

2019/07/11 プログラミング演習

目的

この演習においてはオブジェクト指向プログラミングの理解を深める。

装置/ツール

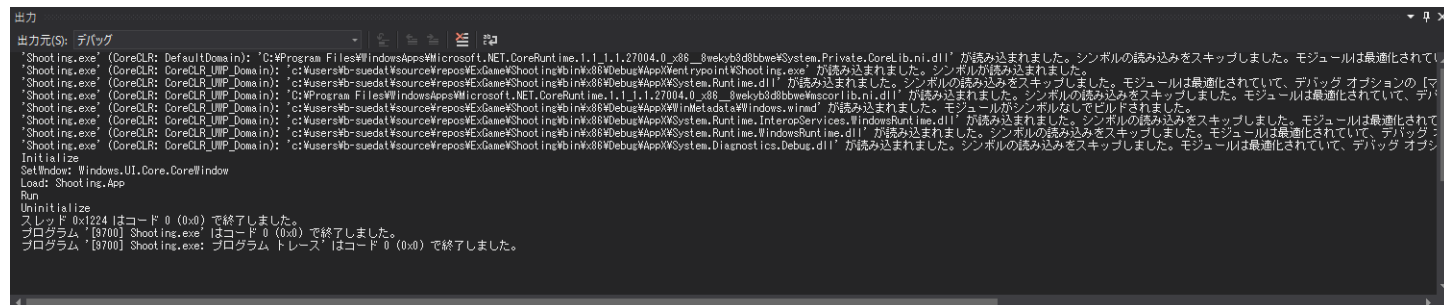
- Visual Studio
- Windows 10 Pro

実験

問題9.1

実験書の図9.1を実行し、デバッグ出力をスクリーンショットで報告しなさい。

デバッグ出力を図9.1に示す。



```
出力
出力元(S): デバッグ
'Shooting.exe' (CoreCLR: DefaultDomain): 'C:\Program Files\WindowsApps\Microsoft.NET.CoreRuntime.1.1.1.27004.0_x86_8wekyb3d8bbwe\System.Private.CoreLib.ni.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されてい
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\WinTryptoint\Shooting.exe' が読み込まれました。シンボルが読み込まれました。
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\System.Runtime.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されていて、デバッグ オプシ
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'C:\Program Files\WindowsApps\Microsoft.NET.CoreRuntime.1.1.1.27004.0_x86_8wekyb3d8bbwe\mscorlib.ni.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されてい
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\WinMetadata\Windows.winmd' が読み込まれました。モジュールがシンボルなしでビルドされました。
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\System.Runtime.InteropServices.WindowsRuntime.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されて
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\System.Runtime.InteropServices.WindowsRuntime.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されてい
'Shooting.exe' (CoreCLR: CoreCLR_Unip_Domain): 'c:\Users\br-suedat\source\repos\ExGame\Shooting\bin\x86\Debug\App\System.Diagnostics.Debug.dll' が読み込まれました。シンボルの読み込みをスキップしました。モジュールは最適化されていて、デバッグ オプシ
Initialize
SetWindow: Windows.UI.Core.CoreWindow
Load: Shooting.App
Run
Uninitialize
スレッド 0x1224 はコード 0 (0x0) で終了しました。
プログラム '[3700] Shooting.exe' はコード 0 (0x0) で終了しました。
プログラム '[3700] Shooting.exe: プログラム トレース' はコード 0 (0x0) で終了しました。
```

図9.1 デバッグ出力

問題9.2

実験書の図9.3を実行し、実行結果をスクリーンショットで報告しなさい。

実行結果を図9.2に示す。

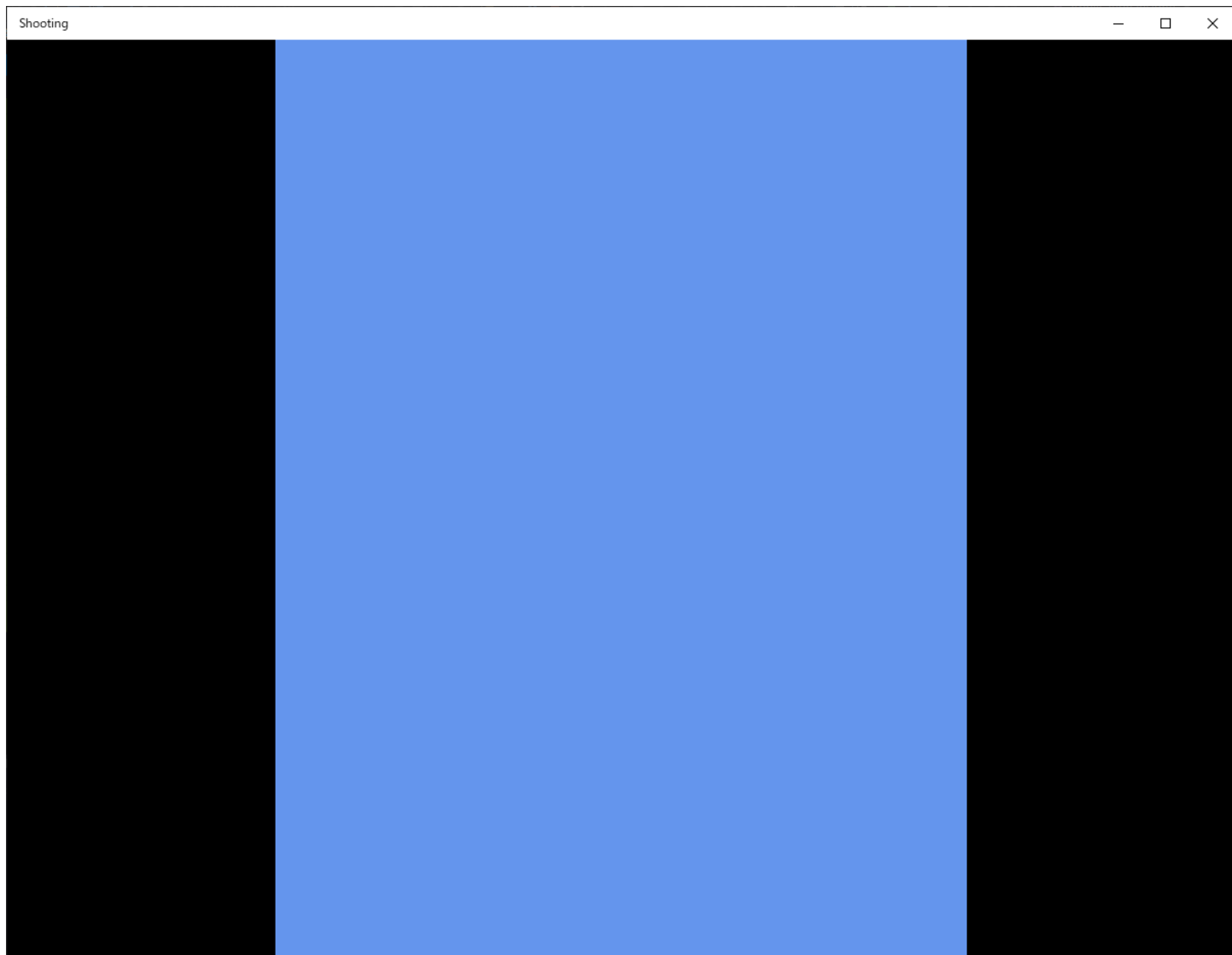


図9.2 実行結果

問題9.3

実験書図9.6から図9.9までの可変を反映しそのスクリーンショットを報告しなさい。

スクリーンショットを図9.3に示す。

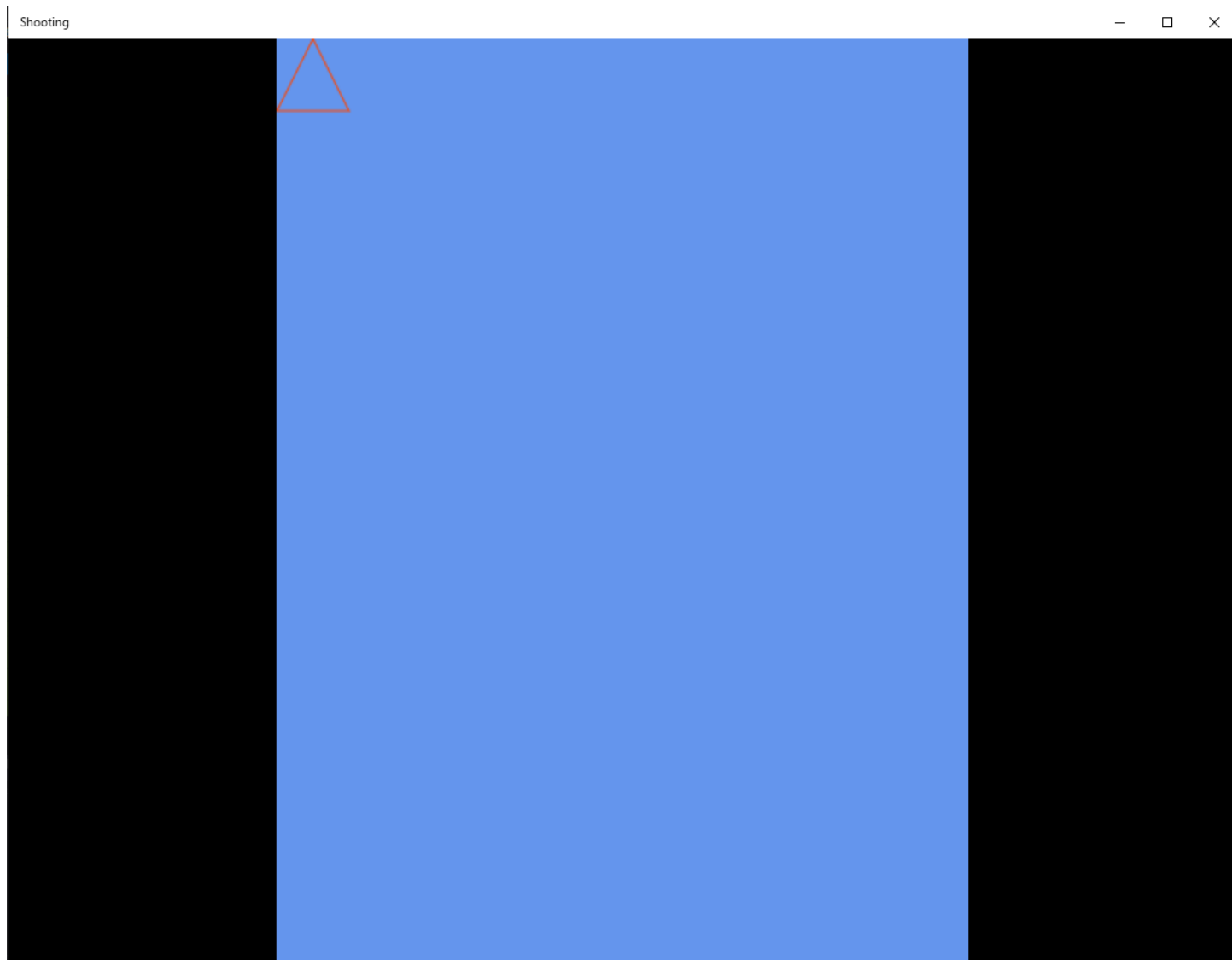


図9.3 スクリーンショット

問題9.4

完成したプログラムを改変して、別の多角形を表示し、その結果の変更箇所およびスクリーンショットを報告しなさい。

問題9.5

実験書の改変箇所をすべて反映させたプログラムを作成し、ソースコードと実行結果と動作結果を報告しなさい。

ソースコードを図9.5.1に示す。

```

using SharpDX;
using SharpDX.Direct2D1;
using SharpDX.Direct3D;
using SharpDX.Direct3D11;
using SharpDX.DXGI;
using System;
using System.Diagnostics;
using Windows.ApplicationModel.Activation;
using Windows.ApplicationModel.Core;
using Windows.Graphics.Display;
using Windows.UI.Core;
using Windows.UI.ViewManagement;

namespace Shooting
{
    class App
    {
        [MTAThread]
        private static void Main()
        {
            var viewFactory = new FrameworkViewSource();

            CoreApplication.Run(viewFactory);
        }

        class FrameworkViewSource : IFrameworkViewSource
        {
            public IFrameworkView CreateView()
            {
                return new FrameworkView();
            }
        }

        class FrameworkView : IFrameworkView
        {
            private SharpDX.Direct2D1.DeviceContext d2dDeviceContext;
            private Bitmap1 d2dTarget;
            private SwapChain1 swapChain;
            private CoreWindow mWindow;

            // 9.6追記
            private TransformedGeometry tFighterPath;
            private SolidColorBrush fighterBrush;

            public void Initialize(CoreApplicationView applicationView)
            {
                Debug.WriteLine("Initialize");

                applicationView.Activated += OnActivated;
            }

            void OnActivated(CoreApplicationView applicationView, IActivatedEventArgs args)
            {
                CoreWindow.GetForCurrentThread().Activate();
            }

            void CreateDeviceResources()
            {
                var defaultDevice = new SharpDX.Direct3D11.Device(DriverType.Hardware, DeviceCreationFlags.CreateDefault);

                var device = defaultDevice.QueryInterface<SharpDX.Direct3D11.Device1>();

                var dxgiDevice2 = device.QueryInterface<SharpDX.DXGI.Device2>();

                var dxgiAdapter = dxgiDevice2.Adapter;
                SharpDX.DXGI.Factory2 dxgiFactory2 = dxgiAdapter.GetParent<SharpDX.DXGI.Factory2>();

                var desc = new SwapChainDescription1();
                desc.Width = 480;
                desc.Height = 640;
                desc.Format = Format.B8G8R8A8_UNorm;
                desc.Stereo = false;
            }
        }
    }
}

```

```

desc.SampleDescription = new SampleDescription(1, 0);
desc.Usage = Usage.RenderTargetOutput;
desc.BufferCount = 2;
desc.Scoring = Scoring.AspectRatioStretch;
desc.SwapEffect = SwapEffect.FlipSequential;
desc.Flags = SwapChainFlags.AllowModeSwitch;

this.swapChain = new SwapChain1(dxgiFactory2, device, new ComObject(mWindow), ref de

var d2dDevice = new SharpDX.Direct2D1.Device(dxgiDevice2);

this.d2dDeviceContext = new SharpDX.Direct2D1.DeviceContext(d2dDevice, DeviceContext

var backBuffer = this.swapChain.GetBackBuffer<Surface>(0);

var displayInfo = DisplayInformation.GetForCurrentView();

this.d2dTarget = new Bitmap1(this.d2dDeviceContext, backBuffer, new BitmapProperties

// 9.7追記
var fighterPath = new PathGeometry(d2dDevice.Factory);

var sink = fighterPath.Open();

sink.BeginFigure(new Vector2(25f, 0f), FigureBegin.Filled);

sink.AddLines(new SharpDX.Mathematics.Interop.RawVector2[]
{
    new Vector2(50f, 50f)
    , new Vector2(0f, 50f)
});
sink.EndFigure(FigureEnd.Closed);
sink.Close();

this.tFighterPath = new TransformedGeometry(d2dDevice.Factory, fighterPath, Matrix3x
this.fighterBrush = new SolidColorBrush(d2dDeviceContext, Color.OrangeRed);
}

public void SetWindow(CoreWindow window)
{
    Debug.WriteLine("SetWndow: " + window);

    this.mWindow = window;
}

public void Load(string entryPoint)
{
    Debug.WriteLine("Load: " + entryPoint);

    this.CreateDeviceResources();
}

public void Run()
{
    Debug.WriteLine("Run");

    // 9.11追記
    var dx = 0;
    var dy = 0;

    while (true)
    {
        this.mWindow.Dispatcher.ProcessEvents(CoreProcessEventsOption.ProcessAllIfPr
        // 入力
        // 9.12追記
        if (this.mWindow.GetAsyncKeyState(Windows.System.VirtualKey.Escape) == CoreVi
        {
            return;
        }
        if (this.mWindow.GetAsyncKeyState(Windows.System.VirtualKey.Right) == CoreVir
        {
            dx = dx + 10;
        }
    }
}

```

```

        if(this.mWindow.GetAsyncKeyState(Windows.System.VirtualKey.Left) == CoreVirt
        {
            dx = dx - 10;
        }
        if(this.mWindow.GetAsyncKeyState(Windows.System.VirtualKey.Down) == CoreVirt
        {
            dy = dy + 10;
        }
        if(this.mWindow.GetAsyncKeyState(Windows.System.VirtualKey.Up) == CoreVirtua
        {
            dy = dy - 10;
        }

        // 出力 未実装
        this.d2dDeviceContext.Target = d2dTarget;
        this.d2dDeviceContext.BeginDraw();
        this.d2dDeviceContext.Clear(Color.CornflowerBlue);

        // 描画
        // 9.13追記
        var fTransform = tFighterPath.Transform;
        fTransform.M31 = dx;
        fTransform.M32 = dy;
        this.d2dDeviceContext.Transform = fTransform;

        // 9.8追記
        this.d2dDeviceContext.DrawGeometry(this.tFighterPath, this.fighterBrush);

        this.d2dDeviceContext.EndDraw();
        this.swapChain.Present(0, PresentFlags.None);
        //待機 未実装
    }
}

public void Uninitialize()
{
    Debug.WriteLine("Uninitialize");

    this.swapChain.Dispose();
    this.d2dDeviceContext.Dispose();
    this.d2dTarget.Dispose();
}
}
}
}

```

図9.5.1 今回のソースコード

実行結果を図9.5.2に示す。

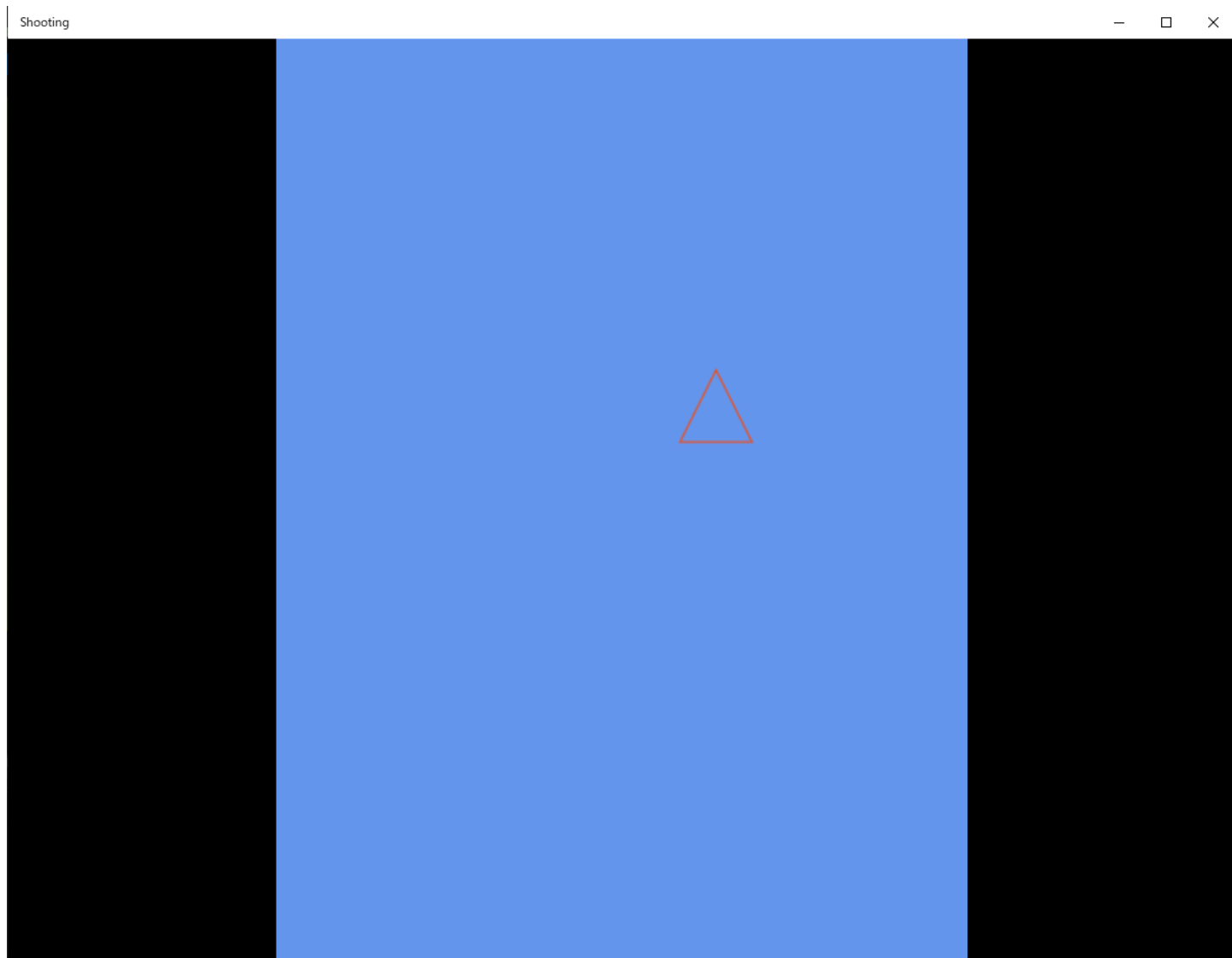


図9.5.2 実行結果

課題

レポート 課題9.1

オブジェクト指向設計を本プログラムに行うとするならば、どのように行うか。説明しなさい。
最初にクラス図を書いて設計する。