# **28 Работа с графикой. Трёхмерная графика**

Задание 1. Изучить и выполнить лабораторную работу №10 из учебного пособия. Листинг программы:

<Canvas>

<Polygon Points="100,100 150,150 50,150" Fill="Red" Stroke="Black" StrokeThickness="2"/>

<Ellipse Canvas.Left="200" Canvas.Top="100" Width="100" Height="50" Fill="Blue" Stroke="Black" StrokeThickness="2"/>

<Ellipse Canvas.Left="350" Canvas.Top="100" Width="100" Height="100" Fill="Green" Stroke="Black" StrokeThickness="2"/>

<Rectangle Canvas.Left="100" Canvas.Top="200" Width="100" Height="50" Fill="Yellow" Stroke="Black" StrokeThickness="2"/>

<Path Canvas.Left="250" Canvas.Top="200" Fill="Orange" Stroke="Black" StrokeThickness="2" Data="M 0,0 L 100,0 A 100,100 0 0 1 50,87 L 0,0"/>

<Ellipse Width="150" Height="150" Stroke="Black" StrokeThickness="2" Canvas.Left="536" Canvas.Top="59"/>

<Ellipse Width="70" Height="72" Canvas.Left="576" Canvas.Top="98" Stroke="Black" StrokeThickness="2"/>

<Ellipse Width="30" Height="31" Canvas.Left="596" Canvas.Top="120" Stroke="Black" StrokeThickness="2"/>

<Rectangle Width="45" Height="91" Canvas.Left="813" Stroke="Black" StrokeThickness="2" Canvas.Top="88"/>

<Rectangle Width="44" Height="91" Canvas.Left="836" Stroke="Black" StrokeThickness="2" Canvas.Top="118" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<Rectangle Width="45" Height="91" Canvas.Left="858" Stroke="Black" StrokeThickness="2" Canvas.Top="150" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<Rectangle Width="45" Height="91" Canvas.Left="885" Stroke="Black" StrokeThickness="2" Canvas.Top="180" HorizontalAlignment="Left" VerticalAlignment="Top"/>

<Canvas x:Name="chessboardCanvas" Width="320" Height="320" Canvas.Left="366" Canvas.Top="268"/>

</Canvas>

Анализ результатов:

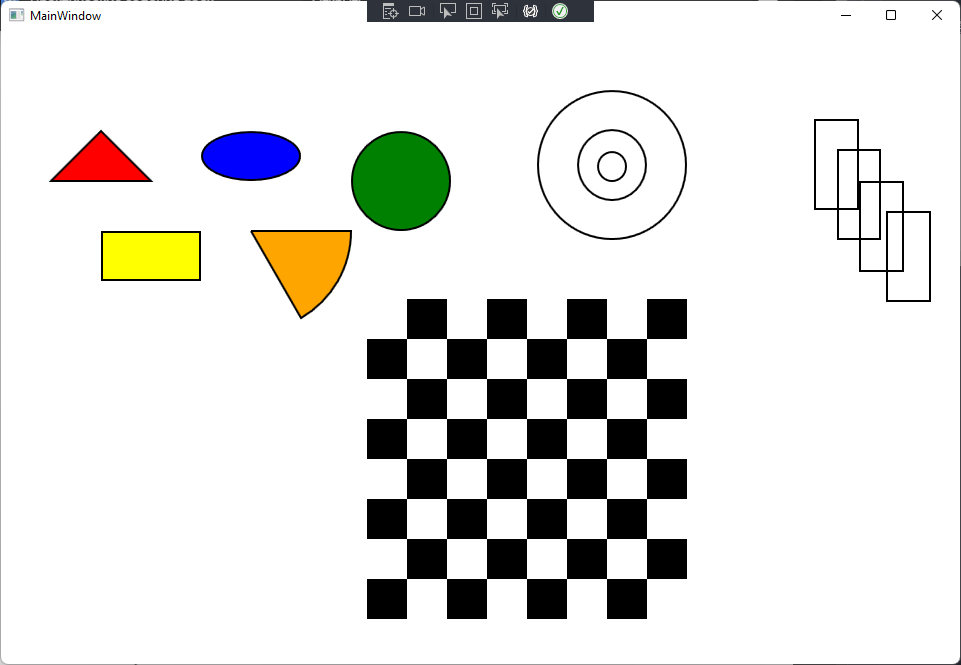


Рисунок 28.1 – Результат работы программы

Задание 2. Выполнить первое задание, используя DrawingContext Листинг программы:

private void MainWindow\_Loaded(object sender, RoutedEventArgs e)

{

DrawingVisual visual = new DrawingVisual();

using (DrawingContext context = visual.RenderOpen())

{

DrawChessboard(context);

Point[] points = { new Point(400, 50), new Point(500, 50), new Point(450, 150) };

Brush brush = Brushes.Red;

DrawTriangle(context, points, brush);

Point ellipseCenter = new Point(100, 100);

double ellipseRadiusX = 50;

double ellipseRadiusY = 30;

Brush ellipseBrush = Brushes.Blue;

DrawEllipse(context, ellipseCenter, ellipseRadiusX, ellipseRadiusY, ellipseBrush);

Point circleCenter = new Point(450, 250);

double circleRadius = 40;

Brush circleBrush = Brushes.Green;

DrawCircle(context, circleCenter, circleRadius, circleBrush);

Rect rectangleRect = new Rect(420, 120, 100, 60);

Brush rectangleBrush = Brushes.Orange;

DrawRectangle(context, rectangleRect, rectangleBrush);

Point sectorCenter = new Point(200, 100);

double sectorRadius = 50;

double startAngle = 45; // Начальный угол в градусах

double sweepAngle = 90; // Угол разворота в градусах

Brush sectorBrush = Brushes.Yellow;

DrawSector(context, sectorCenter, sectorRadius, startAngle, sweepAngle, sectorBrush);

Point center = new Point(400, 100);

double radius1 = 80;

double radius2 = 60;

double radius3 = 40;

brush = Brushes.Transparent;

Pen pen = new Pen(Brushes.Black, 2);

DrawCircle(context, center, radius1, brush, pen);

DrawCircle(context, center, radius2, brush, pen);

DrawCircle(context, center, radius3, brush, pen);

DrawRectangle(context, new Rect(550, 100, 25, 50), brush, pen);

DrawRectangle(context, new Rect(565, 125, 25, 50), brush, pen);

DrawRectangle(context, new Rect(580, 150, 25, 50), brush, pen);

DrawRectangle(context, new Rect(595, 175, 25, 50), brush, pen);

}

RenderTargetBitmap bitmap = new RenderTargetBitmap(1080, 1080, 96, 96, PixelFormats.Pbgra32);

bitmap.Render(visual);

Image image = new Image

{

Source = bitmap,

Width = 1080,

Height = 1080

};

MainCanvas.Children.Add(image);

}

private void DrawChessboard(DrawingContext context)

{

for (int row = 0; row < 8; row++)

{

for (int col = 0; col < 8; col++)

{

Rect rect = new Rect(col \* 40, row \* 40, 40, 40);

Brush brush = (row + col) % 2 == 0 ? Brushes.White : Brushes.Black;

context.DrawRectangle(brush, null, rect);

}

}

}

private void DrawCircle(DrawingContext context, Point center, double radius, Brush brush, Pen strokePen = null)

{

context.DrawEllipse(brush, strokePen, center, radius, radius);

}

private void DrawRectangle(DrawingContext context, Rect rect, Brush brush, Pen strokePen = null)

{

context.DrawRectangle(brush, strokePen, rect);

}

private void DrawEllipse(DrawingContext context, Point center, double radiusX, double radiusY, Brush brush)

{

context.DrawEllipse(brush, null, center, radiusX, radiusY);

}

private void DrawTriangle(DrawingContext context, Point[] points, Brush brush)

{

StreamGeometry geometry = new StreamGeometry();

using (StreamGeometryContext geometryContext = geometry.Open())

{

geometryContext.BeginFigure(points[0], true, true);

geometryContext.LineTo(points[1], true, false);

geometryContext.LineTo(points[2], true, false);

}

context.DrawGeometry(brush, null, geometry);

}

private void DrawSector(DrawingContext context, Point center, double radius, double startAngle, double sweepAngle, Brush brush)

{

StreamGeometry geometry = new StreamGeometry();

using (StreamGeometryContext geometryContext = geometry.Open())

{

geometryContext.BeginFigure(center, true, true);

double startAngleRadians = ConvertToRadians(startAngle);

double endAngleRadians = ConvertToRadians(startAngle + sweepAngle);

Point startPoint = new Point(center.X + radius \* Math.Cos(startAngleRadians), center.Y + radius \* Math.Sin(startAngleRadians));

Point endPoint = new Point(center.X + radius \* Math.Cos(endAngleRadians), center.Y + radius \* Math.Sin(endAngleRadians));

bool isLargeArc = Math.Abs(sweepAngle) > 180;

SweepDirection sweepDirection = sweepAngle > 0 ? SweepDirection.Clockwise : SweepDirection.Counterclockwise;

ArcSegment arcSegment = new ArcSegment(endPoint, new Size(radius, radius), 0, isLargeArc, sweepDirection, true);

geometryContext.LineTo(startPoint, true, true);

geometryContext.ArcTo(arcSegment.Point, arcSegment.Size, arcSegment.RotationAngle, arcSegment.IsLargeArc, arcSegment.SweepDirection, true, false);

}

context.DrawGeometry(brush, null, geometry);

}

private double ConvertToRadians(double degrees)

{

return degrees \* Math.PI / 180;

}

Анализ результатов:

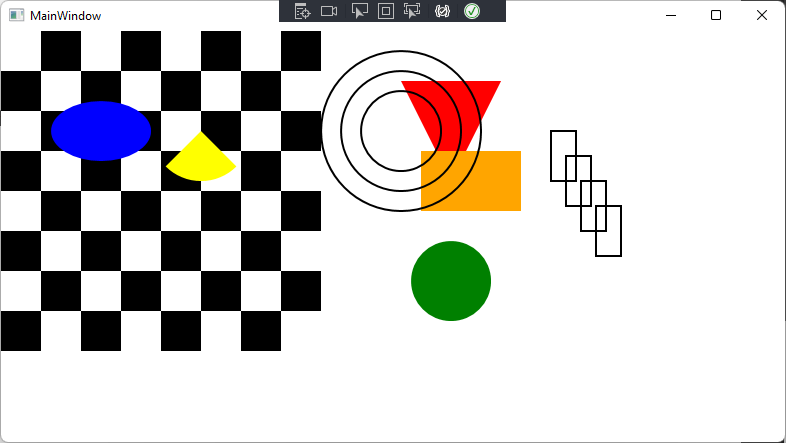


Рисунок 28.2 – Результат работы программы