Bryan Zublin
Assignment #3
CS 420 Section 1001

Assignment #3 - OOP in MATLAB

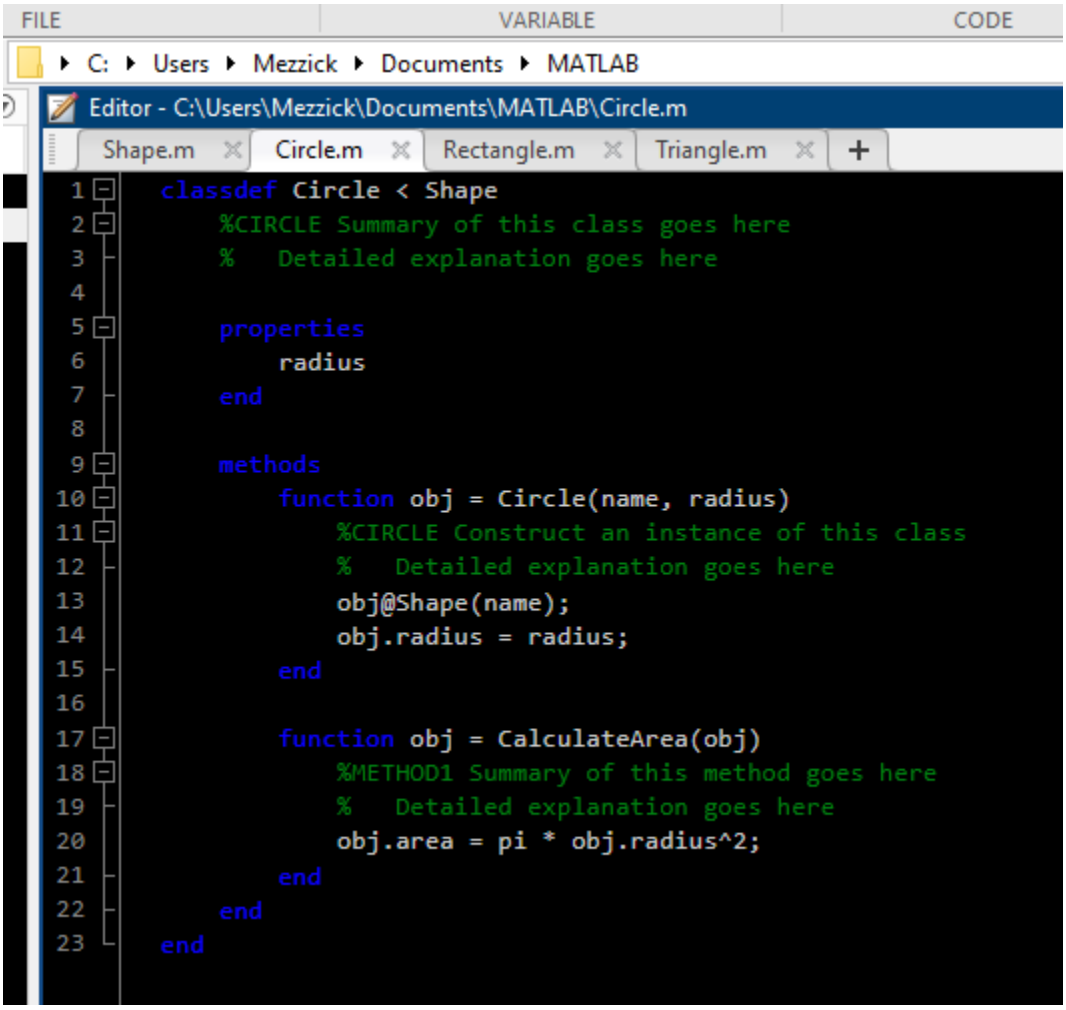
For this assignment, I was able to complete it with Justin Messick from the Monday/Wednesday section. Our contribution was about 50-50. I referred to online MATLAB documentation to learn about the builtin functions for printing and receiving user input, such as disp() and input().

Task 1

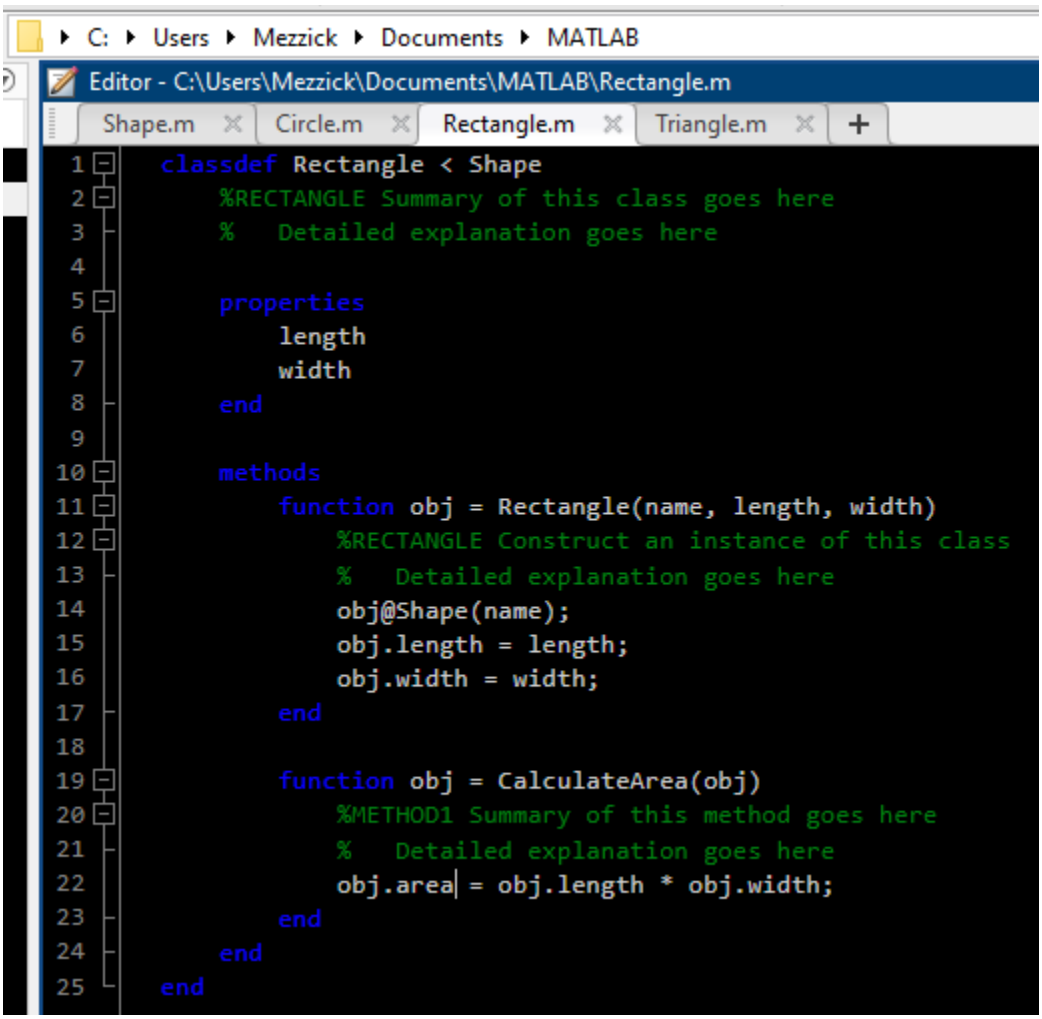
```
classdef Shape
    %SHAPE Summary of this class goes here
    %   Detailed explanation goes here

    properties
        Name
        Area
    end

    methods
        function obj = Shape(name)
            %SHAPE Construct an instance of this class
            %   Detailed explanation goes here
            obj.Name = name;
        end

        function obj = Display(obj)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            fprintf('Name: %s\n', obj.Name);
            fprintf('Area: %f\n', obj.Area);
        end
    end
end
```

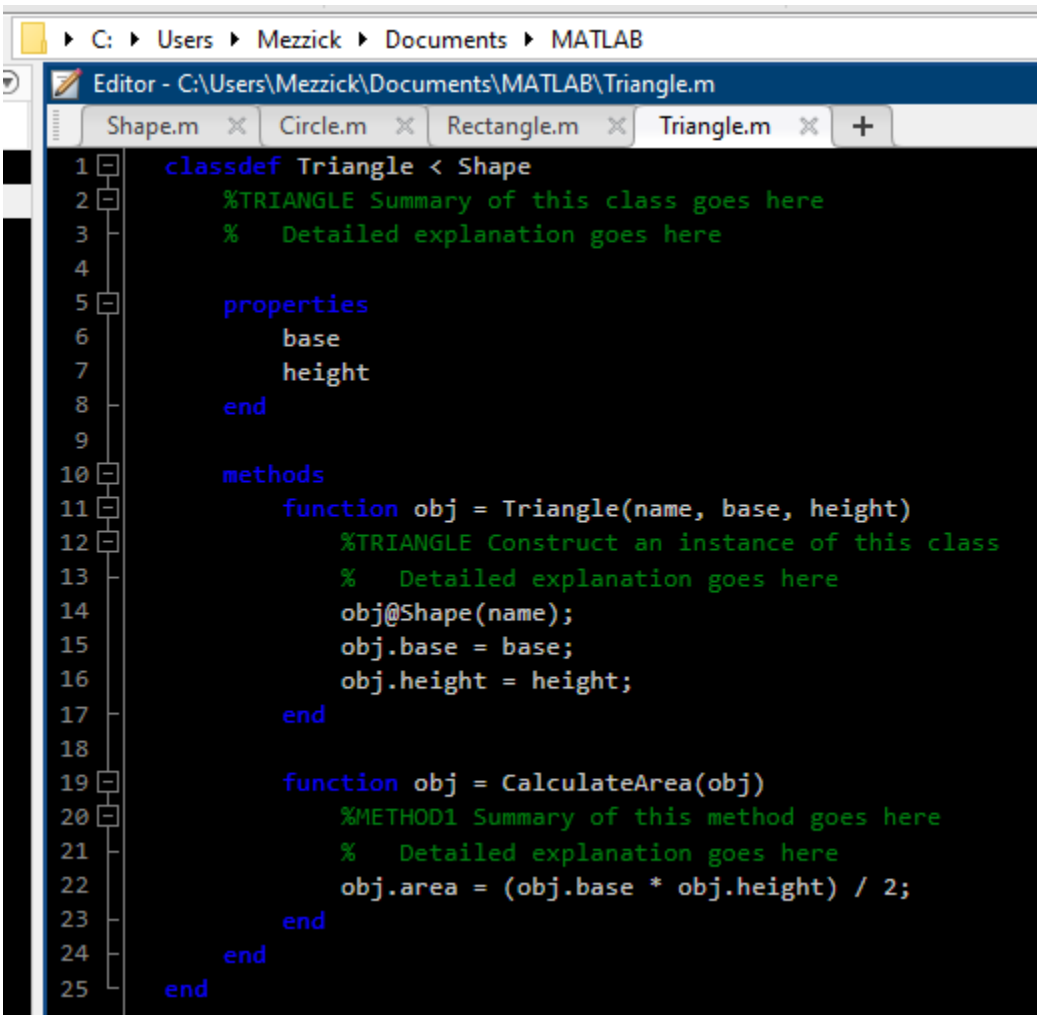
```
>> shape1.Display()
Name: Triangle
Area:

ans =

  Shape with properties:

    Name: "Triangle"
    Area: []
```

Task 2



Circle.m


C: ▸ Users ▸ Mezzick ▸ Documents ▸ MATLAB
Editor - C:\Users\Mezzick\Documents\MATLAB\Rectangle.m
Shape.m
Circle.m
Rectangle.m
Triangle.m
classdef Rectangle < Shape
%RECTANGLE Summary of this class goes here
% Detailed explanation goes here
properties
length
width
end
methods
function obj = Rectangle(name, length, width)
%RECTANGLE Construct an instance of this class
% Detailed explanation goes here
obj@Shape(name);
obj.length = length;
obj.width = width;
end
function obj = CalculateArea(obj)
%METHOD1 Summary of this method goes here
% Detailed explanation goes here
obj.area = obj.length * obj.width;
end
end
end
Rectangle.m


C: ▸ Users ▸ Mezzick ▸ Documents ▸ MATLAB
Editor - C:\Users\Mezzick\Documents\MATLAB\Triangle.m
Shape.m
Circle.m
Rectangle.m
Triangle.m
classdef Triangle < Shape
%TRIANGLE Summary of this class goes here
% Detailed explanation goes here
properties
base
height
end
methods
function obj = Triangle(name, base, height)
%TRIANGLE Construct an instance of this class
% Detailed explanation goes here
obj@Shape(name);
obj.base = base;
obj.height = height;
end
function obj = CalculateArea(obj)
%METHOD1 Summary of this method goes here
% Detailed explanation goes here
obj.area = (obj.base * obj.height) / 2;
end
end
end
Triangle.m

```
>> circle1=Circle("circle1", 2),circle2=Circle("circle2", 4),
circle1=CalculateArea(circle1),circle2=CalculateArea(circle2),
rectangle1=Rectangle("rectangle1", 2, 3),rectangle2=Rectangle("rectangle2", 4, 6),
rectangle1=CalculateArea(rectangle1),rectangle2=CalculateArea(rectangle2),
triangle1=Triangle("triangle1", 2, 4),triangle2=Triangle("triangle2", 6, 8),
triangle1=CalculateArea(triangle1),triangle2=CalculateArea(triangle2)

circle1 =

  Circle with properties:

    radius: 2
      name: "circle1"
      area: []


circle2 =

  Circle with properties:

    radius: 4
      name: "circle2"
      area: []


circle1 =

  Circle with properties:

    radius: 2
      name: "circle1"
      area: 12.5664


circle2 =

  Circle with properties:

    radius: 4
      name: "circle2"
      area: 50.2655
```

```
rectangle1 =

  Rectangle with properties:

    length: 2
     width: 3
      name: "rectangle1"
      area: []


rectangle2 =

  Rectangle with properties:

    length: 4
     width: 6
      name: "rectangle2"
      area: []


rectangle1 =

  Rectangle with properties:

    length: 2
     width: 3
      name: "rectangle1"
      area: 6


rectangle2 =

  Rectangle with properties:

    length: 4
     width: 6
      name: "rectangle2"
      area: 24
```

```
triangle1 =

  Triangle with properties:

      base: 2
    height: 4
      name: "triangle1"
      area: []


triangle2 =

  Triangle with properties:

      base: 6
    height: 8
      name: "triangle2"
      area: []


triangle1 =

  Triangle with properties:

      base: 2
    height: 4
      name: "triangle1"
      area: 4


triangle2 =

  Triangle with properties:

      base: 6
    height: 8
      name: "triangle2"
      area: 24
```

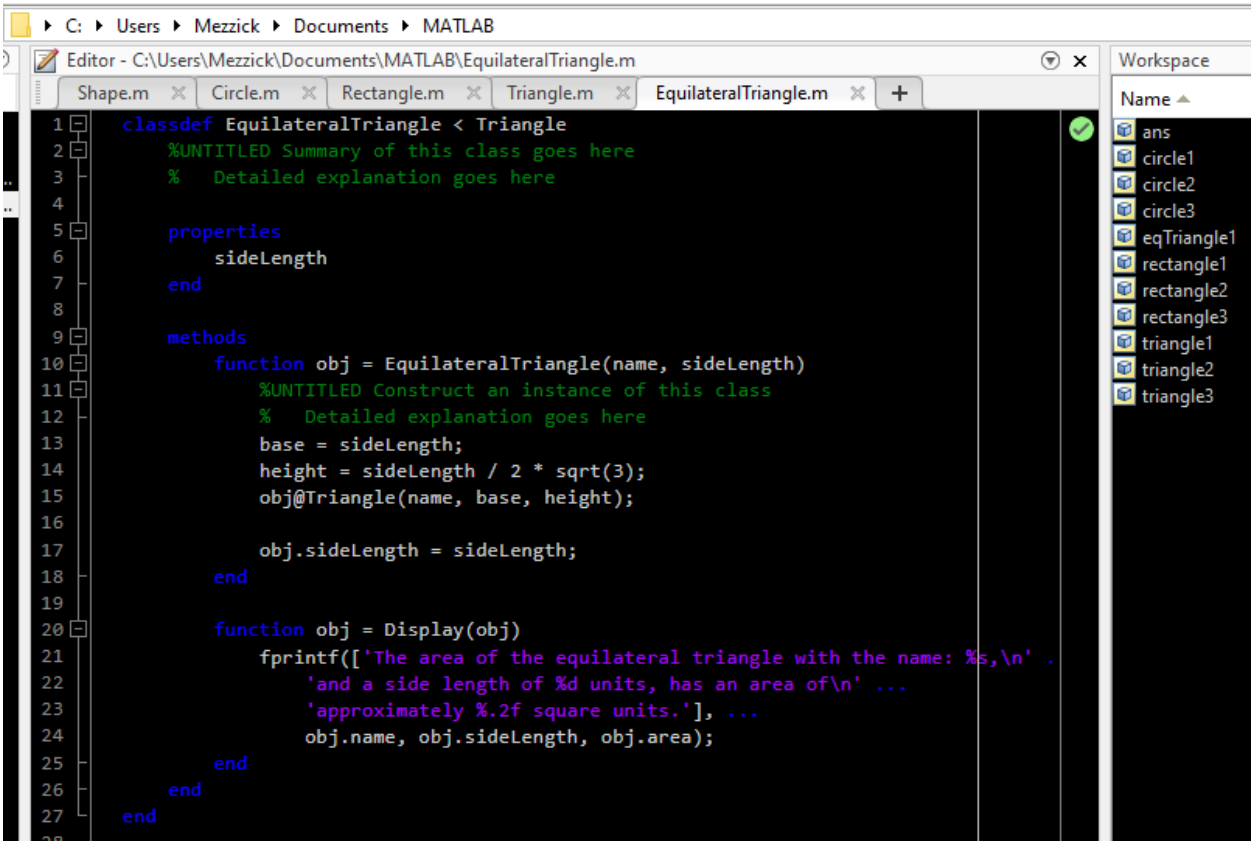
Task 3

```
The area of the circle with name: circle3,
and radius: 8 units, is approximately 201.06
square units.
ans =

  Circle with properties:

    radius: 8
      name: "circle3"
      area: 201.0619
```

```
The area of the rectangle with name: rectangle3,
length: 6 units, and width: 8 units, is
approximately 48.00 square units.
ans =

  Rectangle with properties:

    length: 6
     width: 8
      name: "rectangle3"
      area: 48
```

```
The area of the triangle with name: triangle3,
base: 4 units, and height: 8 units, is
approximately 16.00 square units.
ans =

  Triangle with properties:

      base: 4
    height: 8
      name: "triangle3"
      area: 16
```

Task 4




```
classdef EquilateralTriangle < Triangle
    %UNTITLED Summary of this class goes here
    %   Detailed explanation goes here

    properties
        sideLength
    end

    methods
        function obj = EquilateralTriangle(name, sideLength)
            %UNTITLED Construct an instance of this class
            %   Detailed explanation goes here
            base = sideLength;
            height = sideLength / 2 * sqrt(3);
            obj@Triangle(name, base, height);

            obj.sideLength = sideLength;
        end

        function obj = Display(obj)
            fprintf(['The area of the equilateral triangle with the name: %s,\n' ...
                'and a side length of %d units, has an area of\n' ...
                'approximately %.2f square units.'], ...
                obj.name, obj.sideLength, obj.area);
        end
    end
end
```

EquilateralTriangle.m

```
>> eqTriangle1=EquilateralTriangle("eqTriangle1", 10),
eqTriangle1=CalculateArea(eqTriangle1),
eqTriangle1.Display

eqTriangle1 =

  EquilateralTriangle with properties:

    sideLength: 10
          base: 10
        height: 8.6603
          name: "eqTriangle1"
          area: []


eqTriangle1 =

  EquilateralTriangle with properties:

    sideLength: 10
          base: 10
        height: 8.6603
          name: "eqTriangle1"
          area: 43.3013

The area of the equilateral triangle with the name: eqTriangle1,
and a side length of 10 units, has an area of
approximately 43.30 square units.
ans =

  EquilateralTriangle with properties:

    sideLength: 10
          base: 10
        height: 8.6603
          name: "eqTriangle1"
          area: 43.3013
```

Task 5

```
classdef Circle < Shape & ColorMixin
    %CIRCLE Summary of this class goes here
    %   Detailed explanation goes here

    properties
        radius
    end

    methods
        function obj = Circle(name, radius, color)
            %CIRCLE Construct an instance of this class
            %   Detailed explanation goes here
            obj@Shape(name);
            obj@ColorMixin(color);
            obj.radius = radius;
        end

        function obj = CalculateArea(obj)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            obj.area = pi * obj.radius^2;
        end

        function obj = Display(obj)
            fprintf(['The area of the %s circle with name: %s,\n' ...
                'and radius: %d units, is approximately %.2f\n' ...
                'square units.'], obj.color, obj.name, obj.radius, obj.area);
        end
    end
end
```

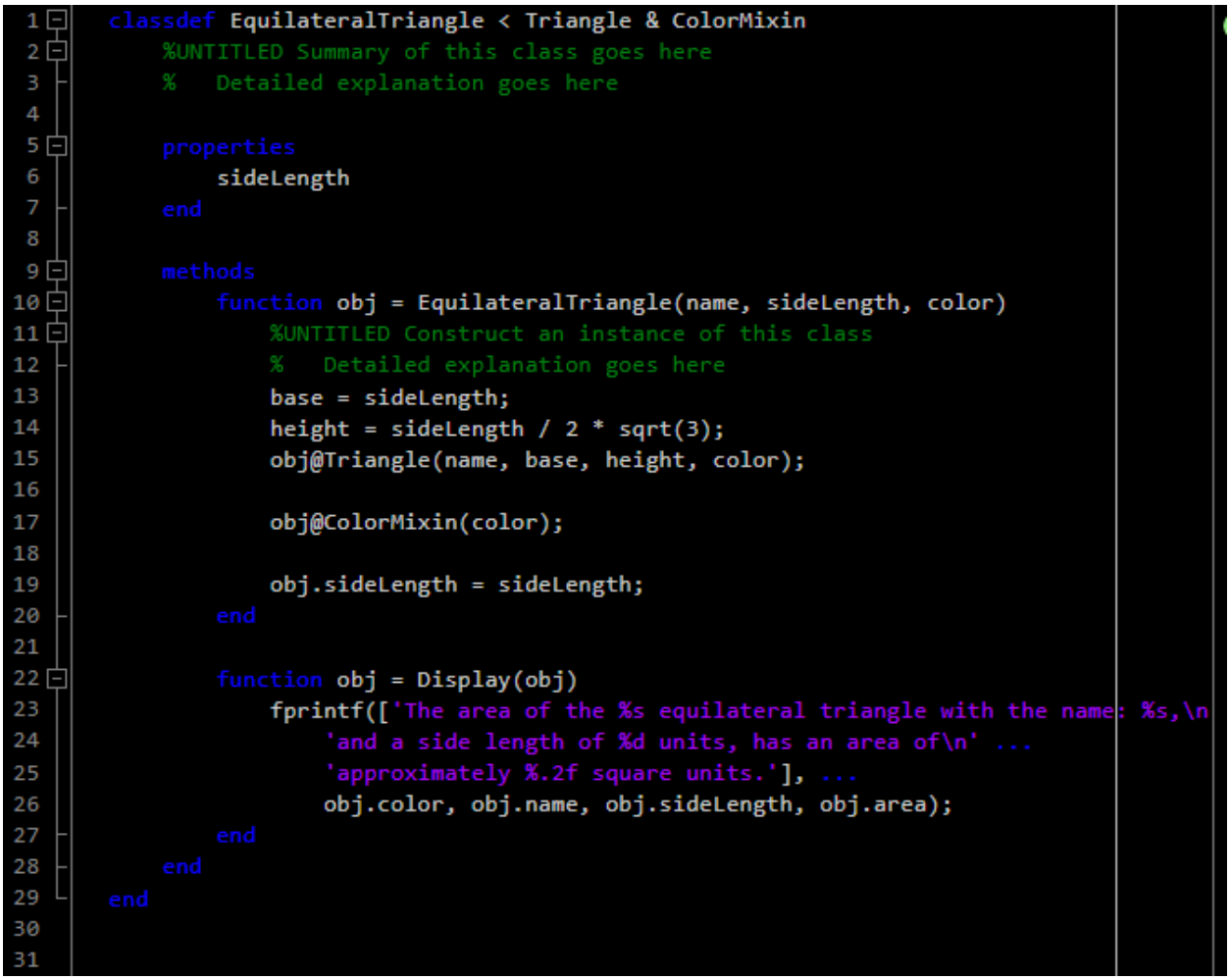
Circle.m

```
classdef Rectangle < Shape & ColorMixin
    %RECTANGLE Summary of this class goes here
    %   Detailed explanation goes here

    properties
        length
        width
    end

    methods
        function obj = Rectangle(name, length, width, color)
            %RECTANGLE Construct an instance of this class
            %   Detailed explanation goes here
            obj@Shape(name);
            obj@ColorMixin(color);
            obj.length = length;
            obj.width = width;
        end

        function obj = CalculateArea(obj)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            obj.area = obj.length * obj.width;
        end

        function obj = Display(obj)
            fprintf(['The area of the %s rectangle with name: %s,\n' ...
                'length: %d units, and width: %d units, is\n' ...
                'approximately %.2f square units.'], ...
                obj.color, obj.name, obj.length, obj.width, obj.area);
        end
    end
end
```

Rectangle.m

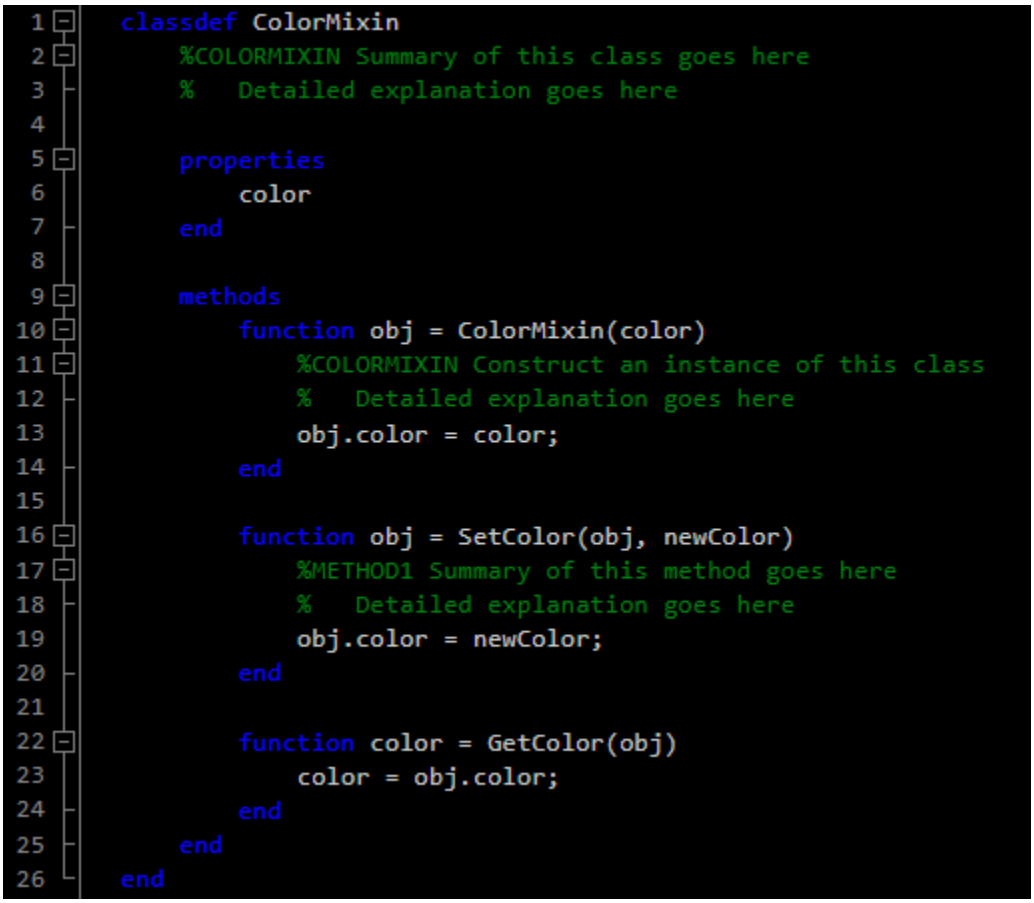
```
classdef Triangle < Shape & ColorMixin
    %TRIANGLE Summary of this class goes here
    %   Detailed explanation goes here

    properties
        base
        height
    end

    methods
        function obj = Triangle(name, base, height, color)
            %TRIANGLE Construct an instance of this class
            %   Detailed explanation goes here
            obj@Shape(name);
            obj@ColorMixin(color);
            obj.base = base;
            obj.height = height;
        end

        function obj = CalculateArea(obj)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            obj.area = (obj.base * obj.height) / 2;
        end

        function obj = Display(obj)
            fprintf(['The area of the %s triangle with name: %s,\n' ...
                'base: %d units, and height: %d units, is\n' ...
                'approximately %.2f square units.'], ...
                obj.color, obj.name, obj.base, obj.height, obj.area);
        end
    end
end
```

Triangle.m



```
classdef EquilateralTriangle < Triangle & ColorMixin
    %UNTITLED Summary of this class goes here
    %   Detailed explanation goes here

    properties
        sideLength
    end

    methods
        function obj = EquilateralTriangle(name, sideLength, color)
            %UNTITLED Construct an instance of this class
            %   Detailed explanation goes here
            base = sideLength;
            height = sideLength / 2 * sqrt(3);
            obj@Triangle(name, base, height, color);

            obj@ColorMixin(color);

            obj.sideLength = sideLength;
        end

        function obj = Display(obj)
            fprintf(['The area of the %s equilateral triangle with the name: %s,\n
                'and a side length of %d units, has an area of\n' ...
                'approximately %.2f square units.'], ...
                obj.color, obj.name, obj.sideLength, obj.area);
        end
    end
end
```

EquilateralTriangle.m



```
classdef ColorMixin
    %COLORMIXIN Summary of this class goes here
    %   Detailed explanation goes here

    properties
        color
    end

    methods
        function obj = ColorMixin(color)
            %COLORMIXIN Construct an instance of this class
            %   Detailed explanation goes here
            obj.color = color;
        end

        function obj = SetColor(obj, newColor)
            %METHOD1 Summary of this method goes here
            %   Detailed explanation goes here
            obj.color = newColor;
        end

        function color = GetColor(obj)
            color = obj.color;
        end
    end
end
```

ColorMixin.m

```
The area of the blue circle with name: circle5,
and radius: 8 units, is approximately 201.06
square units.
ans =

  Circle with properties:

    radius: 8
      name: "circle5"
      area: 201.0619
     color: "blue"

The area of the green rectangle with name: rectangle5,
length: 6 units, and width: 8 units, is
approximately 48.00 square units.
ans =

  Rectangle with properties:

    length: 6
     width: 8
      name: "rectangle5"
      area: 48
     color: "green"
```

```
The area of the red triangle with name: triangle5,
base: 4 units, and height: 8 units, is
approximately 16.00 square units.
ans =

  Triangle with properties:

      base: 4
    height: 8
      name: "triangle5"
      area: 16
     color: "red"

The area of the purple equilateral triangle with the name: eqTriangle5,
and a side length of 8 units, has an area of
approximately 27.71 square units.
ans =

  EquilateralTriangle with properties:

    sideLength: 8
          base: 8
        height: 6.9282
          name: "eqTriangle5"
          area: 27.7128
         color: "purple"
```

After calling SetColor on each shape:

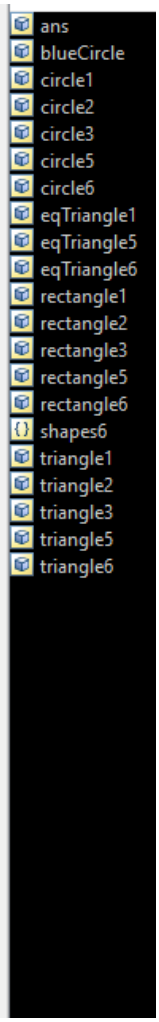
```
>> circle5.SetColor("red"),rectangle5.SetColor("blue"),
triangle5.SetColor("green"),eqTriangle5.SetColor("maroon")

circle5.Display,rectangle5.Display,triangle5.Display,eqTriangle5.Display

ans =

  Circle with properties:

    radius: 8
      name: "circle5"
      area: 201.0619
     color: "red"


ans =

  Rectangle with properties:

    length: 6
     width: 8
      name: "rectangle5"
      area: 48
     color: "blue"


ans =

  Triangle with properties:

      base: 4
    height: 8
      name: "triangle5"
      area: 16
     color: "green"
```

```
ans =

  EquilateralTriangle with properties:

    sideLength: 8
          base: 8
        height: 6.9282
          name: "eqTriangle5"
          area: 27.7128
         color: "maroon"

The area of the blue circle with name: circle5,
and radius: 8 units, is approximately 201.06
square units.
ans =

  Circle with properties:

    radius: 8
      name: "circle5"
      area: 201.0619
     color: "blue"

The area of the green rectangle with name: rectangle5,
length: 6 units, and width: 8 units, is
approximately 48.00 square units.
ans =

  Rectangle with properties:

    length: 6
     width: 8
      name: "rectangle5"
      area: 48
     color: "green"
```

```
The area of the red triangle with name: triangle5,
base: 4 units, and height: 8 units, is
approximately 16.00 square units.
ans =

  Triangle with properties:

      base: 4
    height: 8
      name: "triangle5"
      area: 16
     color: "red"

The area of the purple equilateral triangle with the name: eqTriangle5,
and a side length of 8 units, has an area of
approximately 27.71 square units.
ans =

  EquilateralTriangle with properties:

    sideLength: 8
          base: 8
        height: 6.9282
          name: "eqTriangle5"
          area: 27.7128
         color: "purple"
```

Task 6

```
classdef Shape

    properties
        name
        area
    end

    methods
        function obj = Shape(name)
            obj.name = name;
        end

        function obj = Display(obj)
            fprintf('Name: %s\n', obj.name);
            fprintf('Area: %f\n', obj.area);
        end
    end

    methods (Static)
        function CalculateStatistics(shapes)

            areas = zeros(1, numel(shapes));

            for i = 1:numel(shapes)
                areas(i) = shapes{i}.area;
            end

            meanArea = mean(areas);
            medianArea = median(areas);
            stdDevArea = std(areas);

            fprintf(['Mean area: %.2f\nMedian area: %.2f\n' ...
                'Standard deviation of areas: %.2f\n'] ...
                , meanArea, medianArea, stdDevArea);
        end
    end
end
```

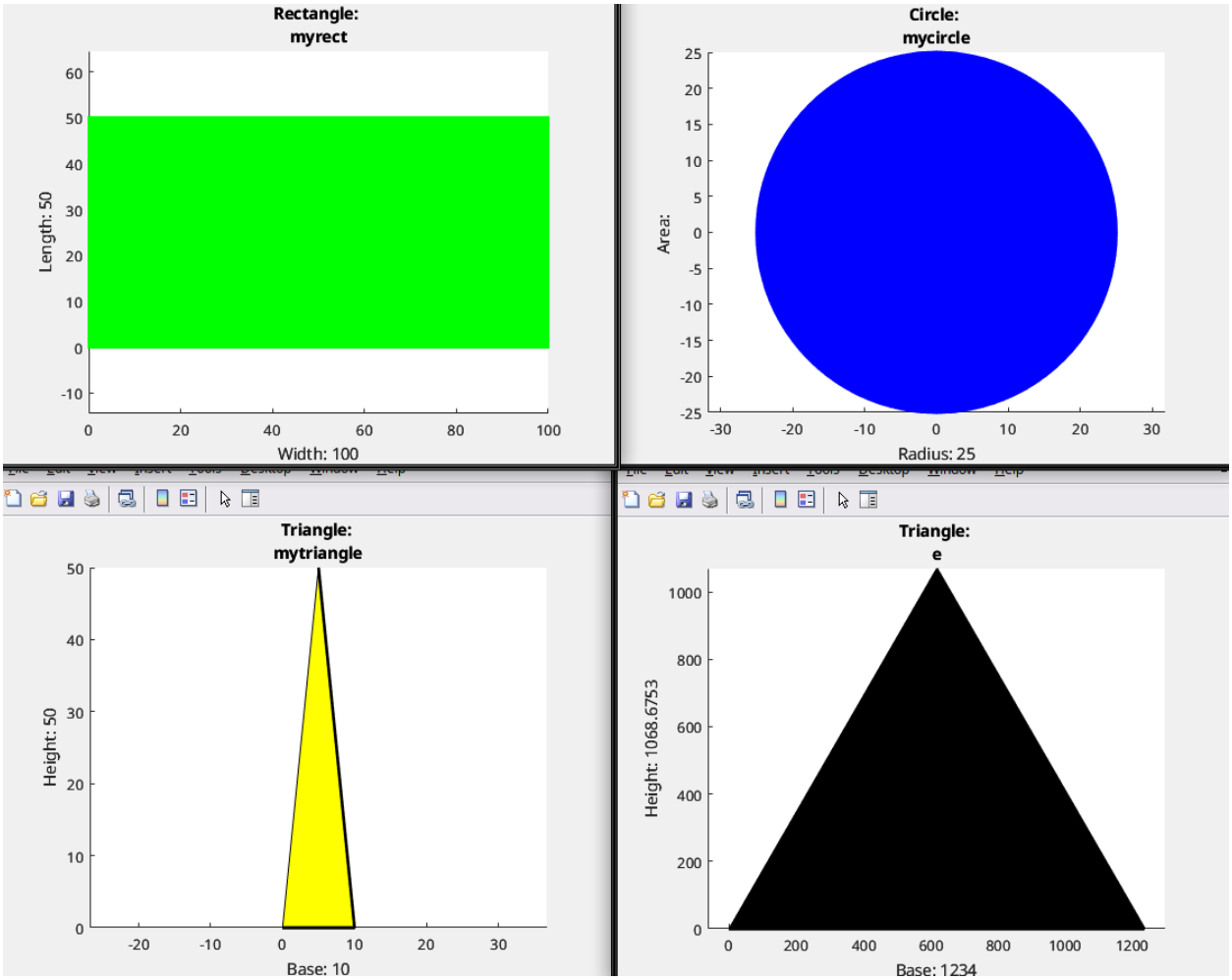


Shape.m

```
>> shapes2={circle6,triangle6,rectangle6,eqTriangle6};
>> Shape.CalculateStatistics(shapes2);
Mean area: 65.86
Median area: 55.18
Standard deviation of areas: 32.82
>>
circle6=Circle("circle6",4,"red"),
rectangle6=Rectangle("rectangle6",5,6,"orange"),
triangle6=Triangle("triangle6",4,8,"blue"),
eqTriangle6=EquilateralTriangle("eqTriangle6",8,"maroon"),

circle6=CalculateArea(circle6),
rectangle6=CalculateArea(rectangle6),
triangle6=CalculateArea(triangle6),
eqTriangle6=CalculateArea(eqTriangle6),

shapes6={circle6,rectangle6,triangle6,eqTriangle6},

Shape.CalculateStatistics(shapes6);
```

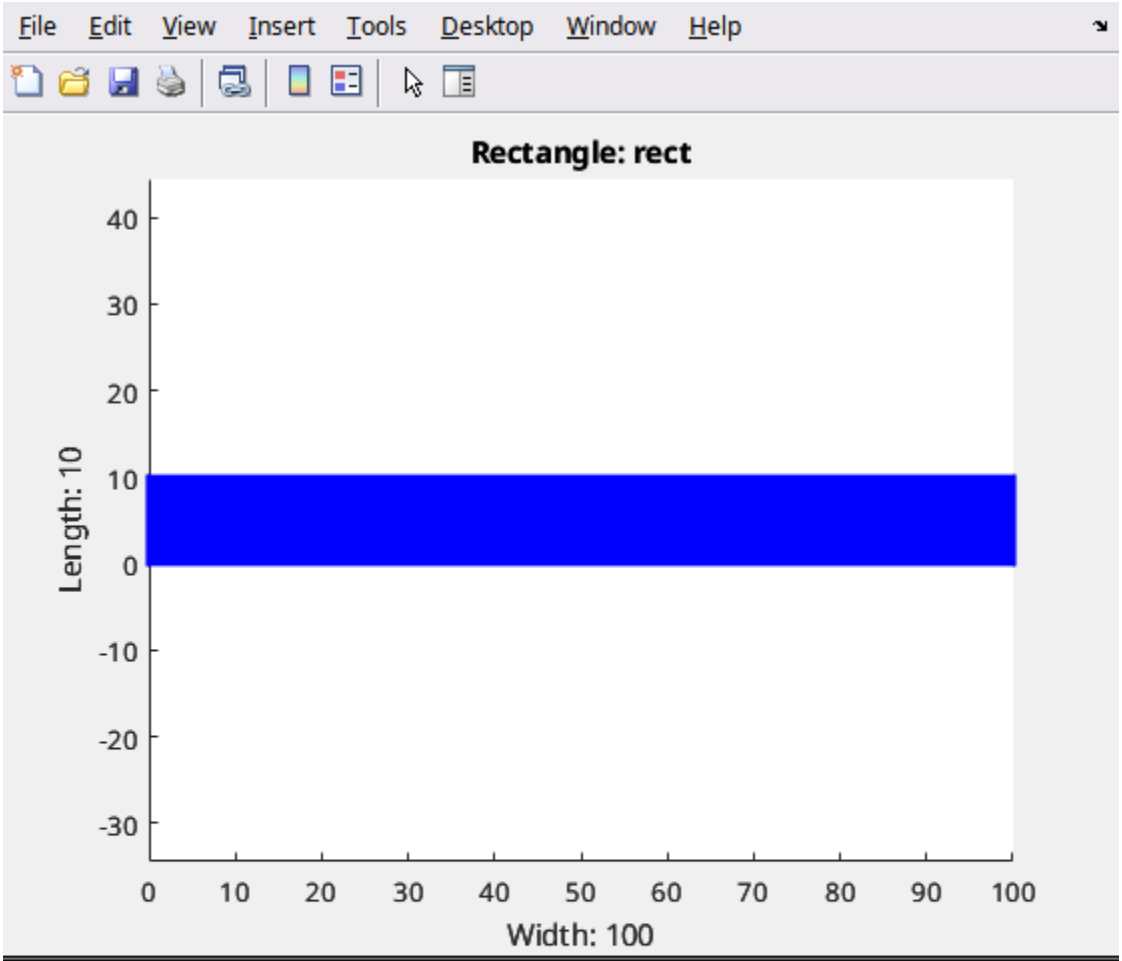
```
shapes6 =

  1×4 cell array

  Columns 1 through 3

    {1×1 Circle}    {1×1 Rectangle}    {1×1 Triangle}

  Column 4

    {1×1 EquilateralTriangle}

Mean area: 30.99
Median area: 28.86
Standard deviation of areas: 14.24
```

Task 7

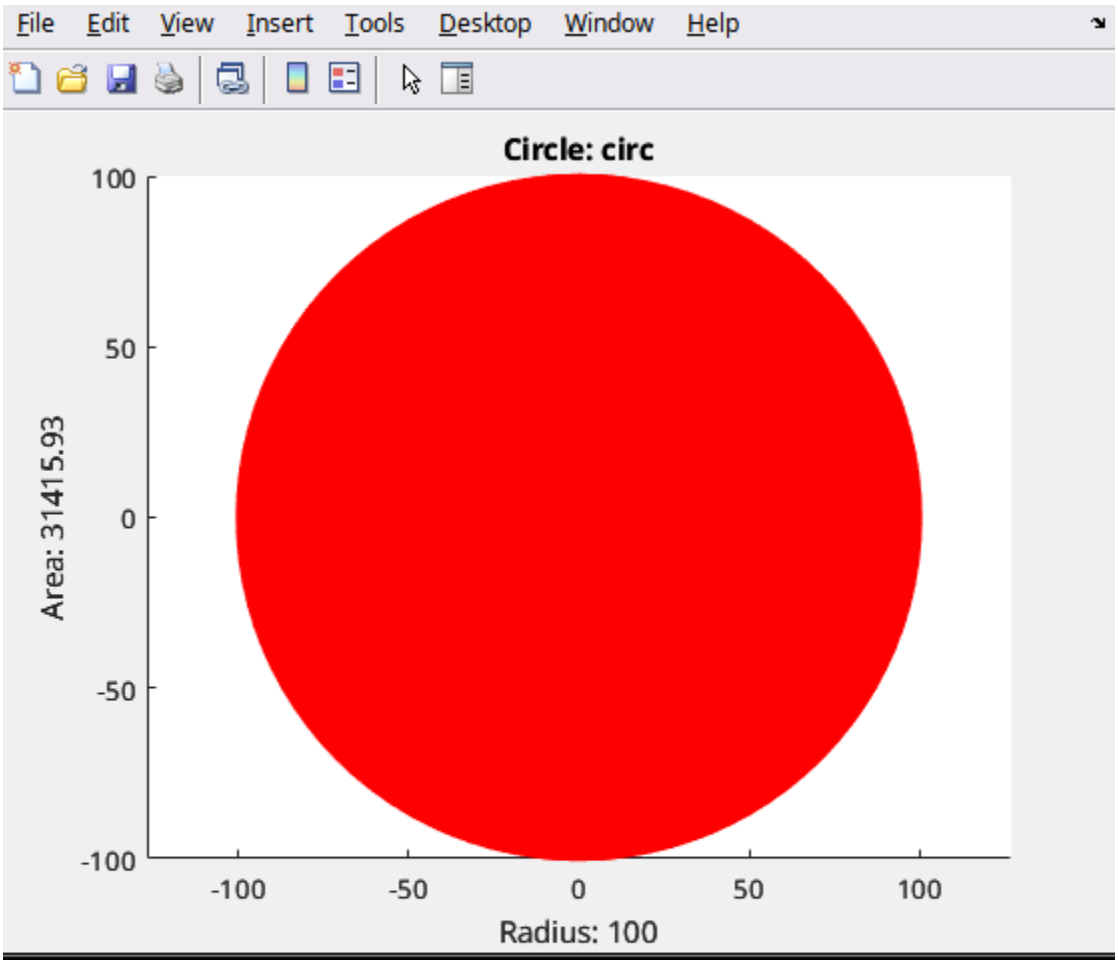


Bottom left is a Triangle, bottom right is an EquilateralTriangle

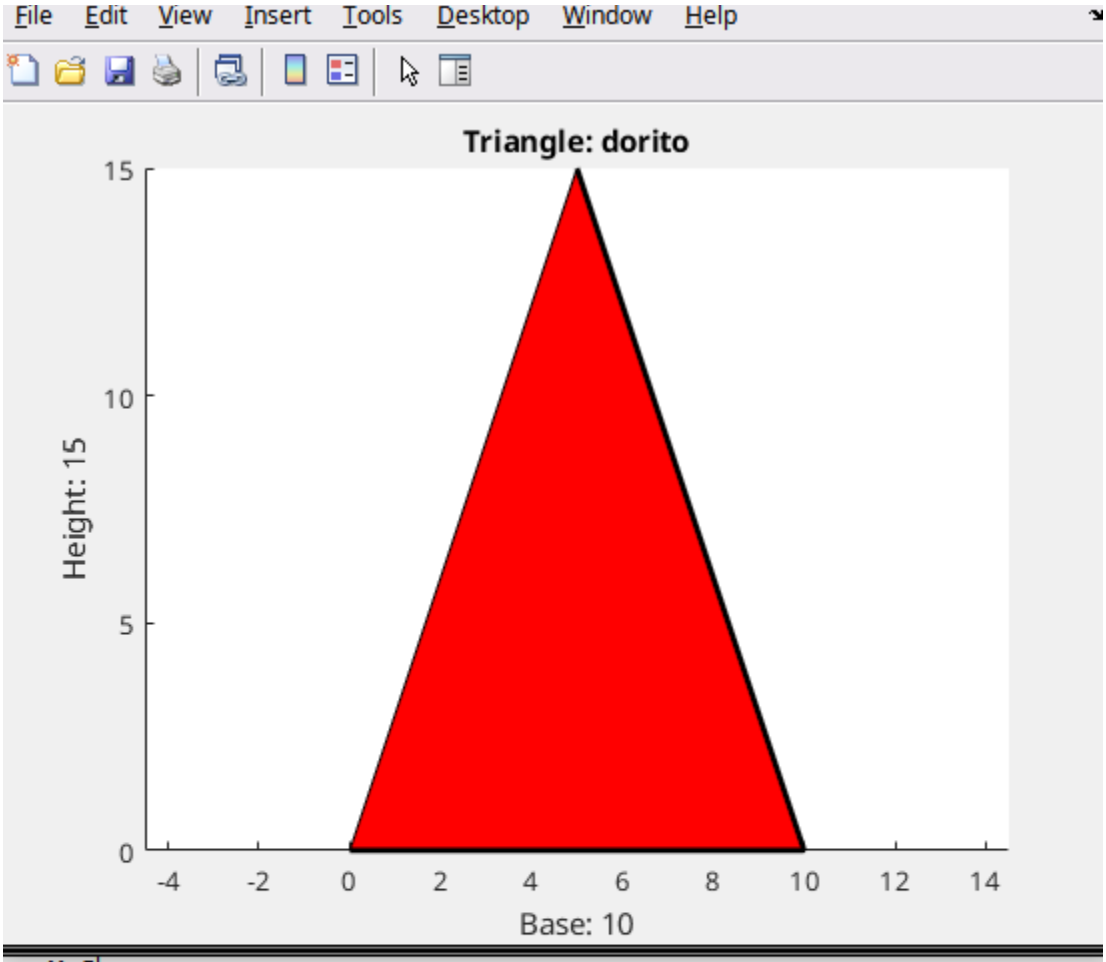
Task 8



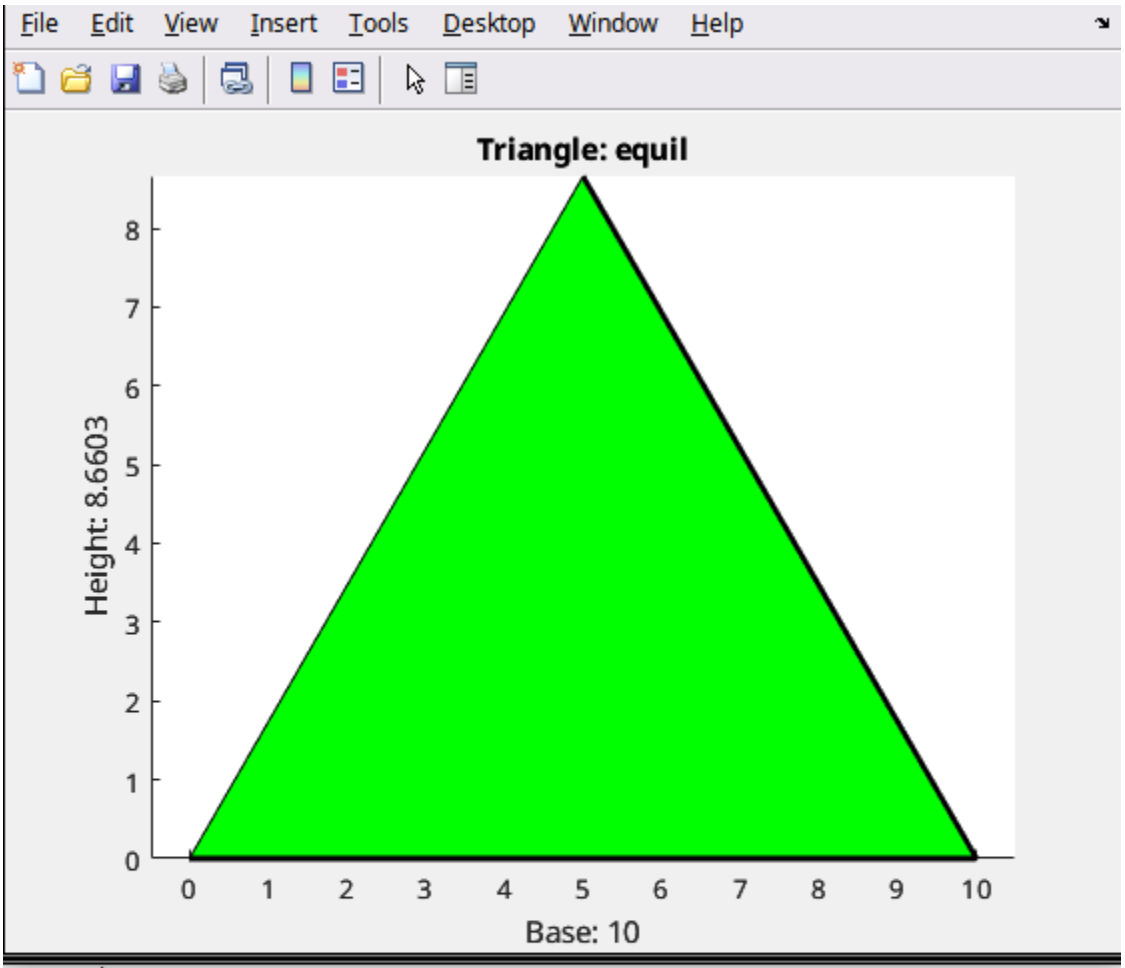

```
>> MyShapes
Available Shapes:
1. Rectangle
2. Circle
3. Triangle
4. Equilateral Triangle
Choose a shape: 1
Enter the shape name: rect
Enter the rectangle's width: 10
Enter the rectangle's height: 100
Enter the color: blue
```




```
>> MyShapes
Available Shapes:
1. Rectangle
2. Circle
3. Triangle
4. Equilateral Triangle
Choose a shape: 2
Enter the shape's name: circ
Enter the circle radius: 100
Enter the color: red
```




```
>> MyShapes
Available Shapes:
1. Rectangle
2. Circle
3. Triangle
4. Equilateral Triangle
Choose a shape: 3
Enter the shape's name: dorito
Enter the triangle's base length: 10
Enter the triangle's height: 15
Enter the color: red
```




```
>> MyShapes
Available Shapes:
1. Rectangle
2. Circle
3. Triangle
4. Equilateral Triangle
Choose a shape: 4
Enter the shape's name: equil
Enter the triangle's side length: 10
Enter the color: green
```