

}

**תרגיל 23**

public static void Ex3()

{

Random rnd = new Random();

int[] arr = { 13, 15, 17, 10, 11, 29, 33, 34, 45, 32, 32, 33, 40, 40, 49 };

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

{

arr[i] = rnd.Next(10, 51);

}

PrintArr(arr);

int length = -1;

int newLength = 1;

int indexStart = 0;

int newIndexStart = 0;

for (int i = 0; i < arr.Length-1; i++)

{

if (arr[i] < arr[i+1])

{

newIndexStart = i;

while (arr[i] < arr[i+1] && i < arr.Length-1)

{

newLength++;

i++;

}

i = newIndexStart;

}

if (newLength > length)

{

length = newLength;

indexStart = newIndexStart;

}

}

**תרגיל** **24**

public static int DigCount(int num)

{

int cnt = 0;

while (num != 0)

{

cnt++;

num /= 10;

}

return cnt;

}

public static void Ex4(int num)

{

int[] arr = new int[10];

while(num != 0)

{

int dig = num % 10;

arr[dig]++;

num /= 10;

}

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

{

if (arr[i] != 0)

{

Console.WriteLine(i + ": " + arr[i] + " ");

}

}

}

**2048**

class My2048

{

private int[] arr;

private int score;

public enum Direction { Right, Left};

public Direction direction;

Random rnd = new Random();

public My2048(int size)

{

this.arr = new int[size];

this.score = 0;

AddNum();

}

private void AddNum()

{

int cnt = 0;

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

{

if (arr[i] == 0)

cnt++;

}

int InsertIndex = rnd.Next(1, cnt + 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.Length && cont; i++)

{

if (arr[i] == 0)

{

if (zeroCnt == InsertIndex)

{

arr[i] = num;

cont = false;

}

else

zeroCnt++;

}

}

}

public void Draw()

{

int xpos = 2;

int ypos = 2;

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

{

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

rec.Draw();

if (arr[i] == 2)

{

Console.BackgroundColor = ConsoleColor.Blue;

}

else if (arr[i] == 4)

{

Console.BackgroundColor = ConsoleColor.Green;

}

else if (arr[i] == 8)

{

Console.BackgroundColor = ConsoleColor.Yellow;

}

else if (arr[i] == 16)

{

Console.BackgroundColor = ConsoleColor.Cyan;

}

else if (arr[i] == 32)

{

Console.BackgroundColor = ConsoleColor.Red;

}

else if (arr[i] == 64)

{

Console.BackgroundColor = ConsoleColor.DarkGreen;

}

else if (arr[i] == 128)

{

Console.BackgroundColor = ConsoleColor.DarkCyan;

}

else if (arr[i] == 256)

{

Console.BackgroundColor = ConsoleColor.DarkMagenta;

}

else if (arr[i] == 512)

{

Console.BackgroundColor = ConsoleColor.DarkYellow;

}

else if (arr[i] == 1024)

{

Console.BackgroundColor = ConsoleColor.Cyan;

}

else if (arr[i] == 2048)

{

Console.BackgroundColor = ConsoleColor.DarkGray;

}

Console.ForegroundColor = ConsoleColor.Black;

Console.SetCursorPosition(xpos + 1, ypos);

if (arr[i] > 0)

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

else

Console.Write(" ");

xpos += 8;

Console.BackgroundColor = ConsoleColor.Black;

}

}

public bool MoveLeft()

{

int index = 0;

bool changed = false;

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

{

if (arr[i] != 0)

{

changed = true;

int save = arr[i];

arr[i] = arr[index];

arr[index] = save;

index++;

}

}

return changed;

}

public bool MoveRight()

{

int index = arr.Length-1;

bool changed = false;

for (int i = arr.Length-1; i >= 0; i--)

{

if (arr[i] != 0)

{

changed = true;

int save = arr[i];

arr[i] = arr[index];

arr[index] = save;

index--;

}

}

return changed;

}

public void RightMerge()

{

for (int i = 0; i < arr.Length-1; i++)

{

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

{

arr[i + 1] \*= 2;

arr[i] = 0;

i++;

}

}

}

public void LeftMerge()

{

for (int i = arr.Length-1; i > 0; i--)

{

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

{

arr[i - 1] \*= 2;

arr[i] = 0;

i--;

}

}

}

public bool Right2048()

{

int[] check = new int[arr.Length];

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

{

check[i] = arr[i];

}

MoveRight();

RightMerge();

MoveRight();

AddNum();

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

{

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

return true;

}

return false;

}

public bool Left2048()

{

int[] check = new int[arr.Length];

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

{

check[i] = arr[i];

}

MoveLeft();

LeftMerge();

MoveLeft();

AddNum();

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

{

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

return true;

}

return false;

}

public bool Move2048(Direction direction)

{

if (direction == Direction.Left)

return Left2048();

else

return Right2048();

}

}

static void Main(string[] args)

{

Console.CursorVisible = false;

My2048 arr = new My2048(10);

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();

}

}

}

Console.Clear();

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("Game Over");

}