```
# SimpleGraphics.py
""" A module that supports the simple drawing of colored
rectangles, disks, stars, line segments, and text. """

import matplotlib.pyplot as plt
from matplotlib.patches import Polygon
import numpy as np
from time import sleep as pause

# Built-in SimpleGraphics colors
YELLOW    = [1.0,1.0,0.0]
CYAN      = [0.0,1.0,1.0]
MAGENTA   = [1.0,0.0,1.0]
RED       = [1.0,0.0,0.0]
GREEN     = [0.0,1.0,0.0]
BLUE      = [0.0,0.0,1.0]
WHITE     = [1.0,1.0,1.0]
BLACK     = [0.0,0.0,0.0]
PURPLE    = [.57,.17,.93]
DARKGRAY  = [.33,.33,.33]
LIGHTGRAY = [.67,.67,.67]
ORANGE    = [1.0,.50,0.0]
PINK      = [1.0,.71,.80]


def MakeWindow(M,labels=True,bgcolor=WHITE):
    """
    Creates a window with x range -M<=x<=M and y range -M<=y<=M

    If labels is set to False, it will turn off the labeled axes.
    Labeling will not look good if M is too large, e.g., M>10.

    The window will have a background color specified by bgcolor.
    The default is white.

    Preconditions: M is positive number, labels is boolean,
    and bgcolor is an rgb array.

    Usage:
      MakeWindow(5)
      MakeWindow(5,labels=False)
      MakeWindow(5,bgcolor=PURPLE)
      MakeWindow(5,labels=False,bgcolor=[0.87 1.00 0.00])
    """

    plt.figure(figsize=(8,8), dpi=80)
    # Where to put the axis ticks.
    plt.xticks(np.linspace(-M, M, 2*M+1, endpoint=True))
    plt.yticks(np.linspace(-M, M, 2*M+1, endpoint=True))
    # The x and y ranges along the axes.
    plt.xlim(-M,M)
    plt.ylim(-M,M)
    # Background color
    axes = plt.gca() #get current axes
    axes.set_axis_bgcolor(bgcolor)
    if not labels:
        # Suppress the ticks
        axes.set_xticks([]) # remove number labels and ticks
        axes.set_yticks([])

def ShowWindow(time=None):
    """
    Display the current window for a specified time before resuming
    execution. The default time is infinite. That means execution
    halts until the window is closed "manually".

    Precondition: t is a nonegative number.
    Usage:
        ShowWindow()
        ShowWindow(5)
    """
    if time==None:
        plt.show()
    else:
        plt.show(block=False)
        pause(time)

def CloseWindow():
    """ Close all windows. """
    plt.close()


def DrawRect(a,b,L,W,theta=0.0,FillColor=None,EdgeColor=BLACK,EdgeWidth=1):
    """
    Displays a rectangle that is rotated theta degrees counter
    clockwise about its center (a,b). The unrotated version of the
    rectangle has horizontal dimension L and vertical dimension W.
    It has fill color FillColor, edge color EdgeColor, edge width EdgeWidth.
    If no fill color is specified, then the rectangle is transparant
    and only its perimeter is displayed. The default edge color is black
    and the default edge width is one.

    Preconditions: a, b, L, and W are numbers, FillColor and EdgeColor are
    rgb lists, EdgeWidth is a positive number, and theta is a number.

    Usage:
        DrawRect(0,0,1,2)
        DrawRect(0,0,1,2,theta=30)
        DrawRect(0,0,1,2,FillColor=CYAN)
        DrawRect(0,0,1,2,EdgeColor=RED)
        DrawRect(0,0,1,2,theta=-30,FillColor=CYAN,EdgeColor=RED,EdgeWidth=5)
    """

    # These arrays specify the (x,y) coordinates of the rectangle corners.
    L = float(L)
    W = float(W)
    theta = theta*np.pi/180
    x1 = np.array([-L/2,L/2,L/2,-L/2,-L/2])
    y1 = np.array([-W/2,-W/2,W/2,W/2,-W/2])
    x = a + np.cos(theta)*x1 - np.sin(theta)*y1
    y = b + np.sin(theta)*x1 + np.cos(theta)*y1
    if FillColor is None:
        # No fill, just draw the perimeter
        plt.plot(x,y,color=EdgeColor,linewidth=EdgeWidth)
    else:
        # Fill and accent the perimeter according to the values of eColor and eWidth.
        plt.fill(x,y,facecolor=FillColor,edgecolor=EdgeColor,linewidth=EdgeWidth)


def DrawDisk(a,b,r,FillColor=None,EdgeColor=BLACK,EdgeWidth=1):
    """
    Draws a disk with center at (a,b), and radius r.
    It has fill color FillColor, edge color EdgeColor, edge width EdgeWidth.
    If no fill color is specified, then the disk is transparant
    and only its perimeter is displayed. The default edge color is black
    and the default edge width is one.

    Preconditions: a, b, and r are numbers, FillColor and EdgeColor are
    rgb lists, and EdgeWidth is a positive number.

    Usage:
        DrawDisk(0,0,2)
        DrawDisk(0,0,2,FillColor=CYAN)
        DrawDisk(0,0,2,EdgeColor=RED)
        DrawDisk(0,0,2,FillColor=CYAN,EdgeColor=RED,EdgeWidth=5)
    """

    theta= np.linspace(0,2*np.pi,256,endpoint=True)
    x = a+r*np.cos(theta)
    y = b+r*np.sin(theta)
    if FillColor is None:
        # No fill, just the perimeter
        plt.plot(x,y,color=EdgeColor,linewidth=EdgeWidth)
    else:
        # Fill and accent the perimeter according to the values of EdgeColor and EdgeWidth.
        plt.fill(x,y,facecolor=FillColor,edgecolor=EdgeColor,linewidth=EdgeWidth)


def DrawStar(a,b,r,theta=0.0,FillColor=None,EdgeColor=BLACK,EdgeWidth=1):
    """
    Displays a star that is rotated theta degrees counter
    clockwise about its center (a,b). The star has radius r.
    It has fill color FillColor, edge color EdgeColor, edge width EdgeWidth.
    If no fill color is specified, then the star is transparant
    and only its perimeter is displayed. The default edge color is black
    and the default edge width is one.

    Preconditions: a, b, r and theta are numbers, FillColor and EdgeColor are
    rgb lists, and EdgeWidth is a positive number.

    Usage:
        DrawStar(0,0,2)
        DrawStar(0,0,2,theta=30)
        DrawStar(0,0,2,FillColor=CYAN)
        DrawStar(0,0,2,EdgeColor=RED)
        DrawStar(0,0,2,theta=-30,FillColor=CYAN,EdgeColor=RED,EdgeWidth=5)
    """

    # The radius of the inner 5 vertices..
    r2 = r/(2*(1+np.sin(np.pi/10)))
    # Compute the 10 vertices
    tau = np.linspace(0,2*np.pi,11,endpoint=True) + np.pi/10 + theta*np.pi/180
    x = np.cos(tau); x[0:11:2]= r*x[0:11:2]; x[1:11:2]=r2*x[1:11:2]; x = x+a
    y = np.sin(tau); y[0:11:2]= r*y[0:11:2]; y[1:11:2]=r2*y[1:11:2]; y = y+b
    # Display...
    if FillColor is None:
        # No fill, just the perimeter
        plt.plot(x,y,color=EdgeColor,linewidth=EdgeWidth)
    else:
        # Fill and accent the perimeter according to the values of eColor and eWidth.
        plt.fill(x,y,facecolor=FillColor,edgecolor=EdgeColor,linewidth=EdgeWidth)


def DrawLineSeg(x0,y0,x1,y1,LineColor=BLACK,LineWidth=1):
    """
    Displays a line segment that connects (x0,y0) and (x1,y1). The
    line segment has color LineColor and width LineWidth.

    Preconditions: x0,y0,x1,y1 are numbers, c is an rgb list, and LW is a positive
    number.

    Usage:
        DrawLineSeg(1,2,3,4)
        DrawLineSeg(1,2,3,4,LineWidth=4)
        DrawLineSeg(1,2,3,4,LineColor=MAGENTA)
        DrawLineSeg(1,2,3,4,LineColor=BLUE,LineWidth=2)

    """
    plt.plot([x0,x1],[y0,y1],linewidth=LineWidth,color=LineColor)

def DrawText(x,y,s,FontColor=BLACK,FontSize=10):
    """
    Displays string s at (x,y) with color FontColor and size FontSize. The default color
    is BLACK and the default size is 10.

    PreConditions: x and y are numbers, FontColor is an rgb list, and FontSize is a positive int.

    Usage:
        DrawText(1,2,'Message')
        DrawText(1,2,'Message',FontColor=RED)
        DrawText(1,2,'Message',FontSize=18)
        DrawText(1,2,'Message',FontSize=18,FontColor=RED)
    """
    plt.text(x,y,s,color=FontColor,fontsize=FontSize)

def Title(s,FontColor=BLACK,FontSize=18):
    """
    Displays string s as a title above the figure window with color FontColor and size FontSize.
    The default color is BLACK and the default size is 18.

    PreConditions: s is a string, FontColor is an rgb list, and FontSize is a positive int.

    Usage:
        Title('Message')
        Title('Message',FontColor=RED)
        Title('Message',FontSize=24)
        Title('Message',FontSize=24,FontColor=RED)
    """
    plt.title(s,fontsize=FontSize,color=FontColor)

def DrawPoly(x,y, FillColor=None, EdgeWidth=1, EdgeColor=BLACK):
```

```
""" Draws a polygon whose vertices are specified by th
lists x and y.

The fill color can be one of the 13 built-in colors YELLOW, CYAN, MAGENTA,
RED, GREEN, BLUE, WHITE, BLACK, PURPLE, LIGHTFGRAY, DARKGRAY, ORANGE, or PINK
or an rgb array. The default value for color is None and in this case
the rectangle is transparent.

The perimeter display width is specified through the argument stroke.
The default value is 1. Larger values create a wider black outline of
the displayed rectangle. For no perimeter highlighting, set stroke=0.

Preconditions: a,b,L, and W are float or int and positive. color
is an rgb array, stroke is a positive float or int, and rotate is
a float or int that specifies the clockwise rotation angle in degrees.

Sample calls:
        DrawPoly([-2,2,2,-2],[-2,-2,2,2],color=PINK)

"""

# These arrays specify the (x,y) coordinates of the rectangle corners.
u = list(x);u.append(u[0])
v = list(y);v.append(v[0])
if FillColor is None:
    # No fill, just draw the perimeter
    plt.plot(u, v,linewidth=EdgeWidth,color=EdgeColor)
else:
    # Fill and accent the perimeter according to the value of stroke.
    plt.fill(u, v, facecolor=FillColor, edgecolor=EdgeColor, linewidth=EdgeWidth)
```