Problem 17

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Problem17

{

public partial class Form1 : Form

{

public class CActorRocket

{

public int X, Y;

public Bitmap im;

}

public class CActorLaser

{

public int X1, Y1, X2, Y2;

public int visible;

}

public class CActorStar

{

public int X, Y;

public Bitmap im;

public int speed;

}

Bitmap off;

Timer tt = new Timer();

int ctTick = 0;

int rocketAcc = 5;

List<CActorLaser> LLaser = new List<CActorLaser>();

List<CActorRocket> LRocket = new List<CActorRocket>();

List<CActorStar> LStars = new List<CActorStar>();

public Form1()

{

this.WindowState = FormWindowState.Maximized;

this.Load += Form1\_Load;

this.Paint += Form1\_Paint;

this.KeyDown += Form1\_KeyDown;

this.MouseDown += Form1\_MouseDown;

tt.Interval = 100;

tt.Start();

tt.Tick += Tt\_Tick;

}

private void Tt\_Tick(object sender, EventArgs e)

{

if (ctTick % 15 == 0)

{

CreateStar();

rocketAcc += 5;

}

MoveStars();

for (int i=0; i < LLaser.Count; i++)

{

LLaser[i].visible--;

}

ctTick++;

DrawDubb(this.CreateGraphics());

}

private void Form1\_MouseDown(object sender, MouseEventArgs e)

{

DrawDubb(this.CreateGraphics());

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Right)

{

LRocket[0].X += rocketAcc;

}

else if (e.KeyCode == Keys.Left)

{

LRocket[0].X -= rocketAcc;

}

else if (e.KeyCode == Keys.Space)

{

CActorLaser pnn=new CActorLaser();

pnn.X1 = LRocket[0].X + 170/2;

pnn.Y1 = LRocket[0].Y;

pnn.X2 = LRocket[0].X + 160 / 2;

pnn.Y2 = 0;

pnn.visible = 2 ;

LLaser.Add(pnn);

HitStar();

}

//if (LLaser.Count > 0)

{

//LLaser.RemoveAt(LLaser.Count - 1);

}

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

DrawDubb(e.Graphics);

}

private void Form1\_Load(object sender, EventArgs e)

{

off = new Bitmap(this.ClientSize.Width, this.ClientSize.Height);

CreateRocket();

}

void CreateRocket()

{

CActorRocket pnn=new CActorRocket();

pnn.im = new Bitmap("rocket.png");

pnn.im.MakeTransparent();

pnn.X = 150;

pnn.Y = this.ClientSize.Height - 172;

LRocket.Add(pnn);

}

void CreateStar()

{

Random rr = new Random();

CActorStar pnn = new CActorStar();

pnn.im = new Bitmap("star.bmp");

pnn.im.MakeTransparent();

pnn.X = rr.Next(80, this.ClientSize.Width - 80);

pnn.Y = 0;

pnn.speed = 5;

LStars.Add(pnn);

}

int check = 0;

int moveDirection = 1;

void MoveStars()

{

check++;

for (int i = 0; i < LStars.Count; i++)

{

LStars[i].Y += LStars[i].speed;

if (check % 10 == 0)

{

moveDirection \*= -1;

}

else if (check % 17 == 1)

{

moveDirection \*= -1;

}

LStars[i].X += 5 \* moveDirection;

}

if (check == 20)

{

check = 0;

}

}

void HitStar()

{

for (int j = 0; j < LLaser.Count; j++)

{

for (int i = 0; i < LStars.Count; i++)

{

if (LLaser[j].X1 > LStars[i].X && LLaser[j].X1 < LStars[i].X + 50)

{

LStars.RemoveAt(i);

}

}

}

}

void DrawScene(Graphics g)

{

g.Clear(Color.Black);

for (int i = 0; i < LRocket.Count; i++)

{

g.DrawImage(LRocket[i].im, LRocket[i].X, LRocket[i].Y, 170, 170);

}

for (int i = 0; i < LLaser.Count; i++)

{

if (LLaser[i].visible > 0)

{

Pen p = new Pen(Color.Yellow, 3);

g.DrawLine(p, LLaser[i].X1, LLaser[i].Y1, LLaser[i].X2, LLaser[i].Y2);

}

}

for (int i = 0; i < LStars.Count; i++)

{

g.DrawImage(LStars[i].im, LStars[i].X, LStars[i].Y, 50, 50);

}

}

void DrawDubb(Graphics g)

{

Graphics g2 = Graphics.FromImage(off);

DrawScene(g2);

g.DrawImage(off, 0, 0);

}

}

}

Problem 18

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.LinkLabel;

namespace Problem18

{

public partial class Form1 : Form

{

public class CActorBall

{

public int X, Y, W, H;

public Color cl;

}

public class CActorMoveRec

{

public int X, Y, W, H;

public Color cl;

public int speed;

public int dirX, dirY;

}

public class CActorPath

{

public int X1, Y1, X2, Y2;

public Color cl;

}

public Form1()

{

this.WindowState = FormWindowState.Maximized;

this.Load += Form1\_Load;

this.Paint += Form1\_Paint;

this.KeyDown += Form1\_KeyDown;

tt.Interval = 100;

tt.Start();

tt.Tick += Tt\_Tick;

}

Bitmap off;

Timer tt = new Timer();

List<CActorBall> LBall = new List<CActorBall>();

List<CActorPath> LPath = new List<CActorPath>();

List<CActorMoveRec> LMoveRec = new List<CActorMoveRec>();

int c1X = 300, c1Y = 180, c2X = 550, c2Y = 180, c3X = 300, c3Y = 540, c4X = 550, c4Y = 540; //corners

private void Tt\_Tick(object sender, EventArgs e)

{

MoveRec();

BallHitRec();

BallHitPath();

DrawDubb(this.CreateGraphics());

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Up)

{

LBall[0].Y -= 2;

}

else if (e.KeyCode == Keys.Down)

{

LBall[0].Y += 2;

}

else if (e.KeyCode == Keys.Left)

{

LBall[0].X -= 2;

}

else if (e.KeyCode == Keys.Right)

{

LBall[0].X += 2;

}

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

DrawDubb(e.Graphics);

}

private void Form1\_Load(object sender, EventArgs e)

{

off = new Bitmap(this.ClientSize.Width, this.ClientSize.Height);

CreateBall();

CreatePath();

CreateMoveRec();

}

void CreateBall()

{

CActorBall ball = new CActorBall();

ball.X = 100;

ball.Y = 330;

ball.W = 25;

ball.H = 25;

ball.cl = Color.Yellow;

LBall.Add(ball);

}

void CreatePath()

{

//1. first vertical line left

CActorPath pnn = new CActorPath();

pnn.X1 = 70;

pnn.Y1 = 300;

pnn.X2 = 70;

pnn.Y2 = 420;

pnn.cl = Color.White;

LPath.Add(pnn);

//2. horizontal line up of rec left

pnn = new CActorPath();

pnn.X1 = 70;

pnn.Y1 = 300;

pnn.X2 = 300;

pnn.Y2 = 300;

pnn.cl = Color.White;

LPath.Add(pnn);

//3. horizontal line down of rec left

pnn = new CActorPath();

pnn.X1 = 70;

pnn.Y1 = 420;

pnn.X2 = 300;

pnn.Y2 = 420;

pnn.cl = Color.White;

LPath.Add(pnn);

//4. vertical line left of rec up

pnn = new CActorPath();

pnn.X1 = 300;

pnn.Y1 = 180;

pnn.X2 = 300;

pnn.Y2 = 300;

pnn.cl = Color.White;

LPath.Add(pnn);

//5. horizontal line up of rec up

pnn = new CActorPath();

pnn.X1 = 300;

pnn.Y1 = 180;

pnn.X2 = 550;

pnn.Y2 = 180;

pnn.cl = Color.White;

LPath.Add(pnn);

//6. vertical line right of rec up

pnn = new CActorPath();

pnn.X1 = 550;

pnn.Y1 = 180;

pnn.X2 = 550;

pnn.Y2 = 300;

pnn.cl = Color.White;

LPath.Add(pnn);

//7. vertical line left of rec down

pnn = new CActorPath();

pnn.X1 = 300;

pnn.Y1 = 420;

pnn.X2 = 300;

pnn.Y2 = 540;

pnn.cl = Color.White;

LPath.Add(pnn);

//8. horizontal line down of rec down

pnn = new CActorPath();

pnn.X1 = 300;

pnn.Y1 = 540;

pnn.X2 = 550;

pnn.Y2 = 540;

pnn.cl = Color.White;

LPath.Add(pnn);

//9. vertical line right of rec down

pnn = new CActorPath();

pnn.X1 = 550;

pnn.Y1 = 420;

pnn.X2 = 550;

pnn.Y2 = 540;

pnn.cl = Color.White;

LPath.Add(pnn);

//10. horizontal line up of rec right

pnn = new CActorPath();

pnn.X1 = 550;

pnn.Y1 = 300;

pnn.X2 = 800;

pnn.Y2 = 300;

pnn.cl = Color.White;

LPath.Add(pnn);

//11. vertical line right of rec right

pnn = new CActorPath();

pnn.X1 = 800;

pnn.Y1 = 300;

pnn.X2 = 800;

pnn.Y2 = 420;

pnn.cl = Color.White;

LPath.Add(pnn);

//12. horizontal line down of rec right

pnn = new CActorPath();

pnn.X1 = 550;

pnn.Y1 = 420;

pnn.X2 = 800;

pnn.Y2 = 420;

pnn.cl = Color.White;

LPath.Add(pnn);

}

void CreateMoveRec()

{

CActorMoveRec pnn = new CActorMoveRec();

pnn.X = 300;

pnn.Y = 180;

pnn.W = 120;

pnn.H = 120;

pnn.cl = Color.White;

pnn.speed = 2;

pnn.dirX = 1;

pnn.dirY = 0;

LMoveRec.Add(pnn);

pnn = new CActorMoveRec();

pnn.X = 430;

pnn.Y = 420;

pnn.W = 120;

pnn.H = 120;

pnn.cl = Color.White;

pnn.speed = 2;

pnn.dirX = -1;

pnn.dirY = 0;

LMoveRec.Add(pnn);

}

void BallHitPath()

{

int ballRight = LBall[0].X + LBall[0].W;

int ballLeft = LBall[0].X;

int ballTop = LBall[0].Y;

int ballBottom = LBall[0].Y + LBall[0].H;

for (int i = 0; i < LPath.Count; i++)

{

CActorPath path = LPath[i];

if ((ballRight >= path.X1 && ballLeft <= path.X1 && ballTop <= path.Y2 && ballBottom >= path.Y1) ||

(ballLeft <= path.X2 && ballRight >= path.X2 && ballTop <= path.Y2 && ballBottom >= path.Y1) ||

(ballTop <= path.Y1 && ballBottom >= path.Y1 && ballLeft <= path.X2 && ballRight >= path.X1) ||

(ballBottom >= path.Y2 && ballTop <= path.Y2 && ballLeft <= path.X2 && ballRight >= path.X1))

{

tt.Stop();

}

}

}

void BallHitRec()

{

for (int i = 0; i < LMoveRec.Count; i++)

{

int ballRight = LBall[0].X + LBall[0].W;

int ballLeft = LBall[0].X;

int ballTop = LBall[0].Y;

int ballBottom = LBall[0].Y + LBall[0].H;

int recRight = LMoveRec[i].X + LMoveRec[i].W;

int recLeft = LMoveRec[i].X;

int recTop = LMoveRec[i].Y;

int recBottom = LMoveRec[i].Y + LMoveRec[i].H;

if (ballRight >= recLeft && ballLeft <= recRight && ballBottom >= recTop && ballTop <= recBottom)

{

tt.Stop();

}

}

}

void MoveRec()

{

for (int i = 0; i < LMoveRec.Count; i++)

{

if (LMoveRec[i].X == c1X && LMoveRec[i].Y == c1Y)

{

LMoveRec[i].dirX = 1;

LMoveRec[i].dirY = 0;

}

else if (LMoveRec[i].X + LMoveRec[i].W == c2X && LMoveRec[i].Y == c2Y)

{

LMoveRec[i].dirX = 0;

LMoveRec[i].dirY = 1;

}

else if (LMoveRec[i].X == c3X && LMoveRec[i].Y + LMoveRec[i].H == c3Y)

{

LMoveRec[i].dirX = 0;

LMoveRec[i].dirY = -1;

}

else if (LMoveRec[i].X + LMoveRec[i].W == c4X && LMoveRec[i].Y + LMoveRec[i].H == c4Y)

{

LMoveRec[i].dirX = -1;

LMoveRec[i].dirY = 0;

}

if (LMoveRec[i].dirX == 0 && LMoveRec[i].dirY == 1)

{

LMoveRec[i].Y += LMoveRec[i].speed;

}

else if (LMoveRec[i].dirX == 0 && LMoveRec[i].dirY == -1)

{

LMoveRec[i].Y -= LMoveRec[i].speed;

}

else if (LMoveRec[i].dirX == 1 && LMoveRec[i].dirY == 0)

{

LMoveRec[i].X += LMoveRec[i].speed;

}

else if (LMoveRec[i].dirX == -1 && LMoveRec[i].dirY == 0)

{

LMoveRec[i].X -= LMoveRec[i].speed;

}

}

}

void DrawScene(Graphics g)

{

g.Clear(Color.Black);

for (int i = 0; i < LBall.Count; i++)

{

Pen pn = new Pen(LBall[i].cl);

g.DrawEllipse(pn, LBall[i].X, LBall[i].Y, LBall[i].W, LBall[i].H);

SolidBrush brush = new SolidBrush(LBall[i].cl);

g.FillEllipse(brush, LBall[i].X, LBall[i].Y, LBall[i].W, LBall[i].H);

}

for (int i = 0; i < LMoveRec.Count; i++)

{

Pen pn = new Pen(LMoveRec[i].cl);

g.DrawRectangle(pn, LMoveRec[i].X, LMoveRec[i].Y, LMoveRec[i].W, LMoveRec[i].H);

SolidBrush brush = new SolidBrush(LMoveRec[i].cl);

g.FillRectangle(brush, LMoveRec[i].X, LMoveRec[i].Y, LMoveRec[i].W, LMoveRec[i].H);

}

for (int i = 0; i < LPath.Count; i++)

{

Pen pn = new Pen(LPath[i].cl, 5);

g.DrawLine(pn, LPath[i].X1, LPath[i].Y1, LPath[i].X2, LPath[i].Y2);

}

}

void DrawDubb(Graphics g)

{

Graphics g2 = Graphics.FromImage(off);

DrawScene(g2);

g.DrawImage(off, 0, 0);

}

}

}