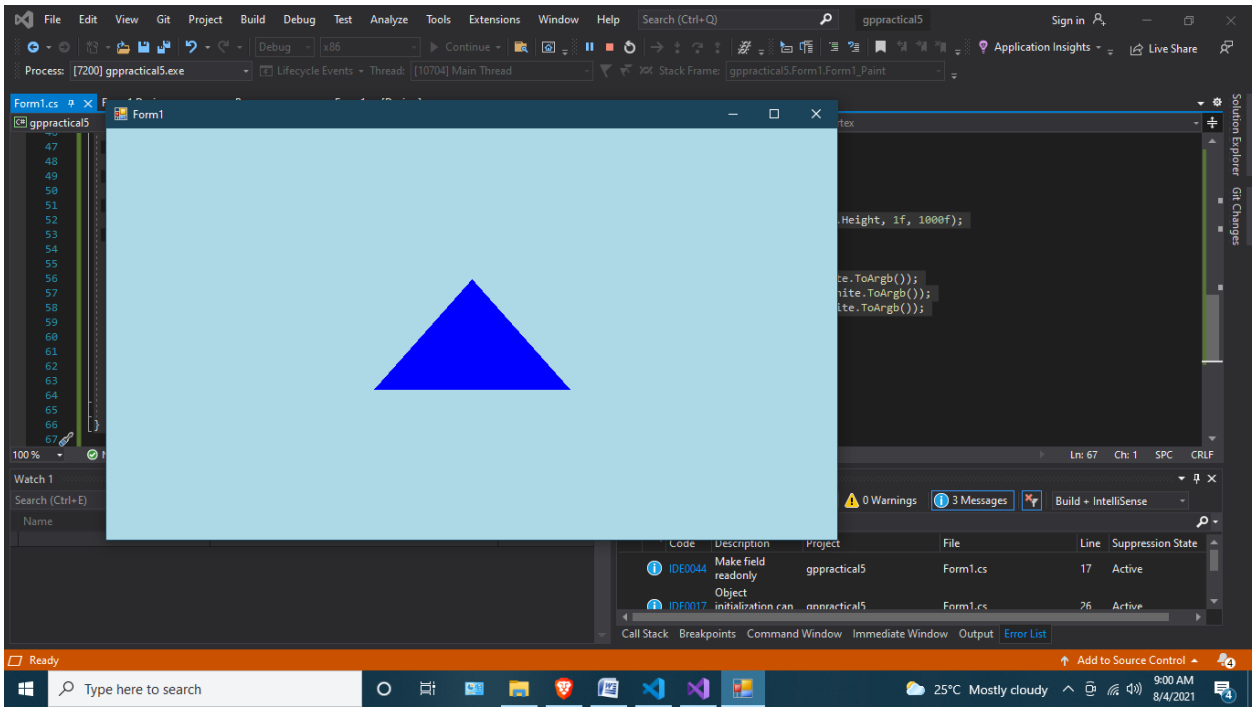
Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Microsoft.DirectX.Direct3D;
using Microsoft.DirectX;


namespace gppractical5
{
    public partial class Form1 : Form
    {
        Microsoft.DirectX.Direct3D.Device device;
        CustomVertex.PositionNormalColored[] vertex = new
CustomVertex.PositionNormalColored[3];
        public Form1()
        {
            InitializeComponent();
            InitDevice();
        }

        public void InitDevice()
        {
            PresentParameters pp = new PresentParameters();
            pp.Windowed = true;
            pp.SwapEffect = SwapEffect.Discard;
            device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);

        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            device.Clear(ClearFlags.Target, Color.LightBlue, 1, 0); device.Present();
            device.BeginScene();
            device.VertexFormat = CustomVertex.PositionNormalColored.Format;
            device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3,
vertex);
            device.EndScene();
            device.Present();
        }
        private void Form1_Load(object sender, System.EventArgs e)
        {
            PresentParameters pp = new PresentParameters();

            pp.Windowed = true;

            pp.SwapEffect = SwapEffect.Discard;

            device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);
```

```
            device.Transform.Projection = Matrix.PerspectiveFovLH(3.14f / 4,
device.Viewport.Width / device.Viewport.Height, 1f, 1000f);

            device.Transform.View = Matrix.LookAtLH(new Vector3(0, 0, 10), new Vector3(),
new Vector3(0, 1, 0));
            device.RenderState.Lighting = false;
            vertex[0] = new CustomVertex.PositionNormalColored(new Vector3(0, 1, 1), new
Vector3(1, 0, 1), Color.White.ToArgb());
            vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1, 1),
new Vector3(2, 0, 1), Color.White.ToArgb());
            vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1, 1), new
Vector3(2, 0, 1), Color.White.ToArgb());
            device.RenderState.Lighting = true;
            device.Lights[0].Type = LightType.Directional;
            device.Lights[0].Diffuse = Color.Blue;
            device.Lights[0].Direction = new Vector3(-1f, 0, -2);
            device.Lights[0].Enabled = true;
        }
    }
}
```

OUTPUT



Form2.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Microsoft.DirectX;
using Microsoft.DirectX.Direct3D;

namespace gppractical5
{
    public partial class Form1 : Form
    {
        Microsoft.DirectX.Direct3D.Device device;
        private CustomVertex.PositionNormalColored[] vertex = new
CustomVertex.PositionNormalColored[6];
        public Form1()
        {
            InitializeComponent();
        }


        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            device.Clear(ClearFlags.Target, Color.Violet, 1.0f, 0);
            device.BeginScene();
            device.VertexFormat = CustomVertex.PositionNormalColored.Format;
            device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3,
vertex);
            device.EndScene();
            device.Present();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            PresentParameters pp = new PresentParameters();
            pp.Windowed = true;
            pp.SwapEffect = SwapEffect.Discard;
            device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);
            device.Transform.Projection = Matrix.PerspectiveFovLH(3.14f / 4,
device.Viewport.Width / device.Viewport.Height, 1f, 1000f);
            device.Transform.View = Matrix.LookAtLH(new Vector3(0, 0, 10), new Vector3(),
new Vector3(0, 1, 0));

            device.RenderState.Lighting = true;
            vertex[0] = new CustomVertex.PositionNormalColored(new Vector3(0, 0, 4), new
Vector3(-1, 0, -1), Color.Pink.ToArgb());
            vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(2, 0, 4), new
Vector3(1, 0, 1), Color.Pink.ToArgb());
            vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(2, 2, 4), new
Vector3(-1, 0, -1), Color.Pink.ToArgb());
```

```
            vertex[3] = new CustomVertex.PositionNormalColored(new Vector3(0, 2, 4), new
Vector3(1, 0, 1), Color.Pink.ToArgb());
            vertex[4] = new CustomVertex.PositionNormalColored(new Vector3(0, 0, 4), new
Vector3(-1, 0, -1), Color.Pink.ToArgb());
            vertex[5] = new CustomVertex.PositionNormalColored(new Vector3(2, 2, 4), new
Vector3(-1, 0, -1), Color.Pink.ToArgb());

            device.Lights[0].Type = LightType.Directional;
            device.Lights[0].Diffuse = Color.Red;
            device.Lights[0].Direction = new Vector3(-1f, 1, -1);
            device.Lights[0].Enabled = true;
        }    }
}
```