```
using System;
using SplashKitSDK;
using System.IO;
using System.Xml.Linq;

namespace MyGame
{
    public class Program
    {
        private enum ShapeKind
        {
            Rectangle,
            Circle,
            Line
        }
        public static void Main()
        {
            ShapeKind kindToAdd = ShapeKind.Circle;

            Window window = new Window("Shape Drawer", 800, 600);

            Drawing myDrawing = new Drawing();

            int count=0;
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                if(SplashKit.KeyTyped(KeyCode.RKey))
                {
                    kindToAdd = ShapeKind.Rectangle;
                    Console.WriteLine("rectangle");
                    count = 0;
                }

                if(SplashKit.KeyTyped(KeyCode.CKey))
                {
                    kindToAdd = ShapeKind.Circle;
                    Console.WriteLine("circle");
                    count = 0;
                }

                if (SplashKit.KeyTyped(KeyCode.LKey))
                {
                    kindToAdd = ShapeKind.Line;
                    Console.WriteLine("line");
                    count = 0;
                }
```

```
50
51                  if (SplashKit.MouseClicked(MouseButton.LeftButton) &&
                      count<3)
52                  {
53                      Shape newShape;
54
55                      switch(kindToAdd)
56                      {
57                          case ShapeKind.Circle:
58                              newShape = new MyCircle();
59                              break;
60
61                          case ShapeKind.Line:
62                              newShape = new MyLine();
63                              count++;
64                              break;
65
66                          default:
67                              newShape = new MyRectangle();
68                              break;
69
70                      }
71
72
73                      newShape.X = SplashKit.MouseX();
74                      newShape.Y = SplashKit.MouseY();
75                      myDrawing.AddShape(newShape);
76                  }
77
78                  if(SplashKit.KeyTyped(KeyCode.SpaceKey))
79                  {
80                      myDrawing.BackGround = SplashKit.RandomColor();
81                  }
82
83                  if(SplashKit.MouseClicked(MouseButton.RightButton))
84                  {
85                      myDrawing.SelectShapeAt(SplashKit.MousePosition());
86                  }
87
88                  string file_path = "E:/COS20007/week5/Task5_3C/
                      MultipleShapeKinds/TestDrawing.txt";
89
90                  if (SplashKit.KeyTyped(KeyCode.DeleteKey) ||
                      SplashKit.KeyTyped(KeyCode.BackspaceKey))
91                  {
92                      foreach(Shape s in myDrawing.SelectedShapes)
93                      {
94                          myDrawing.RemoveShape(s);
95                      }
```

```
 96                  }
 97
 98                  if(SplashKit.KeyTyped(KeyCode.SKey))
 99                  {
100
101                      myDrawing.Save(file_path);
102
103                      Console.WriteLine($"Drawing saved to {file_path}");
104                  }
105
106                  if(SplashKit.KeyTyped(KeyCode.OKey))
107                  {
108                      try
109                      {
110                          myDrawing.Load(file_path);
111                      }
112                      catch(Exception e)
113                      {
114                          Console.Error.WriteLine("Error loading file: {0}", ↵
                        e.Message);
115                      }
116                  }
117
118                  myDrawing.Draw();
119                  SplashKit.RefreshScreen();
120              } while (!window.CloseRequested);
121          }
122
123      }
124  }
125
```

```
1
1
1
5
Circle
0
0
1
433
208
123
Rectangle
0
0.5
0
222
420
173
173
Rectangle
0
0.5
0
611
391
173
173
Line
1
0
0
71
181
0
0
Line
1
0
0
252
159
0
0

```

```
1  using System;
2  using System.IO;
3  using SplashKitSDK;
4
5  namespace MyGame
6  {
7      public static class ExtensionMethod
8      {
9          public static int ReadInteger(this StreamReader reader)
10         {
11             return Convert.ToInt32(reader.ReadLine());
12         }
13
14         public static float ReadSingle(this StreamReader reader)
15         {
16             return Convert.ToSingle(reader.ReadLine());
17         }
18
19         public static Color ReadColor(this StreamReader reader)
20         {
21             return Color.RGBColor(reader.ReadSingle(), reader.ReadSingle(), ↵
                   reader.ReadSingle());
22         }
23
24         public static void WriteColor(this StreamWriter writer, Color clr)
25         {
26             writer.WriteLine("{0}\n{1}\n{2}", clr.R, clr.G, clr.B);
27         }
28     }
29 }
30
```

```
1  using SplashKitSDK;
2  using System;
3  using System.IO;
4
5  namespace MyGame
6  {
7      public abstract class Shape
8      {
9          private Color _color;
10         private float _x;
11         private float _y;
12         private bool _selected;
13
14         public Shape() : this(Color.Yellow)
15         {
16
17         }
18
19         public Shape(Color color)
20         {
21             _color = color;
22             _x = 0.0f;
23             _y = 0.0f;
24         }
25
26         public Color Color
27         {
28             get
29             {
30                 return _color;
31             }
32             set
33             {
34                 _color = value;
35             }
36         }
37
38         public float X
39         {
40             get
41             {
42                 return _x;
43             }
44             set
45             {
46                 _x = value;
47             }
48         }
49
```

```
50          public float Y
51          {
52              get
53              {
54                  return _y;
55              }
56              set
57              {
58                  _y = value;
59              }
60          }
61
62          public bool Selected
63          {
64              get
65              {
66                  return _selected;
67              }
68              set
69              {
70                  _selected = value;
71              }
72          }
73
74          public abstract void Draw();
75
76          public abstract Boolean IsAt(Point2D pt);
77
78          public abstract void DrawOutline();
79
80          public virtual void SaveTo(StreamWriter writer)
81          {
82              writer.WriteColor(Color);
83              writer.WriteLine(X);
84              writer.WriteLine(Y);
85          }
86
87          public virtual void LoadFrom(StreamReader reader)
88          {
89              Color = reader.ReadColor();
90              X = reader.ReadInteger();
91              Y = reader.ReadInteger();
92          }
93      }
94  }
95
```

```
 1
 2  using SplashKitSDK;
 3  using System.ComponentModel;
 4  using System.Runtime.CompilerServices;
 5  using System.IO;
 6
 7  namespace MyGame
 8  {
 9      public class Drawing
10      {
11          private readonly List<Shape> _shapes;
12          private Color _background;
13
14          public Drawing() : this(Color.White)
15          {
16
17          }
18
19          public Drawing(Color background)
20          {
21              _shapes = new List<Shape>();
22              _background = background;
23          }
24
25          public List<Shape> SelectedShapes
26          {
27              get
28              {
29                  List<Shape> _selectedShapes = new List<Shape>();
30                  foreach (Shape s in _shapes)
31                  {
32                      if (s.Selected)
33                      {
34                          _selectedShapes.Add(s);
35                      }
36                  }
37                  return _selectedShapes;
38              }
39          }
40
41          public int ShapeCount
42          {
43              get
44              {
45                  return _shapes.Count;
46              }
47          }
48
49          public Color BackGround
```

```
50          {
51              get
52              {
53                  return _background;
54              }
55              set
56              {
57                  _background = value;
58              }
59          }
60
61          public void Draw()
62          {
63              SplashKit.ClearScreen(_background);
64              for (int i = 0; i < ShapeCount; i++)
65              {
66                  _shapes[i].Draw();
67              }
68          }
69
70          public void SelectShapeAt(Point2D pt)
71          {
72              foreach (Shape s in _shapes)
73              {
74                  if (s.IsAt(pt))
75                  {
76                      s.Selected = true;
77                  }
78                  else
79                  {
80                      s.Selected = false;
81                  }
82              }
83          }
84
85          public void AddShape(Shape s)
86          {
87              _shapes.Add(s);
88          }
89
90          public void RemoveShape(Shape s)
91          {
92              _ = _shapes.Remove(s);
93          }
94
95          public void Save(String filename)
96          {
97              StreamWriter writer = new StreamWriter(filename);
98
```

```
 99              try
100              {
101                  writer.WriteColor(BackGround);
102                  writer.WriteLine(ShapeCount);
103
104                  foreach (Shape s in _shapes)
105                  {
106                      s.SaveTo(writer);
107                  }
108              }
109              finally
110              {
111                  writer.Close();
112              }
113          }
114
115          public void Load(string filename)
116          {
117              StreamReader reader = new StreamReader(filename);
118
119              try
120              {
121                  int count;
122                  Shape s;
123                  string kind;
124
125                  BackGround = reader.ReadColor();
126                  count = reader.ReadInteger();
127                  _shapes.Clear();
128
129                  for (int i = 0; i < count; i++)
130                  {
131                      kind = reader.ReadLine();
132                      switch (kind)
133                      {
134                          case "Rectangle":
135                              s = new MyRectangle();
136                              break;
137                          case "Circle":
138                              s = new MyCircle();
139                              break;
140                          case "Line":
141                              s = new MyLine();
142                              break;
143                          default:
144                              throw new InvalidDataException("Unknown shape
                        kind: " + kind);
145                      }
146                      s.LoadFrom(reader);
```

```
147                         AddShape(s);
148                     }
149                 }
150                 finally
151                 {
152                     reader.Close();
153                 }
154             }
155         }
156 }
157
```

```
 1  using System;
 2  using System.IO;
 3  using SplashKitSDK;
 4
 5  namespace MyGame
 6  {
 7      public class MyCircle:Shape
 8      {
 9          private int _radius;
10
11          public MyCircle() : this(Color.Blue, 0.0f, 0.0f, 50+73)
12          {
13
14          }
15
16          public MyCircle(Color color, float x, float y, int radius) : base
              (color)
17          {
18              _radius = radius;
19          }
20
21          public int Radius
22          {
23              get
24              {
25                  return _radius;
26              }
27              set
28              {
29                  _radius = value;
30              }
31          }
32
33          public override void Draw()
34          {
35              if(Selected)
36              {
37                  DrawOutline();
38              }
39              SplashKit.FillCircle(Color, X, Y, _radius);
40          }
41
42          public override void DrawOutline()
43          {
44              SplashKit.FillCircle(Color.Black, X, Y, _radius + 2);
45          }
46
47          public override bool IsAt(Point2D pt)
48          {
```

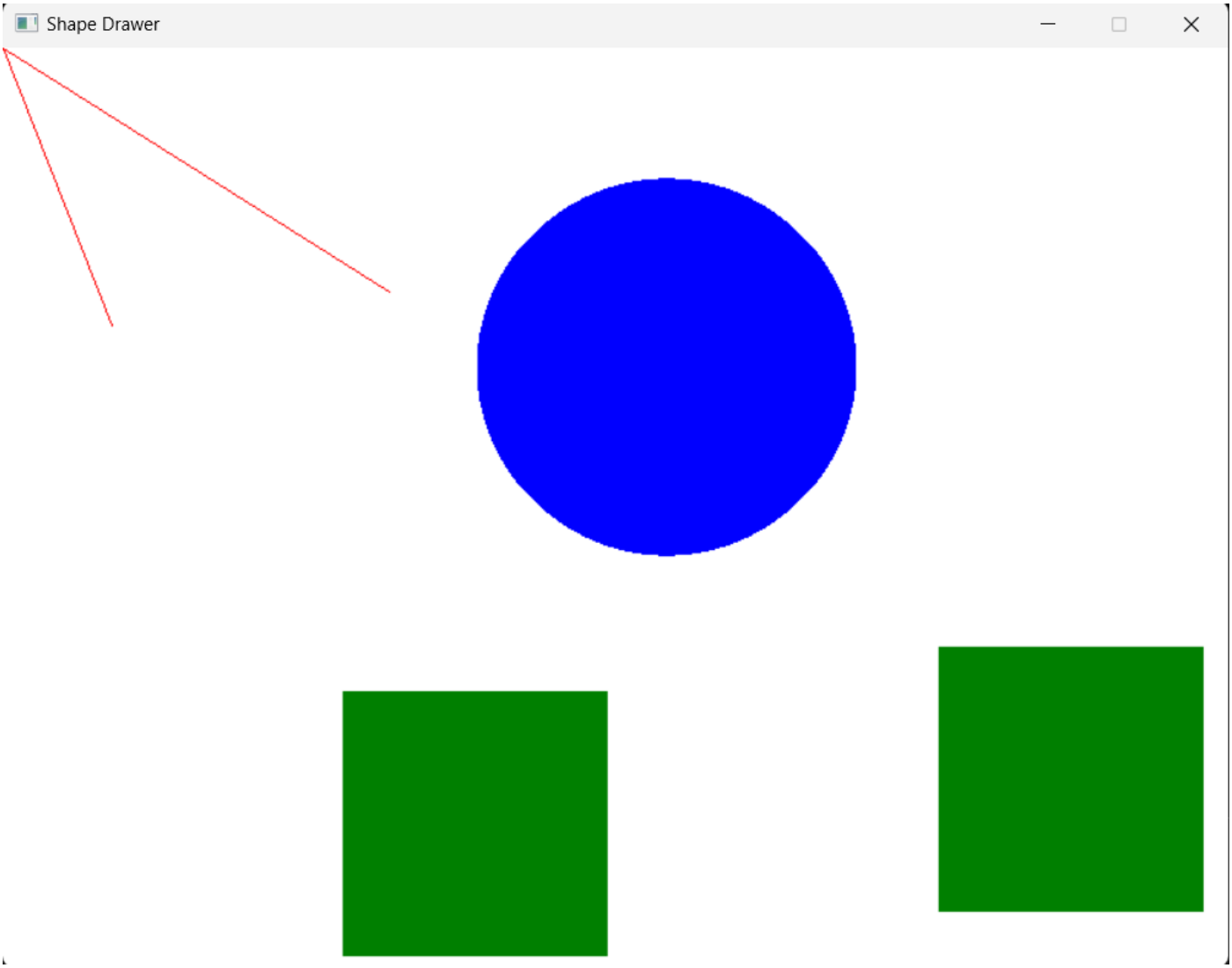
```
49              if (Math.Abs(pt.X - X) < _radius && Math.Abs(pt.Y - Y) <
                  _radius)
50              {
51                  return true;
52              }
53              else return false;
54          }
55          public override void SaveTo(StreamWriter writer)
56          {
57              writer.WriteLine("Circle");
58              base.SaveTo(writer);
59              writer.WriteLine(_radius);
60          }
61
62          public override void LoadFrom(StreamReader reader)
63          {
64              base.LoadFrom(reader);
65              _radius = reader.ReadInteger();
66          }
67      }
68  }
69
```

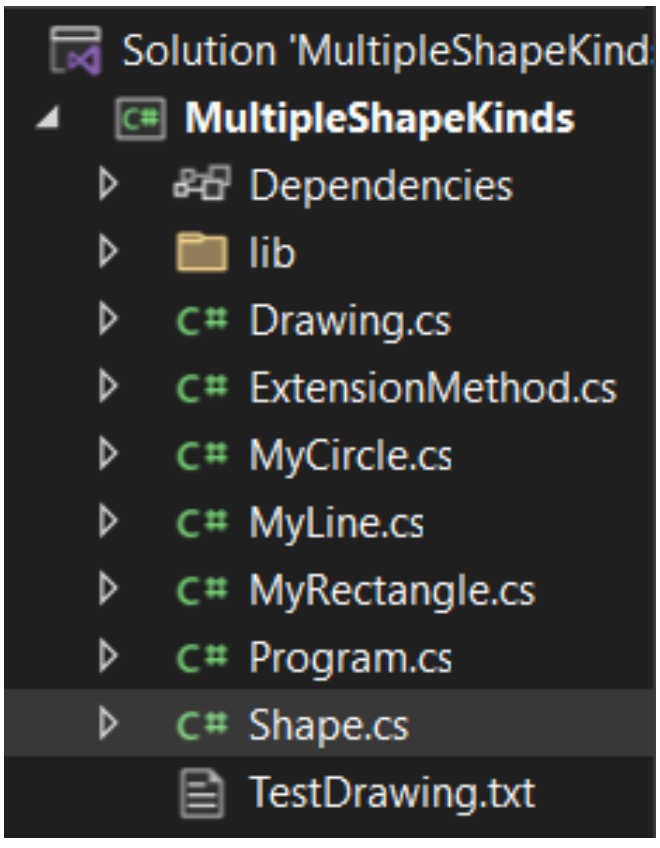
```
 1  using System;
 2  using System.IO;
 3  using SplashKitSDK;
 4
 5  namespace MyGame
 6  {
 7      public class MyLine:Shape
 8      {
 9          private float _endX;
10          private float _endY;
11
12          public MyLine() : this(Color.Red, 0.0f, 0.0f, 10, 10)
13          {
14
15          }
16
17          public MyLine(Color color, float startX, float startY, float endX,
              float endY) : base(color)
18          {
19              X = startX;
20              Y = startY;
21          }
22
23          public float EndX
24          {
25              get
26              {
27                  return _endX;
28              }
29              set
30              {
31                  _endX = value;
32              }
33          }
34
35          public float EndY
36          {
37              get
38              {
39                  return _endY;
40              }
41              set
42              {
43                  _endY = value;
44              }
45          }
46
47          public override void Draw()
48          {
```

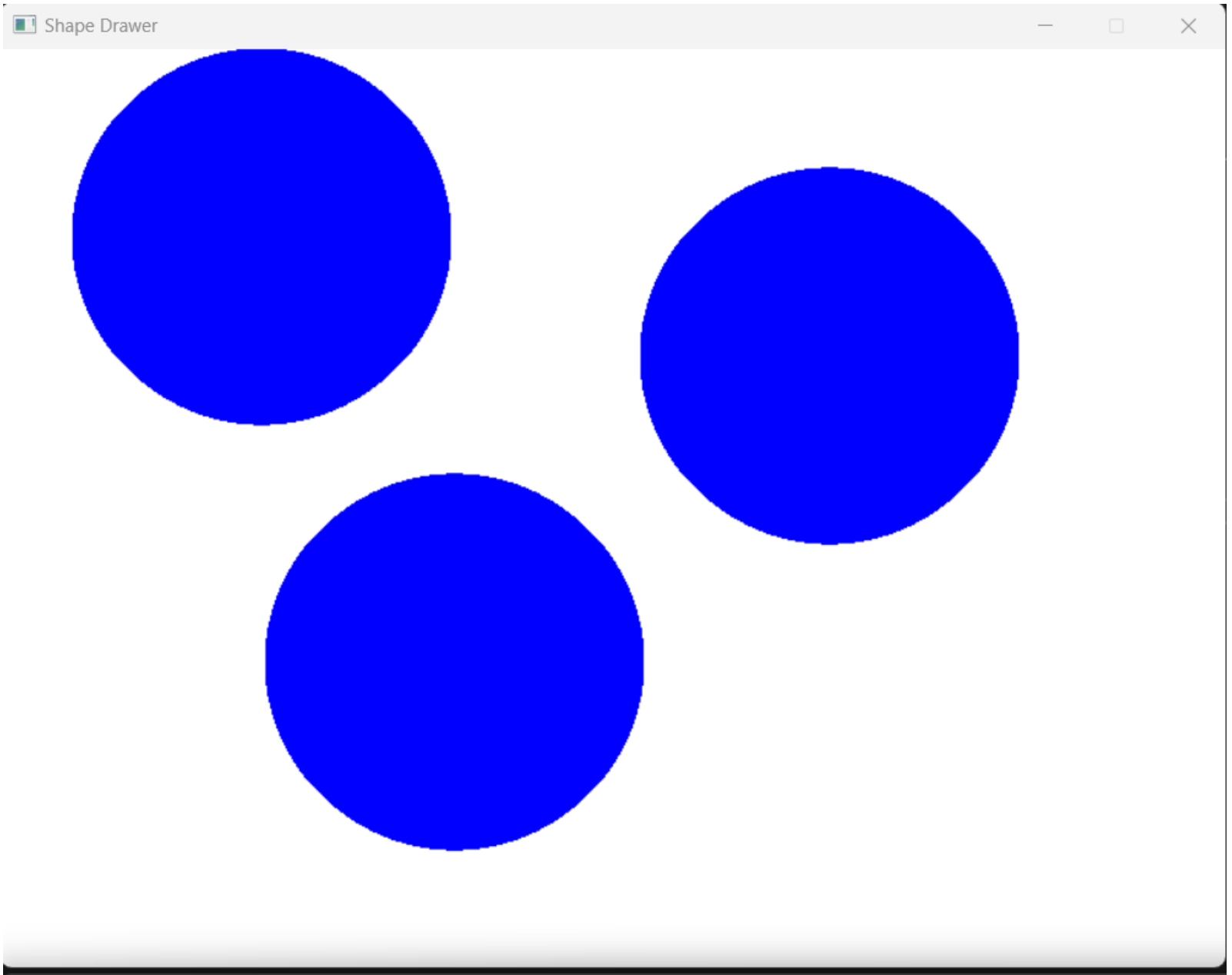
```
49              if (Selected)
50              {
51                  DrawOutline();
52              }
53              SplashKit.DrawLine(Color, X, Y, _endX, _endY);
54          }
55
56          public override void DrawOutline()
57          {
58              SplashKit.FillCircle(Color.Black, X, Y, 2);
59              SplashKit.FillCircle(Color.Black, _endX, _endY, 2);
60          }
61
62          public override bool IsAt(Point2D pt)
63          {
64              if (SplashKit.PointOnLine(pt, SplashKit.LineFrom(X, Y, _endX,
                  _endY)))
65              {
66                  return true;
67              }
68              else return false;
69          }
70
71          public override void SaveTo(StreamWriter writer)
72          {
73              writer.WriteLine("Line");
74              base.SaveTo(writer);
75              writer.WriteLine(_endX);
76              writer.WriteLine(_endY);
77          }
78
79          public override void LoadFrom(StreamReader reader)
80          {
81              base.LoadFrom(reader);
82              _endX = reader.ReadInteger();
83              _endY = reader.ReadInteger();
84          }
85      }
86  }
87
```

```
using SplashKitSDK;
using System;
using System.IO;

namespace MyGame
{
    public class MyRectangle:Shape
    {
        private int _width;
        private int _height;
        public MyRectangle() : this(Color.Green, 0.0f, 0.0f, 173, 173)
        {

        }

        public MyRectangle(Color color, float x, float y, int width, int
          height) : base(color)
        {
            X = x;
            Y = y;
            _width = width;
            _height = height;
        }

        public int Width
        {
            get
            {
                return _width;
            }
            set
            {
                _width = value;
            }
        }

        public int Height
        {
            get
            {
                return _height;
            }
            set
            {
                _height = value;
            }
        }

        public override void Draw()
```

```
49          {
50              if (Selected)
51              {
52                  DrawOutline();
53              }
54              SplashKit.FillRectangle(Color, X, Y, _width, _height);
55          }
56
57          public override void DrawOutline()
58          {
59              SplashKit.FillRectangle(Color.Black , X - (5 + 3), Y - (5 + 3),
                    _width + 2 * (5 + 3), _height + 2 * (5 + 3));
60          }
61
62          public override bool IsAt(Point2D pt)
63          {
64              if (pt.X > X && pt.X < X + _width && pt.Y > Y && pt.Y < Y +
                  _height)
65              {
66                  return true;
67              }
68              else
69              {
70                  return false;
71              }
72          }
73
74          public override void SaveTo(StreamWriter writer)
75          {
76              writer.WriteLine("Rectangle");
77              base.SaveTo(writer);
78              writer.WriteLine(_width);
79              writer.WriteLine(_height);
80          }
81
82          public override void LoadFrom(StreamReader reader)
83          {
84              base.LoadFrom(reader);
85              _width = reader.ReadInteger();
86              _height = reader.ReadInteger();
87          }
88      }
89  }
90
```

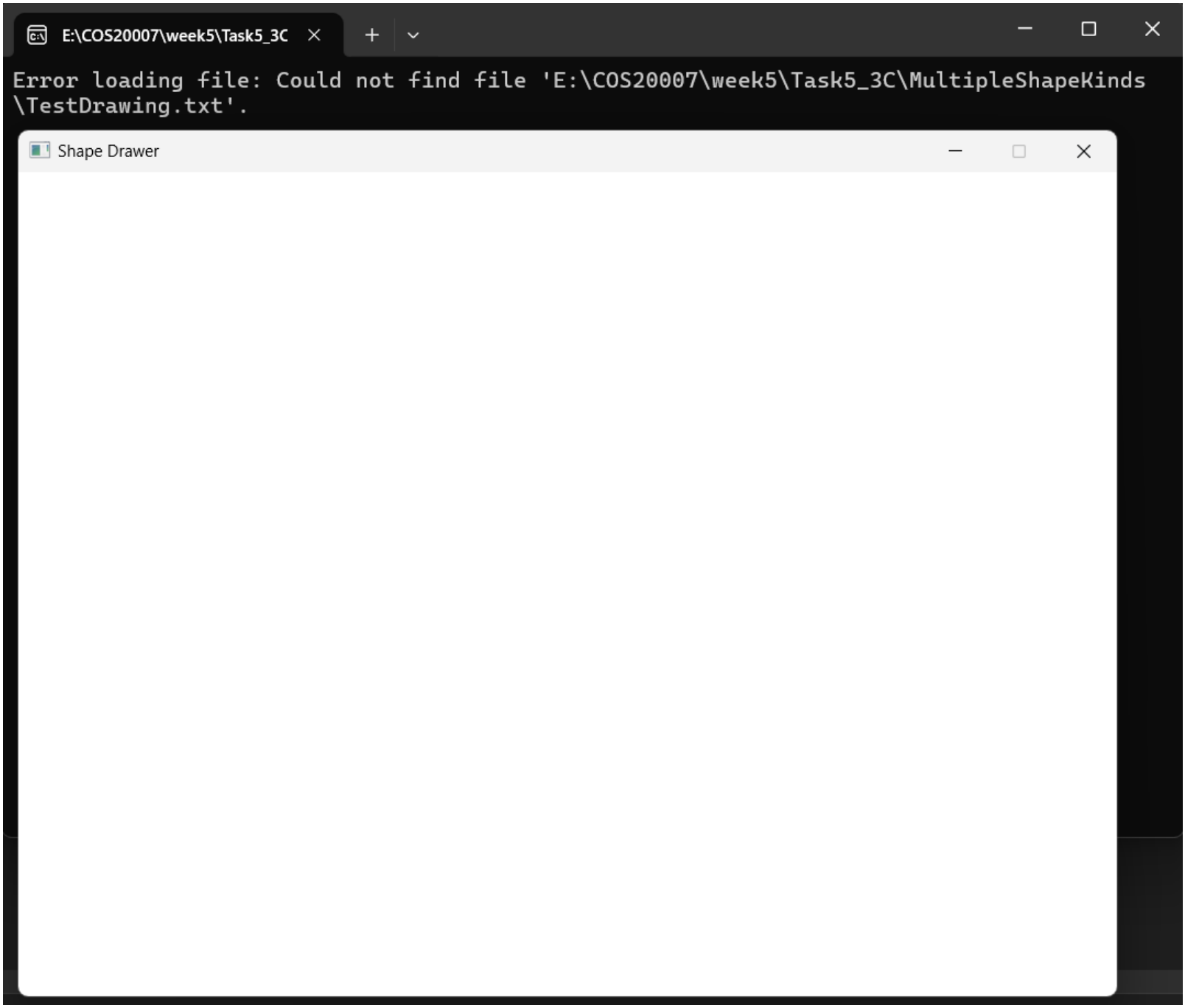

Shape Drawer

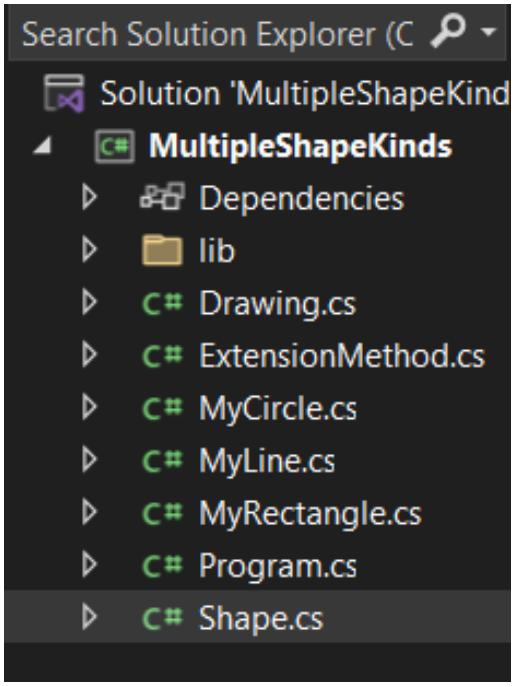

Solution 'MultipleShapeKind
MultipleShapeKinds
Dependencies
lib
Drawing.cs
ExtensionMethod.cs
MyCircle.cs
MyLine.cs
MyRectangle.cs
Program.cs
Shape.cs
TestDrawing.txt

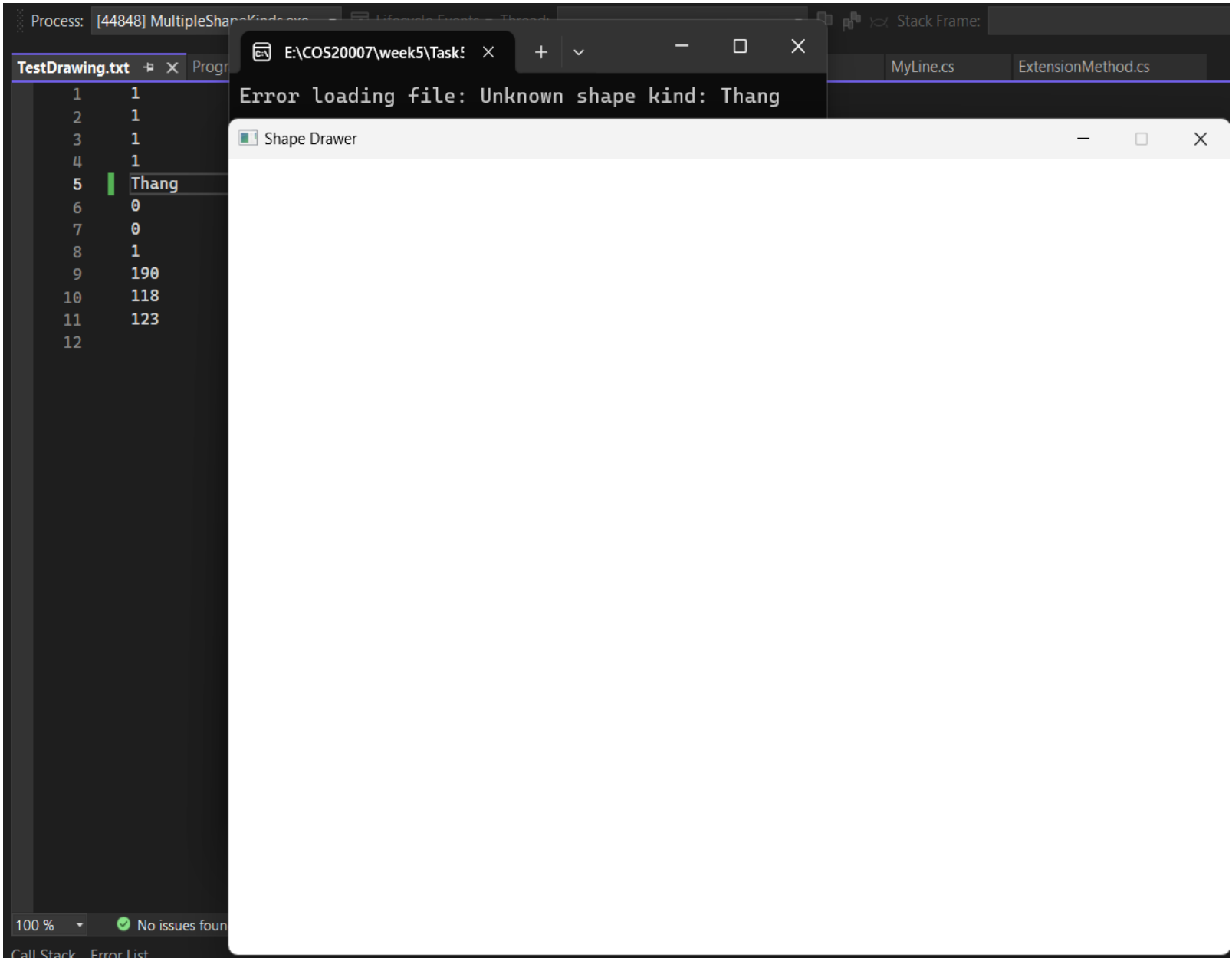


E:\COS20007\week5\Task5_3C

```
Drawing saved to E:/COS20007/week5/Task5_3C/MultipleShapeKinds/TestDrawing.txt
```


E:\COS20007\week5\Task5_3C
Error loading file: Could not find file 'E:\COS20007\week5\Task5_3C\MultipleShapeKinds
\TestDrawing.txt'.
Shape Drawer


Search Solution Explorer (C
Solution 'MultipleShapeKind
MultipleShapeKinds
Dependencies
lib
Drawing.cs
ExtensionMethod.cs
MyCircle.cs
MyLine.cs
MyRectangle.cs
Program.cs
Shape.cs


Process: [44848] MultipleShapeKinds.exe
Stack Frame:
TestDrawing.txt
MyLine.cs
ExtensionMethod.cs
1
1
1
1
Thang
0
0
1
190
118
123
E:\COS20007\week5\Task5
Error loading file: Unknown shape kind: Thang
Shape Drawer
100 %
No issues found
Call Stack
Error List