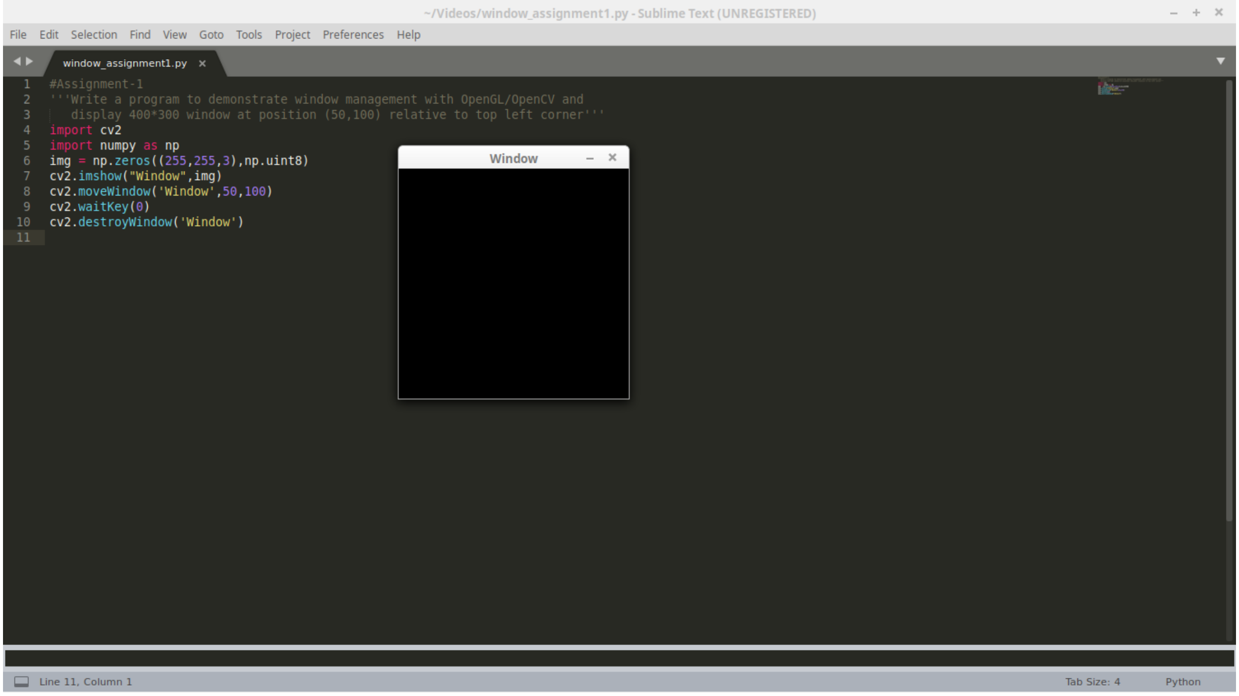
1. Write a program to demonstrate window management and display 400*300 window at position (50,100) relative to top left corner

In [ ]:

```
#Assignment-1
'''Write a program to demonstrate window management with OpenGL/OpenCV and
   display 400*300 window at position (50,100) relative to top left corner''
'
import cv2
import numpy as np
img = np.zeros((255,255,3),np.uint8)
cv2.imshow("Window",img)
cv2.moveWindow('Window',50,100)
cv2.waitKey(0)
cv2.destroyWindow('Window')
```

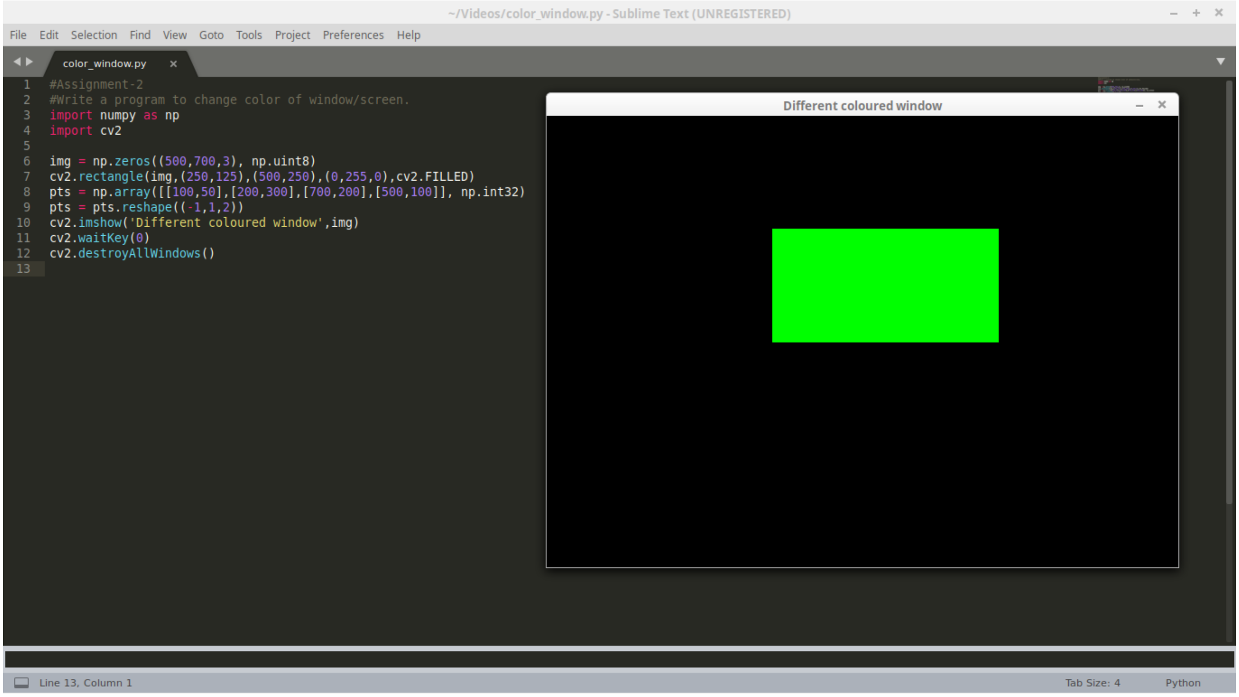

~/Videos/window_assignment1.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
window_assignment1.py
Window
Line 11, Column 1
Tab Size: 4
Python

```
#Assignment-1
'''Write a program to demonstrate window management with OpenGL/OpenCV and
   display 400*300 window at position (50,100) relative to top left corner'''
import cv2
import numpy as np
img = np.zeros((255,255,3),np.uint8)
cv2.imshow("Window",img)
cv2.moveWindow('Window',50,100)
cv2.waitKey(0)
cv2.destroyWindow('Window')
```

2. Write a program to change color of window/screen.

In [ ]:

```
#Assignment-2
#Write a program to change color of window/screen.
import numpy as np
import cv2

img = np.zeros((500,700,3), np.uint8)
cv2.rectangle(img,(250,125),(500,250),(0,255,0),cv2.FILLED)
pts = np.array([[100,50],[200,300],[700,200],[500,100]], np.int32)
pts = pts.reshape((-1,1,2))
cv2.imshow('Different coloured window',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

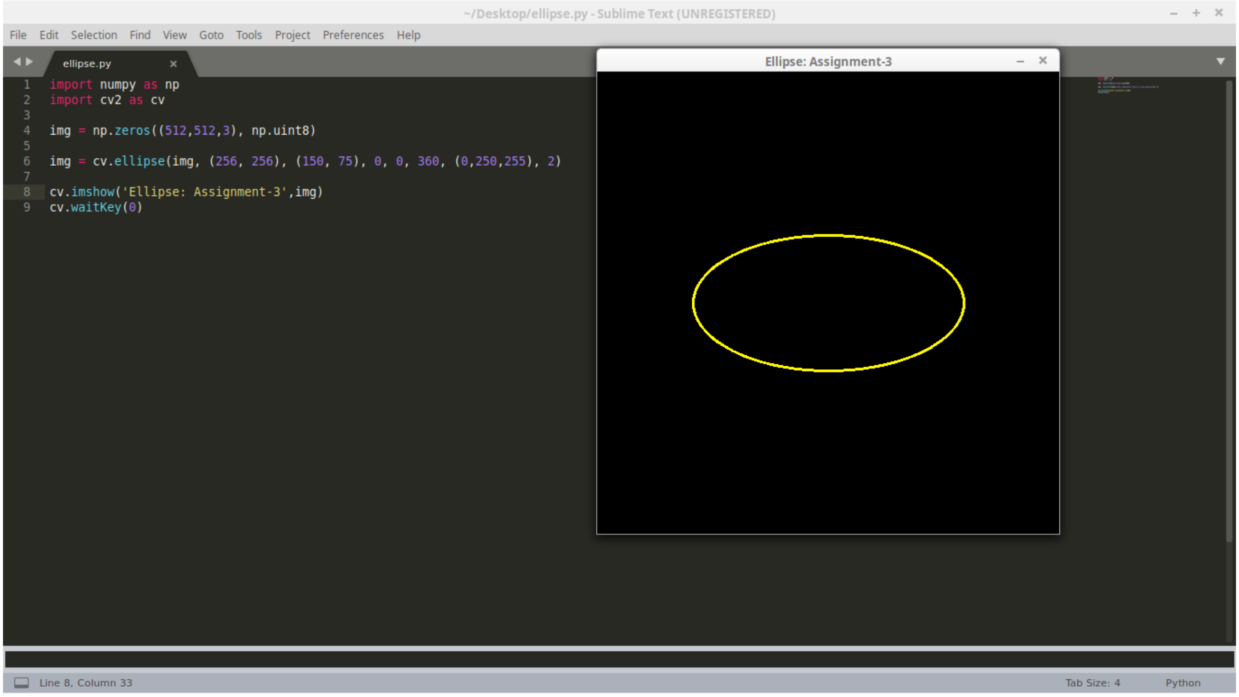

~/Videos/color_window.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
color_window.py
Different coloured window
Line 13, Column 1
Tab Size: 4
Python

```
#Assignment-2
#Write a program to change color of window/screen.
import numpy as np
import cv2

img = np.zeros((500,700,3), np.uint8)
cv2.rectangle(img,(250,125),(500,250),(0,255,0),cv2.FILLED)
pts = np.array([[100,50],[200,300],[700,200],[500,100]], np.int32)
pts = pts.reshape((-1,1,2))
cv2.imshow('Different coloured window',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3. Write a program to draw an ellipse.

In [ ]:

```
#Assignment-3
#Draw an ellipse.

import numpy as np
import cv2 as cv

img = np.zeros((512,512,3), np.uint8)

img = cv.ellipse(img, (256, 256), (150, 75), 0, 0, 360, (0,250,255), 2)

cv.imshow('Ellipse: Assignment-3',img)
cv.waitKey(0)
```



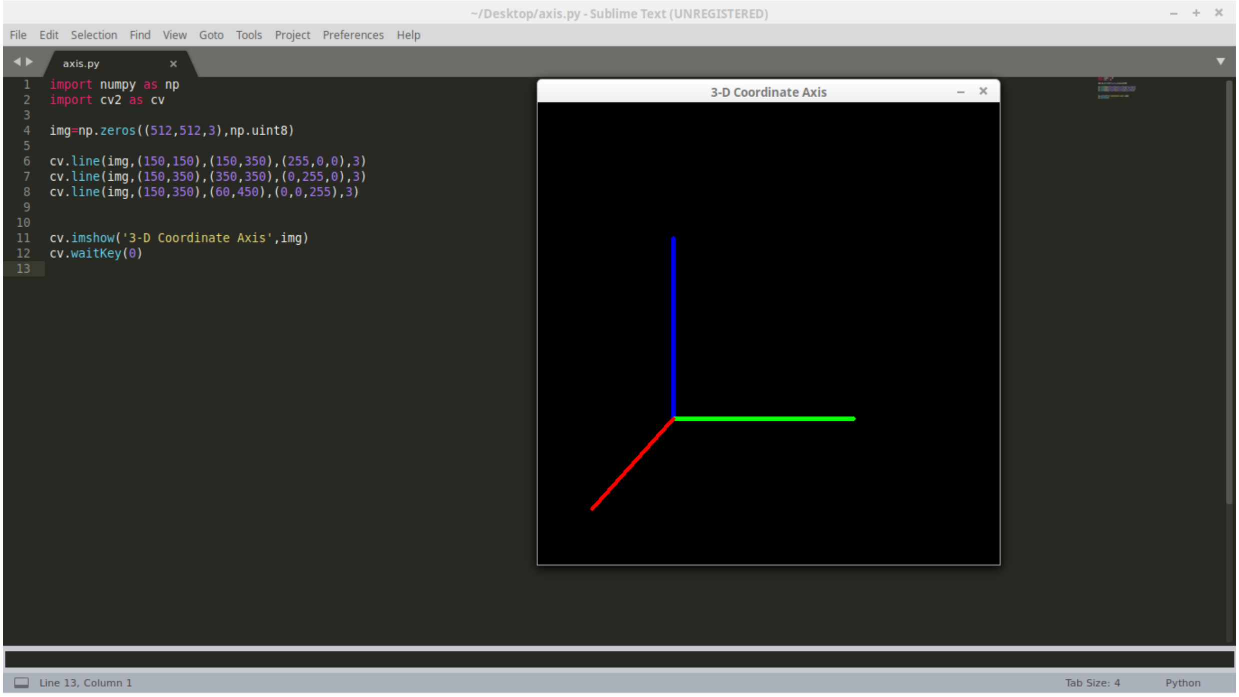
~/Desktop/ellipse.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
ellipse.py
Ellipse: Assignment-3
Line 8, Column 33
Tab Size: 4
Python

```
import numpy as np
import cv2 as cv

img = np.zeros((512,512,3), np.uint8)

img = cv.ellipse(img, (256, 256), (150, 75), 0, 0, 360, (0,250,255), 2)

cv.imshow('Ellipse: Assignment-3',img)
cv.waitKey(0)
```

4. Write a program to draw X-Y-Z Coordinate axis system.

In [ ]:

```
#Assignment-4
#Draw X-Y-Z Coordinate axis system

import numpy as np
import cv2 as cv

img=np.zeros((512,512,3),np.uint8)

cv.line(img,(150,150),(150,350),(255,0,0),3)
cv.line(img,(150,350),(350,350),(0,255,0),3)
cv.line(img,(150,350),(60,450),(0,0,255),3)


cv.imshow('3-D Coordinate Axis',img)
cv.waitKey(0)
```

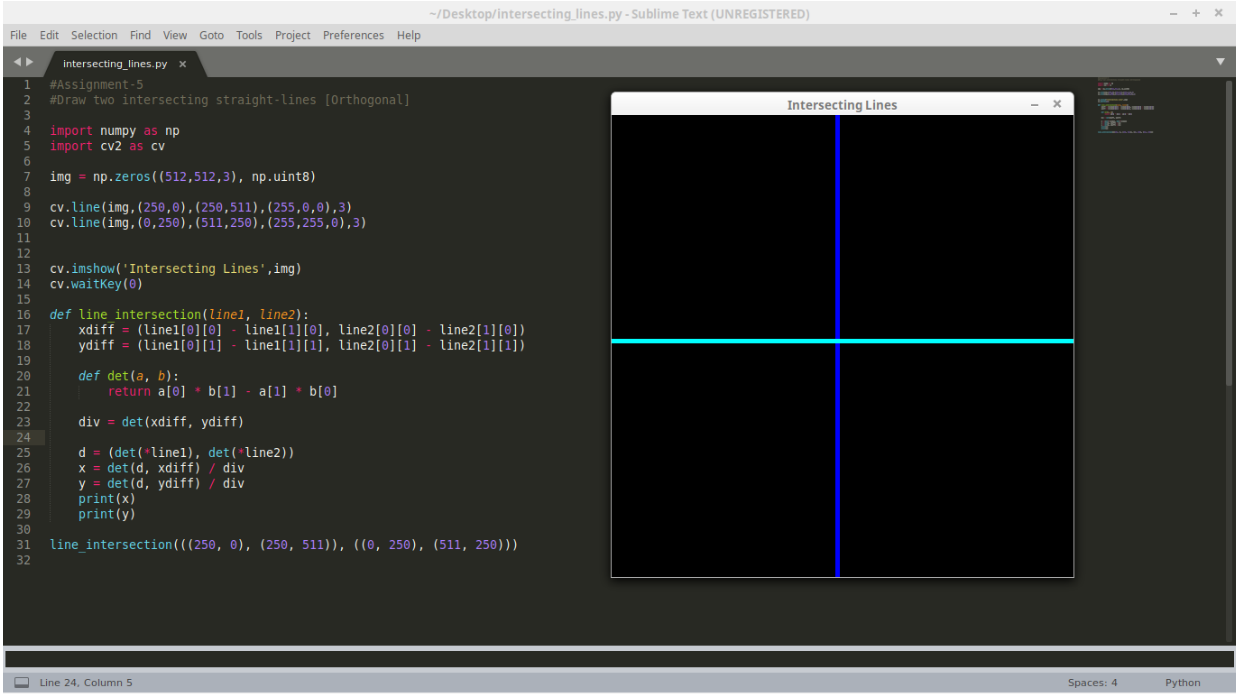

~/Desktop/axis.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
axis.py
3-D Coordinate Axis
Line 13, Column 1
Tab Size: 4
Python

```
import numpy as np
import cv2 as cv

img=np.zeros((512,512,3),np.uint8)

cv.line(img,(150,150),(150,350),(255,0,0),3)
cv.line(img,(150,350),(350,350),(0,255,0),3)
cv.line(img,(150,350),(60,450),(0,0,255),3)


cv.imshow('3-D Coordinate Axis',img)
cv.waitKey(0)
```

5. Write a program to draw two intersecting straight-lines [Orthogonal]

In [ ]:

```
#Assignment-5
#Draw two intersecting straight-lines [Orthogonal]

import numpy as np
import cv2 as cv

img = np.zeros((512,512,3), np.uint8)

cv.line(img,(250,0),(250,511),(255,0,0),3)
cv.line(img,(0,250),(511,250),(255,255,0),3)


cv.imshow('Intersecting Lines',img)
cv.waitKey(0)

def line_intersection(line1, line2):
    xdiff = (line1[0][0] - line1[1][0], line2[0][0] - line2[1][0])
    ydiff = (line1[0][1] - line1[1][1], line2[0][1] - line2[1][1])

    def det(a, b):
        return a[0] * b[1] - a[1] * b[0]

    div = det(xdiff, ydiff)

    d = (det(*line1), det(*line2))
    x = det(d, xdiff) / div
    y = det(d, ydiff) / div
    print(x)
    print(y)

line_intersection(((250, 0), (250, 511)), ((0, 250), (511, 250)))
```

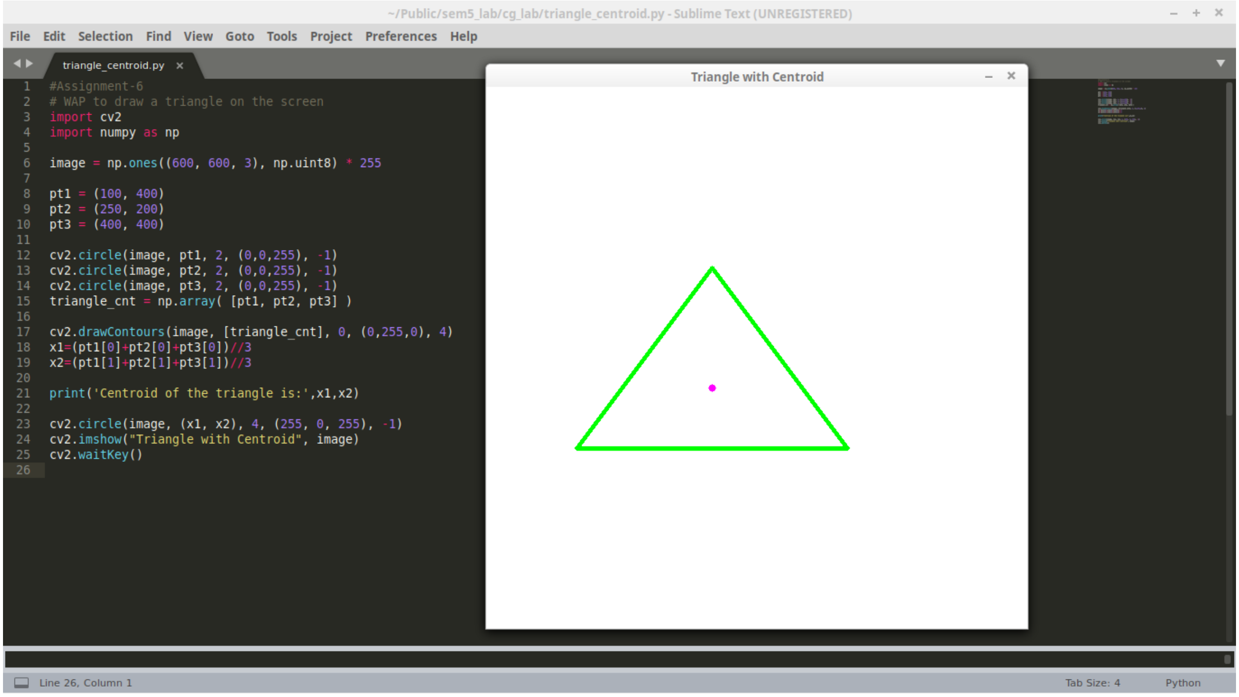

~/Desktop/intersecting_lines.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
intersecting_lines.py
Intersecting Lines
Line 24, Column 5
Spaces: 4
Python

```
#Assignment-5
#Draw two intersecting straight-lines [Orthogonal]

import numpy as np
import cv2 as cv

img = np.zeros((512,512,3), np.uint8)

cv.line(img,(250,0),(250,511),(255,0,0),3)
cv.line(img,(0,250),(511,250),(255,255,0),3)


cv.imshow('Intersecting Lines',img)
cv.waitKey(0)

def line_intersection(line1, line2):
    xdiff = (line1[0][0] - line1[1][0], line2[0][0] - line2[1][0])
    ydiff = (line1[0][1] - line1[1][1], line2[0][1] - line2[1][1])

    def det(a, b):
        return a[0] * b[1] - a[1] * b[0]

    div = det(xdiff, ydiff)

    d = (det(*line1), det(*line2))
    x = det(d, xdiff) / div
    y = det(d, ydiff) / div
    print(x)
    print(y)

line_intersection(((250, 0), (250, 511)), ((0, 250), (511, 250)))
```

6. Write a program to to draw a triangle on the screen and find its centroid.

In [ ]:

```
#Assignment-6
# WAP to draw a triangle on the screen
import cv2
import numpy as np

image = np.ones((600, 600, 3), np.uint8) * 255

#Coordinates of the Vertices of the triangle
pt1 = (100, 400)
pt2 = (250, 200)
pt3 = (400, 400)

cv2.circle(image, pt1, 2, (0,0,255), -1)
cv2.circle(image, pt2, 2, (0,0,255), -1)
cv2.circle(image, pt3, 2, (0,0,255), -1)
triangle_cnt = np.array( [pt1, pt2, pt3] )

cv2.drawContours(image, [triangle_cnt], 0, (0,255,0), 4)
x1=(pt1[0]+pt2[0]+pt3[0])//3
x2=(pt1[1]+pt2[1]+pt3[1])//3

print('Centroid of the triangle is:',x1,x2)

#Centroid is (250,333.33)

cv2.circle(image, (x1, x2), 4, (255, 0, 255), -1)
cv2.imshow("Triangle with Centroid", image)
cv2.waitKey()
```

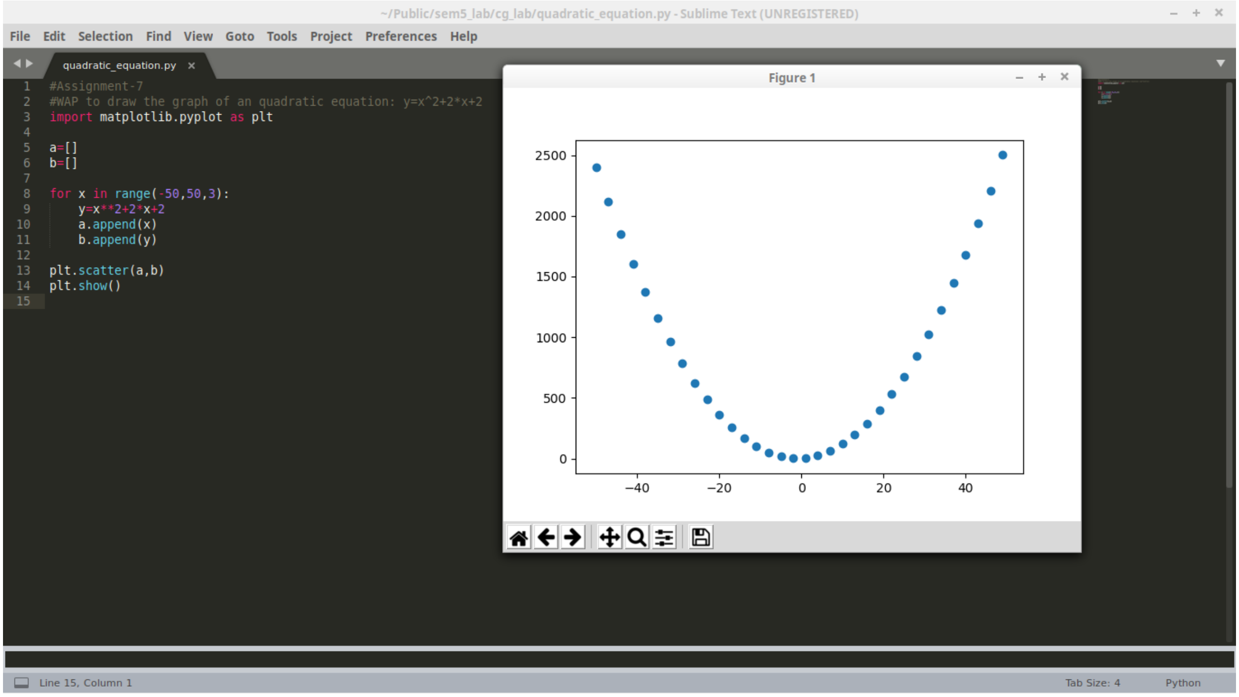

~/Public/sem5_lab/cg_lab/triangle_centroid.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
triangle_centroid.py
Triangle with Centroid
Line 26, Column 1
Tab Size: 4
Python

```
#Assignment-6
# WAP to draw a triangle on the screen
import cv2
import numpy as np

image = np.ones((600, 600, 3), np.uint8) * 255

pt1 = (100, 400)
pt2 = (250, 200)
pt3 = (400, 400)

cv2.circle(image, pt1, 2, (0,0,255), -1)
cv2.circle(image, pt2, 2, (0,0,255), -1)
cv2.circle(image, pt3, 2, (0,0,255), -1)
triangle_cnt = np.array( [pt1, pt2, pt3] )

cv2.drawContours(image, [triangle_cnt], 0, (0,255,0), 4)
x1=(pt1[0]+pt2[0]+pt3[0])//3
x2=(pt1[1]+pt2[1]+pt3[1])//3

print('Centroid of the triangle is:',x1,x2)

cv2.circle(image, (x1, x2), 4, (255, 0, 255), -1)
cv2.imshow("Triangle with Centroid", image)
cv2.waitKey()
```

7. Write a program to draw the graph of an quadratic equation: y=x^2+2*x+2

In [ ]:

```
#Assignment-7
#WAP to draw the graph of an quadratic equation: y=x^2+2*x+2
import matplotlib.pyplot as plt
from tkinter import *

a=[]
b=[]

for x in range(-50,50,3):
    y=x**2+2*x+2
    a.append(x)
    b.append(y)

plt.scatter(a,b)
plt.show()
```



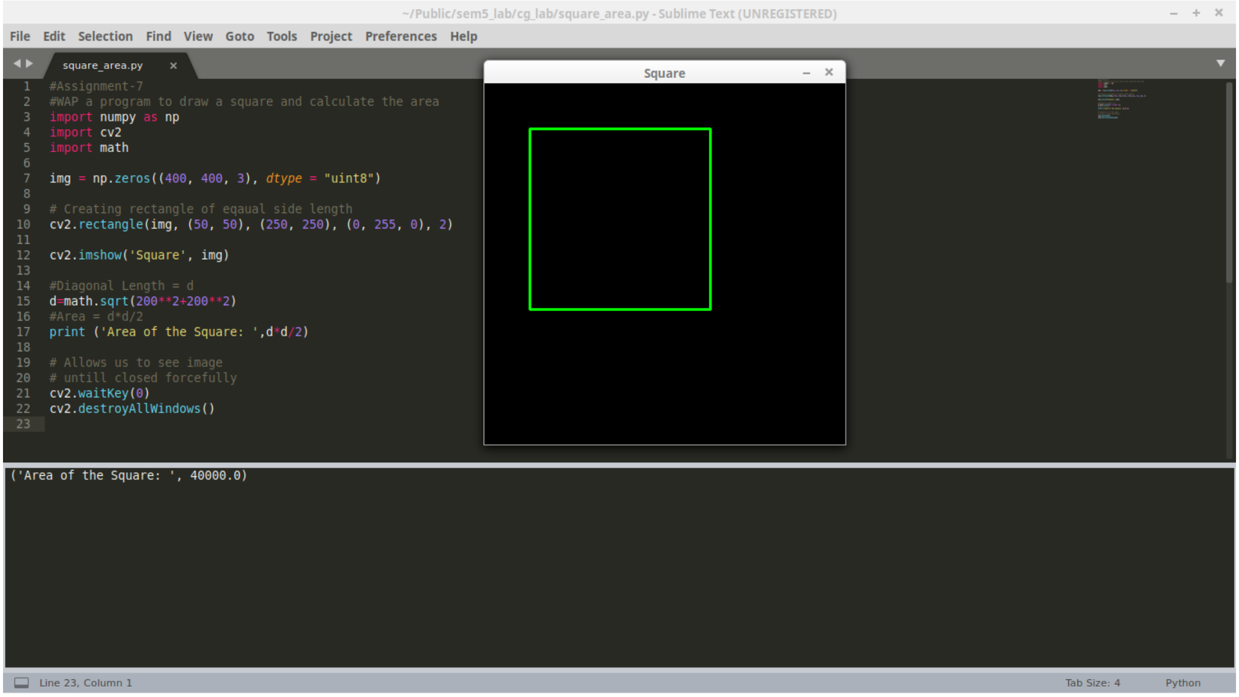
~/Public/sem5_lab/cg_lab/quadratic_equation.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
quadratic_equation.py
Figure 1
2500
2000
1500
1000
500
0
−40
−20
0
20
40
Line 15, Column 1
Tab Size: 4
Python

```
#Assignment-7
#WAP to draw the graph of an quadratic equation: y=x^2+2*x+2
import matplotlib.pyplot as plt

a=[]
b=[]

for x in range(-50,50,3):
    y=x**2+2*x+2
    a.append(x)
    b.append(y)

plt.scatter(a,b)
plt.show()
```

8. Write a program to draw a square and calculate the area

In [ ]:

```
#Assignment-8
#WAP a program to draw a square and calculate the area
import numpy as np
import cv2
import math

img = np.zeros((400, 400, 3), dtype = "uint8")

# Creating rectangle of eqaual side length
cv2.rectangle(img, (50, 50), (250, 250), (0, 255, 0), 2)

cv2.imshow('Square', img)

#Diagonal Length = d
d=math.sqrt(200**2+200**2)
#Area = d*d/2
print ('Area of the Square: ',d*d/2)

# Allows us to see image
# untill closed forcefully
cv2.waitKey(0)
cv2.destroyAllWindows()
```


~/Public/sem5_lab/cg_lab/square_area.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
square_area.py
Square
Line 23, Column 1
Tab Size: 4
Python

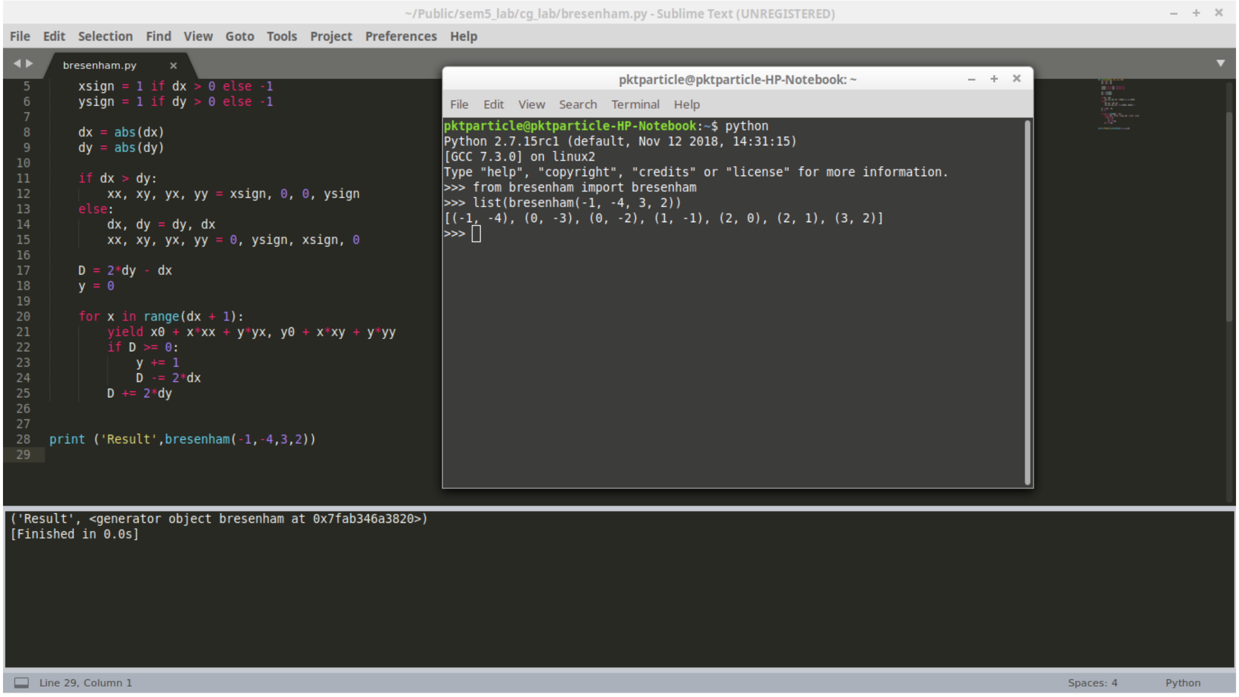
```python
#Assignment-7
#WAP a program to draw a square and calculate the area
import numpy as np
import cv2
import math

img = np.zeros((400, 400, 3), dtype = "uint8")

# Creating rectangle of eqaual side length
cv2.rectangle(img, (50, 50), (250, 250), (0, 255, 0), 2)

cv2.imshow('Square', img)

#Diagonal Length = d
d=math.sqrt(200**2+200**2)
#Area = d*d/2
print ('Area of the Square: ',d*d/2)

# Allows us to see image
# untill closed forcefully
cv2.waitKey(0)
cv2.destroyAllWindows()

```

```
('Area of the Square: ', 40000.0)
```

9. Write a program to implement Bresenham's line algorithm and test it for a given point.

In [ ]:

```python
#Assignment-9
#WAP to implement Bresenham's line algorithm and test it for a given point.
def bresenham(x0, y0, x1, y1):
    dx = x1 - x0
    dy = y1 - y0

    xsign = 1 if dx > 0 else -1
    ysign = 1 if dy > 0 else -1

    dx = abs(dx)
    dy = abs(dy)

    if dx > dy:
        xx, xy, yx, yy = xsign, 0, 0, ysign
    else:
        dx, dy = dy, dx
        xx, xy, yx, yy = 0, ysign, xsign, 0

    D = 2*dy - dx
    y = 0

    for x in range(dx + 1):
        yield x0 + x*xx + y*yx, y0 + x*xy + y*yy
        if D >= 0:
            y += 1
            D -= 2*dx
        D += 2*dy


print ('Result',bresenham(-1,-4,3,2))
```


~/Public/sem5_lab/cg_lab/bresenham.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
bresenham.py
pktparticle@pktparticle-HP-Notebook: ~
File Edit View Search Terminal Help
Line 29, Column 1
Spaces: 4
Python

```
    xsign = 1 if dx > 0 else -1
    ysign = 1 if dy > 0 else -1

    dx = abs(dx)
    dy = abs(dy)

    if dx > dy:
        xx, xy, yx, yy = xsign, 0, 0, ysign
    else:
        dx, dy = dy, dx
        xx, xy, yx, yy = 0, ysign, xsign, 0

    D = 2*dy - dx
    y = 0

    for x in range(dx + 1):
        yield x0 + x*xx + y*yx, y0 + x*xy + y*yy
        if D >= 0:
            y += 1
            D -= 2*dx
        D += 2*dy


print ('Result',bresenham(-1,-4,3,2))
```

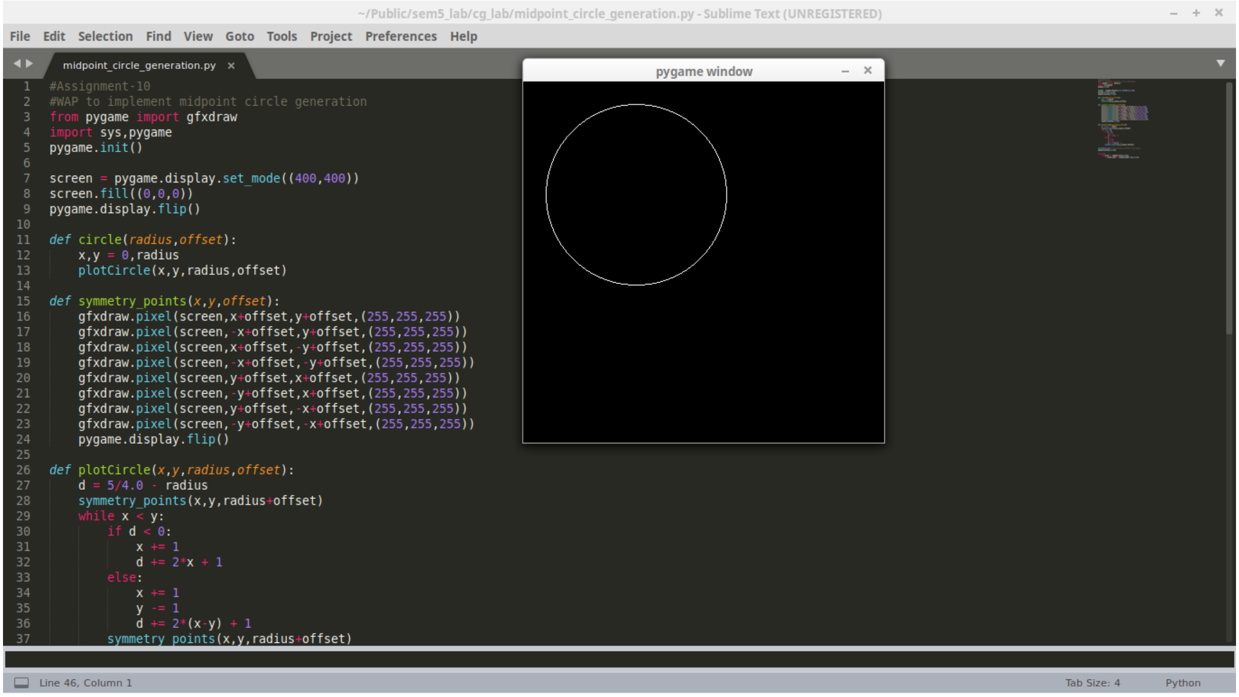
```
pktparticle@pktparticle-HP-Notebook:~$ python
Python 2.7.15rc1 (default, Nov 12 2018, 14:31:15)
[GCC 7.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from bresenham import bresenham
>>> list(bresenham(-1, -4, 3, 2))
[(-1, -4), (0, -3), (0, -2), (1, -1), (2, 0), (2, 1), (3, 2)]
>>> 
```

```
('Result', <generator object bresenham at 0x7fab346a3820>)
[Finished in 0.0s]
```

10. Write a program to implement midpoint circle generation.

In [ ]:

```
#Assignment-10
#WAP to implement midpoint circle generation
from pygame import gfxdraw
import sys,pygame
pygame.init()

screen = pygame.display.set_mode((400,400))
screen.fill((0,0,0))
pygame.display.flip()

def circle(radius,offset):
        x,y = 0,radius
        plotCircle(x,y,radius,offset)

def symmetry_points(x,y,offset):
        gfxdraw.pixel(screen,x+offset,y+offset,(255,255,255))
        gfxdraw.pixel(screen,-x+offset,y+offset,(255,255,255))
        gfxdraw.pixel(screen,x+offset,-y+offset,(255,255,255))
        gfxdraw.pixel(screen,-x+offset,-y+offset,(255,255,255))
        gfxdraw.pixel(screen,y+offset,x+offset,(255,255,255))
        gfxdraw.pixel(screen,-y+offset,x+offset,(255,255,255))
        gfxdraw.pixel(screen,y+offset,-x+offset,(255,255,255))
        gfxdraw.pixel(screen,-y+offset,-x+offset,(255,255,255))
        pygame.display.flip()

def plotCircle(x,y,radius,offset):
        d = 5/4.0 - radius
        symmetry_points(x,y,radius+offset)
        while x < y:
                if d < 0:
                        x += 1
                        d += 2*x + 1
                else:
                        x += 1
                        y -= 1
                        d += 2*(x-y) + 1
                symmetry_points(x,y,radius+offset)

circle(100,25) # circle(radius,<offset from edge>)
pygame.display.flip()

while 1:
        for event in pygame.event.get():
                if event.type == pygame.QUIT: sys.exit()
```


~/Public/sem5_lab/cg_lab/midpoint_circle_generation.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
midpoint_circle_generation.py ×
pygame window
Line 46, Column 1
Tab Size: 4
Python

```python
#Assignment-10
#WAP to implement midpoint circle generation
from pygame import gfxdraw
import sys,pygame
pygame.init()

screen = pygame.display.set_mode((400,400))
screen.fill((0,0,0))
pygame.display.flip()

def circle(radius,offset):
    x,y = 0,radius
    plotCircle(x,y,radius,offset)

def symmetry_points(x,y,offset):
    gfxdraw.pixel(screen,x+offset,y+offset,(255,255,255))
    gfxdraw.pixel(screen,-x+offset,y+offset,(255,255,255))
    gfxdraw.pixel(screen,x+offset,-y+offset,(255,255,255))
    gfxdraw.pixel(screen,-x+offset,-y+offset,(255,255,255))
    gfxdraw.pixel(screen,y+offset,x+offset,(255,255,255))
    gfxdraw.pixel(screen,-y+offset,x+offset,(255,255,255))
    gfxdraw.pixel(screen,y+offset,-x+offset,(255,255,255))
    gfxdraw.pixel(screen,-y+offset,-x+offset,(255,255,255))
    pygame.display.flip()

def plotCircle(x,y,radius,offset):
    d = 5/4.0 - radius
    symmetry_points(x,y,radius+offset)
    while x < y:
        if d < 0:
            x += 1
            d += 2*x + 1
        else:
            x += 1
            y -= 1
            d += 2*(x-y) + 1
        symmetry_points(x,y,radius+offset)
```