No.1

import turtle

x0, y0 = eval(input("Enter x0 and y0 for p0: "))

x1, y1 = eval(input("Enter x1 and y1 for p1: "))

x2, y2 = eval(input("Enter x2 and y2 for p2: "))

c = (x1 - x0)\*(y2 - y0) - (x2 - x0)\*(y1 - y0)

# mid point

mid\_pointx = (x0 + x1) / 2

mid\_pointy = (y0 + y1) / 2

if (x1 > x0):

    first\_pointx = x0

    first\_pointy = y0

    second\_pointx = x1

    second\_pointy = y1

elif(x0 > x1):

    first\_pointx = x1

    first\_pointy = y1

    second\_pointx = x0

    second\_pointy = y0

print("first\_pointx, first\_pointy", first\_pointx, first\_pointy)

print("second\_pointx, second\_pointy", second\_pointx, second\_pointy)

# Display two points, point 1 and point 2 and the connecting line

turtle.penup()

turtle.goto(x0, y0) # Move to (x1, y1)

turtle.pendown()

turtle.write("p0")

turtle.goto(x1, y1) # Draw a line to (x2, y2)

turtle.write("p1")

turtle.penup()

turtle.goto(x2, y2)

turtle.write("p2")

style = ('Courier', 12, 'bold')

if (c > 0):

    turtle.goto(x2, y2-y2-y2)

    turtle.write("p2 is on the left side of the line from p0 to  p1.", font=style,  align='left')

elif (c == 0):

    if (x2 >= second\_pointx and y2 >= second\_pointy):

        turtle.goto(x2, y2-y2-y2)

        turtle.write("p2 is on the same line from p0 to p1 and on the right side.", font=style,  align='left')

    elif (x2 <= first\_pointx and y2 <= first\_pointy):

        turtle.goto(x2, y2-y2-y2)

        turtle.write("p2 is on the same line from p0 to p1 and on the left side.", font=style,  align='left')

    else:

        turtle.goto(x2+x2, y2+y2)

        turtle.write("p2 is on the same line from p0 to p1 and It's between the p0 and p1.", font=style,  align='left')

elif (c < 0):

    turtle.goto(x2, y2-y2-y2)

    turtle.write("p2 is on the right side of the line from p0 to p1.", font=style,  align='left')

turtle.done()

No. 2

from tkinter import E

import turtle, math

# User input center x-, y- coordinates, width and height of two rectangles

r1x = eval(input("Please enter rectangle #1's center x-coordinate: "))

r1y = eval(input("Please enter rectangle #1's center y-coordinate: "))

r1Width = eval(input("Please enter rectangle #1's width: "))

r1Height = eval(input("Please enter rectangle #1's height: "))

r2x = float(input("Please enter rectangle #2's center x-coordinate: "))

r2y = float(input("Please enter rectangle #2's center y-coordinate: "))

r2Width = float(input("Please enter rectangle #2's width: "))

r2Height =  float(input("Please enter rectangle #2's height: "))

#Finding points to draw the sides of both rectangles

Center1X = r1x

Center1Y = r1y

Width1 = r1Width

Height1 = r1Height

Center2X = r2x

Center2Y = r2y

Width2 = r2Width

Height2 = r2Height

# first rectangle's points

r1\_point1x = Center1X - (Width1/2)

r1\_point1y = Center1Y + (Height1/2)

r1\_point2x = Center1X + (Width1/2)

r1\_point2y = Center1Y + (Height1/2)

r1\_point3x = Center1X + (Width1/2)

r1\_point3y = Center1Y - (Height1/2)

r1\_point4x = Center1X - (Width1/2)

r1\_point4y = Center1Y - (Height1/2)

# second rectangle's points

r2\_point1x = Center2X - (Width2/2)

r2\_point1y = Center2Y + (Height2/2)

r2\_point2x = Center2X + (Width2/2)

r2\_point2y = Center2Y + (Height2/2)

r2\_point3x = Center2X + (Width2/2)

r2\_point3y = Center2Y - (Height2/2)

r2\_point4x = Center2X - (Width2/2)

r2\_point4y = Center2Y - (Height2/2)

# Draw two rectangles

turtle.penup()

# First rectangle

turtle.goto(Center1X, Center1Y)

turtle.write("x1, y1", align="center")

turtle.goto(r1\_point1x, r1\_point1y)

turtle.write("First Triangle")

turtle.pendown()

turtle.goto(r1\_point2x, r1\_point2y)

turtle.goto(r1\_point3x, r1\_point3y)

turtle.goto(r1\_point4x, r1\_point4y)

turtle.goto(r1\_point1x, r1\_point1y)

turtle.penup()

# Second rectangle

turtle.goto(Center2X, Center2Y)

turtle.write("x2, y2", align="center")

turtle.goto(r2\_point1x, r2\_point1y)

turtle.write("Second Triangle")

turtle.pendown()

turtle.goto(r2\_point2x, r2\_point2y)

turtle.goto(r2\_point3x, r2\_point3y)

turtle.goto(r2\_point4x, r2\_point4y)

turtle.goto(r2\_point1x, r2\_point1y)

turtle.penup()

# Checking which rectangle is inside or overlap with each other

turtle.pensize(6)

turtle.color("blue")

if(r1x > r2x):

    a = r1x

    b = r2x

else:

    a = r2x

    b = r1x

if(Width1 > Width2):

    c = Width1

    d = Width2

else:

    c = Width2

    d = Width1

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if(r1y > r2y):

    A = r1y

    B = r2y

else:

    A = r2y

    B = r1y

if(Height1 > Height2):

    C = Height1

    D = Height2

else:

    C = Height2

    D = Height1

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

x\_distance = a - b

Width\_distance = c - d

y\_distance = A - B

Height\_distance = C - D

if((x\_distance <= Width\_distance / 2) and (y\_distance <= Height\_distance / 2)):

    if((math.pow(math.pow(r2y - r1y, 2), .05) + r2Height / 2 <= r1Height / 2) and

    (math.pow(math.pow(r2x - r1x, 2), .05) + r2Width / 2 <= r1Width / 2) and

    (r1Height / 2 + r2Height / 2 <= r1Height) and

    (r1Width / 2 + r2Width / 2 <= r1Width)):

        turtle.goto(r2\_point4x, r2\_point4y -50)

        turtle.write("r2 is inside r1", font=("Arial", 16, "normal"))

    elif((math.pow(math.pow(r2y - r1y, 2), .05) + r1Height / 2 <= r2Height / 2) and

    (math.pow(math.pow(r2x - r1x, 2), .05) + r1Width / 2 <= r2Width / 2) and

    (r2Height / 2 + r1Height / 2 <= r2Height) and

    (r2Width / 2 + r1Width / 2 <= r2Width)):

        turtle.goto(r2\_point4x, r2\_point4y- 50)

        turtle.write("r1 is inside r2", font=("Arial", 16, "normal"))

x\_distance = a - b

Width\_distance = c + d

y\_distance = A - B

Height\_distance = C + D

if ((x\_distance <= Width\_distance / 2) and (y\_distance <= Height\_distance / 2)):

    turtle.goto(r1\_point4x, r1\_point4y - 30)

    turtle.write("Both rectangles overlap with each other", font=("Arial", 16, "normal"))

else:

    turtle.goto(r1\_point4x, r1\_point4y)

    turtle.write("Both rectangles do not overlap with each other", font=("Arial", 16, "normal"))

turtle.done()