COS20007 - Object Oriented Programming

Student name: Nguyen Duc Manh
ID: 105547489

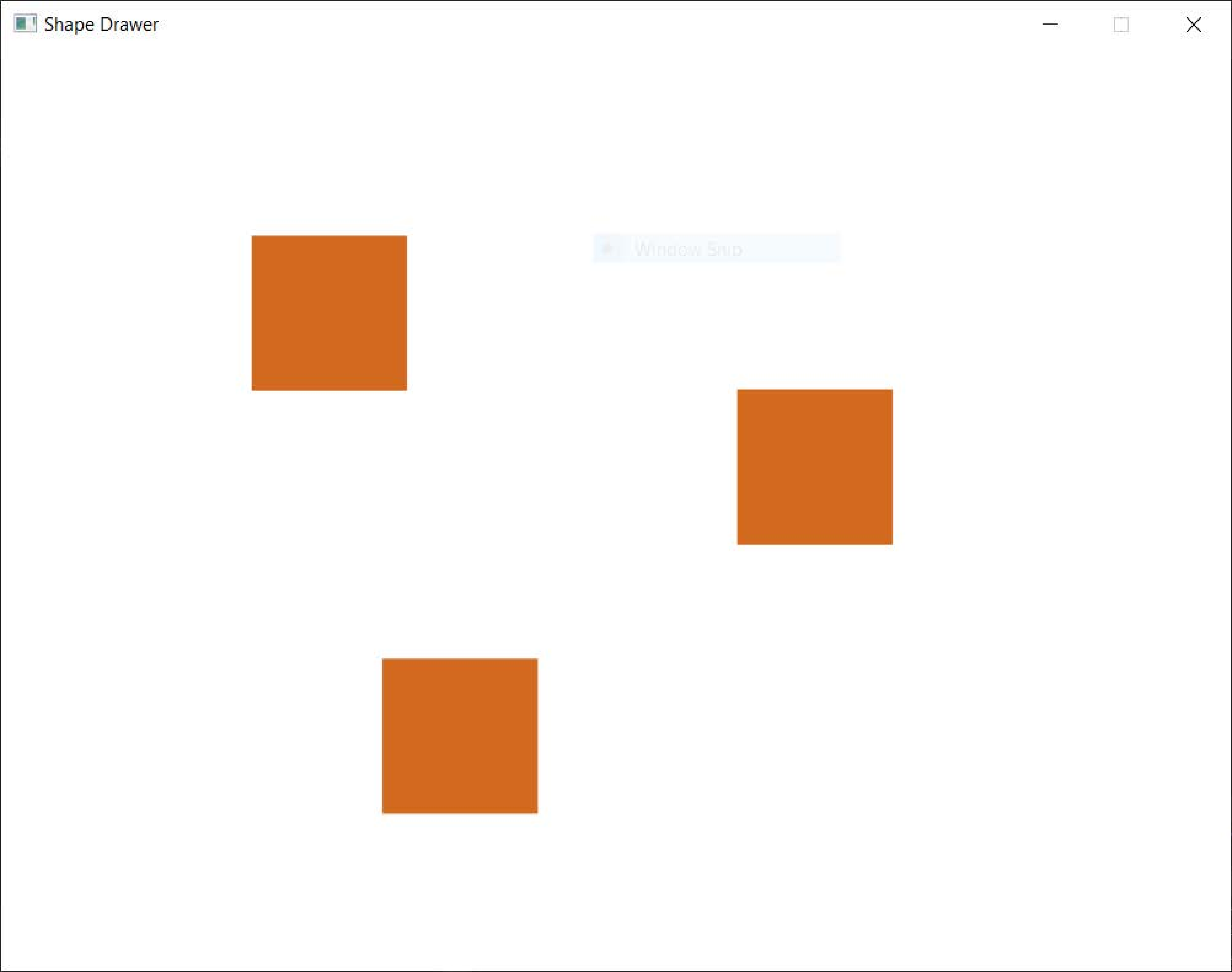
3.3P - Drawing Program - Drawing Class with your own attributes

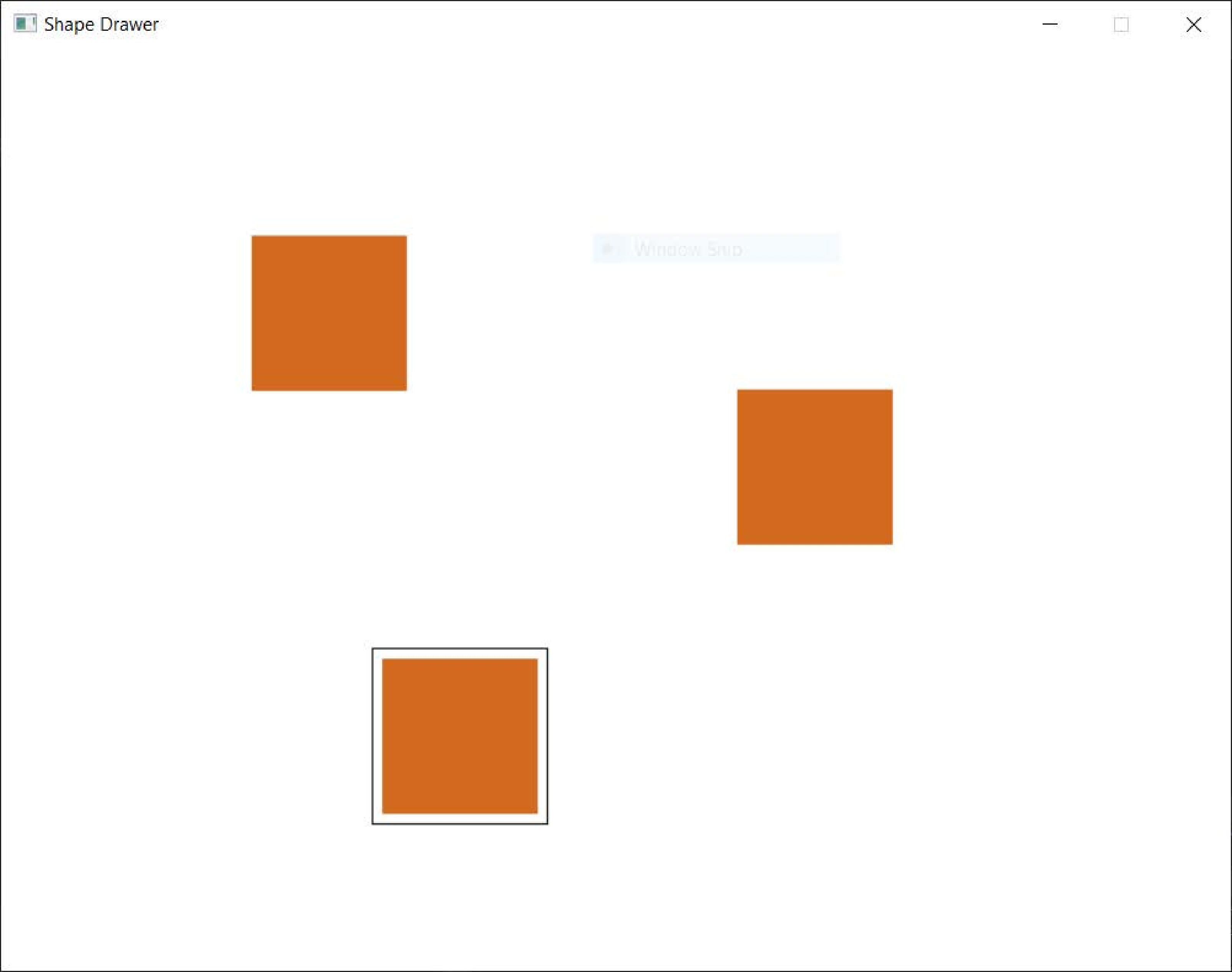
SWINBURNE UNIVERSITY OF TECHNOLOGY

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Program
    {
        public static void Main()
        {
            ////the first Shape is the type of object, which is the class i've ↩
              made earlier
            //Shape myShape = new Shape(); //Shape() to call the constructor
            Window window = new Window("Shape Drawer", 800, 600);
            Drawing myDrawing = new Drawing();
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();
                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    int X = (int)SplashKit.MouseX();
                    int Y = (int)SplashKit.MouseY();
                    Shape newShape = new Shape();
                    newShape.X = X; newShape.Y = Y;
                    myDrawing.AddShape(newShape);

                }
                if (SplashKit.KeyDown(KeyCode.SpaceKey))
                {
                    myDrawing.Background = Color.RandomRGB(255);
                }
                if (SplashKit.MouseClicked(MouseButton.RightButton))
                {
                    myDrawing.SelectShapeAt(SplashKit.MousePosition());

                }
                if (SplashKit.KeyDown(KeyCode.DeleteKey) || SplashKit.KeyDown  ↩
                  (KeyCode.BackspaceKey))
                {
                    foreach (Shape newShape in myDrawing.SelectedShapes)
                    {
                        myDrawing.RemoveShape(newShape);
                    }
                }
                myDrawing.Draw();
                SplashKit.RefreshScreen();
            } while (!window.CloseRequested);
        }
    }
}
```

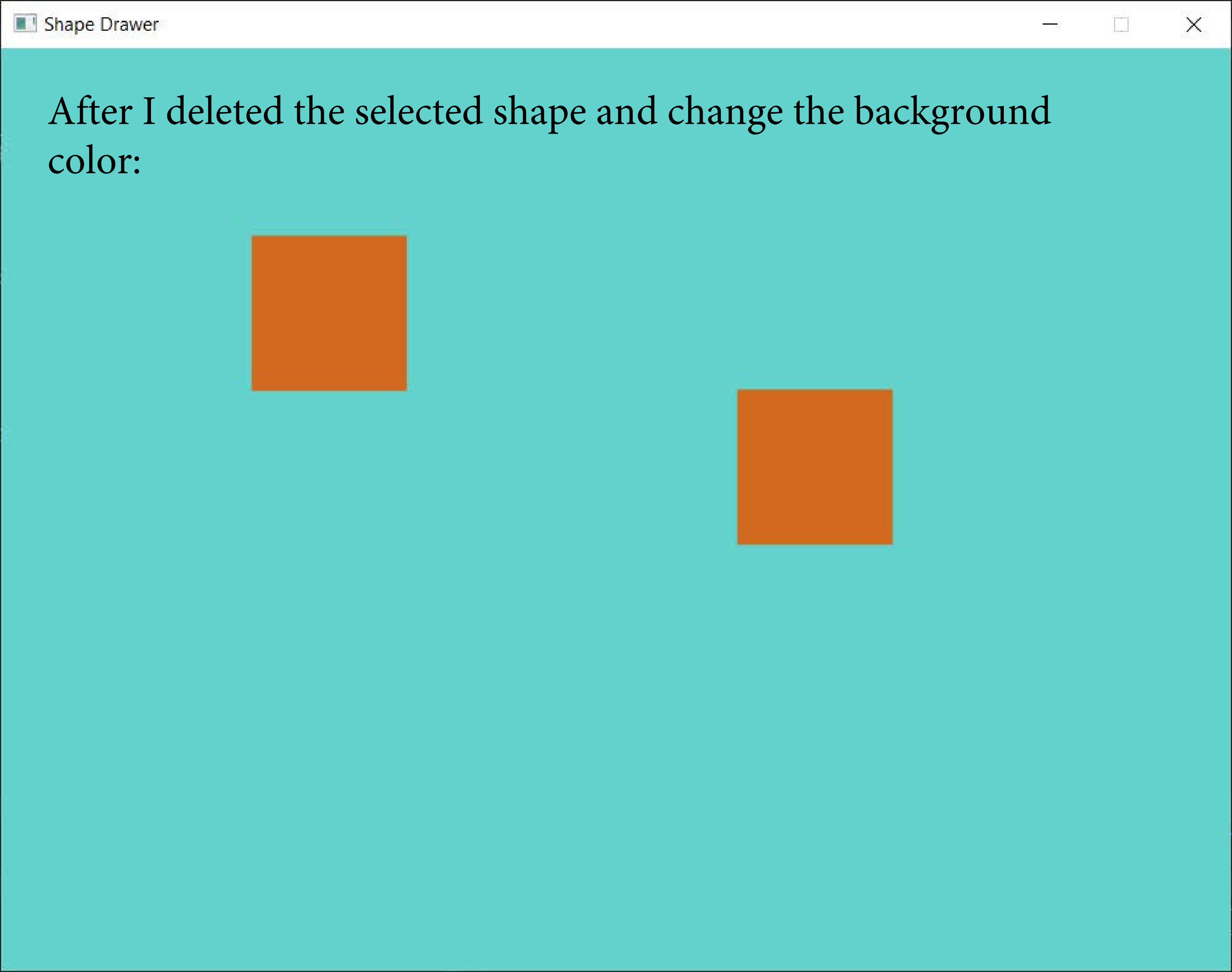
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Shape
    {
        //private fields
        Color _color;
        float _x, _y;
        int _width, _height;
        private bool _selected; //bool field is "false" by default
        public Shape() //Constructor
        {
        _color = Color.Chocolate;
        _x = _y = 0.0f;
        _width = _height = 101;
        }
        //Properties
        public Color FillColor
        {
            get
            {
                return _color;
            }
            set
            {
                _color = value;
            }
        }
        public float
        X
        {
            get
            {
                return _x;
            }
            set
            {
                _x = value
                ;
            }
        }
        public float Y
        {
            get
            {
                return _y;
            }
            set
            {
                _y = value;
            }
        }
        public bool Selected
        {
            get { return _selected; }
            set { _selected = value; }
        }

        //methods
        public void Draw()
        {
            SplashKit.FillRectangle(_color, _x, _y, _width, _height);
            if (Selected) { DrawOutline(); }
        }
        public bool IsAt(Point2D pt)
        {
            return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom(X,    ⮐
              Y, _width, _height));
        }

        public void DrawOutline()
        {
            SplashKit.DrawRectangle(Color.Black, _x - 7, _y - 7, _width + 14 , ⮐
              _height + 14); //105547489
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Drawing
    {
        private readonly List<Shape> _shapes;
        private Color _background;

        public Drawing(Color background)
        {
            _shapes = new List<Shape>();
            _background = background;
        }

        public Drawing(): this(Color.White) //default constructor -
          initializes objs with predefined values
        {
            //other steps
        }

        //methods
        public void AddShape(Shape shape)
        {
            _shapes.Add(shape);
        }

        public void RemoveShape(Shape shape)
        {
            _ = _shapes.Remove(shape); //to discard the value it returns
        }

        public void Draw()
        {
            SplashKit.ClearScreen(_background);
            foreach (Shape shape in _shapes)
            {
                shape.Draw();
            }
        }

        public void SelectShapeAt(Point2D pt)
        {
            foreach (Shape shape in _shapes)
            {
                shape.Selected = shape.IsAt(pt);
            }
        }

        //properties
        public Color Background
        {
            get
            {
                return _background;
            }
            set
            {
                _background = value;
            }
        }

        public int ShapeCount => _shapes.Count;

        public List<Shape> SelectedShapes
        {
            get
            {
                List<Shape> result = new List<Shape>();
                foreach (Shape shape in _shapes)
                {
                    if (shape.Selected)
                    {
                        result.Add(shape);
                    }
                } return result;
            }
        }
    }
}
```


Shape Drawer
Window Snip


Shape Drawer
Window Snip

After I deleted the selected shape and change the background color:


Shape Drawer