Lab 05

Using Functions in Graphics

Turtle Graphics: Modularizing Code with Functions

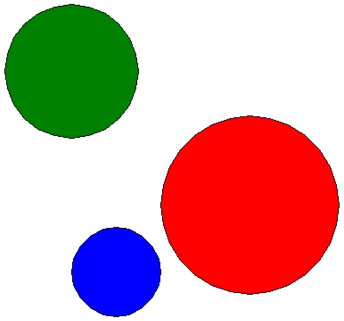
- The following function draws a circle. The parameters specify the location, radius, and color.

```
def circle(x, y, radius, color):
    turtle.penup()             # Raise the pen
    turtle.goto(x, y - radius) # Position the turtle
    turtle.fillcolor(color)    # Set the fill color
    turtle.pendown()           # Lower the pen
    turtle.begin_fill()        # Start filling
    turtle.circle(radius)      # Draw a circle
    turtle.end_fill()          # End filling
```

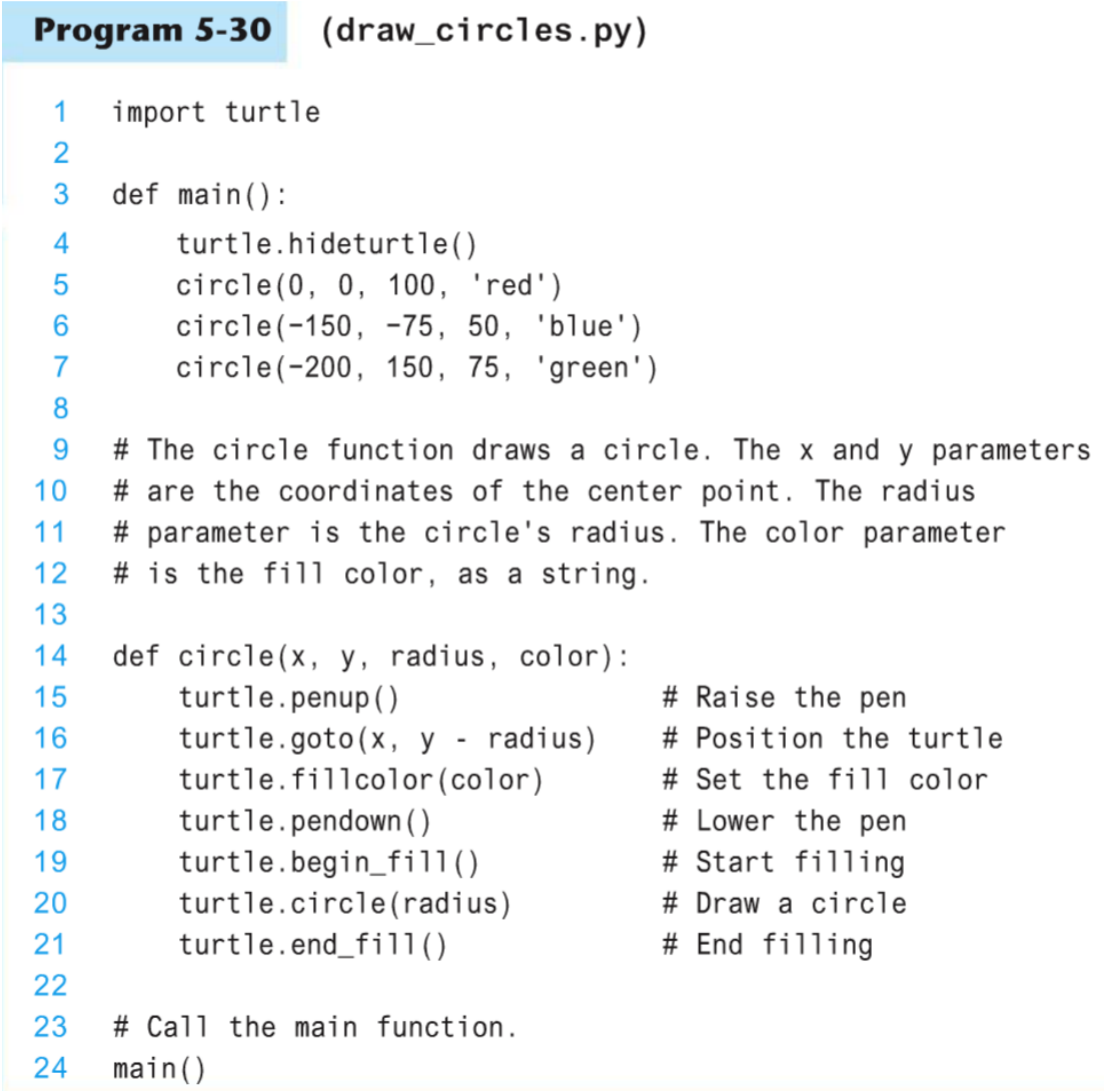
Turtle Graphics: Modularizing Code with Functions

- The following code calls the previously shown `circle` function to draw three circles:

```
circle(0, 0, 100, 'red')
circle(-150, -75, 50, 'blue')
circle(-200, 150, 75, 'green')
```



Program 5-30 Draw_ Circles.py



Program 5-30 (draw_circles.py)

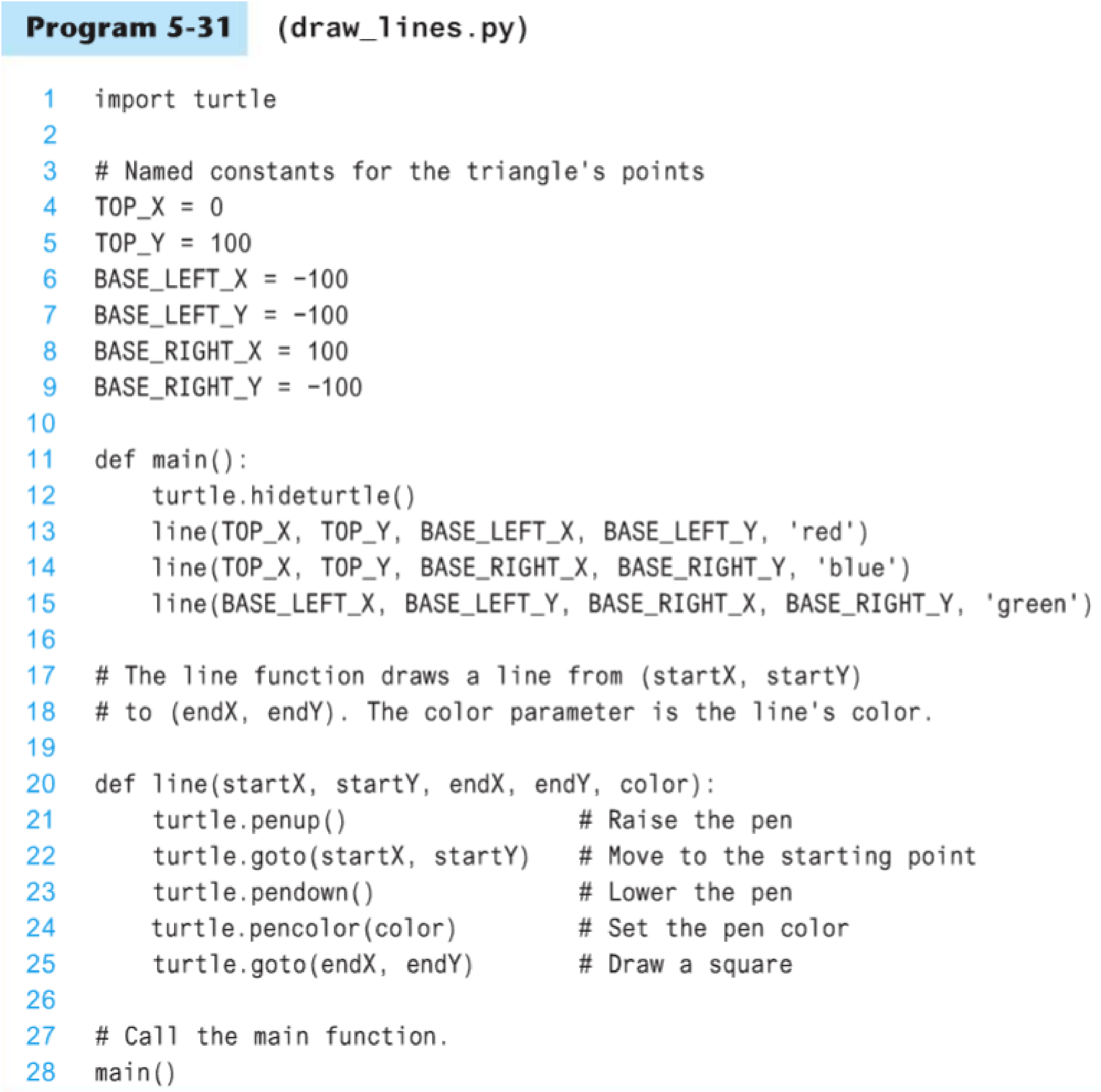
```
import turtle

def main():
    turtle.hideturtle()
    circle(0, 0, 100, 'red')
    circle(-150, -75, 50, 'blue')
    circle(-200, 150, 75, 'green')

# The circle function draws a circle. The x and y parameters
# are the coordinates of the center point. The radius
# parameter is the circle's radius. The color parameter
# is the fill color, as a string.

def circle(x, y, radius, color):
    turtle.penup()                 # Raise the pen
    turtle.goto(x, y - radius)     # Position the turtle
    turtle.fillcolor(color)        # Set the fill color
    turtle.pendown()               # Lower the pen
    turtle.begin_fill()            # Start filling
    turtle.circle(radius)          # Draw a circle
    turtle.end_fill()              # End filling

# Call the main function.
main()
```

Turtle Graphics: Modularizing Code with Functions

- The following function draws a line. The parameters specify the starting and ending locations, and color.

```
def line(startX, startY, endX, endY, color):
    turtle.penup()              # Raise the pen
    turtle.goto(startX, startY) # Move to the starting point
    turtle.pendown()            # Lower the pen
    turtle.pencolor(color)      # Set the pen color
    turtle.goto(endX, endY)     # Draw a square
```

Program 5-31 Draw_ Lines.py

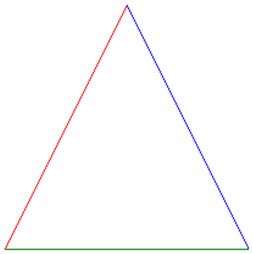


Program 5-31 (draw_lines.py)

```
import turtle

# Named constants for the triangle's points
TOP_X = 0
TOP_Y = 100
BASE_LEFT_X = -100
BASE_LEFT_Y = -100
BASE_RIGHT_X = 100
BASE_RIGHT_Y = -100

def main():
    turtle.hideturtle()
    line(TOP_X, TOP_Y, BASE_LEFT_X, BASE_LEFT_Y, 'red')
    line(TOP_X, TOP_Y, BASE_RIGHT_X, BASE_RIGHT_Y, 'blue')
    line(BASE_LEFT_X, BASE_LEFT_Y, BASE_RIGHT_X, BASE_RIGHT_Y, 'green')

# The line function draws a line from (startX, startY)
# to (endX, endY). The color parameter is the line's color.

def line(startX, startY, endX, endY, color):
    turtle.penup()                  # Raise the pen
    turtle.goto(startX, startY)     # Move to the starting point
    turtle.pendown()                # Lower the pen
    turtle.pencolor(color)          # Set the pen color
    turtle.goto(endX, endY)         # Draw a square

# Call the main function.
main()
```

Turtle Graphics: Modularizing Code with Functions

- The following code calls the previously shown `line` function to draw a triangle:

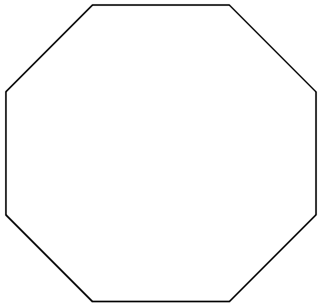


```
TOP_X = 0
TOP_Y = 100
BASE_LEFT_X = -100
BASE_LEFT_Y = -100
BASE_RIGHT_X = 100
BASE_RIGHT_Y = -100
line(TOP_X, TOP_Y, BASE_LEFT_X, BASE_LEFT_Y, 'red')
line(TOP_X, TOP_Y, BASE_RIGHT_X, BASE_RIGHT_Y, 'blue')
line(BASE_LEFT_X, BASE_LEFT_Y, BASE_RIGHT_X, BASE_RIGHT_Y, 'green')
```

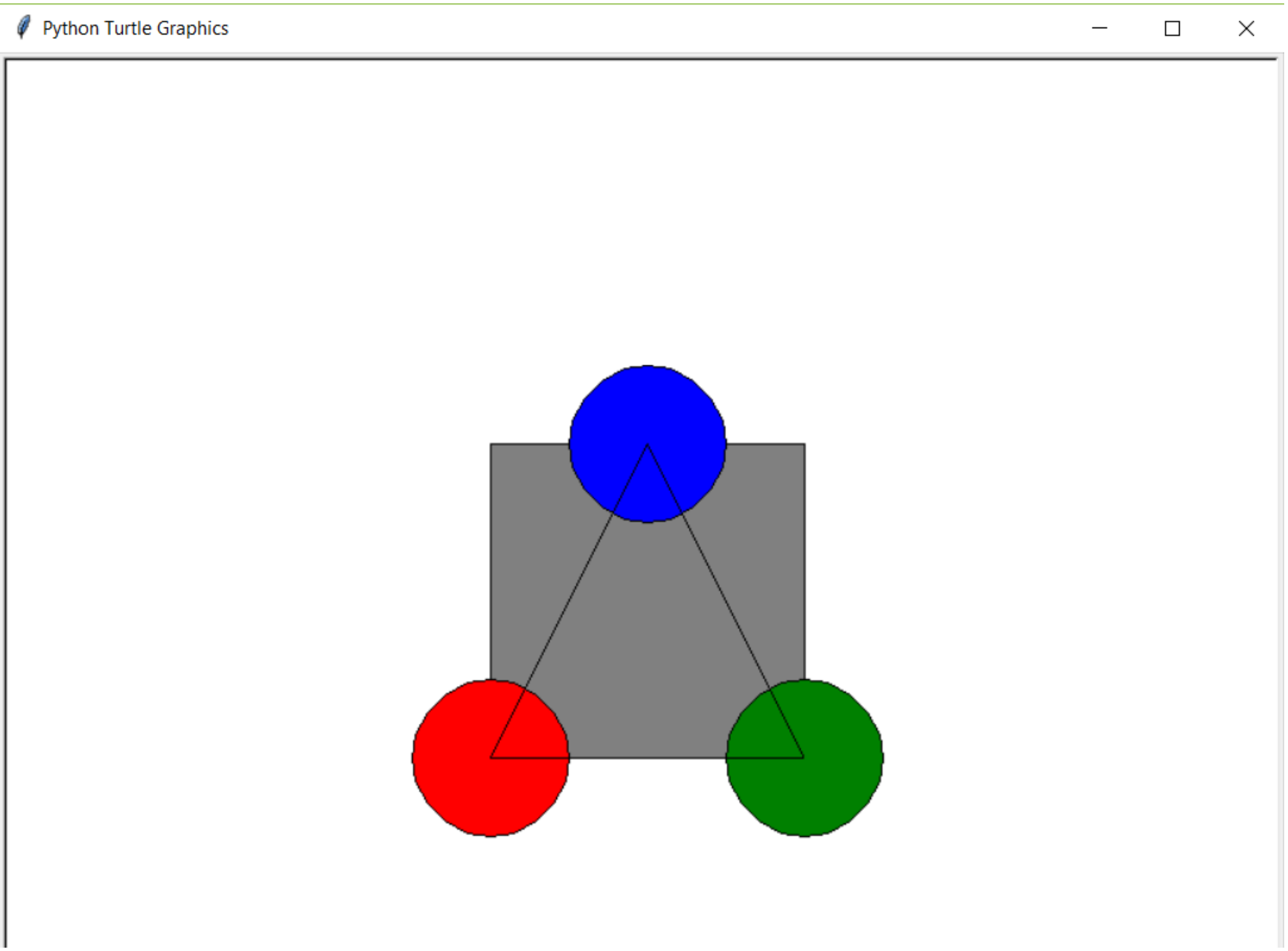
Turtle Graphics: Using Loops to Draw Designs

- This `for` loop iterates eight times to draw the octagon:

```
for x in range(8):
    turtle.forward(100)
    turtle.right(45)
```

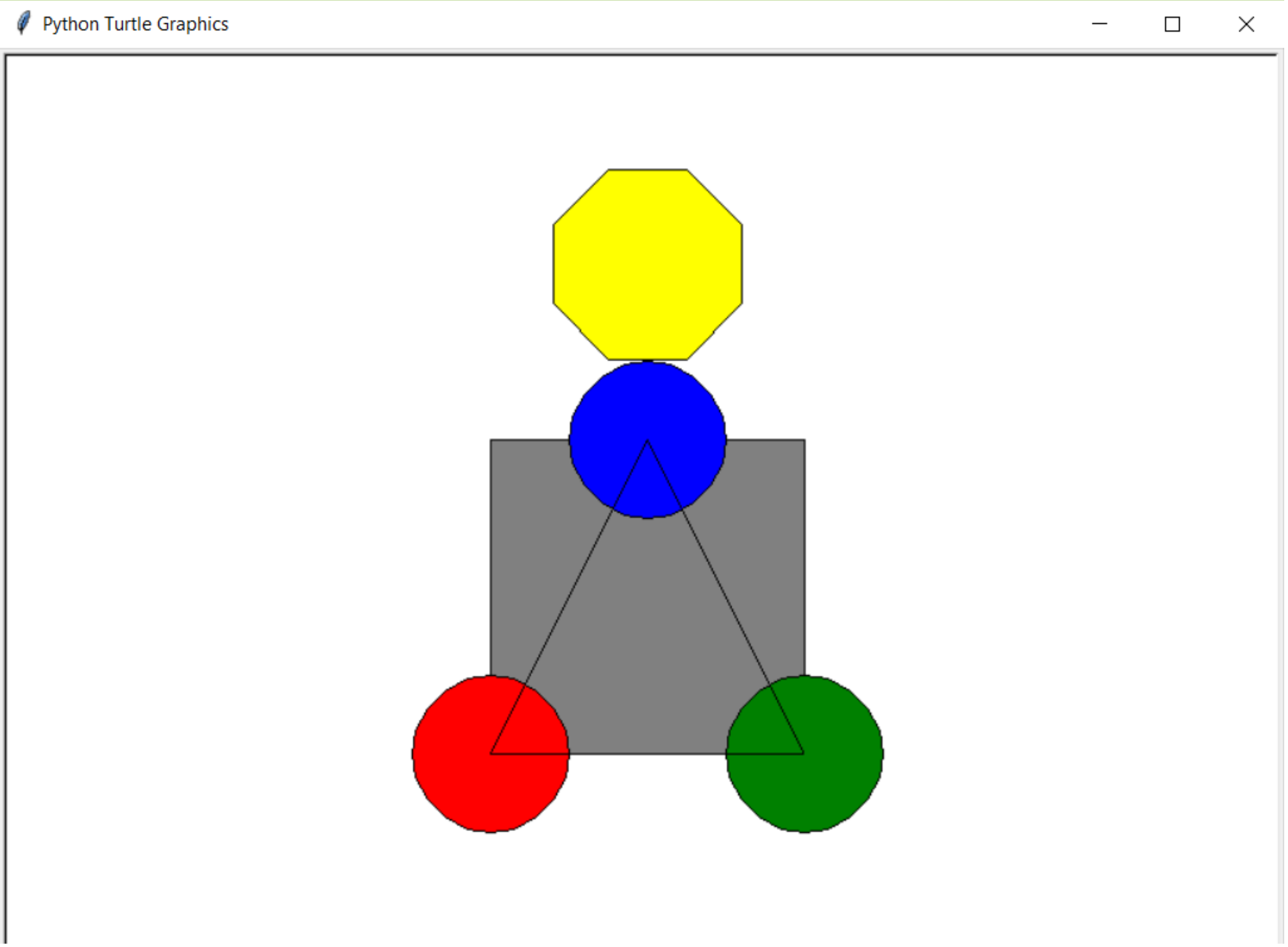


Laboratory 05A



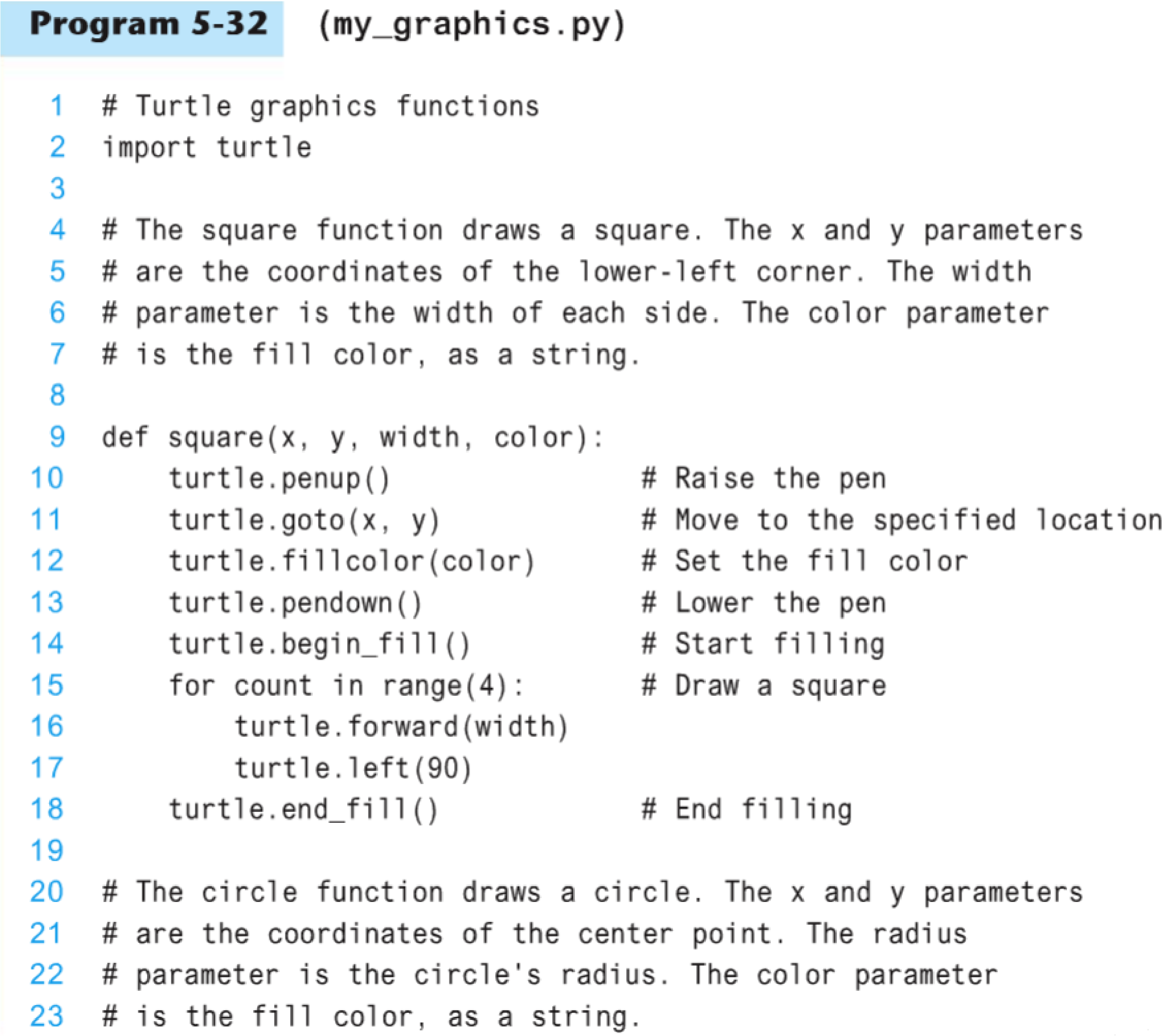
- Using Program_5-32 and Program_5-33 as guides, type in these programs in order to obtain the result shown in the figure to the left.
- When you have got this result working, then call over the Instructor to give you credit for your work.

Laboratory 05B



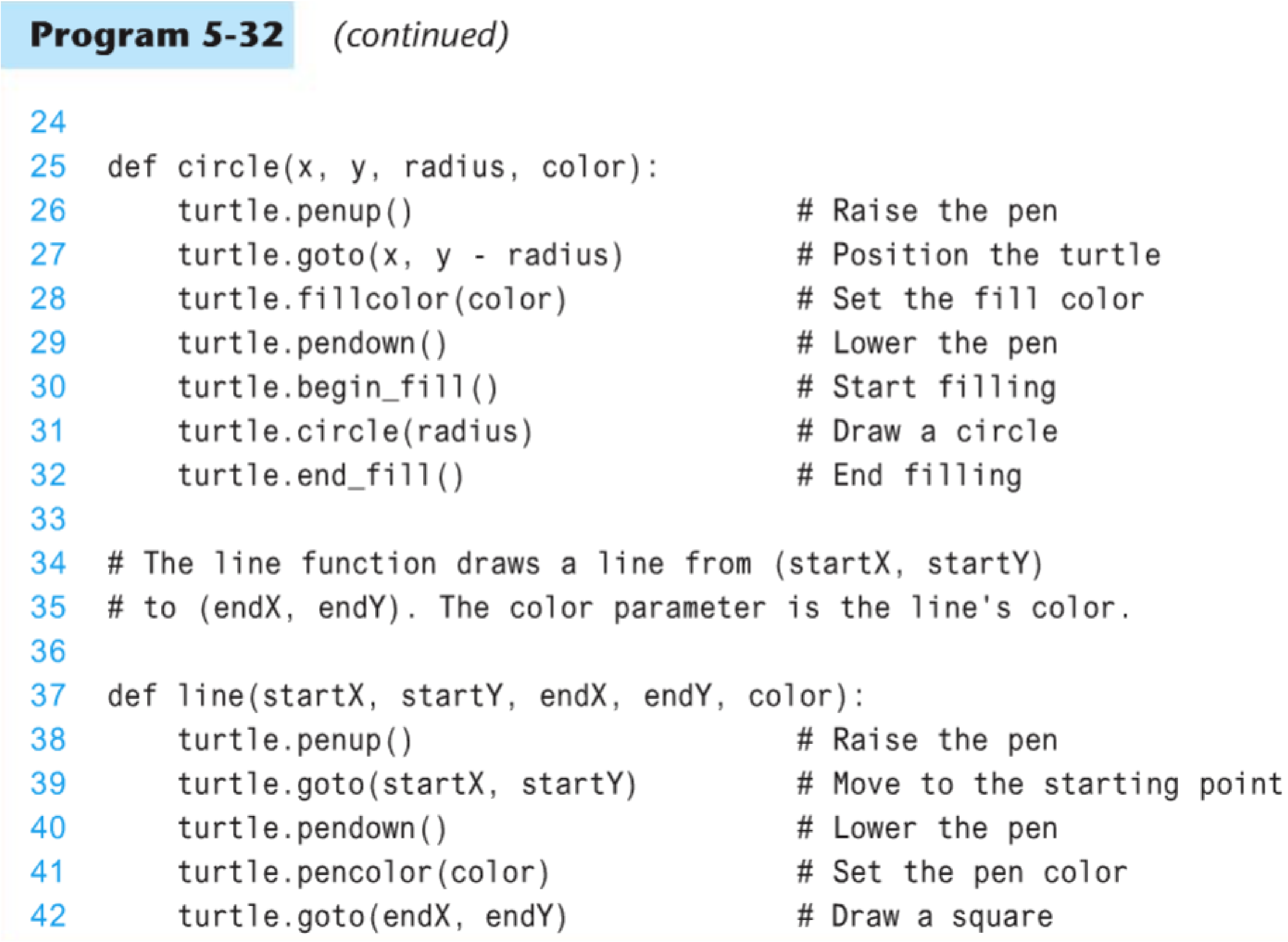
- Now we want to add one more function called: **octagon(beginX, beginY, side, color)** to the functions contained in Program5-32.
- The value of **side** = 50.
- The value of **color** = Yellow.
- Choose values for **beginX** and **beginY** so that the Octagon is positioned **precisely** as shown at left.
- Add the code to implement the Octagon in Program5-33.
- Run the program, and then call the instructor to evaluate your work.

Program 5-32 My_graphics.py



Program 5-32 (my_graphics.py)

```
# Turtle graphics functions
import turtle

# The square function draws a square. The x and y parameters
# are the coordinates of the lower-left corner. The width
# parameter is the width of each side. The color parameter
# is the fill color, as a string.

def square(x, y, width, color):
    turtle.penup()                # Raise the pen
    turtle.goto(x, y)             # Move to the specified location
    turtle.fillcolor(color)       # Set the fill color
    turtle.pendown()              # Lower the pen
    turtle.begin_fill()           # Start filling
    for count in range(4):        # Draw a square
        turtle.forward(width)
        turtle.left(90)
    turtle.end_fill()             # End filling

# The circle function draws a circle. The x and y parameters
# are the coordinates of the center point. The radius
# parameter is the circle's radius. The color parameter
# is the fill color, as a string.
```

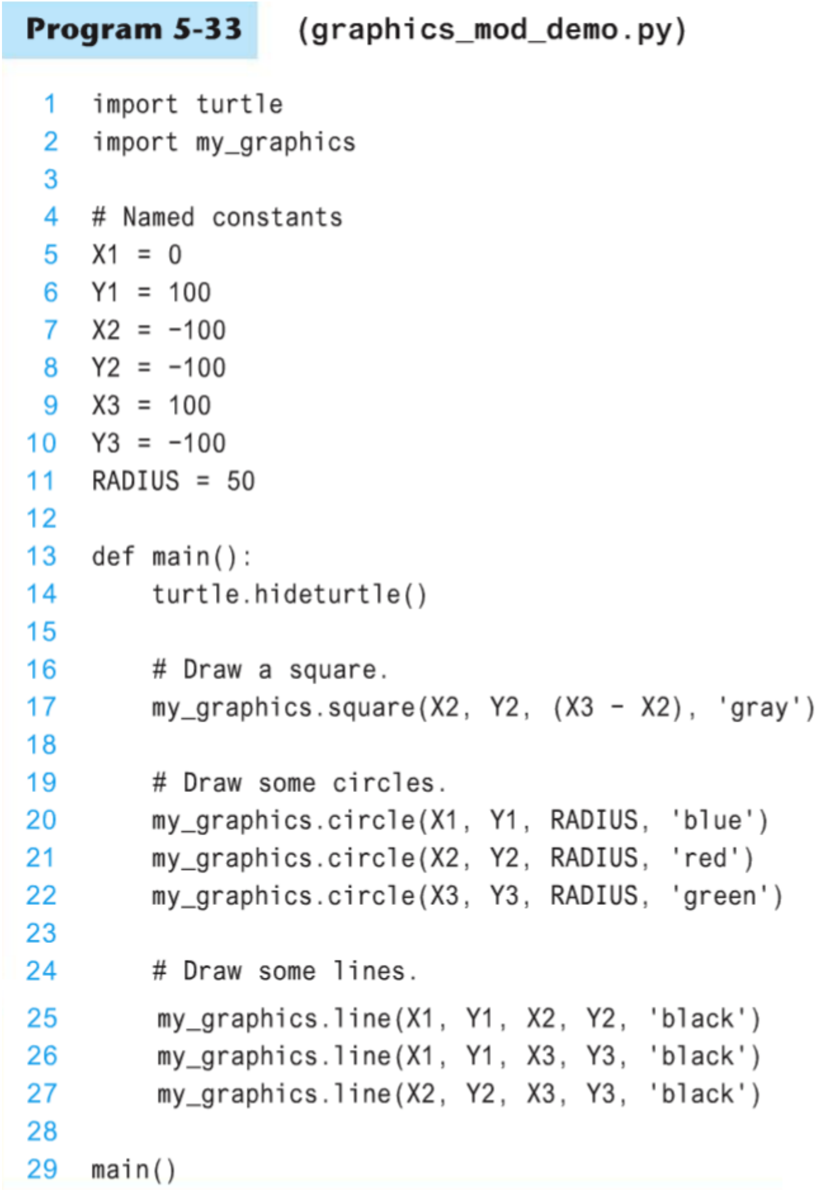


Program 5-32 *(continued)*

```python

def circle(x, y, radius, color):
    turtle.penup()                  # Raise the pen
    turtle.goto(x, y - radius)      # Position the turtle
    turtle.fillcolor(color)         # Set the fill color
    turtle.pendown()                # Lower the pen
    turtle.begin_fill()             # Start filling
    turtle.circle(radius)           # Draw a circle
    turtle.end_fill()               # End filling

# The line function draws a line from (startX, startY)
# to (endX, endY). The color parameter is the line's color.

def line(startX, startY, endX, endY, color):
    turtle.penup()                  # Raise the pen
    turtle.goto(startX, startY)     # Move to the starting point
    turtle.pendown()                # Lower the pen
    turtle.pencolor(color)          # Set the pen color
    turtle.goto(endX, endY)         # Draw a square
```

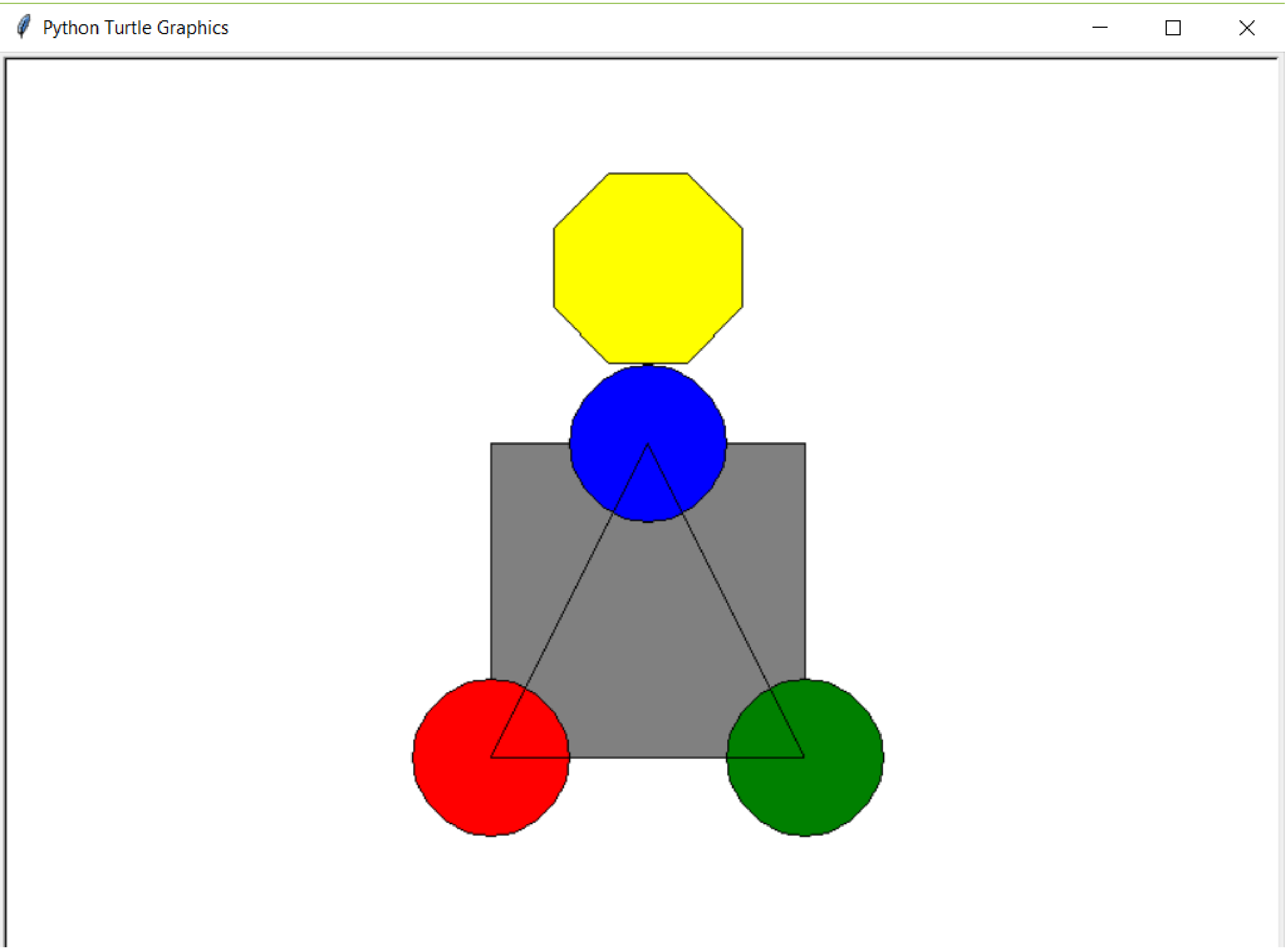
Program 5-32 My_graphics.py

Program 5-33
graphics_mod_demo.py



Program 5-33 (graphics_mod_demo.py)

```python
import turtle
import my_graphics

# Named constants
X1 = 0
Y1 = 100
X2 = -100
Y2 = -100
X3 = 100
Y3 = -100
RADIUS = 50

def main():
    turtle.hideturtle()

    # Draw a square.
    my_graphics.square(X2, Y2, (X3 - X2), 'gray')

    # Draw some circles.
    my_graphics.circle(X1, Y1, RADIUS, 'blue')
    my_graphics.circle(X2, Y2, RADIUS, 'red')
    my_graphics.circle(X3, Y3, RADIUS, 'green')

    # Draw some lines.
    my_graphics.line(X1, Y1, X2, Y2, 'black')
    my_graphics.line(X1, Y1, X3, Y3, 'black')
    my_graphics.line(X2, Y2, X3, Y3, 'black')

main()
```



Lab05B Final Result