Practical no 5

AIM: Lightning (Programmable Diffuse Lightning using Direct3D 11)

Steps:-

1. Create a new project and select a windows form application(.Net Framework 2.0-3.5).
2. Right click on the properties → click on open → click build → select platform target → x86 or add new
3. Click on view code on form 1(design) or press F7.
4. Go to the solution explorer → right click on project name → select add reference .
5. Click on browse and add the required dll files.
6. Code the required files.
7. Add the paint method for changing the appearance .
8. Change the window name and icon if possible.
9. Disable the Exception Settings option such as LoaderLock.
10. Run the app.

Code:-

Program.cs file

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;
using Microsoft.DirectX.Direct3D;

namespace WindowsFormsApp6
{
    static class Program
    {

        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

Form1.cs file

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

namespace WindowsFormsApp6
{
    public partial class Form1 : Form
    {
        private Microsoft.DirectX.Direct3D.Device device;
        // Microsoft.DirectX.Direct3D.Device device;
        private CustomVertex.PositionNormalColored[] vertex
             = new CustomVertex.PositionNormalColored[3];
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            device.Clear(ClearFlags.Target, Color.Black, 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)
        {
            PresentParameters pp = new PresentParameters();
            pp.Windowed = true;
            pp.SwapEffect = SwapEffect.Discard;
            device = new Device(0, DeviceType.Hardware, this, CreateFlags.HardwareVertexPro-
cessing, pp);
            device.Transform.Projection = Matrix.PerspectiveFovLH(3.14f / 4 , device.View-
port.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 Vec-
tor3(1, 0, 1),
             Color.OrangeRed.ToArgb());

            vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1, 1), new Vec-
tor3(1, 0, 1),
             Color.Orange.ToArgb());
```

```
            vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1, 1), new Vec-
tor3(-1, 0, 1),
             Color.Red.ToArgb());

            device.RenderState.Lighting = true;
            device.Lights[0].Type = LightType.Directional;
            device.Lights[0].Diffuse = Color.White;
            device.Lights[0].Direction = new Vector3(0.8f, 0, -1);
            device.Lights[0].Enabled = true;


        }
    }
}
```

Output: