ROLL NO : 449 NAME: ROHIT PRAJAPATI CLASS: TYCS

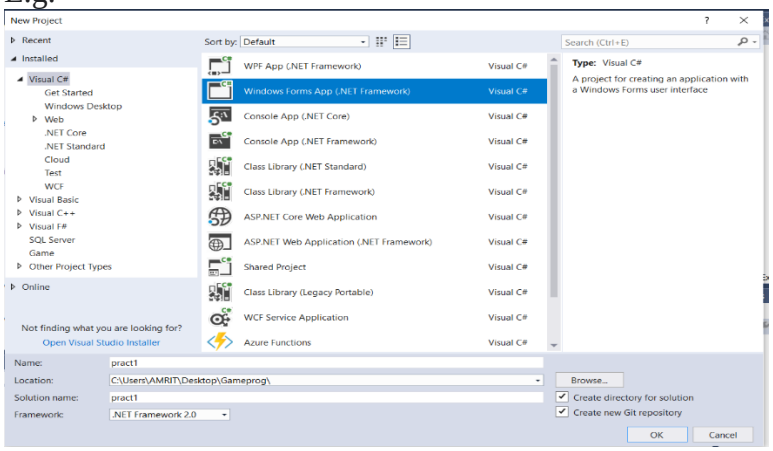
SRNO	NAME	DATE	PG NO	SIGN
1	Initializing Windows Form Using Direct X.	12/09/22	02	
2	Draw Transformed colored Triangle using DirectX.	12/09/22	07	
3	Creating a triangle and texturizing.	03/10/22	12	
4	Lightening (Programmable Diffuse Lightening using direct 3D11).	03/10/22	18	
5	Lightening (Programmable Specular Lightening using direct 3D11).	20/10/22	22	
6	Loading models into Direct X 11 and Rendering.	24/10/22	26	
7	Creating a 2-Dimensional UFO game using unity.	07/11/22	30	
8	Create a 3D Roll a Ball using Unity	07/11/22	61	

ROLL NO : 449 NAME: ROHIT PRAJAPATI CLASS: TYCS

PRACTICAL NO:01

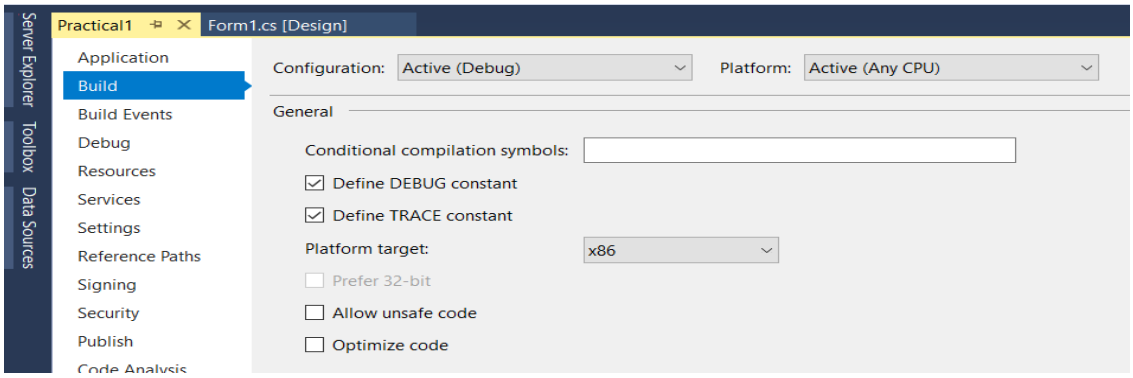
AIM: Initialising Windows Form Using Direct X.

STEPS:
1: Open Visual Studio→Create New Project→Select Windows Forms App (.Net Framework)→Change Name & Location According to You→Framework Select : .Net Framework 2.0→OK.
E.g.

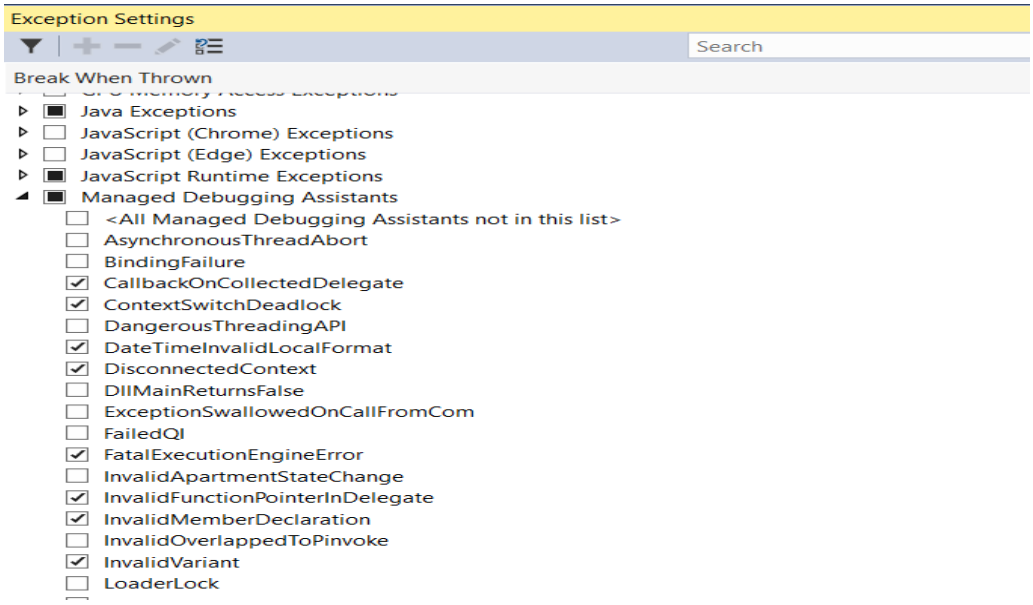


2: Right Click on Name of Your Project In Solution Explorer→Click Add→References→Browse→(C:→Windows→Microsoft.Net→DirectX for Managed Code→1.0.2902.0→ctrl+A→Add→ok).

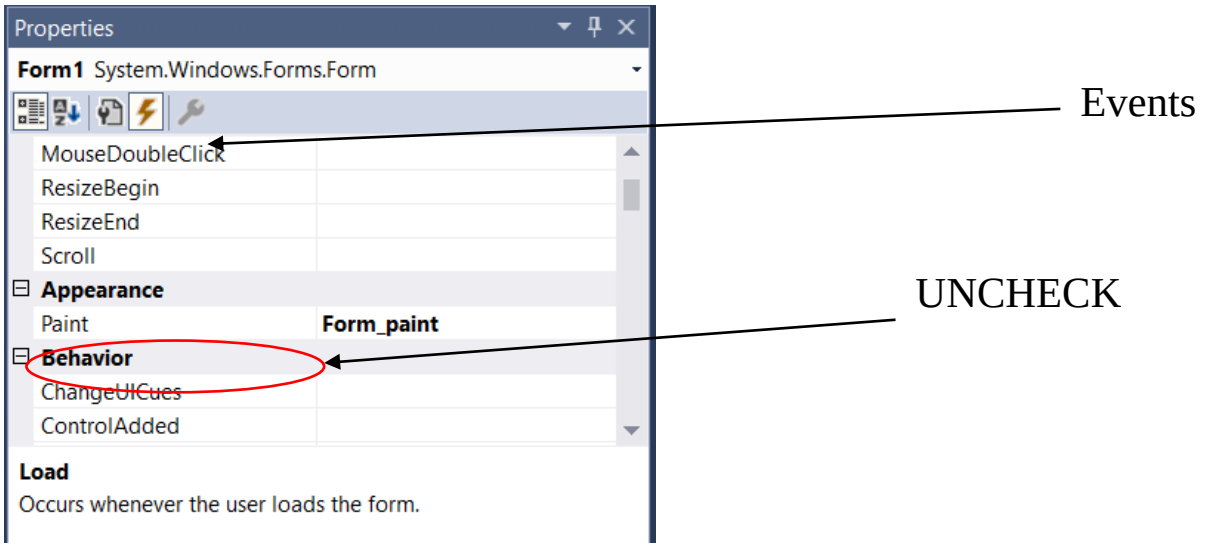
3: Right Click on Name of Your Project In Solution Explorer→Properties→Build→Platform Target(change this from Any CPU to x86)→ctrl+s.



4: Go to Debug→windows→Exception Setting→Managed debugging Assistant→uncheck Loderlock.



5: Go To Properties→Click on Events→Appearance→double Click On Paint, A paint Method Is created Automatically.



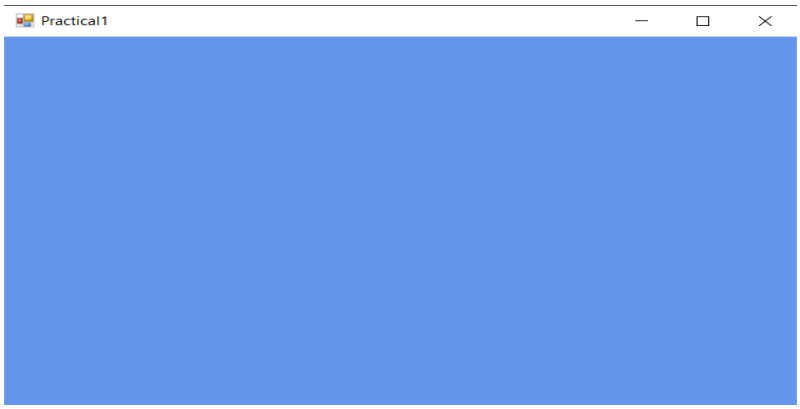
A Simple Program To Initialize Window with DirectX.

Code:

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

```
namespace Practical1
{
   public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
      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 Render()
      {
         device.Clear(ClearFlags.Target,Color.CornflowerBlue, 0, 1);
         device.Present();
      }


      private void Form1_Load(object sender, EventArgs e)
      {

      }

      private void Form_paint(object sender, PaintEventArgs e)
      {
         Render();
      }
   }
}
```

OUTPUT:-



ROLL NO : 449 NAME: ROHIT PRAJAPATI CLASS: TYCS

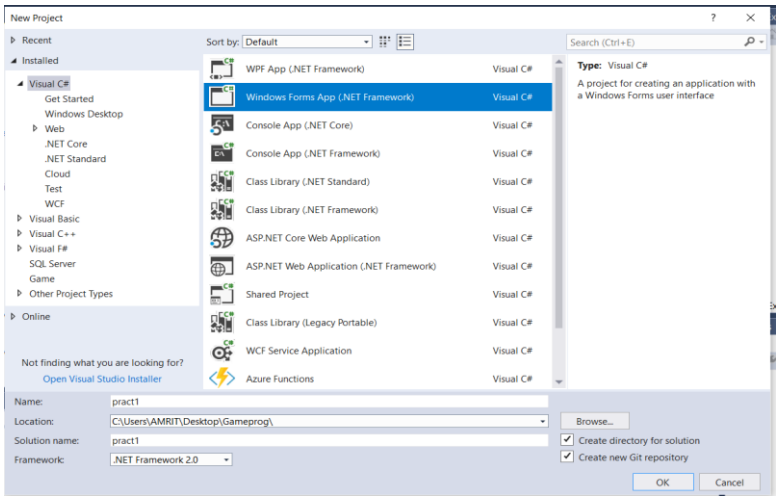
PRACTICAL NO:02

AIM: Draw Transformed coloured Triangle using DirectX.

STEPS:

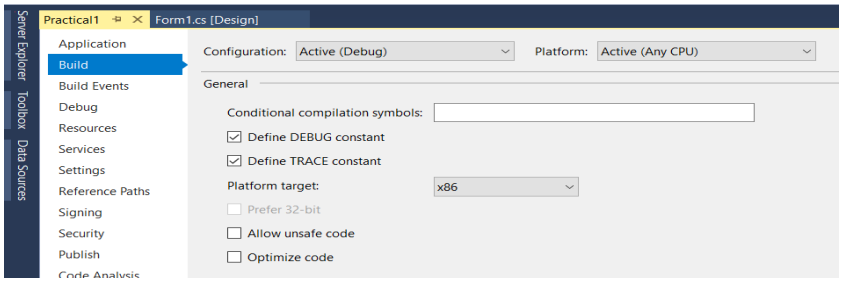
1: Open Visual Studio→Create New Project→Select Windows Forms App (.Net Framework)→Change Name & Location According to You→Framework Select : .Net Framework 2.0→OK.

E.g.

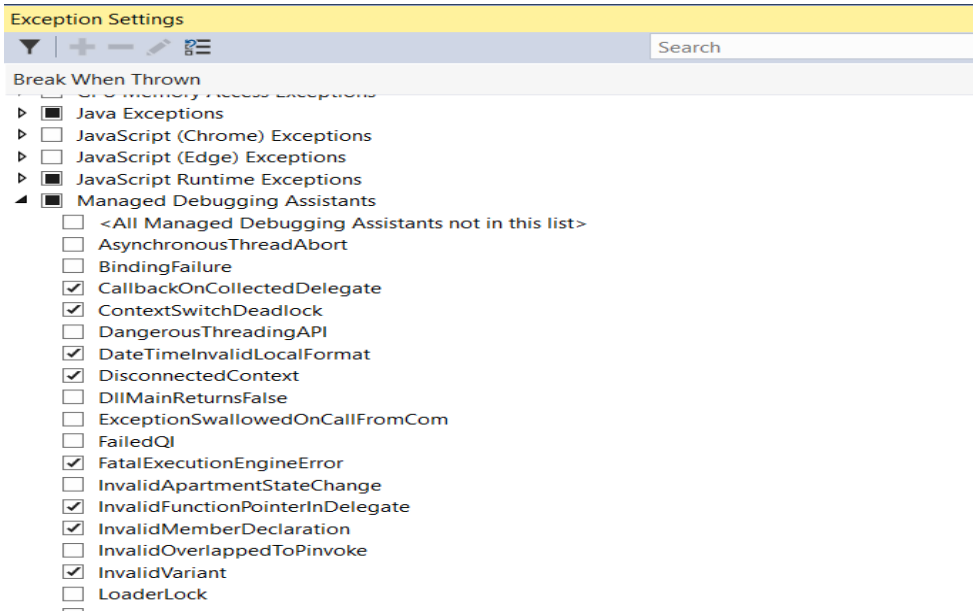


2: Right Click on Name of Your Project In Solution Explorer→Click Add→References→Browse→(C:→Windows→Microsoft.Net→DirectX for Managed Code→1.0.2902.0→ctrl+A→Add→ok).

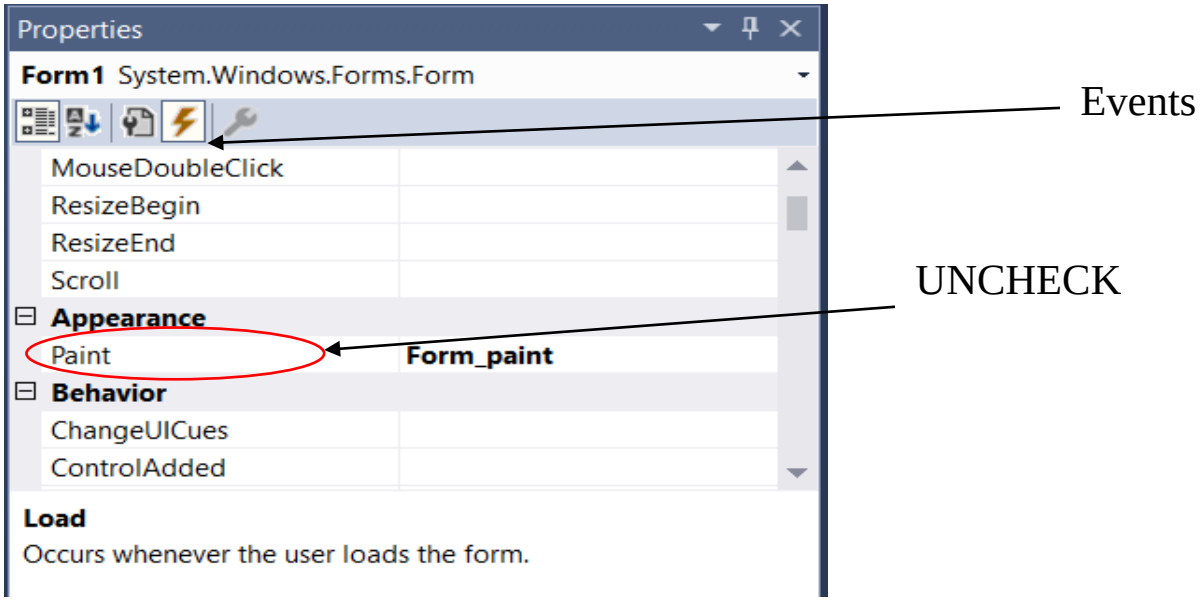
3: Right Click on Name of Your Project In Solution Explorer→Properties→Build→Platform Target(change this from Any CPU to x86)→ctrl+s.



4: Go to Debug→windows→Exception Setting→Managed debugging Assistant→uncheck Loderlock.



5: Go To Properties→Click on Events→Appearance→double Click On Paint, A paint Method Is created Automatically.



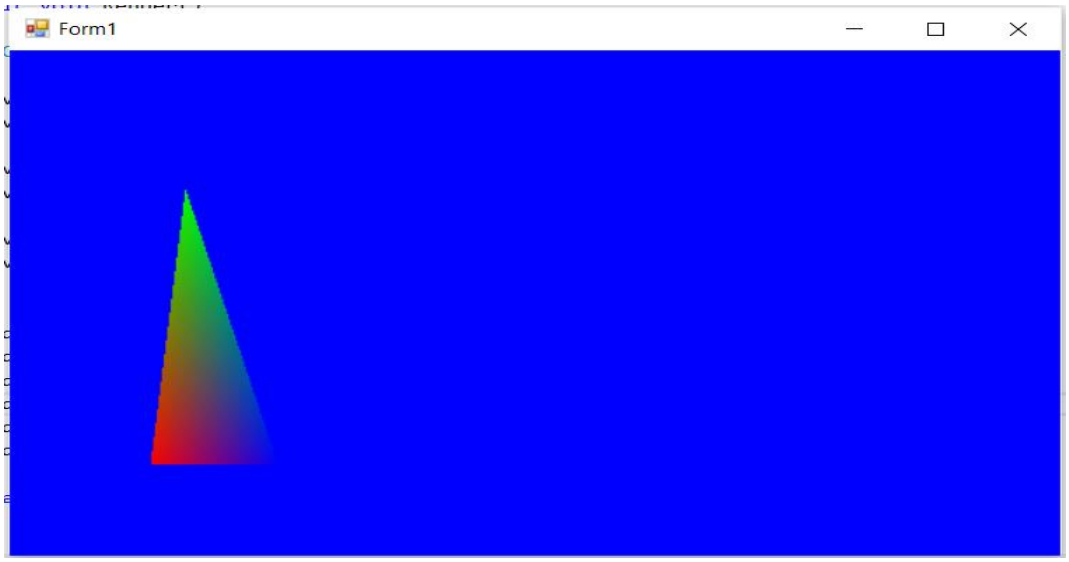
CODE:

```
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 triangle pract2
{
```

```
   public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
      public Form1()
      {
         InitializeComponent();
         InitDevice();
      }

      private void InitDevice()
      {
         PresentParameters pp = new PresentParameters();
         pp.Windowed = true;
         pp.SwapEffect = SwapEffect.Discard;
         device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);
      }
      public void Render()
      {
         CustomVertex.TransformedColored[] vertexes = new
CustomVertex.TransformedColored[3];

         vertexes[0].Position = new Vector4(100, 100, 0, 1.0f);//first point
         vertexes[0].Color = System.Drawing.Color.FromArgb(0, 255, 0).ToArgb();

         vertexes[1].Position = new Vector4(153, 300, 0, 1.0f);//second point
         vertexes[1].Color = System.Drawing.Color.FromArgb(0, 0, 255).ToArgb();

         vertexes[2].Position = new Vector4(80, 300, 0, 1.0f);//third point
         vertexes[2].Color = System.Drawing.Color.FromArgb(255, 0, 0).ToArgb();


         device.Clear(ClearFlags.Target, Color.Blue, 0, 1);
         device.BeginScene();
         device.VertexFormat = CustomVertex.TransformedColored.Format;
         device.DrawUserPrimitives(PrimitiveType.TriangleList, 1, vertexes);
         device.EndScene();
         device.Present();
      }
      private void Form1_Load(object sender, EventArgs e)
      {

      }

      private void Form1_Paint(object sender, PaintEventArgs e)
      {
         Render();
      }
   }
}
```

ROLL NO : 449 NAME: ROHIT PRAJAPATI CLASS: TYCS

OUTPUT:



ROLL NO : 449 NAME: ROHIT PRAJAPATI CLASS: TYCS

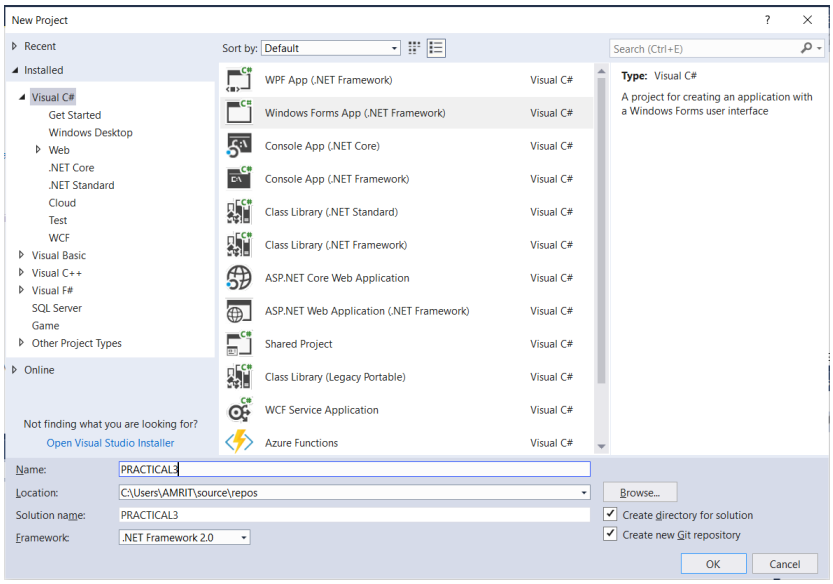
PRACTICAL NO:03

AIM: Texturing (Texture the Triangle using Direct 3D 11)

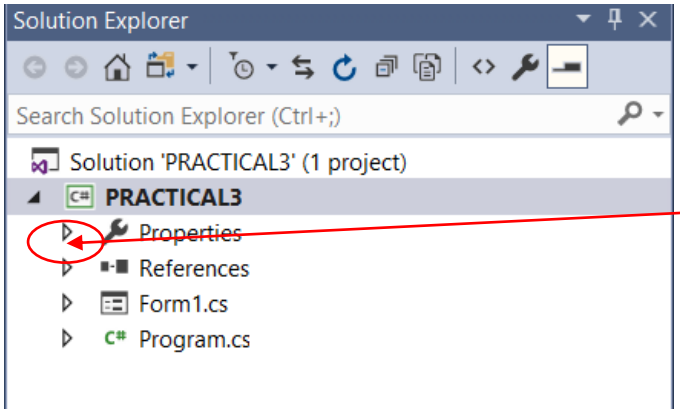
STEPS:

1: Open Visual Studio→Create New Project→Select Windows Forms App (.Net Framework)→Change Name & Location According to You→Framework Select : .Net Framework 2.0→OK.

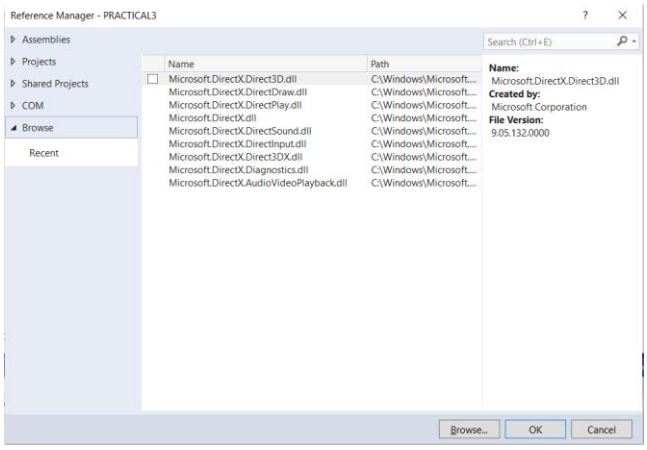
E.g.



2: Right Click on Name of Your Project In Solution Explorer→Click Add→References→Browse→(C:→Windows→Microsoft.Net→DirectX for Managed Code→1.0.2902.0→ctrl+A→Add→ok).

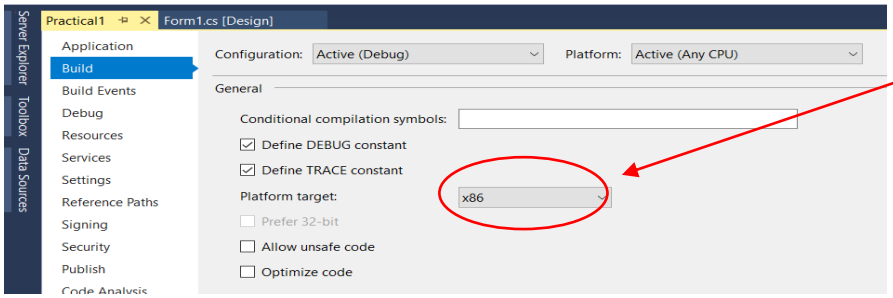


RIGHT CLICK

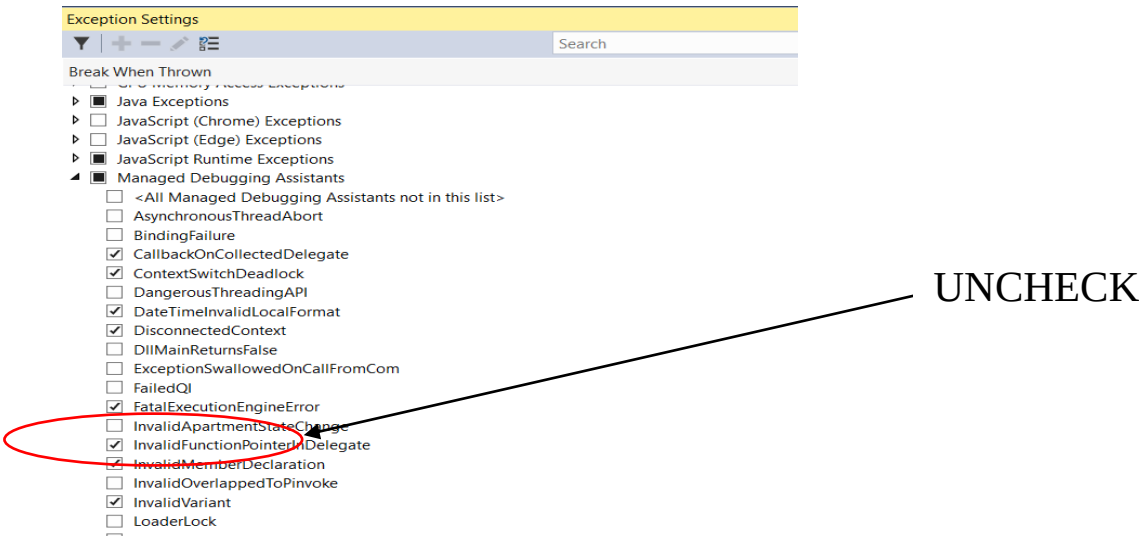


3: Right Click on Name of Your Project In Solution Explorer→Properties→Target(change this from Any CPU to x86)→ctrl+s.

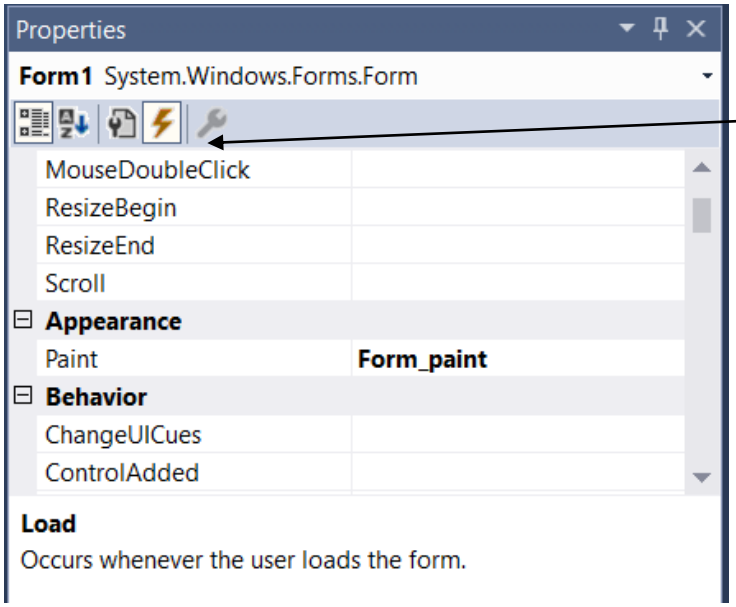
PUT PATH SPECIFIED & then click OK



4: Go to Debug→windows→Exception Setting→Managed debugging Assistant→uncheck Loderlock.



5: Go To Properties→Click on Events→Appearance→double Click On Paint, A paint Method Is created Automatically.



6: Go To Browser→Type Texture Images→Select Any Texture Image Of Your Choice→Save the image in following way→(save→Select Your Project Folder (Practical3)→Practical3→Now create A New Folder name it As Image→inside this Folder Save Your Texture Image)

7.Now in Visual Studio Go To File→Open→Folder→now From this go to the Folder (Image) where You have Saved Your Texture Image→copy The Path and Paste it in texture = new Texture (device, newBitmap(@"C:\Users\AMRIT\Desktop\Gameprog\practical3\practical3\images\texture.jpg "), 0, Pool.Managed);

8.Add @Symbol at Beginning of path or else Add ''\\'' in path

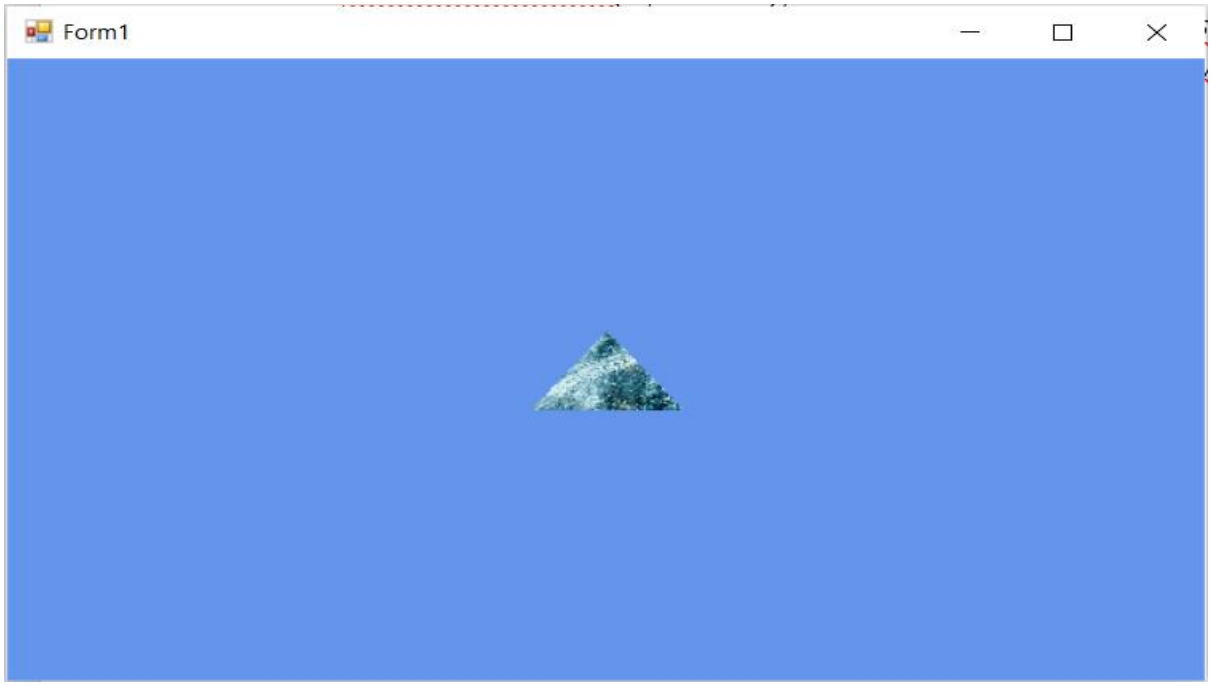
---

CODE:

```
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 practical3
{
   public partial class Form1 : Form
```

```
    {
        Microsoft.DirectX.Direct3D.Device device;
        private CustomVertex.PositionTextured[] vertex = new
CustomVertex.PositionTextured[3];
        private Texture texture;
        public Form1()
        {
            InitializeComponent();
            InitDevice();
        }
        private void InitDevice()
        {
            PresentParameters pp = new PresentParameters();
            pp.Windowed = true;
            pp.SwapEffect = SwapEffect.Discard;
            device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);

        }
        public void Render()
        {
            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, 20), new Vector3(),
new Vector3(0, 1, 0));
            device.RenderState.Lighting = false;
            vertex[0] = new CustomVertex.PositionTextured(new Vector3(0, 1, 1), 0, 0);
            vertex[1] = new CustomVertex.PositionTextured(new Vector3(-1, -1, 1), -1, 0);
            vertex[2] = new CustomVertex.PositionTextured(new Vector3(1, -1, 1), 0, -1);
            texture = new Texture(device, new
Bitmap(@"C:\Users\AMRIT\Desktop\Gameprog\practical3\practical3\images\texture.jpg"),
0, Pool.Managed);
            device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1, 0);
            device.BeginScene();
            device.SetTexture(0, texture);
            device.VertexFormat = CustomVertex.PositionTextured.Format;
            device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3, vertex);
            device.EndScene();
            device.Present();
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {

            Render();
        }
    }
}
```

OUTPUT:


Form1

PRACTICAL NO:04

AIM: Programmable Diffuse Lightening using Direct3D

STEPS:
Open Visual studio: File → New → Project → Visual C# →Select Windows Form application
Framework: .Net Framework 2.0

Add References Right Click on References Add references →Browse → Click on browse button Go to C → Windows → Windows.Net → DirectX for Managed Code → 1.0.2902.0 →Select the following .dll files

- Direct 3D.dll(3rd file)
- Direct 3DX.dll(4th file)
- Direct X.dll(Last File)

CODE:

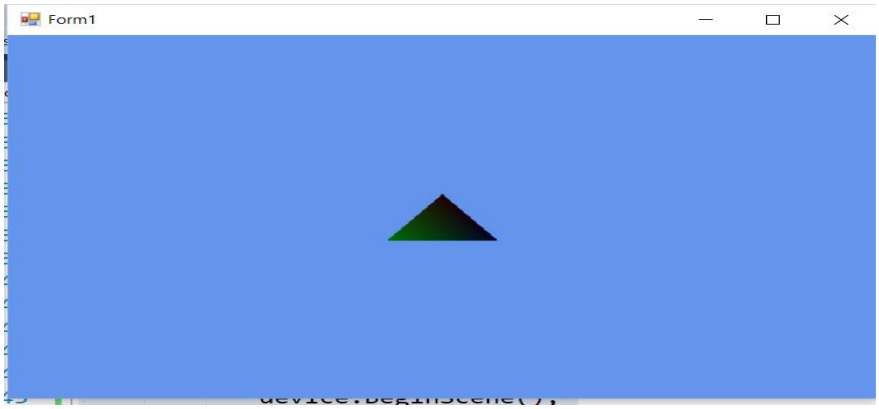
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Microsoft.DirectX;
using Microsoft.DirectX.Direct3D;

namespace Practicalno4
{

   public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
      private CustomVertex.PositionNormalColored[] vertex = new
CustomVertex.PositionNormalColored[3];
      public Form1()
      {
         InitializeComponent();
         InitDevice();
      }
      private void InitDevice()
      {
         PresentParameters pp = new PresentParameters();
         pp.Windowed = true;
         pp.SwapEffect = SwapEffect.Discard;
         device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);
```

```
        }
        public void Render()
        {
            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, 20), 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.Red.ToArgb());
            vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1, 1), new
Vector3(1, 0, 1), Color.Blue.ToArgb());
            vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1, 1), new
Vector3(-1, 0, 1), Color.Green.ToArgb());
            device.RenderState.Lighting = true;
            device.Lights[0].Type = LightType.Directional;
            device.Lights[0].Diffuse = Color.Plum;
            device.Lights[0].Direction = new Vector3(0.8f, 0, -1);
            device.Lights[0].Enabled = true;
            device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1, 0);
            device.BeginScene();
            device.VertexFormat = CustomVertex.PositionNormalColored.Format;
            device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3, vertex);
            device.EndScene();
            device.Present();
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            Render();
        }
    }
}
```

OUTPUT:

STEPS:

To run the Code: First] Right click on the project →Properties → Build →Set Platform target to x86 Second] Go to Debug → Exception Settings → Expand →Managed Debugging Assistants → UNCHECK LOADER LOCK Then Run


Form1

PRACTICAL NO:05

AIM: Lighting (Programmable Specular Lighting using Direct 3D11)

STEPS:
Open Visual studio: File → New → Project → Visual C# →Select Windows Form application
Framework: .Net Framework 2.0

Add References Right Click on References Add references →Browse → Click on browse button Go to C → Windows → Windows.Net → DirectX for Managed Code → 1.0.2902.0 →Select the following .dll files

- Direct 3D.dll(3rd file)
- Direct 3DX.dll(4th file)
- Direct X.dll(Last File)

CODE:

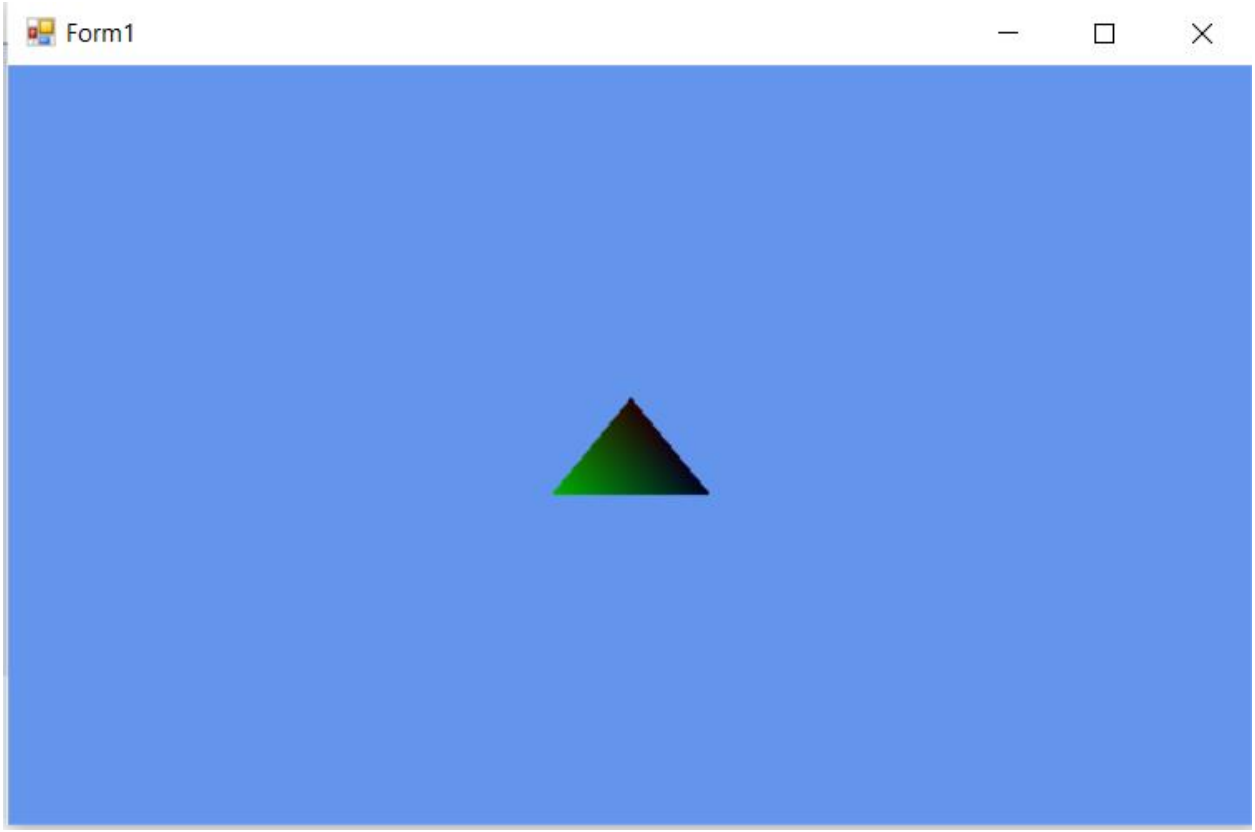
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Microsoft.DirectX;
using Microsoft.DirectX.Direct3D;

namespace practical5
{

   public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
      private CustomVertex.PositionNormalColored[] vertex = new
CustomVertex.PositionNormalColored[3];
      public Form1()
      {
         InitializeComponent();
         InitDevice();
      }
      private void InitDevice()
      {
         PresentParameters pp = new PresentParameters();
         pp.Windowed = true;
         pp.SwapEffect = SwapEffect.Discard;
         device = new Device(0, DeviceType.Hardware, this,
CreateFlags.HardwareVertexProcessing, pp);

      }
      public void Render()
```

```
        {
            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, 20), 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.Red.ToArgb());
            vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1, 1), new
Vector3(1, 0, 1), Color.Blue.ToArgb());
            vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1, 1), new
Vector3(-1, 0, 1), Color.Green.ToArgb());
            device.RenderState.Lighting = true;
            device.Lights[0].Type = LightType.Directional;
            device.Lights[0].Specular = Color.DarkGoldenrod;
            device.Lights[0].Direction = new Vector3(0.8f, 0, -1);
            device.Lights[0].Enabled = true;
            device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1, 0);
            device.BeginScene();
            device.VertexFormat = CustomVertex.PositionNormalColored.Format;
            device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3, vertex);
            device.EndScene();
            device.Present();
        }
        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            Render();
        }
    }
}
```

OUTPUT:

STEPS:

To run the Code: First] Right click on the project →Properties → Build →Set Platform target to x86 Second] Go to Debug → Exception Settings → Expand →Managed Debugging Assistants → UNCHECK LOADER LOCK Then Run


Form1

PRACTICAL NO:06

AIM: Loading models into Direct X 11 and Rendering

STEPS:
Open Visual studio: File → New → Project → Visual C# →Select Windows Form application
Framework: .Net Framework 2.0

Add References Right Click on References Add references →Browse → Click on browse button Go to C → Windows → Windows.Net → DirectX for Managed Code → 1.0.2902.0 →Select the following .dll files

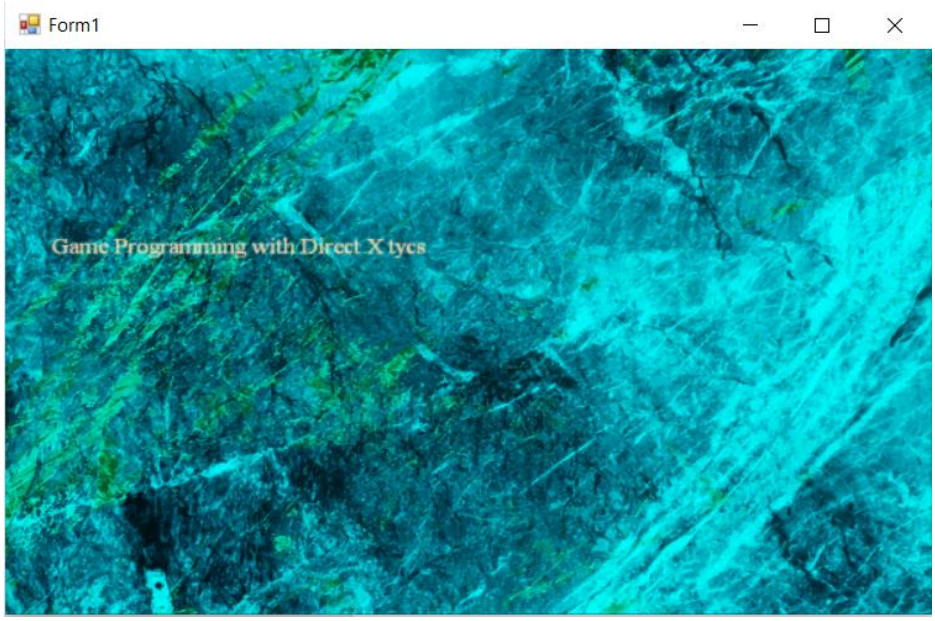
- Direct 3D.dll(3rd file)
- Direct 3DX.dll(4th file)
- Direct X.dll(Last File)

CODE:

```
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 practical6
{
   public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
      Microsoft.DirectX.Direct3D.Texture texture;
      Microsoft.DirectX.Direct3D.Font font;

      public Form1()
      {
         InitializeComponent();
         InitDevice();
         InitFont();
         LoadTexture();
      }

      private void LoadTexture()
      {
        texture = new Texture(device, new
Bitmap(@"C:\Users\AMRIT\Desktop\Gameprog\practical6\practical6\images\texture.jpg"),
0, Pool.Managed);
      }
```

```
        private void InitFont()
        {
            System.Drawing.Font f = new System.Drawing.Font("Times New Roman", 16f,
FontStyle.Regular);
            font = new Microsoft.DirectX.Direct3D.Font(device, f);
        }

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

        }
        public void Render()
        {

            device.Clear(ClearFlags.Target, Color.CornflowerBlue, 0,1);
            device.BeginScene();
            using (Sprite s = new Sprite(device))
            {
                s.Begin(SpriteFlags.AlphaBlend);
                s.Draw2D(texture, new Rectangle(0, 0, 0, 0), new Rectangle(0, 0,
device.Viewport.Width, device.Viewport.Height), new Point(0, 0), 0f, new Point(0, 0),
Color.Aqua);
                font.DrawText(s, "Game Programming with Direct X tycs", new Point(40, 175),
Color.Wheat);
                s.End();
            }
            device.EndScene();
            device.Present();
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {

            Render();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

OUTPUT:


Form1
Game Programming with Direct X tycs