**29** **Анимация**

Задание 1. Разработайте программу Clock моделирующую движение секундной стрелки.

Листинг программы:

namespace Task3

{

public partial class Form1 : Form

{

private List<Point> points = new List<Point>(); // Список точек для отображения пути

private Random random = new Random(); // Генератор случайных чисел для цветов

private int x1, y1, x2, y2;

private double a, t, fi;

private Pen pen = new Pen(Color.DarkRed, 2);

public Form1()

{

InitializeComponent();

this.Load += Form1\_Load; // Подключаем обработчик события Load

this.Paint += Form1\_Paint; // Подключаем обработчик события Paint

timer1.Interval = 10; // Устанавливаем интервал времени (в миллисекундах)

timer1.Tick += timer1\_Tick; // Подключаем обработчик события Tick

timer1.Start(); // Запускаем таймер

this.DoubleBuffered = true;

}

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

{

x1 = ClientSize.Width / 2;

y1 = ClientSize.Height / 2;

a = 150;

fi = -0.5;

t = Math.Sin(fi);

x2 = x1 + (int)((3 \* a \* t) / (1 + t \* t \* t));

y2 = y1 - (int)((3 \* a \* t \* t) / (1 + t \* t \* t));

this.BackColor = Color.White; // Установка цвета фона формы

}

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

{

Graphics g = e.Graphics;

// Рисуем линии, соединяющие точки в списке points

for (int i = 1; i < points.Count; i++)

{

Color lineColor = Color.FromArgb(random.Next(256), random.Next(256), random.Next(256));

Pen linePen = new Pen(lineColor, 2);

g.DrawLine(linePen, points[i - 1], points[i]);

}

// Рисуем текущую позицию в виде круга

g.DrawEllipse(pen, x2, y2, 20, 20);

}

private void timer1\_Tick(object sender, EventArgs e)

{

fi += 0.05; // Увеличение шага изменения угла

t = Math.Tan(fi);

x2 = x1 + (int)((3 \* a \* t) / (1 + t \* t \* t));

y2 = y1 - (int)((3 \* a \* t \* t) / (1 + t \* t \* t));

// Добавляем текущую позицию в список points

points.Add(new Point(x2 + 10, y2 + 10));

// Ограничиваем количество точек, чтобы не рисовать слишком длинный путь

if (points.Count > 100)

{

points.RemoveAt(0);

}

Invalidate();

}

}

}

Анализ результатов:

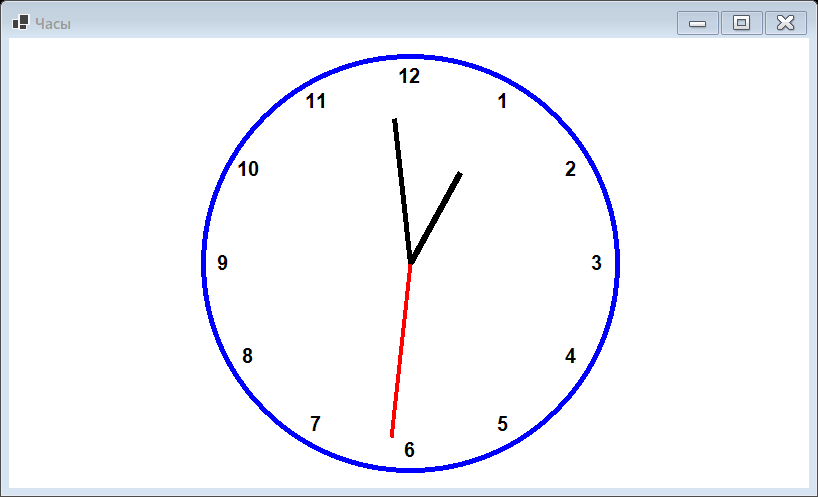


Рисунок 29.1 – Результат работы программы

Задание 2. Движение по траектории.

Листинг программы:

namespace Task3

{

public partial class Form1 : Form

{

private List<Point> points = new List<Point>(); // Список точек для отображения пути

private Random random = new Random(); // Генератор случайных чисел для цветов

private int x1, y1, x2, y2;

private double a, t, fi;

private Pen pen = new Pen(Color.DarkRed, 2);

public Form1()

{

InitializeComponent();

this.Load += Form1\_Load; // Подключаем обработчик события Load

this.Paint += Form1\_Paint; // Подключаем обработчик события Paint

timer1.Interval = 10; // Устанавливаем интервал времени (в миллисекундах)

timer1.Tick += timer1\_Tick; // Подключаем обработчик события Tick

timer1.Start(); // Запускаем таймер

this.DoubleBuffered = true;

}

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

{

x1 = ClientSize.Width / 2;

y1 = ClientSize.Height / 2;

a = 150;

fi = -0.5;

t = Math.Sin(fi);

x2 = x1 + (int)((3 \* a \* t) / (1 + t \* t \* t));

y2 = y1 - (int)((3 \* a \* t \* t) / (1 + t \* t \* t));

this.BackColor = Color.White; // Установка цвета фона формы

}

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

{

Graphics g = e.Graphics;

// Рисуем линии, соединяющие точки в списке points

for (int i = 1; i < points.Count; i++)

{

Color lineColor = Color.FromArgb(random.Next(256), random.Next(256), random.Next(256));

Pen linePen = new Pen(lineColor, 2);

g.DrawLine(linePen, points[i - 1], points[i]);

}

// Рисуем текущую позицию в виде круга

g.DrawEllipse(pen, x2, y2, 20, 20);

}

private void timer1\_Tick(object sender, EventArgs e)

{

fi += 0.05; // Увеличение шага изменения угла

t = Math.Tan(fi);

x2 = x1 + (int)((3 \* a \* t) / (1 + t \* t \* t));

y2 = y1 - (int)((3 \* a \* t \* t) / (1 + t \* t \* t));

// Добавляем текущую позицию в список points

points.Add(new Point(x2 + 10, y2 + 10));

// Ограничиваем количество точек, чтобы не рисовать слишком длинный путь

if (points.Count > 100)

{

points.RemoveAt(0);

}

Invalidate();

}

}

}

Анализ результатов:

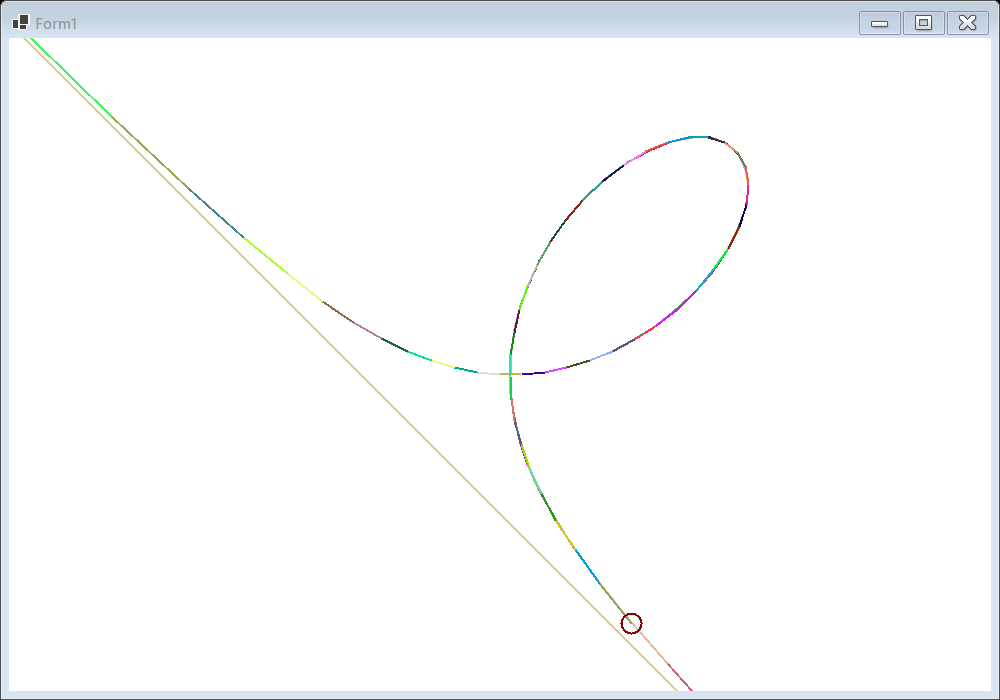


Рисунок 29.2 – Результат работы программы