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

namespace ShapeCalculator
{
    public class Program
    {
        private enum ShapeKind
        {
            Square,
            Triangle,
            Circle,
            Trapezium,
            Rectangle
        }

        public static void Main()
        {
            // WIll create a directory for the saved drawings
            Directory.CreateDirectory("SavedDrawings");

            while (true)
            {
                int choice = GetValidatedChoice();

                // Exit the program if 7 is entered
                if (choice == 7)
                {
                    break;
                }

                // Load a saved shape if 6 is entered
                if (choice == 6)
                {
                    Console.Write("Enter the filename to load: ");
                    string fileName = Console.ReadLine();
                    ShapeLoader loader = new
                     ShapeLoader($"SavedDrawings/{fileName}.txt");
                    Shape loadedShape = loader.LoadShape();
                    if (loadedShape != null)
                    {
                        Console.WriteLine($"Loaded shape:
                         {loadedShape.GetType().Name}");
                        Console.WriteLine($"Area:
                         {loadedShape.CalculateArea()}");
                        Console.WriteLine($"Perimeter:
                         {loadedShape.CalculatePerimeter()}");
                        DrawShape(loadedShape);
                    }
                    else
                    {
                        Console.WriteLine("Failed to load shape.");
                    }
```

```
            continue; // Skip the rest of the loop and ask for
             choice again
        }

        // Create a shape based on the user's choice
        ShapeKind kind = (ShapeKind)(choice - 1);
        Shape shape = CreateShape(kind);

        if (shape != null)
        {
            Console.WriteLine($"Area: {shape.CalculateArea()}");
            Console.WriteLine($"Perimeter:
             {shape.CalculatePerimeter()}");

            Console.WriteLine("Do you want a visual representation?
             (yes/no)");
            string response = Console.ReadLine();

            // Draw the shape if the user wants a visual
             representation
            if (response.ToLower() == "yes")
            {
                DrawShape(shape);
            }

            // Save the shape if the user wants to save it
            Console.WriteLine("Do you want to save the shape?
             (yes/no)");
            response = Console.ReadLine();

            if (response.ToLower() == "yes")
            {
                Console.Write("Enter the filename to save: ");
                string fileName = Console.ReadLine();
                ShapeSaver saver = new
                 ShapeSaver($"SavedDrawings/{fileName}.txt");
                saver.SaveShape(shape);
                Console.WriteLine("Shape saved successfully.");
            }
        }
    }
}

// Get a validated choice from the user
private static int GetValidatedChoice()
{
    int choice;
    while (true)
    {
        Console.WriteLine("Choose a shape to create:");
        Console.WriteLine("1. Square");
        Console.WriteLine("2. Triangle");
        Console.WriteLine("3. Circle");
        Console.WriteLine("4. Trapezium");
        Console.WriteLine("5. Rectangle");
```

```
            Console.WriteLine("6. Load a shape");
            Console.WriteLine("7. Exit");

            if (int.TryParse(Console.ReadLine(), out choice) && choice
             >= 1 && choice <= 7)
            {
                return choice;
            }
            else
            {
                Console.WriteLine("Invalid choice. Please enter a
                 number between 1 and 7.");
            }
        }
    }

    // Create a shape based on the user's choice
    private static Shape CreateShape(ShapeKind kind)
    {
        Shape shape = null;
        string color = InputValidator.GetValidatedColor("Enter the
         color of the shape: ");
        string unit = InputValidator.GetValidatedUnit("Enter the unit
         (inches/centimeters): ");
        bool isInches = unit == "inches";

        // Assign propertires to the shape based on its type
        switch (kind)
        {
            case ShapeKind.Square:
                shape = new Square();
                double sideLength =
                 InputValidator.GetValidatedDouble("Enter the side
                 length: ");
                ((Square)shape).SideLength = isInches ?
                 UnitConverter.InchesToCentimeters(sideLength) :
                 sideLength;
                break;

            case ShapeKind.Triangle:
                shape = new Triangle();
                double baseLength =
                 InputValidator.GetValidatedDouble("Enter the base
                 length: ");
                double height =
                 InputValidator.GetValidatedDouble("Enter the height:
                 ");
                ((Triangle)shape).Base = isInches ?
                 UnitConverter.InchesToCentimeters(baseLength) :
                 baseLength;
                ((Triangle)shape).Height = isInches ?
                 UnitConverter.InchesToCentimeters(height) : height;
                break;

            case ShapeKind.Circle:
```

```
            shape = new Circle();
            double radius =
             InputValidator.GetValidatedDouble("Enter the radius:
             ");
            ((Circle)shape).Radius = isInches ?
             UnitConverter.InchesToCentimeters(radius) : radius;
            break;

        case ShapeKind.Trapezium:
            shape = new Trapezium();
            double base1 = InputValidator.GetValidatedDouble("Enter
             the first base length: ");
            double base2 = InputValidator.GetValidatedDouble("Enter
             the second base length: ");
            double trapeziumHeight =
             InputValidator.GetValidatedDouble("Enter the height:
             ");
            ((Trapezium)shape).Base1 = isInches ?
             UnitConverter.InchesToCentimeters(base1) : base1;
            ((Trapezium)shape).Base2 = isInches ?
             UnitConverter.InchesToCentimeters(base2) : base2;
            ((Trapezium)shape).Height = isInches ?
             UnitConverter.InchesToCentimeters(trapeziumHeight) :
             trapeziumHeight;
            break;

        case ShapeKind.Rectangle:
            shape = new Rectangle();
            double width = InputValidator.GetValidatedDouble("Enter
             the width: ");
            double rectangleHeight =
             InputValidator.GetValidatedDouble("Enter the height:
             ");
            ((Rectangle)shape).Width = isInches ?
             UnitConverter.InchesToCentimeters(width) : width;
            ((Rectangle)shape).Height = isInches ?
             UnitConverter.InchesToCentimeters(rectangleHeight) :
             rectangleHeight;
            break;
    }

    // Set the color of the shape
    if (shape != null)
    {
        shape.Color = color.ToColor();
        //Console.WriteLine($"Shape color set to:
         {SplashKit.ColorToString(shape.Color)}"); // Debug print
    }

    return shape;
}

// Draws the shape on a Splashkit window
private static void DrawShape(Shape shape)
{
```

```
            Window window = new Window("Shape Drawer", 800, 600);
            while (!window.CloseRequested)
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen(Color.White);

                shape.Draw();
                SplashKit.RefreshScreen();
            }
            window.Close();
        }
    }
}
```