**Лабораторная работа No8**

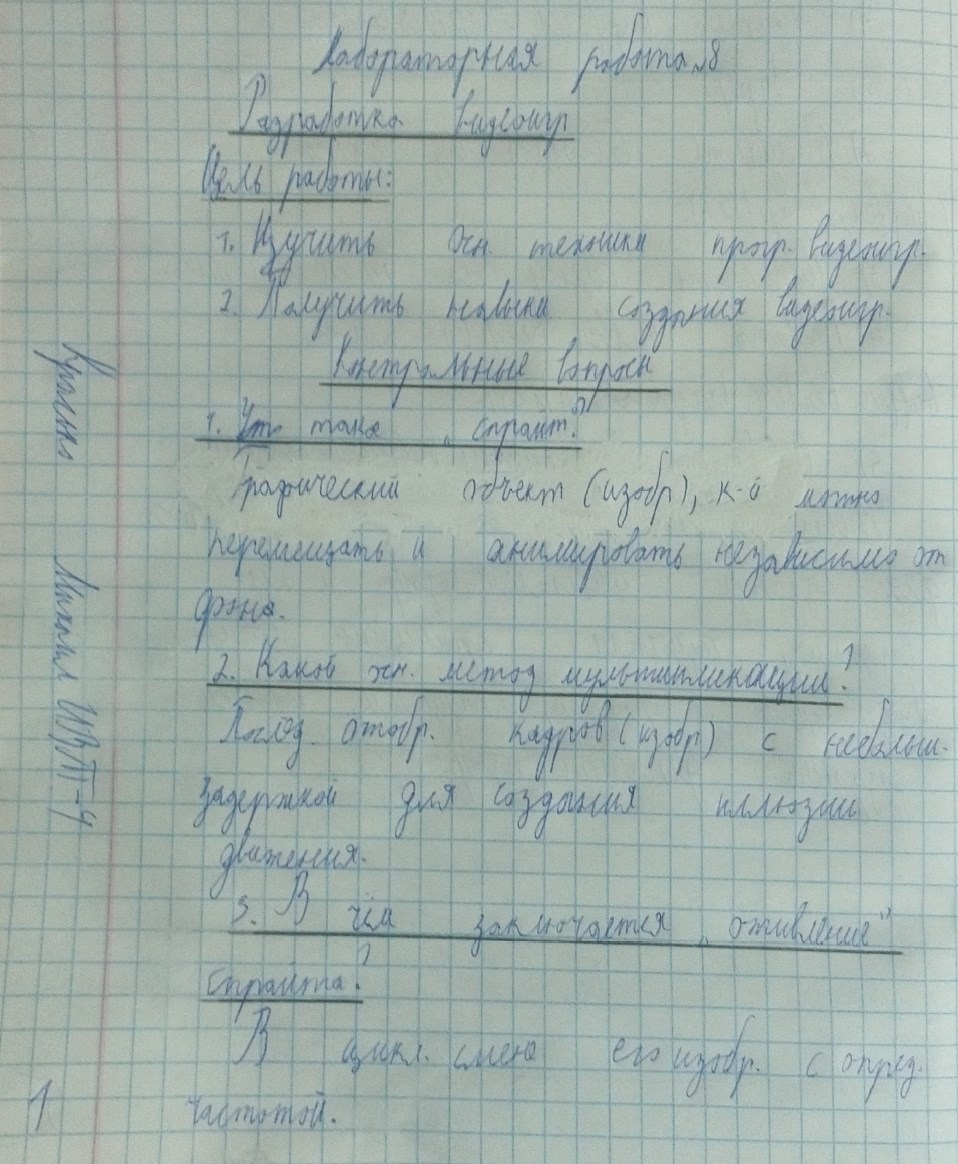
**Разработка видеоигр**

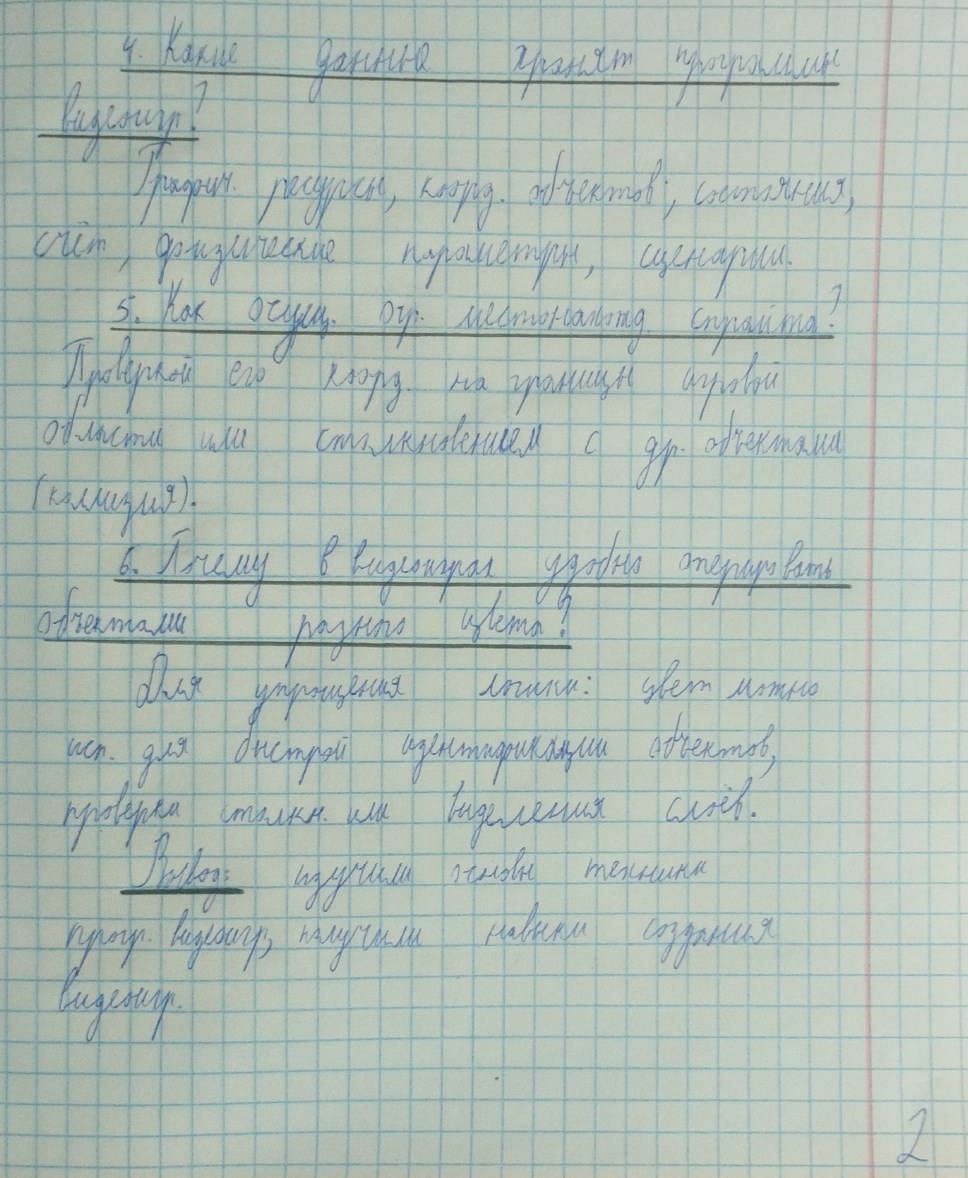
**Цель работы:**

1. Изучить основы техники программирования видеоигр.

2. Получить навыкисоздания видеоигр.

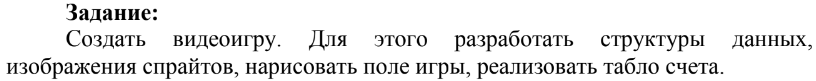
**Контрольные вопросы**

****

****

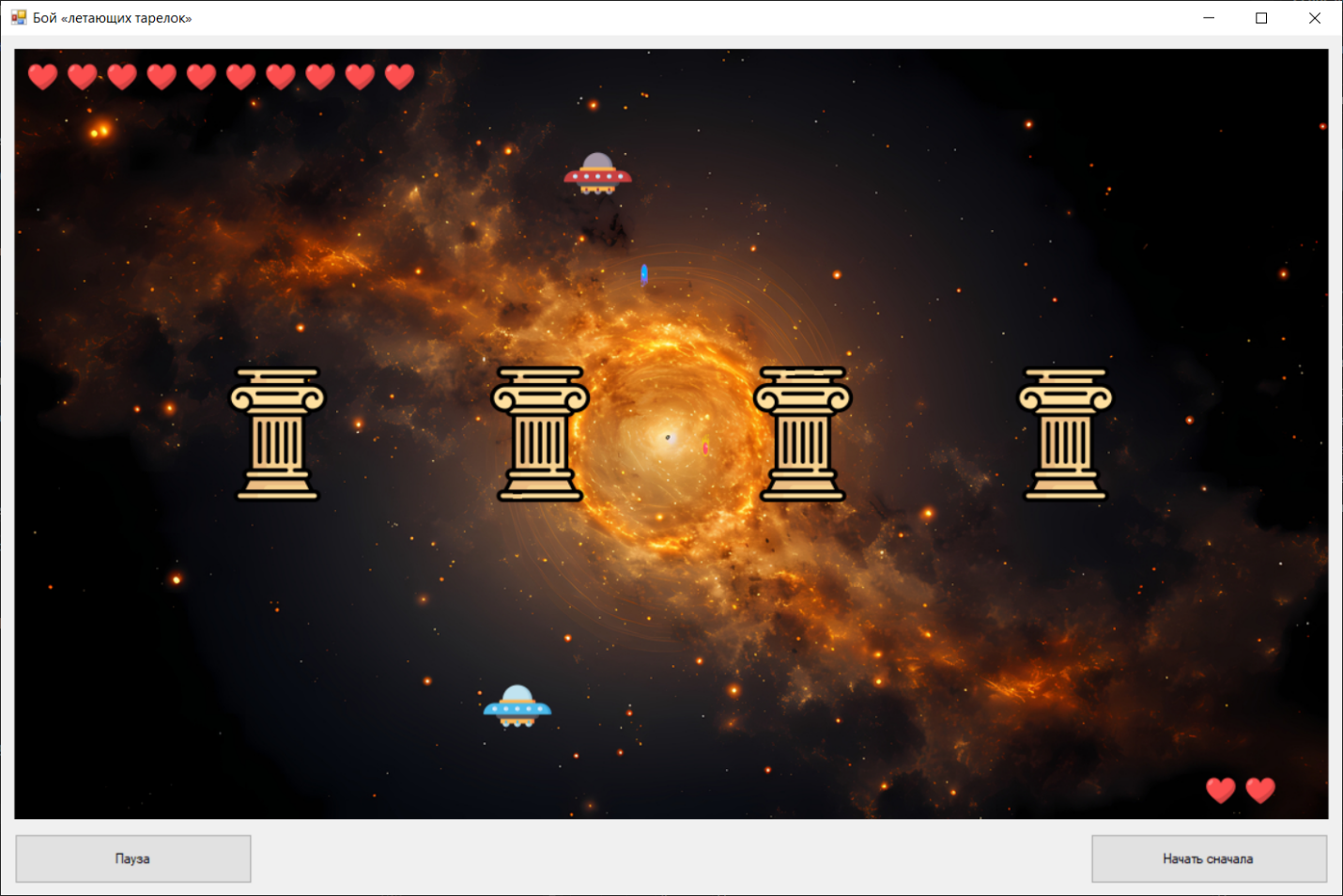
**Ход работы**

**Вариант 5**

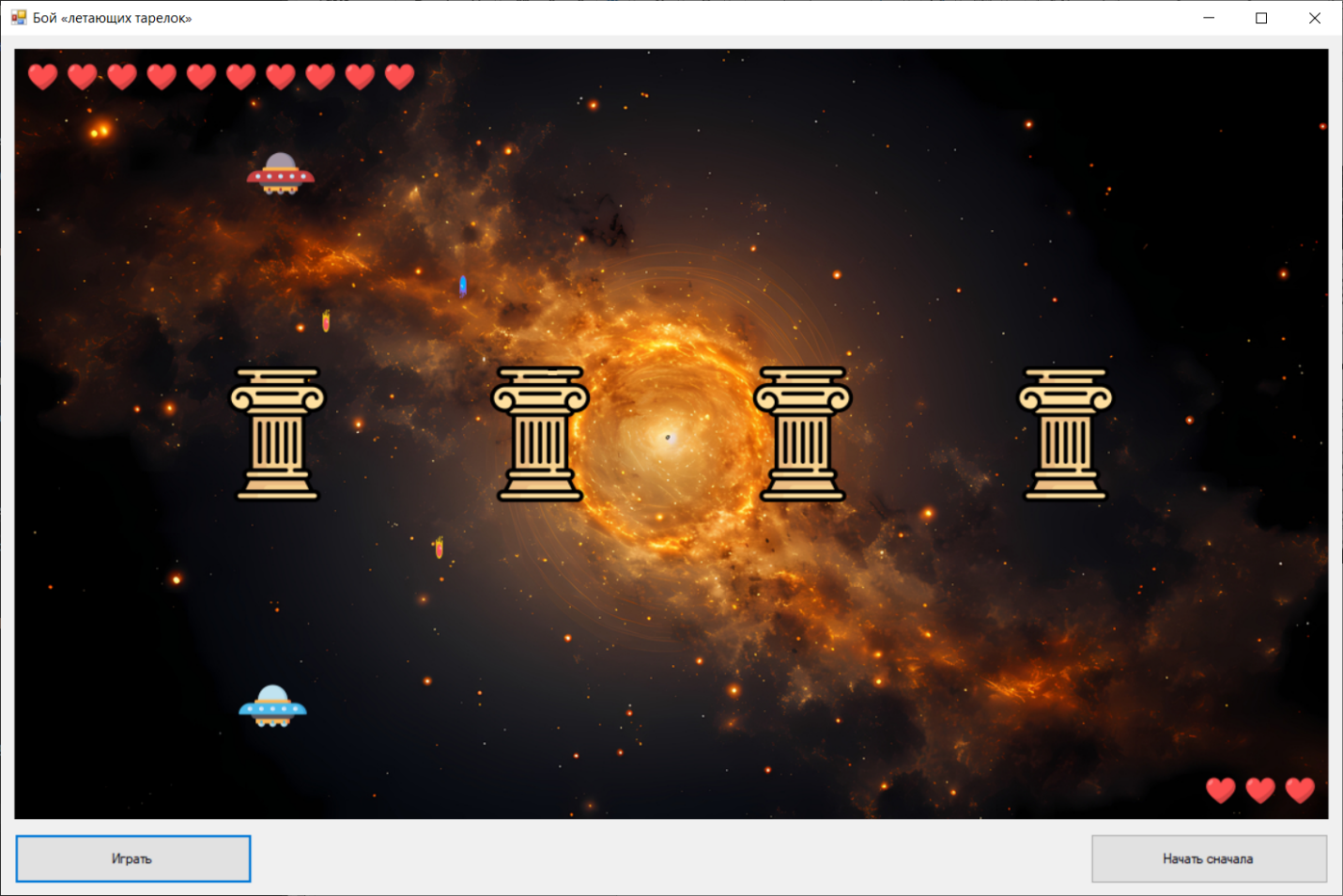




Бой «летающий тарелок», функционал: игрок и враг перемещаются по диагонали, игрок стреляет при нажатии Space и перемещается стрелками. Враг может менять направление движения при достижении конца окна либо произвольно с заданной частотой. Колоны блокируют выстрелы. Если у игрока или врага заканчивается HP, то на экран выводится соответствующая надпись о победе или поражении. Имеется возможность остановки игры (Паузы).







**Код проекта:**

using System;

using System.Collections.Generic;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace КГ\_ЛР8\_Красько\_ИВТ\_4\_2курс

{

public partial class FormUFOFight : Form

{

private PictureBox player;

private PictureBox enemy;

private List<PictureBox> obstacles = new List<PictureBox>();

private List<PictureBox> playerBullets = new List<PictureBox>();

private List<PictureBox> enemyBullets = new List<PictureBox>();

private Image playerImage;

private Image enemyImage;

private Image backgroundImage;

private Image columnImage;

private Image playerBulletImage;

private Image enemyBulletImage;

private Image heartImage;

private List<PictureBox> playerHearts = new List<PictureBox>();

private List<PictureBox> enemyHearts = new List<PictureBox>();

private bool isGameActive = false;

private int playerSpeed = 8;

private bool moveLeft = false;

private bool moveRight = false;

private bool shootPressed = false;

private int enemySpeed = 5;

private int enemyDirection = 1;

private Random random = new Random();

private int enemyMoveCounter = 0;

private int playerFireCooldown = 0;

private int enemyFireCooldown = 0;

private int playerLives = 3;

private int enemyLives = 10;

public FormUFOFight()

{

InitializeComponent();

}

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

{

InitializeGame();

timerGameStep.Stop();

timerEnemy.Stop();

this.Focus();

}

private void InitializeGame()

{

LoadGameImages();

CreateGameObjectsWithImages();

playerLives = 3;

enemyLives = 10;

enemyMoveCounter = 0;

playerFireCooldown = 0;

enemyFireCooldown = 0;

CreateHearts();

}

private void LoadGameImages()

{

playerImage = Image.FromFile("ufo\_player.png");

enemyImage = Image.FromFile("ufo\_enemy.png");

backgroundImage = Image.FromFile("space.jpg");

pictureBoxGame.BackgroundImage = backgroundImage;

pictureBoxGame.BackgroundImageLayout = ImageLayout.Stretch;

columnImage = Image.FromFile("column.png");

playerBulletImage = Image.FromFile("player\_bullet.png");

enemyBulletImage = Image.FromFile("enemy\_bullet.png");

heartImage = Image.FromFile("HP\_icon.png");

}

private void CreateGameObjectsWithImages()

{

pictureBoxGame.Controls.Clear();

obstacles.Clear();

ClearBullets();

ClearHearts();

if (playerImage != null)

{

player = new PictureBox();

player.Image = playerImage;

player.SizeMode = PictureBoxSizeMode.StretchImage;

player.Size = new Size(60, 60);

player.BackColor = Color.Transparent;

player.Location = new Point(pictureBoxGame.Width / 2 - 30, 550);

pictureBoxGame.Controls.Add(player);

player.BringToFront();

}

if (enemyImage != null)

{

enemy = new PictureBox();

enemy.Image = enemyImage;

enemy.SizeMode = PictureBoxSizeMode.StretchImage;

enemy.Size = new Size(60, 60);

enemy.BackColor = Color.Transparent;

enemy.Location = new Point(pictureBoxGame.Width / 2 - 30, 80);

pictureBoxGame.Controls.Add(enemy);

enemy.BringToFront();

}

CreateObstacles();

CreateHearts();

}

private void ClearBullets()

{

foreach (var bullet in playerBullets)

{

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

}

playerBullets.Clear();

foreach (var bullet in enemyBullets)

{

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

}

enemyBullets.Clear();

}

private void ClearHearts()

{

foreach (var heart in playerHearts)

{

pictureBoxGame.Controls.Remove(heart);

heart.Dispose();

}

playerHearts.Clear();

foreach (var heart in enemyHearts)

{

pictureBoxGame.Controls.Remove(heart);

heart.Dispose();

}

enemyHearts.Clear();

}

private void CreateHearts()

{

ClearHearts();

if (heartImage == null) return;

// сердечки врага

int enemyHeartsStartX = 10;

int enemyHeartsY = 10;

for (int i = 0; i < 10; i++)

{

PictureBox heart = new PictureBox();

heart.Image = heartImage;

heart.SizeMode = PictureBoxSizeMode.StretchImage;

heart.Size = new Size(30, 30);

heart.BackColor = Color.Transparent;

heart.Location = new Point(enemyHeartsStartX + i \* (30 + 5), enemyHeartsY);

heart.Visible = (i < enemyLives);

pictureBoxGame.Controls.Add(heart);

heart.BringToFront();

enemyHearts.Add(heart);

}

// Сердечки игрока

int playerHeartsTotalWidth = (3 \* 30) + ((3 - 1) \* 5);

int playerHeartsStartX = pictureBoxGame.Width - playerHeartsTotalWidth - 10;

int playerHeartsY = pictureBoxGame.Height - 30 - 10;

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

{

PictureBox heart = new PictureBox();

heart.Image = heartImage;

heart.SizeMode = PictureBoxSizeMode.StretchImage;

heart.Size = new Size(30, 30);

heart.BackColor = Color.Transparent;

heart.Location = new Point(playerHeartsStartX + i \* (30 + 5), playerHeartsY);

heart.Visible = (i < playerLives);

pictureBoxGame.Controls.Add(heart);

heart.BringToFront();

playerHearts.Add(heart);

}

if (player != null) player.BringToFront();

if (enemy != null) enemy.BringToFront();

}

private void UpdateHearts()

{

if (heartImage == null) return;

for (int i = 0; i < 10; i++)

{

if (i < enemyHearts.Count)

{

enemyHearts[i].Visible = (i < enemyLives);

}

}

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

{

if (i < playerHearts.Count)

{

playerHearts[i].Visible = (i < playerLives);

}

}

}

private void CreateObstacles()

{

obstacles.Clear();

int obstacleCount = 4;

int obstacleWidth = 90;

int obstacleHeight = 120;

int spacing = pictureBoxGame.Width / (obstacleCount + 1);

for (int i = 1; i <= obstacleCount; i++)

{

PictureBox obstacle = new PictureBox();

if (columnImage != null)

{

obstacle.Image = columnImage;

obstacle.SizeMode = PictureBoxSizeMode.StretchImage;

obstacle.BackColor = Color.Transparent;

}

else

{

obstacle.BackColor = Color.FromArgb(150, 100, 100, 100);

}

obstacle.Size = new Size(obstacleWidth, obstacleHeight);

obstacle.Location = new Point(spacing \* i - obstacleWidth / 2, pictureBoxGame.Height / 2 - obstacleHeight / 2);

obstacles.Add(obstacle);

pictureBoxGame.Controls.Add(obstacle);

}

}

private void buttonPlay\_Click(object sender, EventArgs e)

{

if (!isGameActive)

{

if (player == null || enemy == null)

{

InitializeGame();

}

if (player == null || enemy == null)

{

MessageBox.Show("Не удалось создать игровые объекты.");

return;

}

isGameActive = true;

buttonPlay.Text = "Пауза";

timerGameStep.Start();

timerEnemy.Start();

pictureBoxGame.Focus();

}

else

{

isGameActive = false;

buttonPlay.Text = "Играть";

timerGameStep.Stop();

timerEnemy.Stop();

this.Focus();

}

}

private void buttonGiveUp\_Click(object sender, EventArgs e)

{

isGameActive = false;

timerGameStep.Stop();

timerEnemy.Stop();

buttonPlay.Text = "Играть";

InitializeGame();

this.Focus();

}

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

{

if (isGameActive && player != null)

{

if (e.KeyCode == Keys.Left)

{

moveLeft = true;

e.Handled = true;

}

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

{

moveRight = true;

e.Handled = true;

}

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

{

shootPressed = true;

e.Handled = true;

}

}

if (e.KeyCode == Keys.Enter)

{

buttonPlay\_Click(sender, e);

e.Handled = true;

}

}

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

{

if (isGameActive)

{

if (e.KeyCode == Keys.Left)

{

moveLeft = false;

e.Handled = true;

}

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

{

moveRight = false;

e.Handled = true;

}

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

{

shootPressed = false;

e.Handled = true;

}

}

}

protected override bool ProcessCmdKey(ref Message msg, Keys keyData)

{

if (isGameActive)

{

if (keyData == Keys.Left || keyData == Keys.Right || keyData == Keys.Space)

{

if (msg.Msg == 0x100)

{

Form1\_KeyDown(this, new KeyEventArgs(keyData));

}

else if (msg.Msg == 0x101)

{

Form1\_KeyUp(this, new KeyEventArgs(keyData));

}

return true;

}

}

if (keyData == Keys.Enter)

{

buttonPlay\_Click(this, EventArgs.Empty);

return true;

}

return base.ProcessCmdKey(ref msg, keyData);

}

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

{

if (!isGameActive || player == null || enemy == null) return;

if (moveLeft && player.Left > 0)

{

player.Left -= playerSpeed;

}

if (moveRight && player.Right < pictureBoxGame.Width)

{

player.Left += playerSpeed;

}

MoveEnemy();

if (shootPressed && playerFireCooldown <= 0)

{

CreatePlayerBullet();

playerFireCooldown = 15;

}

if (playerFireCooldown > 0) playerFireCooldown--;

if (enemyFireCooldown <= 0)

{

CreateEnemyBullet();

enemyFireCooldown = 20;

}

if (enemyFireCooldown > 0) enemyFireCooldown--;

MovePlayerBullets();

MoveEnemyBullets();

CheckCollisions();

}

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

{

if (!isGameActive || enemy == null) return;

enemyMoveCounter++;

int distanceToLeftEdge = enemy.Left;

int distanceToRightEdge = pictureBoxGame.Width - (enemy.Left + enemy.Width);

if ((enemyDirection == -1 && distanceToLeftEdge < 100) ||

(enemyDirection == 1 && distanceToRightEdge < 100))

{

if (random.Next(0, 100) < 80)

{

enemyDirection \*= -1;

enemyMoveCounter = 0;

}

}

if (enemyMoveCounter >= 50)

{

if (random.Next(0, 100) < 40)

{

enemyDirection = enemyDirection \* -1;

}

if (random.Next(0, 100) < 30)

{

enemySpeed = random.Next(3, 7);

}

enemyMoveCounter = 0;

}

if (random.Next(0, 100) < 25)

{

CreateEnemyBullet();

}

}

private void MoveEnemy()

{

if (enemy == null) return;

int newX = enemy.Left + (enemySpeed \* enemyDirection);

if (newX < 0)

{

newX = 0;

enemyDirection = 1;

}

else if (newX + enemy.Width > pictureBoxGame.Width)

{

newX = pictureBoxGame.Width - enemy.Width;

enemyDirection = -1;

}

enemy.Left = newX;

if (random.Next(0, 1000) < 20)

{

int currentPos = enemy.Left + enemy.Width / 2;

if (currentPos > pictureBoxGame.Width / 3 && currentPos < pictureBoxGame.Width \* 2 / 3)

{

enemyDirection \*= -1;

}

}

}

private void CreatePlayerBullet()

{

PictureBox bullet = new PictureBox();

bullet.Image = playerBulletImage;

bullet.SizeMode = PictureBoxSizeMode.StretchImage;

bullet.Size = new Size(8, 20);

bullet.BackColor = Color.Transparent;

bullet.Location = new Point(player.Left + player.Width / 2 - 4, player.Top - 20);

pictureBoxGame.Controls.Add(bullet);

bullet.BringToFront();

playerBullets.Add(bullet);

}

private void CreateEnemyBullet()

{

if (enemy == null || enemyBulletImage == null) return;

PictureBox bullet = new PictureBox();

bullet.Image = enemyBulletImage;

bullet.SizeMode = PictureBoxSizeMode.StretchImage;

bullet.Size = new Size(8, 20);

bullet.BackColor = Color.Transparent;

bullet.Location = new Point(enemy.Left + enemy.Width / 2 - 4, enemy.Top + enemy.Height);

pictureBoxGame.Controls.Add(bullet);

bullet.BringToFront();

enemyBullets.Add(bullet);

}

private void MovePlayerBullets()

{

for (int i = playerBullets.Count - 1; i >= 0; i--)

{

PictureBox bullet = playerBullets[i];

bullet.Top -= 10;

bool hitObstacle = false;

foreach (PictureBox obstacle in obstacles)

{

if (bullet.Bounds.IntersectsWith(obstacle.Bounds))

{

hitObstacle = true;

break;

}

}

if (bullet.Top + bullet.Height < 0 || hitObstacle)

{

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

playerBullets.RemoveAt(i);

}

}

}

private void MoveEnemyBullets()

{

for (int i = enemyBullets.Count - 1; i >= 0; i--)

{

PictureBox bullet = enemyBullets[i];

bullet.Top += 10;

bool hitObstacle = false;

foreach (PictureBox obstacle in obstacles)

{

if (bullet.Bounds.IntersectsWith(obstacle.Bounds))

{

hitObstacle = true;

break;

}

}

if (bullet.Top > pictureBoxGame.Height || hitObstacle)

{

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

enemyBullets.RemoveAt(i);

}

}

}

private void CheckCollisions()

{

for (int i = playerBullets.Count - 1; i >= 0; i--)

{

PictureBox bullet = playerBullets[i];

if (enemy != null && bullet.Bounds.IntersectsWith(enemy.Bounds))

{

enemyLives--;

UpdateHearts();

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

playerBullets.RemoveAt(i);

if (enemyLives <= 0)

{

GameOver("Вы победили! Враг уничтожен.");

}

}

}

for (int i = enemyBullets.Count - 1; i >= 0; i--)

{

PictureBox bullet = enemyBullets[i];

if (player != null && bullet.Bounds.IntersectsWith(player.Bounds))

{

playerLives--;

UpdateHearts();

pictureBoxGame.Controls.Remove(bullet);

bullet.Dispose();

enemyBullets.RemoveAt(i);

if (playerLives <= 0)

{

GameOver("Вы проиграли! Ваша тарелка уничтожена.");

}

}

}

}

private void GameOver(string message)

{

isGameActive = false;

timerGameStep.Stop();

timerEnemy.Stop();

buttonPlay.Text = "Играть";

ClearBullets();

MessageBox.Show(message, "Конец игры", MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Focus();

}

}

}

**Вывод:** изучили основы техники программирования видеоигр, получили навыки создания видеоигр.