In [2]:
```
import numpy as np
```

In [3]:
```
a=np.array([[9.0,8.0,7.0],[1.0,2.0,3.0]])
print(a)
a.ndim
```
```
[[9. 8. 7.]
 [1. 2. 3.]]
```
Out[3]:
```
2
```

In [4]:
```
a.shape
```
Out[4]:
```
(2, 3)
```

In [5]:
```
a.dtype
```
Out[5]:
```
dtype('float64')
```

In [6]:
```
a.itemsize
```
Out[6]:
```
8
```

In [7]:
```
a.size
```
Out[7]:
```
6
```

In [8]:
```
b=np.array([[12,10,14,13,11,15],[4,5,6,1,7,8]])
print(b)
b.shape
```
```
[[12 10 14 13 11 15]
 [ 4  5  6  1  7  8]]
```
Out[8]:
```
(2, 6)
```

In [9]:
```
#getting the elements (row,coloumn)#
b[0,-1]
```
Out[9]:
```
15
```

In [10]:
```
#getting the spcefic row
b[1,  : ]
```
Out[10]:
```
array([4, 5, 6, 1, 7, 8])
```

In [11]:
```
#getting the spcefic coloumn
b[:  ,0]
```
Out[11]:
```
array([12,  4])
```

In [12]:
```
#getting the elements like[start:end:stepindex]
b[0, 1:5:1]
```
Out[12]:
```
array([10, 14, 13, 11])
```

In [13]:
```
#adding the number in the second row at index 5
b[1,5]=20
print(b)

b[:, 5]=[99,100]
print(b)
```
```
[[12 10 14 13 11 15]
 [ 4  5  6  1  7 20]]
[[ 12  10  14  13  11  99]
 [  4   5   6   1   7 100]]
```

In [14]:
```
c=np.array([[[1,2],[3,4]],[[5,6],[7,8]]])
print(c)
c.ndim
```
```
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]
```
Out[14]:
```
3
```

In [15]:
```
c[ 0,1,1]
```
Out[15]:
```
4
```

In [16]:
```
c[1,1,1]
```
Out[16]:
```
8
```

In [17]:
```
c[1,0,1]
```
Out[17]:
```
6
```

In [18]:
```
c[1,0,0]
```
Out[18]:
```
5
```

In [19]:
```
#replace
c[:,1,:]=[[3,4],[7,6]]
print(c)
```
```
[[[1 2]
  [3 4]]

 [[5 6]
  [7 6]]]
```

In [20]:
```
#all zeros matrix
np.zeros((2,2,6))
```
Out[20]:
```
array([[[0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0.]],

       [[0., 0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0., 0.]]])
```

In [21]:
```
np.ones((2,4,3))
```
Out[21]:
```
array([[[1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.]],

       [[1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.]]])
```

In [22]:
```
np.full((5,10),60)
```
Out[22]:
```
array([[60, 60, 60, 60, 60, 60, 60, 60, 60, 60],
       [60, 60, 60, 60, 60, 60, 60, 60, 60, 60],
       [60, 60, 60, 60, 60, 60, 60, 60, 60, 60],
       [60, 60, 60, 60, 60, 60, 60, 60, 60, 60],
       [60, 60, 60, 60, 60, 60, 60, 60, 60, 60]])
```

In [50]:
```
np.random.randint(1,10 , size=(2,5))
```
Out[50]:
```
array([[8, 4, 9, 6, 7],
       [2, 3, 3, 8, 3]])
```

In [24]:
```
np.identity(10)
```
Out[24]:
```
array([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])
```

In [25]:
```
arr=np.array([1,2,3])
r1=np.repeat(arr,5)
print(r1)
```
```
[1 1 1 1 1 2 2 2 2 2 3 3 3 3 3]
```

In [26]:
```
a=np.array([1,2,3,5,6,7])
print(a)
```
```
[1 2 3 5 6 7]
```

In [27]:
```
a+5
```
Out[27]:
```
array([ 6,  7,  8, 10, 11, 12])
```

In [31]:
```
a-1
```
Out[31]:
```
array([[8., 7., 6.],
       [0., 1., 2.]])
```

In [32]:
```
a*3
```
Out[32]:
```
array([[27., 24., 21.],
       [ 3.,  6.,  9.]])
```

In [33]:
```
np.random.randint(0,1 , size=(4,5))
```
Out[33]:
```
array([[0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0]])
```

In [34]:
```
c=np.array([[[1,2],[3,4]],[[5,6],[7,8]]])
print(c)
c.ndim
c.size
```
```
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]
```
Out[34]:
```
8
```

In [35]:
```
c.itemsize
```
Out[35]:
```
4
```

In [36]:
```
#shape is used to axis dimensions in the array
c.shape
```
Out[36]:
```
(2, 2, 2)
```

In [36]:
```
np.arange(30).reshape(2,3,5)
```
Out[36]:
```
array([[[ 0,  1,  2,  3,  4],
        [ 5,  6,  7,  8,  9],
        [10, 11, 12, 13, 14]],

       [[15, 16, 17, 18, 19],
        [20, 21, 22, 23, 24],
        [25, 26, 27, 28, 29]]])
```

In [61]:
```
np.eye(4,6)
```
Out[61]:
```
array([[1., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0.]])
```

In [55]:
```
np.diag([1,2,3])
```
Out[55]:
```
array([[1, 0, 0],
       [0, 2, 0],
       [0, 0, 3]])
```

In [37]:
```
#added one more row an coloumn
np.diag([1, 2, 3], 1)
```
Out[37]:
```
array([[0, 1, 0, 0],
       [0, 0, 2, 0],
       [0, 0, 0, 3],
       [0, 0, 0, 0]])
```

In [38]:
```
#trageting the first two elements in the array and adding via copy command
a = np.array([1, 2, 3, 4, 5, 6])
b = a[:2].copy()
b=b+1
print('a=',a,'b=',b)
```
```
a= [1 2 3 4 5 6] b= [2 3]
```

In [40]:
```
#shape function gives (z,x,y axes)
d=np.array([[[1,2,0],[2,3,0],[5,6,8]],[[4,5,0],[5,6,0],[5,6,8]],[[7,8,0],[9,10,0],[5,6,8]]])
print(d)
d.shape
```
```
[[[ 1  2  0]
  [ 2  3  0]
  [ 5  6  8]]

 [[ 4  5  0]
  [ 5  6  0]
  [ 5  6  8]]

 [[ 7  8  0]
  [ 9 10  0]
  [ 5  6  8]]]
```
Out[40]:
```
(3, 3, 3)
```

In [112…
```
d[2,1,0:2]
```
Out[112]:
```
 array([ 9, 10])
```

In [127…
```
#z axes x:row y:coloumn
d[2,0:2,1]
```
Out[127]:
```
 array([ 8, 10])
```

In [138…
```
d[1,0:]
```
Out[138]:
```
 array([5, 6, 6])
```

In [38]:
```
 z = np.array([[1, 2, 3, 0], [0, 0, 5, 3], [4, 6, 0, 0]])
print(z)
```
```
[[1 2 3 0]
 [0 0 5 3]
 [4 6 0 0]]
```

In [39]:
```
 np.nonzero(z)
```
Out[39]:
```
(array([0, 0, 0, 1, 1, 2, 2], dtype=int64),
 array([0, 1, 2, 2, 3, 0, 1], dtype=int64))
```

In [40]:
```
np.flatnonzero(z)
```
Out[40]:
```
array([0, 1, 2, 6, 7, 8, 9], dtype=int64)
```

In [44]:
```
Z=np.zeros((10,10))
print("%d bytes" % (Z.size * Z.itemsize))
```
```
800 bytes
```

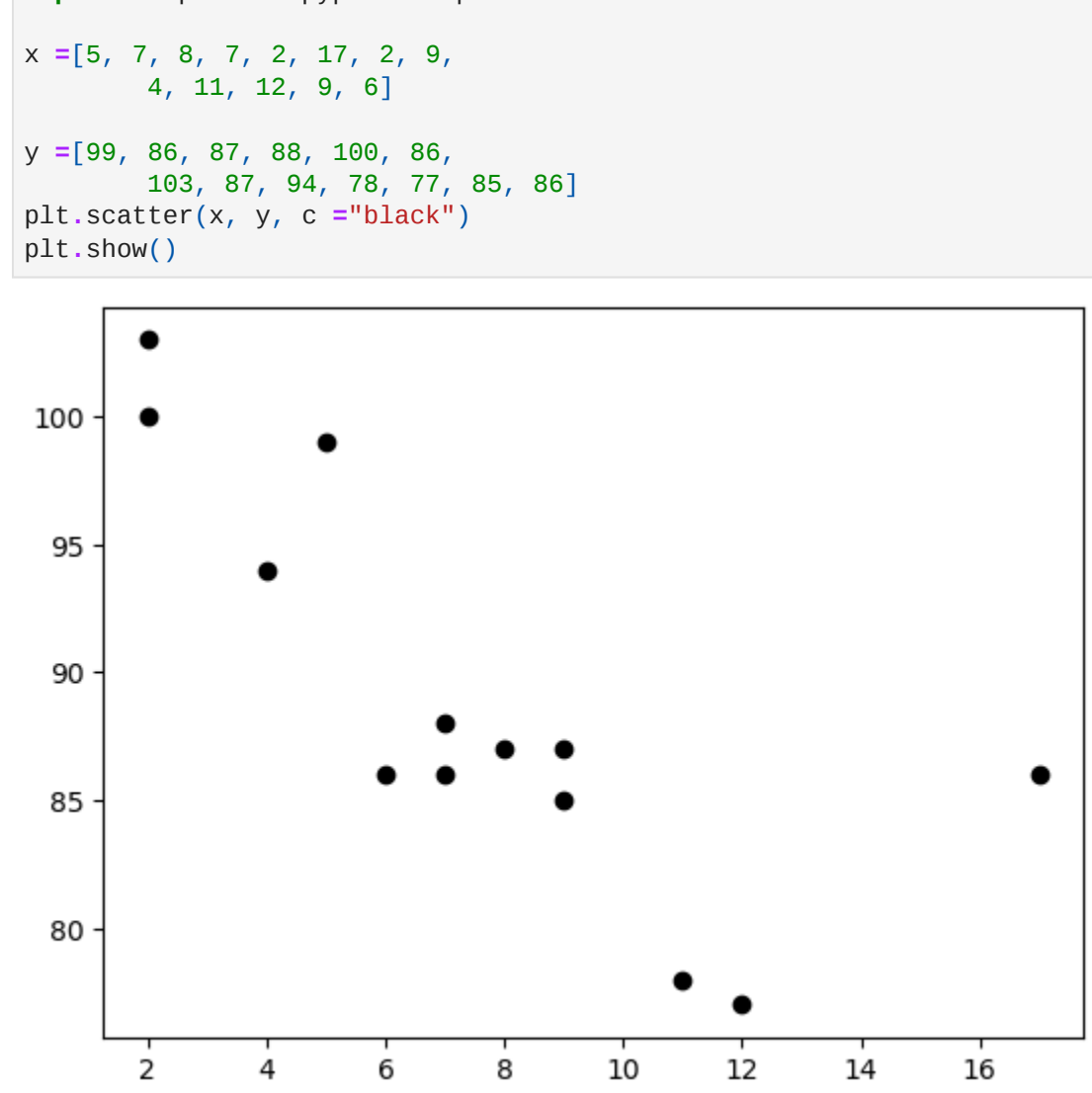
In [45]:
```
#a null vector of size 10 but the fifth value which is 1
Z = np.zeros(10)
Z[4] = 1
print(Z)
```
```
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```

In [49]:
```
Z = np.arange(50)
Z = Z[::-1]
print(Z)
```
```
[49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26
 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2
  1  0]
```

In [48]:
```
Z = np.arange(9).reshape(3, 3)
print(Z)
```
```
[[0 1 2]
 [3 4 5]
 [6 7 8]]
```

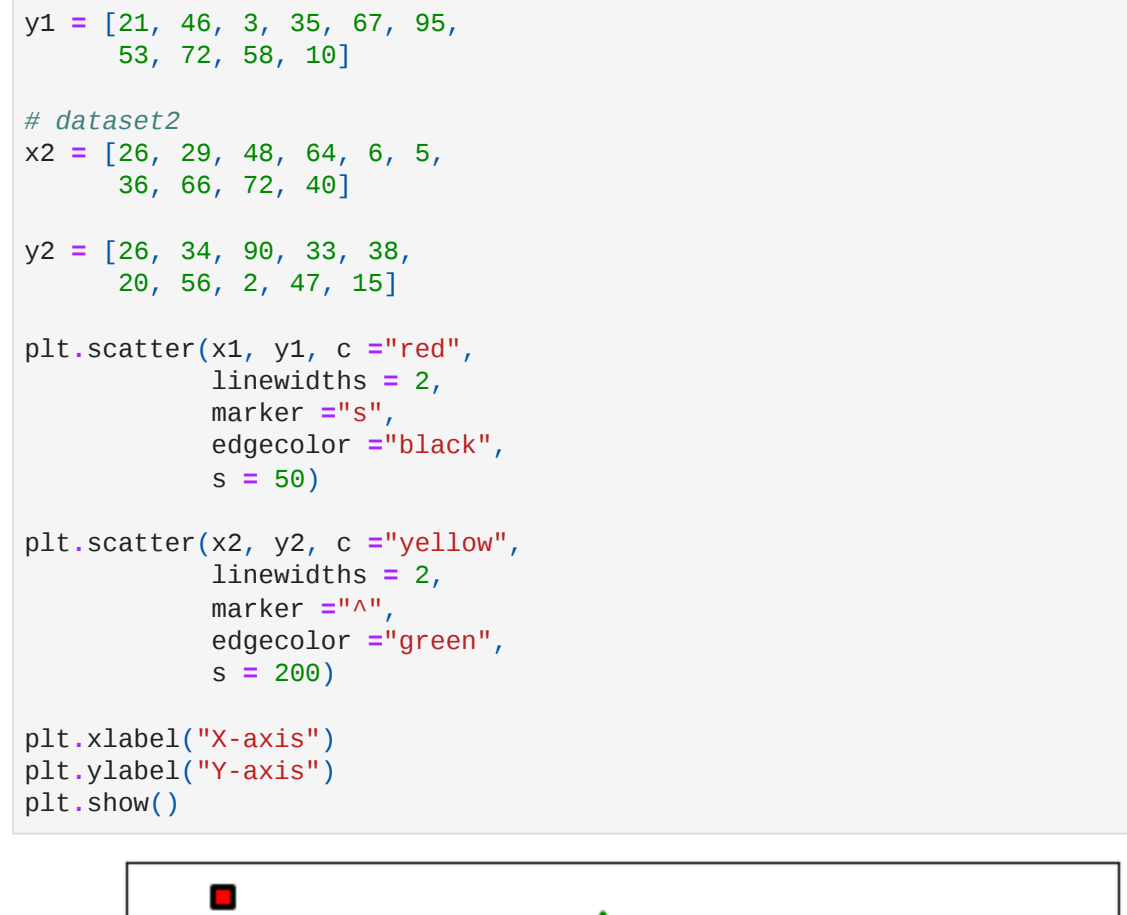
In [52]:
```
#random values in 3*3*3 matrix
Z = np.random.random((3,3,3))
print(Z)
```
```
[[[0.28087893 0.09008563 0.33189387]
  [0.46966046 0.48233036 0.18354132]
  [0.93076387 0.20950017 0.25855131]]

 [[0.20260776 0.01686711 0.29461807]
  [0.42273431 0.09055294 0.40345507]
  [0.61646024 0.05426406 0.94737011]]

 [[0.82240913 0.08018562 0.55827871]
  [0.82121303 0.67661395 0.8408387 ]
  [0.81685496 0.32354503 0.86767249]]]
```

In [1]:
```
import matplotlib.pyplot as plt

price = [2.50, 1.23, 4.02, 3.25, 5.00, 4.40]
sales_per_day = [34, 62, 49, 22, 13, 19]

plt.scatter(price, sales_per_day)
plt.show()
```

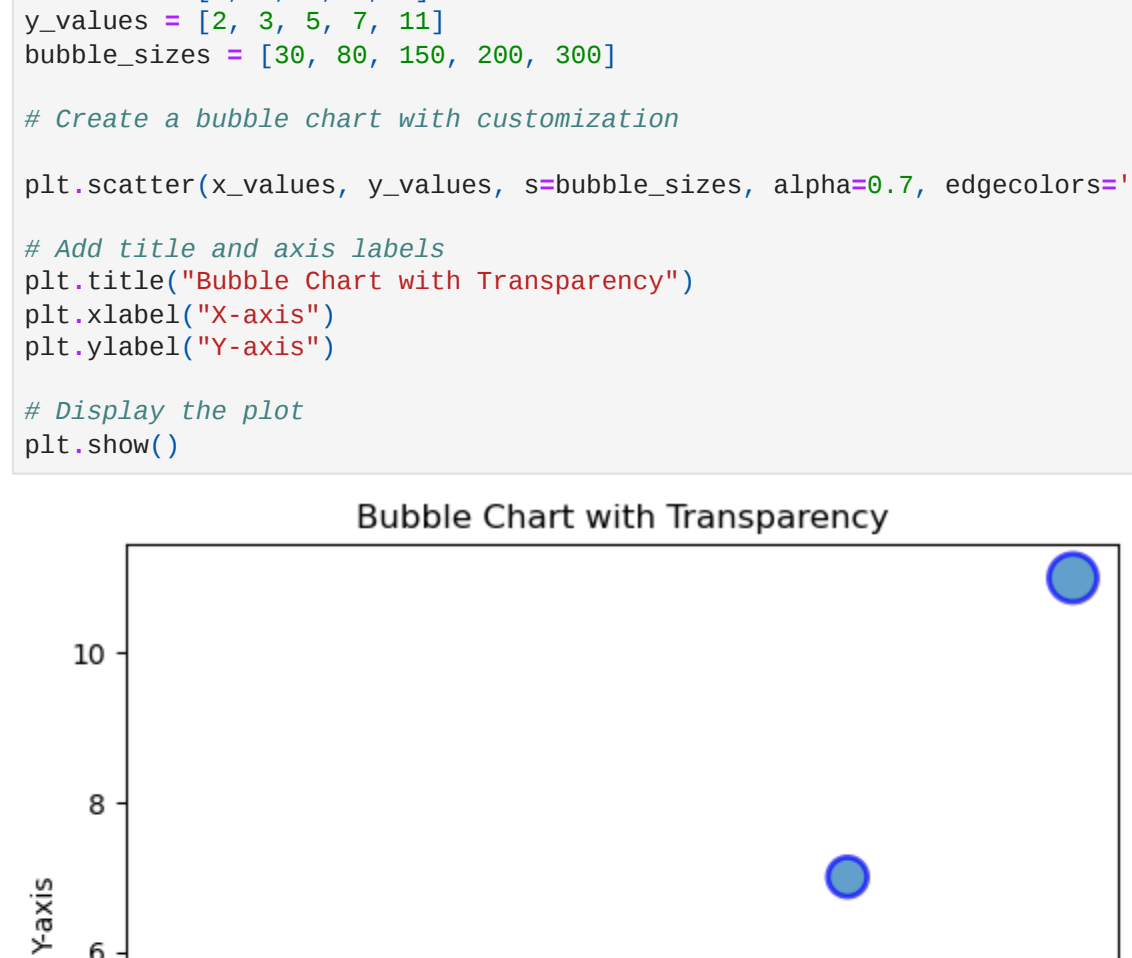


Parameters: x_axis_data: An array containing data for the x-axis.matplotlib s: Marker size, which can be a scalar or an array of size equal to the size of x or y. c: Color of the sequence of colors for markers. marker: Marker style. cmap: Colormap name. linewidths: Width of the marker border. edgecolor: Marker border color. alpha: Blending value, ranging between 0 (transparent) and 1 (opaque).

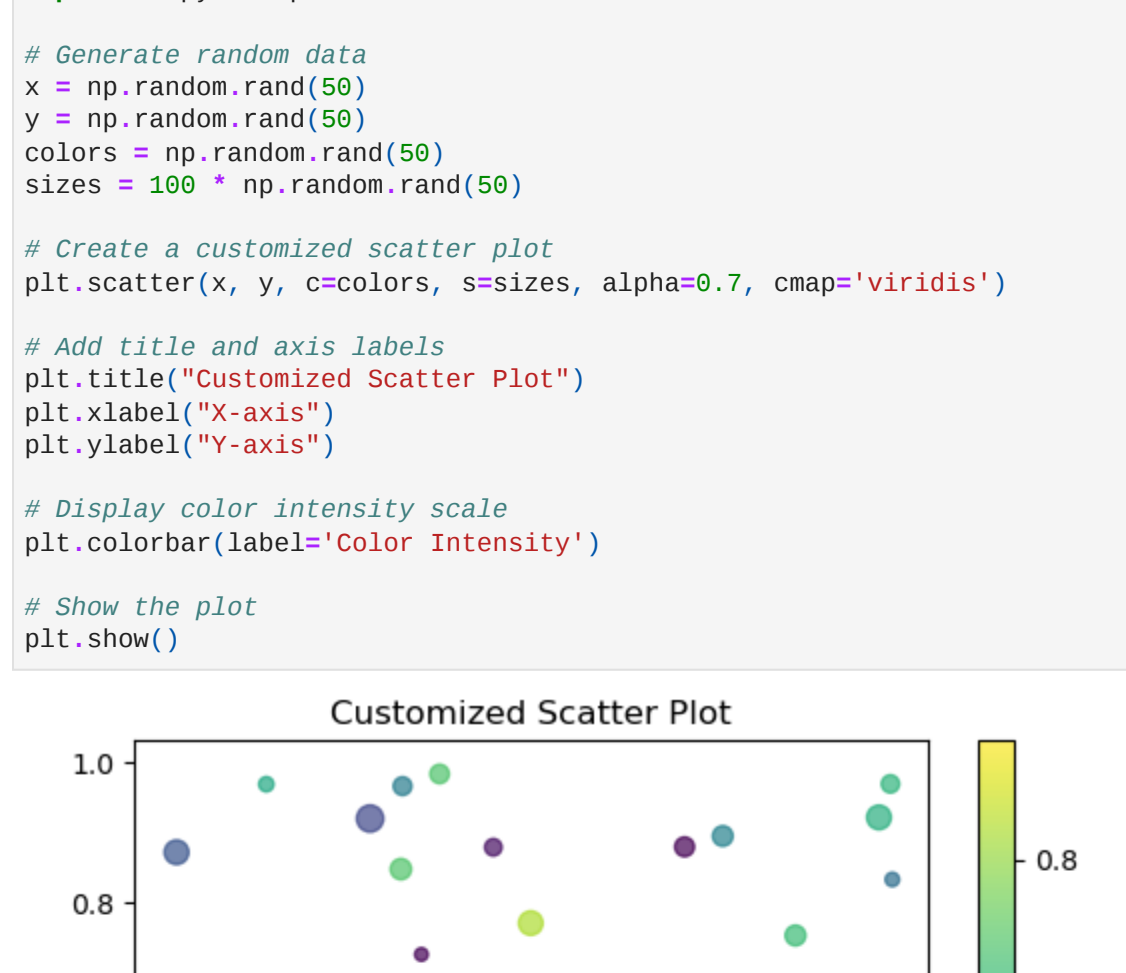
In [42]:
```
import matplotlib.pyplot as plt

x =[5, 7, 8, 7, 2, 17, 2, 9,
        4, 11, 12, 9, 6]

y =[99, 86, 87, 88, 100, 86,
        103, 87, 94, 78, 77, 85, 86]
plt.scatter(x, y, c ="black")
plt.show()
```



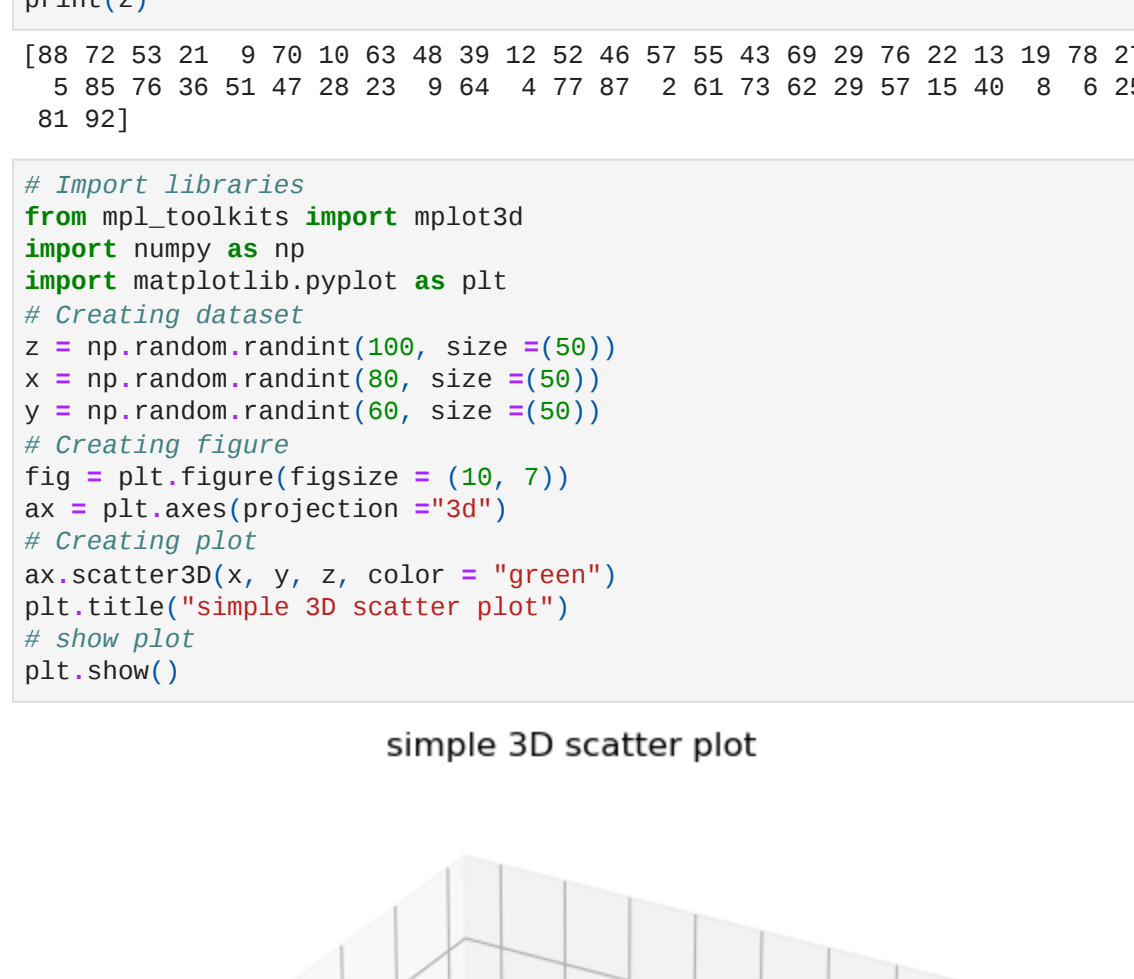
In [43]:
```
# dataset-1
x1 = [89, 43, 36, 36, 95, 10,
      66, 34, 38, 20]

y1 = [21, 46, 3, 35, 67, 95,
      53, 72, 58, 10]

# dataset2
x2 = [26, 29, 48, 64, 6, 5,
      36, 66, 72, 40]

y2 = [26, 34, 90, 33, 38,
      20, 56, 2, 47, 15]

plt.scatter(x1, y1, c ="red",
            linewidths = 2,
            marker ="s",
            edgecolor ="black",
            s = 50)

plt.scatter(x2, y2, c ="yellow",
            linewidths = 2,
            marker ="^",
            edgecolor ="green",
            s = 200)

plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.show()
```



In [44]:
```
# Data
x_values = [1, 2, 3, 4, 5]
y_values = [2, 3, 5, 7, 11]
bubble_sizes = [30, 80, 150, 200, 300]

# Create a bubble chart with customization

plt.scatter(x_values, y_values, s=bubble_sizes, alpha=0.7, edgecolors='b', linewidths=2)

# Add title and axis labels
plt.title("Bubble Chart with Transparency")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

# Display the plot
plt.show()
```

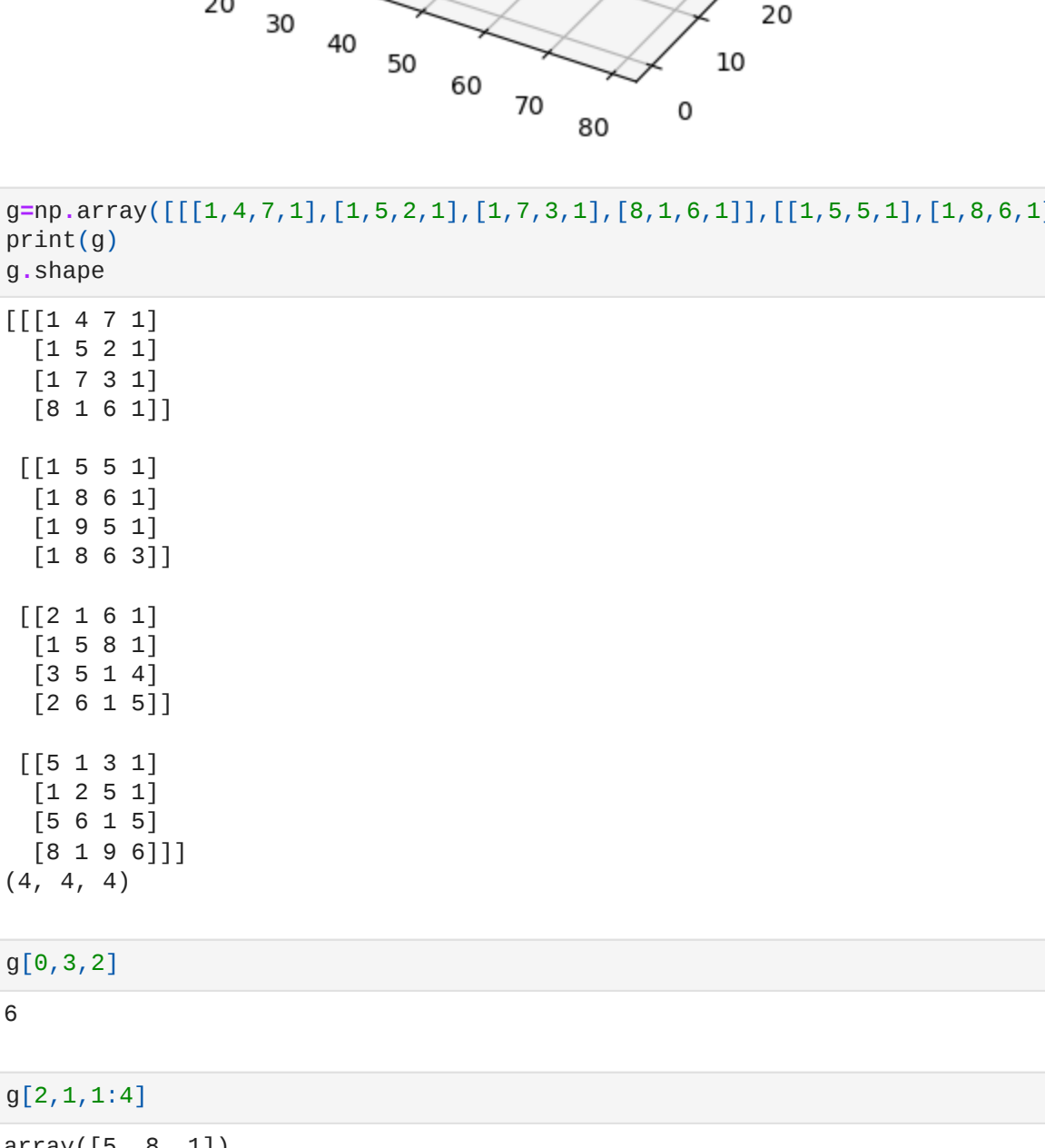


In [46]:
```
import matplotlib.pyplot as plt
import numpy as np

# Generate random data
x = np.random.rand(50)
y = np.random.rand(50)
colors = np.random.rand(50)
sizes = 100 * np.random.rand(50)

# Create a customized scatter plot
plt.scatter(x, y, c=colors, s=sizes, alpha=0.7, cmap='viridis')

# Add title and axis labels
plt.title("Customized Scatter Plot")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

# Display color intensity scale
plt.colorbar(label='Color Intensity')

# Show the plot
plt.show()
```



In [52]:
```
z = np.random.randint(100, size =(50))
print(z)
```
```
[88 72 53 21  9 70 10 63 48 39 12 52 46 57 55 43 69 29 76 22 13 19 78 27
  5 85 76 36 51 47 28 23  9 64  4 77 87  2 61 73 62 29 57 15 40  8  6 25
 81 92]
```

In [53]:
```
# Import libraries
from mpl_toolkits import mplot3d
import numpy as np
import matplotlib.pyplot as plt
# Creating dataset
z = np.random.randint(100, size =(50))
x = np.random.randint(80, size =(50))
y = np.random.randint(60, size =(50))
# Creating figure
fig = plt.figure(figsize = (10, 7))
ax = plt.axes(projection ="3d")
# Creating plot
ax.scatter3D(x, y, z, color = "green")
plt.title("simple 3D scatter plot")
# show plot
plt.show()
```



In [21]:
```
g=np.array([[[1,4,7,1],[1,5,2,1],[1,7,3,1],[8,1,6,1]],[[1,5,5,1],[1,8,6,1],[1,9,5,1],[1,8,6,3]],[[2,1,6,1],[1,5,8,1],[3,5,1,4],[2,6,1,5]],[[5,1,3,1],[1,2,5,1],[5,6,1,5],[8,1,9,6]]])
print(g)
g.shape
```
```
[[[1 4 7 1]
  [1 5 2 1]
  [1 7 3 1]
  [8 1 6 1]]

 [[1 5 5 1]
  [1 8 6 1]
  [1 9 5 1]
  [1 8 6 3]]

 [[2 1 6 1]
  [1 5 8 1]
  [3 5 1 4]
  [2 6 1 5]]

 [[5 1 3 1]
  [1 2 5 1]
  [5 6 1 5]
  [8 1 9 6]]]
```
Out[21]:
```
(4, 4, 4)
```

In [22]:
```
g[0,3,2]
```
Out[22]:
```
6
```

In [24]:
```
g[2,1,1:4]
```
Out[24]:
```
array([5, 8, 1])
```

In [29]:

In [ ]: