Prerequisites:

Install visual studio 2019 or 2017 community

Unity 3d

Steps:

Download DirectX SDK from the link given in the description.

https://www.microsoft.com/en-in/download/details.aspx?id=6812

Open control panel:

If you have Windows 64 bit then uninstall these 2 files first.(Since mine is 64bit..)

► Microsoft Visual C++ 2010 x64 Redistributable

► Microsoft Visual C++ 2010 x86 Redistributable

If you have Windows 32 bit then uninstall this file only.

► Microsoft Visual C++ 2010 x86 Redistributable

If you don't Follow the above steps you will get an "ERROR NO:S103"

Install DirectX SDK wait till it gets installed.

How to check whether it is installed or not ?

Go to Local Disk C:

Windows folder.

Microsoft.Net

DirectX Mangaged code....

If the above folder is present then you have successfully installed

DirectX SDK :)

Practical No. 1:

Aim: Setup DirectX 11, Window Framework and Initialize Direct3D Device

In this practical we are just learning the window framework and initializing a Direct3D device.

Step 1:

i) Create new project, and select "Windows Forms Application", select .NET Framework as 2.0 in Visuals C#.
ii) Right Click on properties Click on open click on build Select Platform Target and Select x86.

Step 2: Click on View Code of Form 1.

Step 3:

Go to Solution Explorer, right click on project name, and select Add Reference. Click on Browse and select the given .dll files which are "Microsoft.DirectX", "Microsoft.DirectX.Direct3D", and "Microsoft.DirectX.DirectX3DX".

Right click on "References" in Solution explorer. ►Add references ►Browse to the "DirectX Managed Code" directory. ►Select these three files. ► Microsoft.DirectX.Direct3DX ► Microsoft.DirectX.Direct3D.dll ► Microsoft.DirectX.dll and add...

Step 4:

Go to Properties Section of Form, select Paint in the Event List and enter as Form1_Paint.

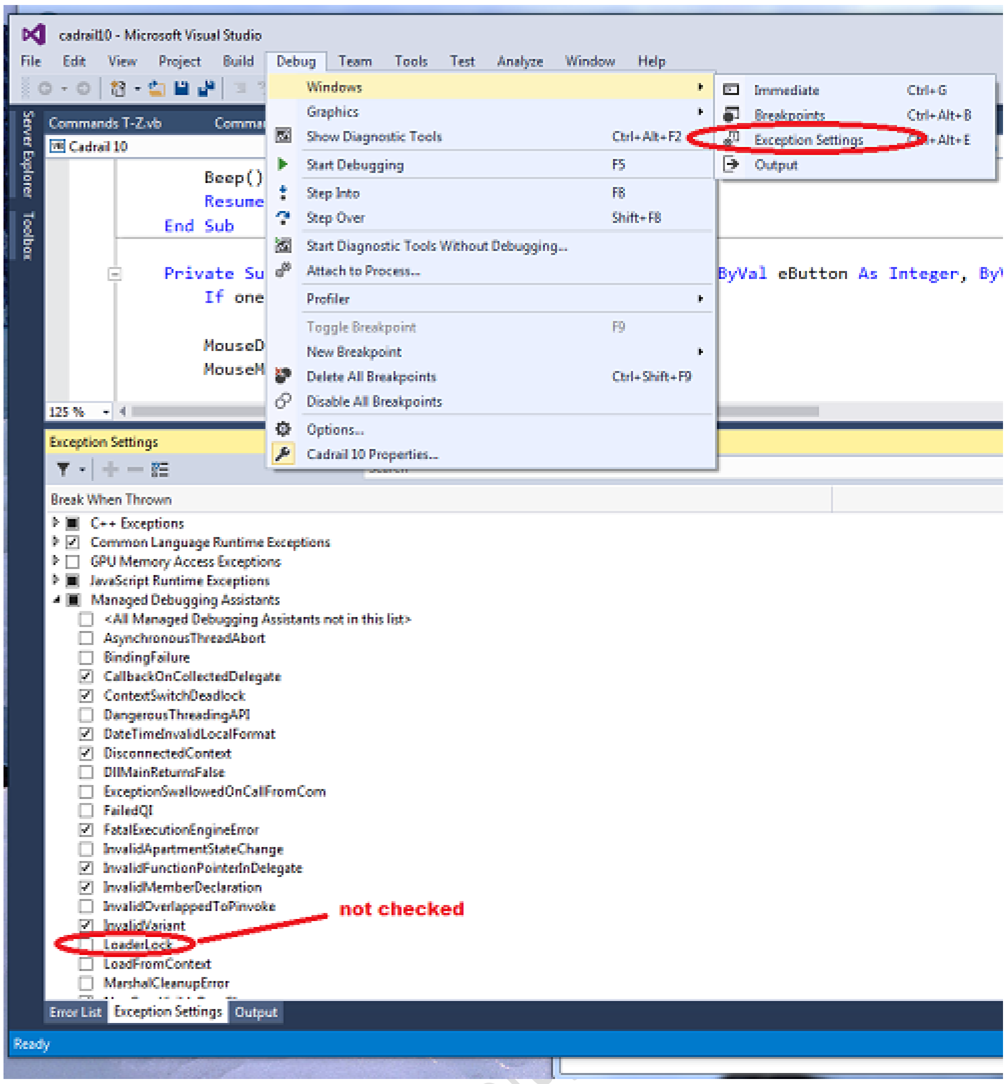
Step 5:

Edit the Form's C# code file. Namespace must be as same as your project name.

```
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 GP_P1
{
public partial class Form1 : Form
   {
      Microsoft.DirectX.Direct3D.Device device;
public Form1()
```

```
        {
InitializeComponent();
InitDevice();
        }

public void InitDevice()
        {
PresentParameterspp = 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.Orange, 0, 1);
device.Present();
        }

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

Step 6: Click on Start. And here is the output. We have initialized 3D Device.

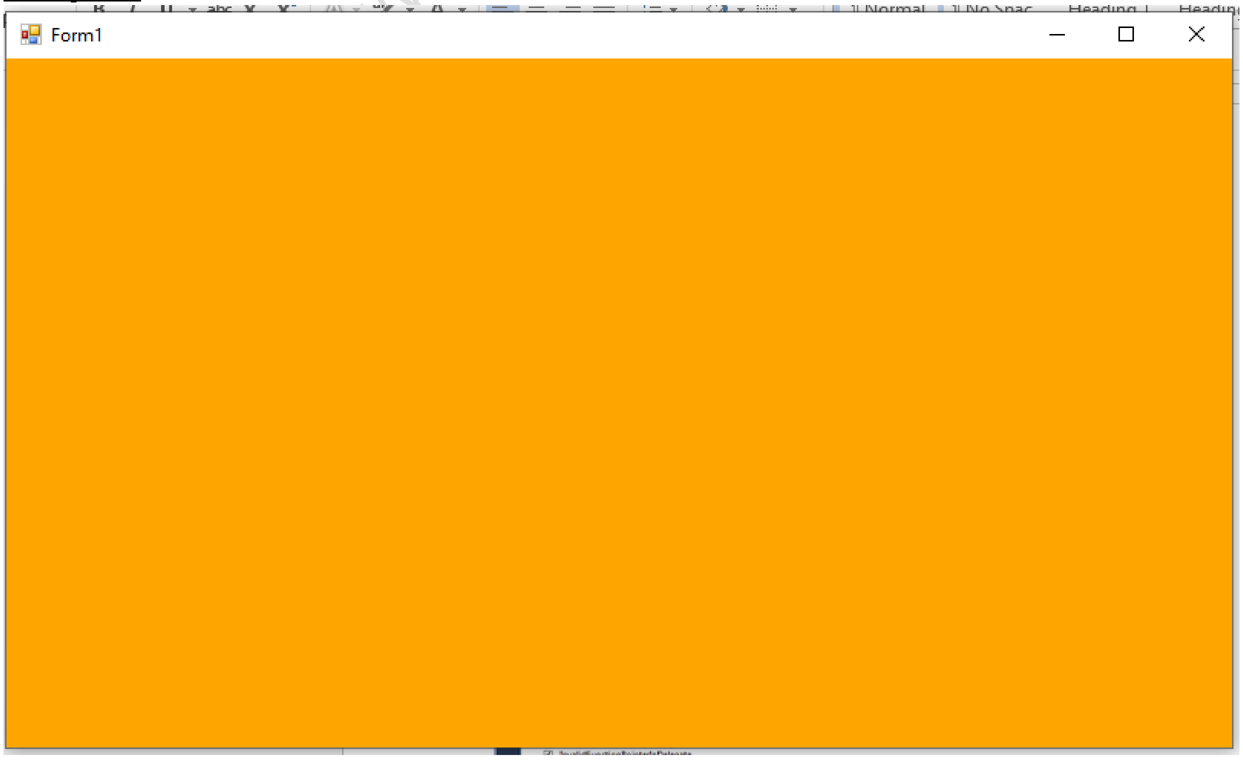


```
        Beep()
        Resume
    End Sub

    Private Su                         ByVal eButton As Integer, By
        If one

        MouseD
        MouseM
```

- ■ C++ Exceptions
- ✓ Common Language Runtime Exceptions
- ☐ GPU Memory Access Exceptions
- ■ JavaScript Runtime Exceptions
- ■ Managed Debugging Assistants
  - ☐ <All Managed Debugging Assistants not in this list>
  - ☐ AsynchronousThreadAbort
  - ☐ BindingFailure
  - ✓ CallbackOnCollectedDelegate
  - ✓ ContextSwitchDeadlock
  - ☐ DangerousThreadingAPI
  - ✓ DateTimeInvalidLocalFormat
  - ✓ DisconnectedContext
  - ☐ DllMainReturnsFalse
  - ☐ ExceptionSwallowedOnCallFromCom
  - ☐ FailedQI
  - ✓ FatalExecutionEngineError
  - ☐ InvalidApartmentStateChange
  - ✓ InvalidFunctionPointerInDelegate
  - ✓ InvalidMemberDeclaration
  - ☐ InvalidOverlappedToPinvoke
  - ✓ InvalidVariant
  - ☐ LoaderLock
  - ☐ LoadFromContext
  - ☐ MarshalCleanupError

NOTE : uncheck loaderlock or else it generates exception

<u>Output:</u>



Practical No. 2:

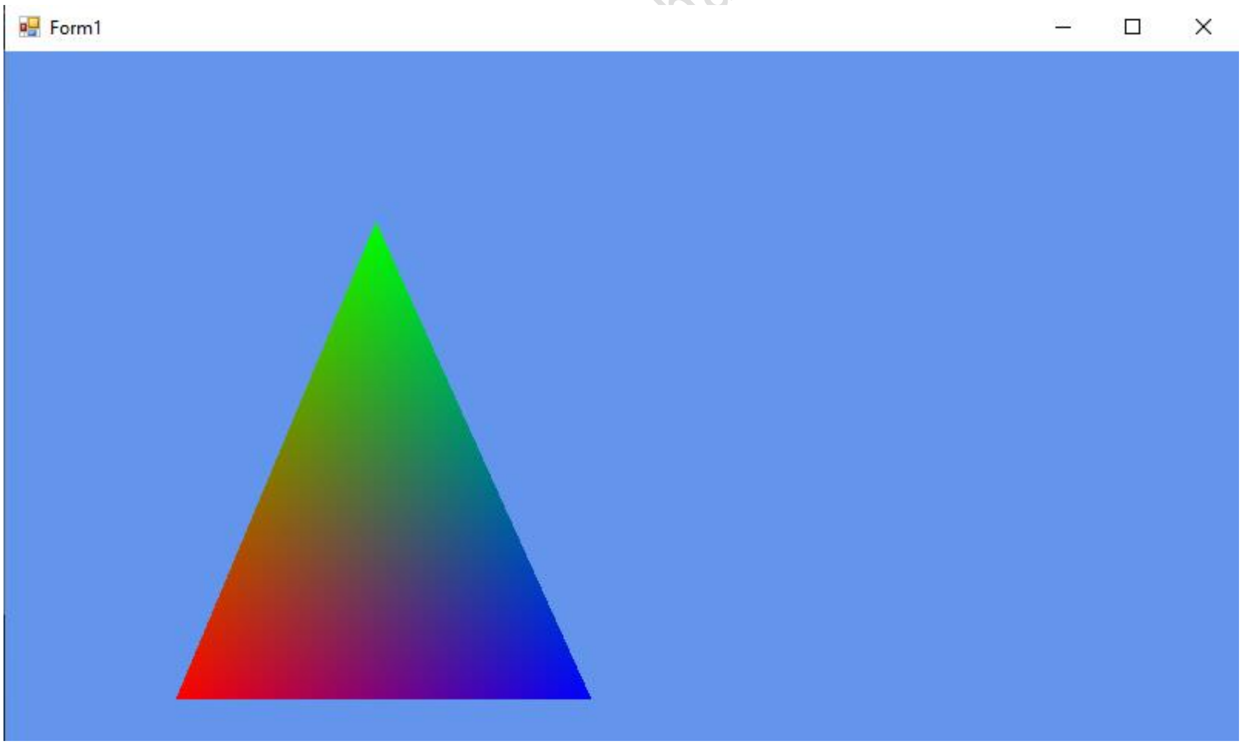
Aim: Draw a triangle using Direct3D 11

Solution:

```
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 GPRACT2
{
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);
       }
private void Render()
       {
CustomVertex.TransformedColored[] vertexes = new
CustomVertex.TransformedColored[3];

vertexes[0].Position = new Vector4(240, 110, 0, 1.0f);//first point
vertexes[0].Color = System.Drawing.Color.FromArgb(0, 255, 0).ToArgb();

vertexes[1].Position = new Vector4(380, 420, 0, 1.0f);//second point
vertexes[1].Color = System.Drawing.Color.FromArgb(0, 0, 255).ToArgb();

vertexes[2].Position = new Vector4(110, 420, 0, 1.0f);//third point
vertexes[2].Color = System.Drawing.Color.FromArgb(255, 0, 0).ToArgb();
device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1.0f, 0);
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();
        }
    }
}
```

Note: Go to Properties Section of Form, select Paint in the Event List and enter as Form1_Paint.

<u>Output:</u>



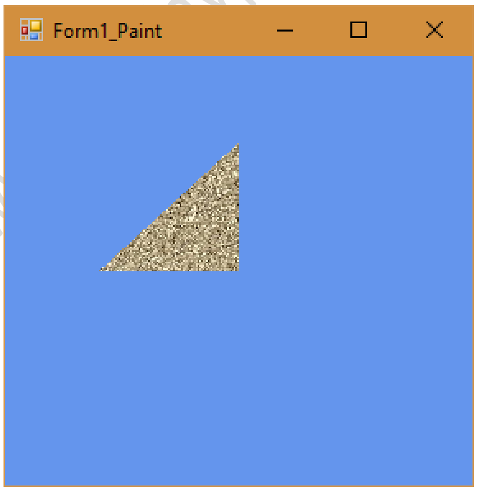
Practical No. 3:

<u>Aim:</u> Texture the triangle using Direct3D 11

<u>Solution:</u>

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

namespace GPPRACT3
{
public partial class Form1 : Form
    {
private Microsoft.DirectX.Direct3D.Device device;
privateCustomVertex.PositionTextured[] vertex = new
CustomVertex.PositionTextured[3];
private Texture texture;
public Form1()
        {
InitializeComponent();
InitDevice();
        }
private void InitDevice()
        {
PresentParameterspp = 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, 20), new
Vector3(),
new Vector3(0, 1, 0));

device.RenderState.Lighting = false;

vertex[0] = new CustomVertex.PositionTextured(new Vector3(0, 0, 0), 0,
0);
vertex[1] = new CustomVertex.PositionTextured(new Vector3(5, 0, 0), 0,
1);
vertex[2] = new CustomVertex.PositionTextured(new Vector3(0, 5, 0),-1,
1);
texture=new Texture (device, new Bitmap ("E:\\TYCS\\images\\img1.jpg"),
0,Pool.Managed );
        }

private void Form1_Load(Object sender, EventArgs e)
        { }

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

Output:



NOTE: don't forget to add image into your solution with proper path specified

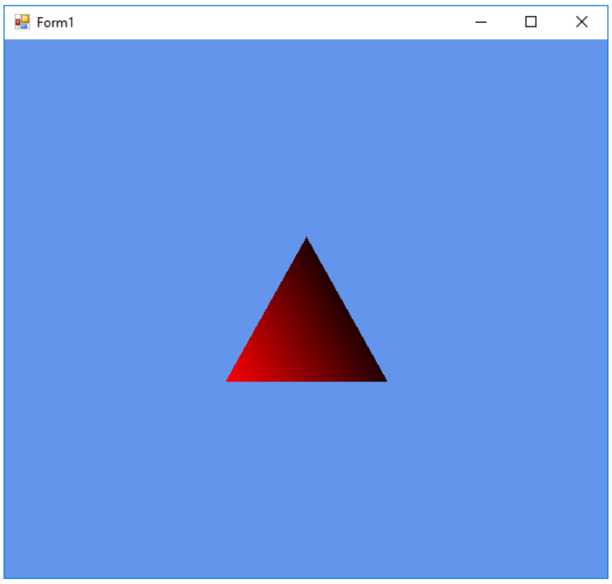
Practical No. 4:

Aim: Programmable Diffuse Lightning using Direct3D 11

Solution:

```
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 GPPRACT4
{
public partial class Form1 : Form
   {

private Microsoft.DirectX.Direct3D.Device device;
privateCustomVertex.PositionNormalColored[] vertex = new
CustomVertex.PositionNormalColored[3];
public Form1()
       {
InitializeComponent();
InitDevice();
       }

public void InitDevice()
       {
PresentParameterspp = 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.Red.ToArgb());
vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1,
1), new Vector3(1, 0, 1), Color.Red.ToArgb());

vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1,
1), new Vector3(-1, 0, 1), Color.Red.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;
        }

public void Render()
        {
        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_Load(object sender, EventArgs e)
        {

        }

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


    }
}
```

<u>Output:</u>

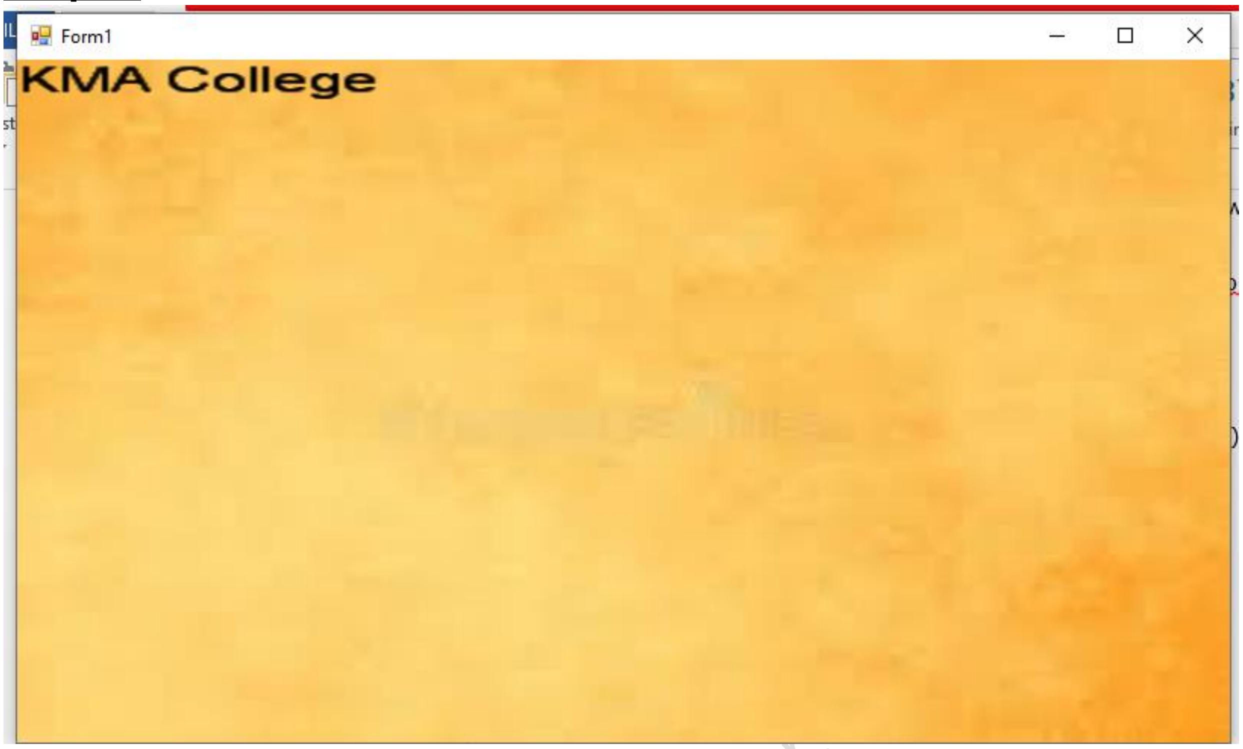


Practical No. 5:

<u>Aim:</u> Loading models into DirectX 11 and rendering

```
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 GPPRACT5
{
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 InitFont()
 {
        System.Drawing.Font f = new System.Drawing.Font("Arial", 16f,
        FontStyle.Regular);
        font = new Microsoft.DirectX.Direct3D.Font(device, f);
 }

private void LoadTexture()
 {
        texture =
        TextureLoader.FromFile(device,"E:\\TYCS\\images\\img1.jpg",400,
        400, 1, 0, Format.A8B8G8R8, Pool.Managed, Filter.Point, Filter.Point,
        Color.Transparent.ToArgb());
 }

private void InitDevice()
 {
        PresentParameterspp = 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.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.White);
        font.DrawText(s, "KMA College", new Point(0, 0), Color.Black);
        s.End();
 }
device.EndScene();
device.Present();
 }
private void Form1_Paint(object sender, PaintEventArgs e)
 {
Render();
 }
 }
}
```

Output:


Form1
KMA College