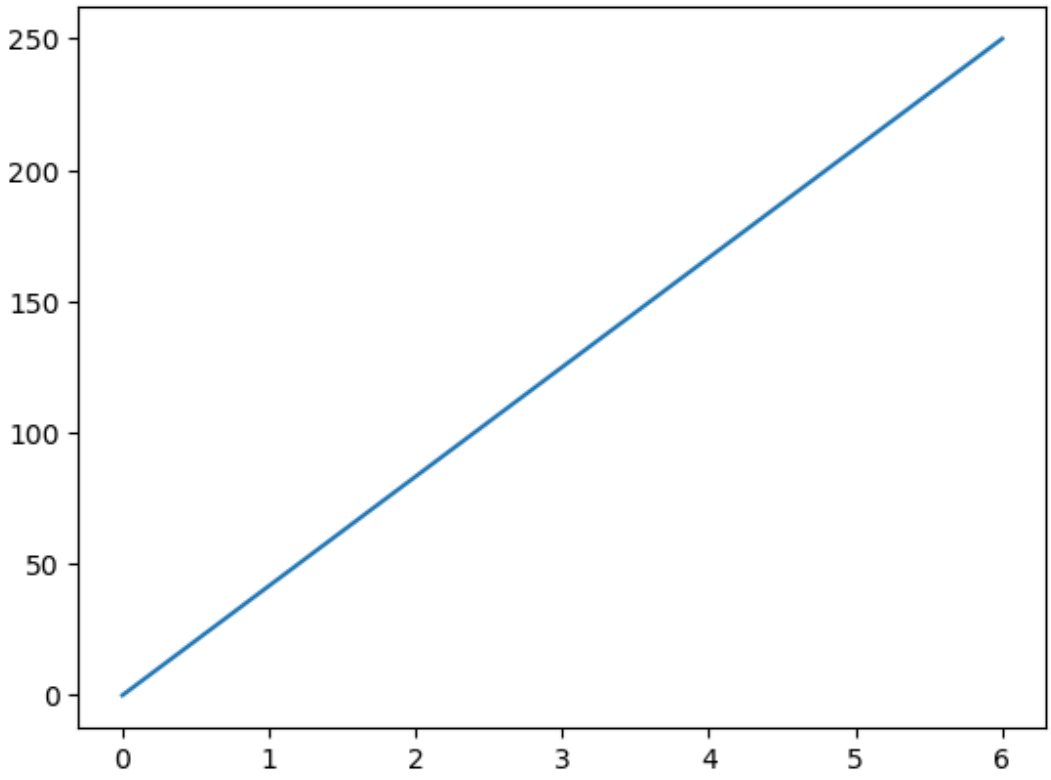
Lab 8

Jenit Harnesha

21052158

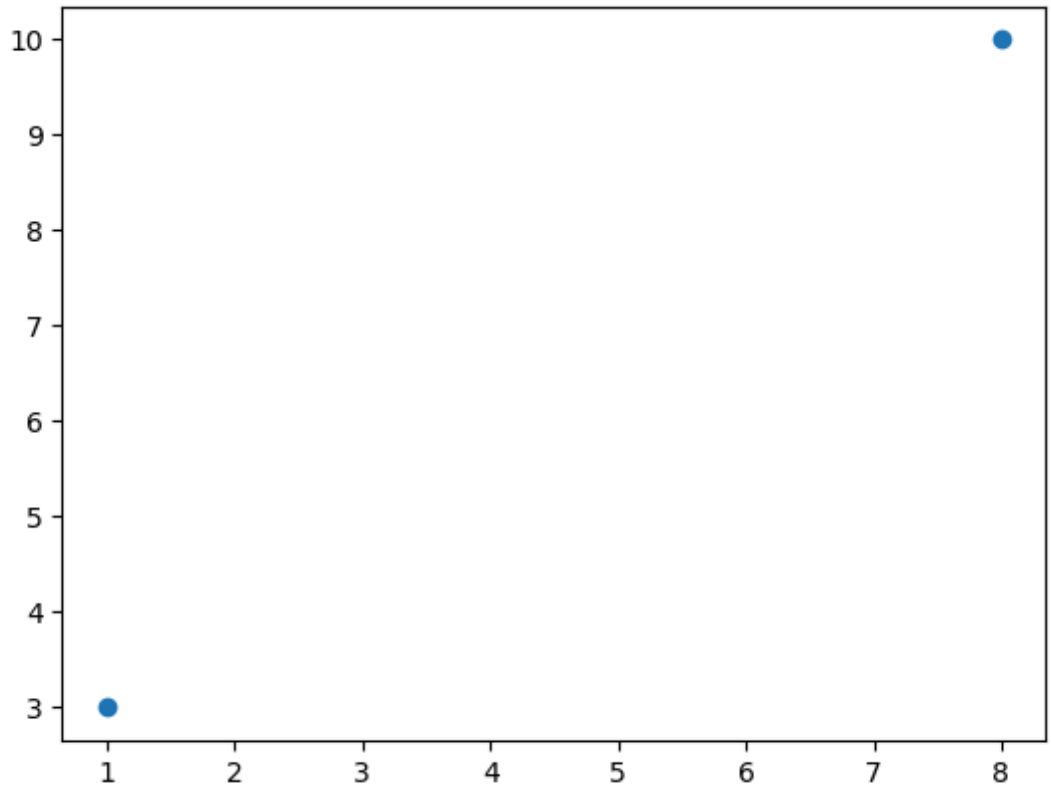
In [1]:
```
import matplotlib
```

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([0,6])
ypoints = np.array([0,250])
plt.plot(xpoints,ypoints)
plt.show()
```



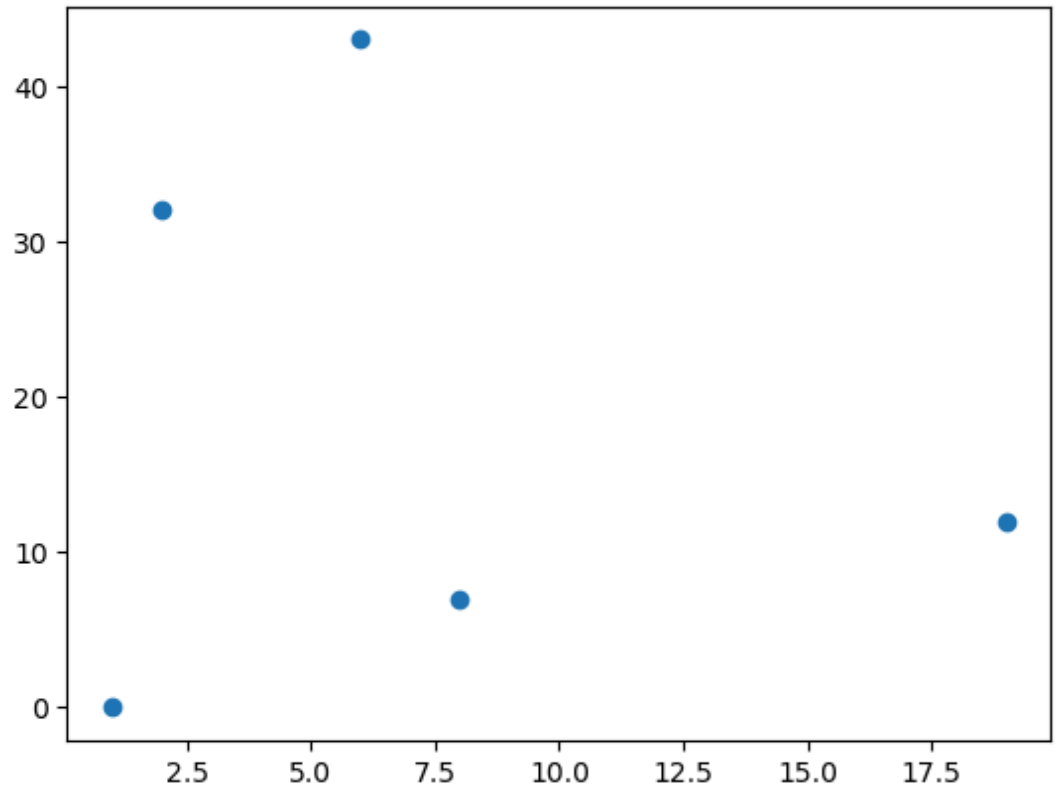
In [3]:

```
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([1,8])
ypoints = np.array([3,10])
plt.plot(xpoints,ypoints,'o')
plt.show()
```



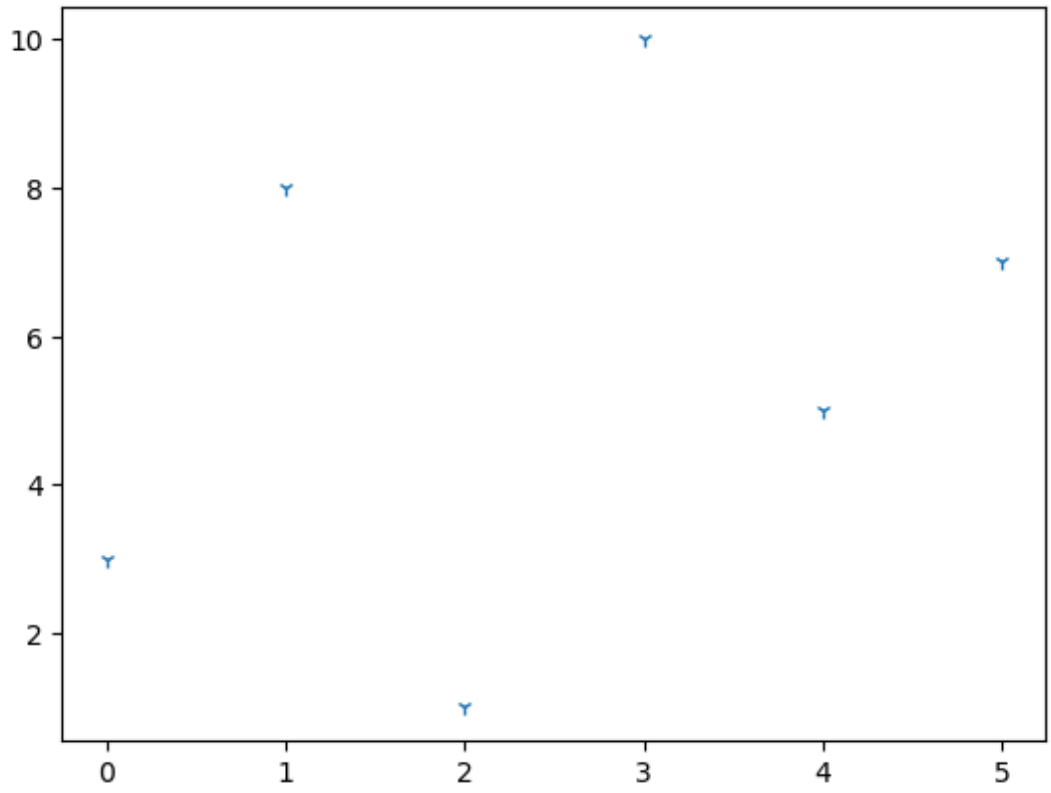
In [4]:

```
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([1,2,6,8,19])
ypoints = np.array([0,32,43,7,12])
plt.plot(xpoints,ypoints,'o')
plt.show()
```



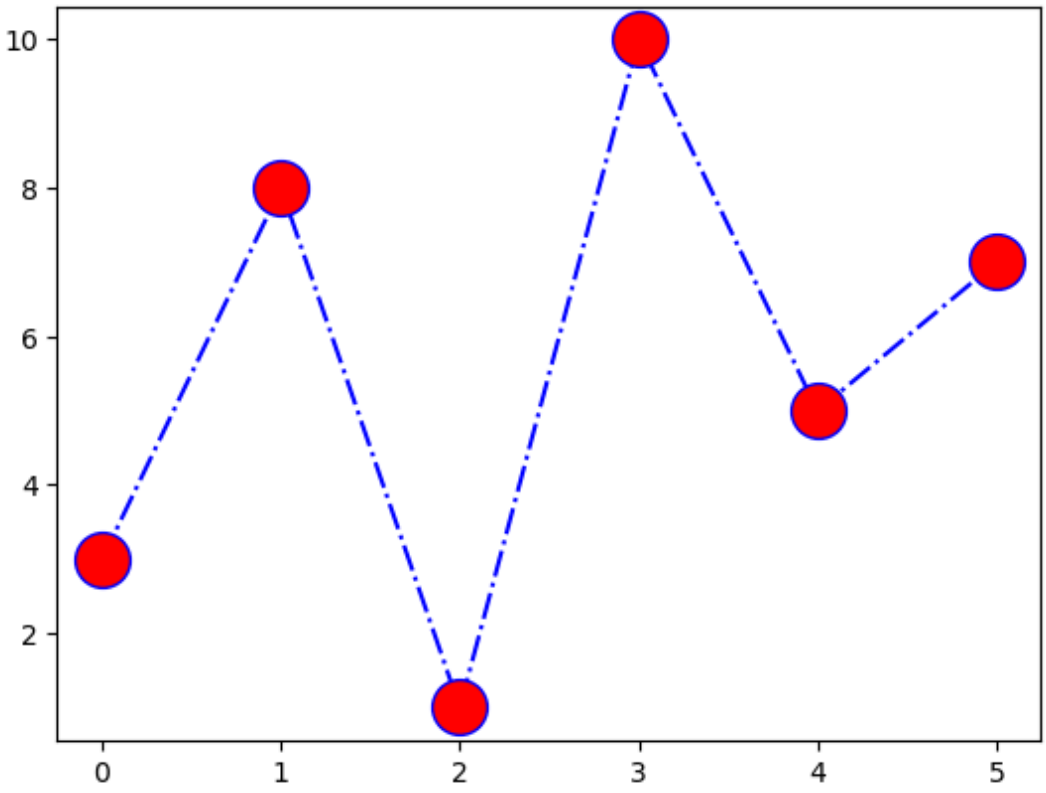
In [5]:

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3,8,1,10,5,7])
plt.plot(ypoints,'1')
plt.show()
```



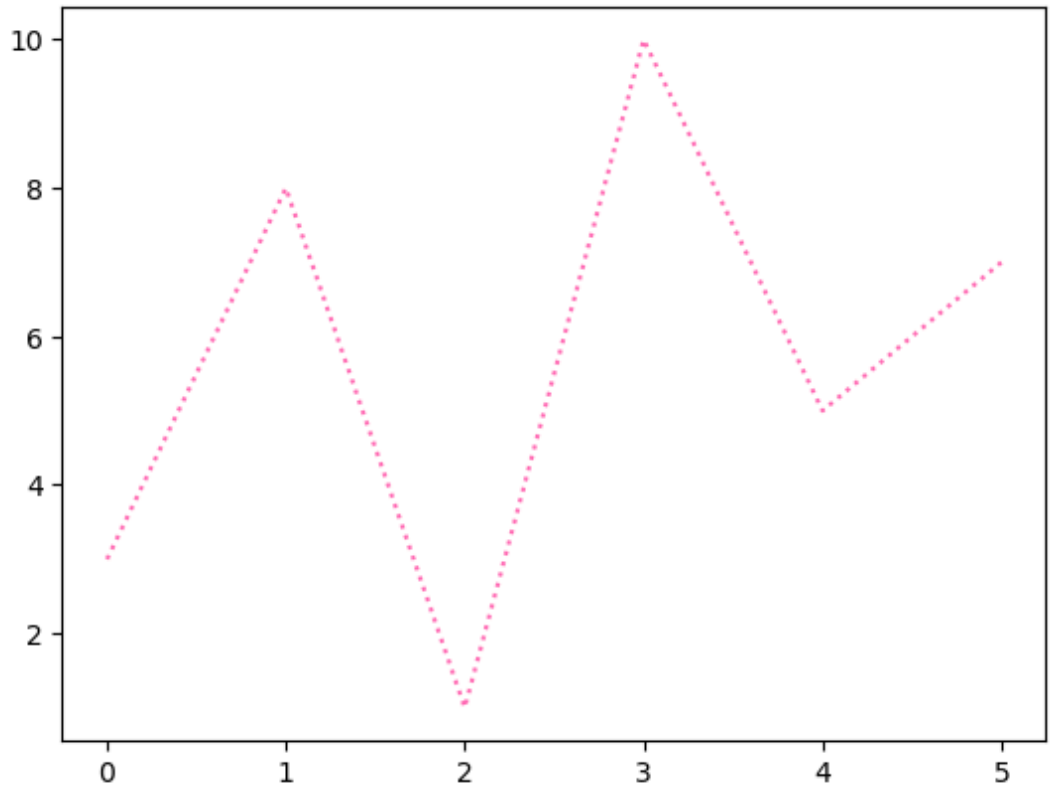
In [6]:

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3,8,1,10,5,7])
plt.plot(ypoints,'o-.b',ms=20,mfc='r')
plt.show()
```



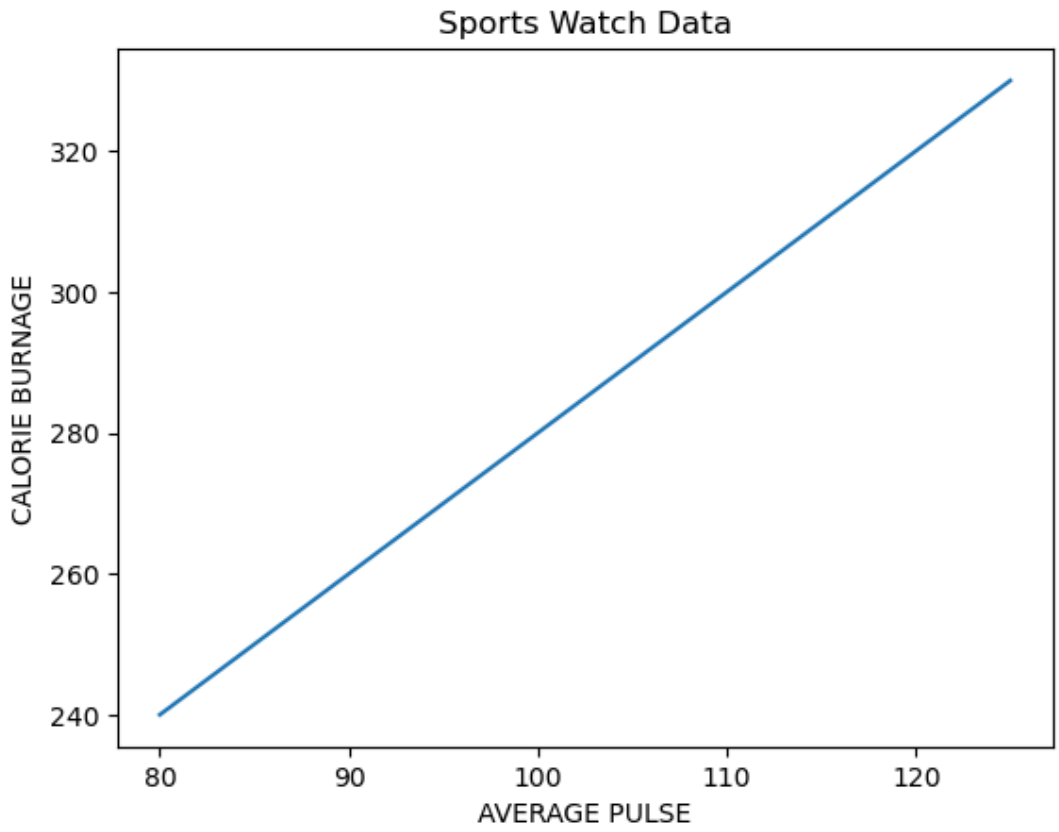
In [7]:

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3,8,1,10,5,7])
plt.plot(ypoints,linestyle='dotted',color='hotpink')
plt.show()
```



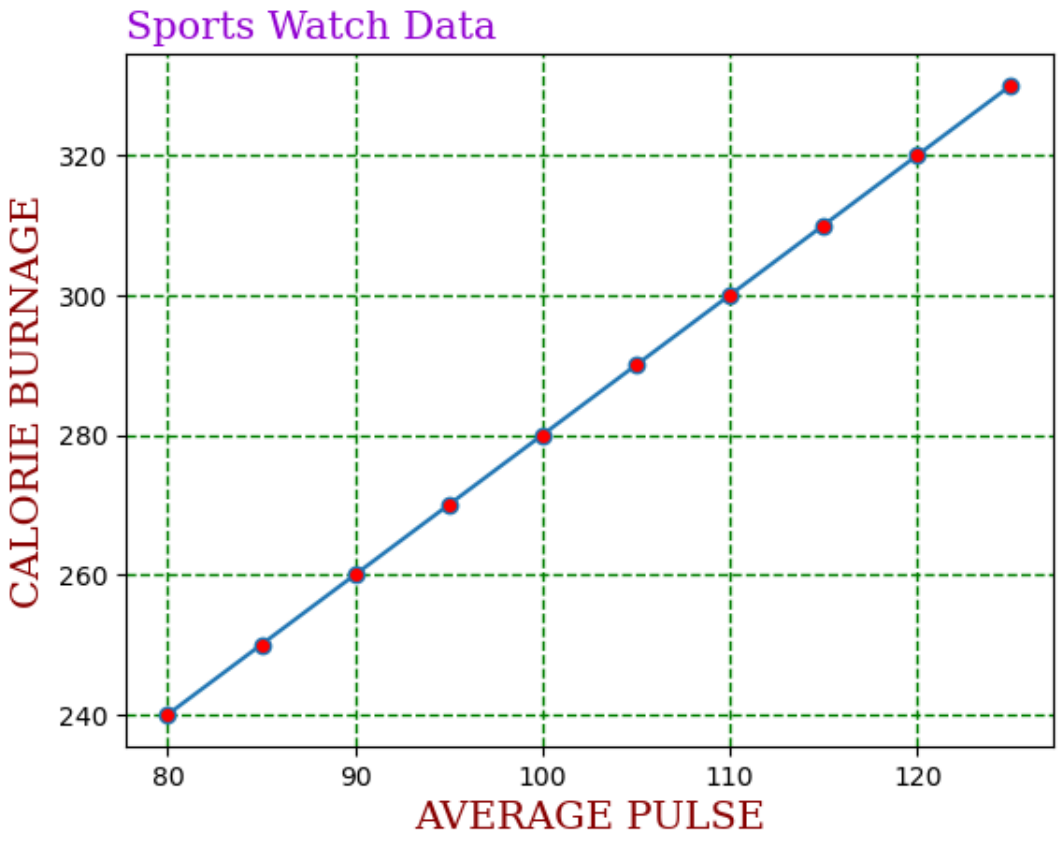
In [8]:
```
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([80,85,90,95,100,105,110,115,120,125])
ypoints = np.array([240,250,260,270,280,290,300,310,320,330])
plt.plot(xpoints,ypoints)
plt.title("Sports Watch Data")
plt.xlabel("AVERAGE PULSE")
plt.ylabel("CALORIE BURNAGE")
```

Out[8]:
```
Text(0, 0.5, 'CALORIE BURNAGE')
```



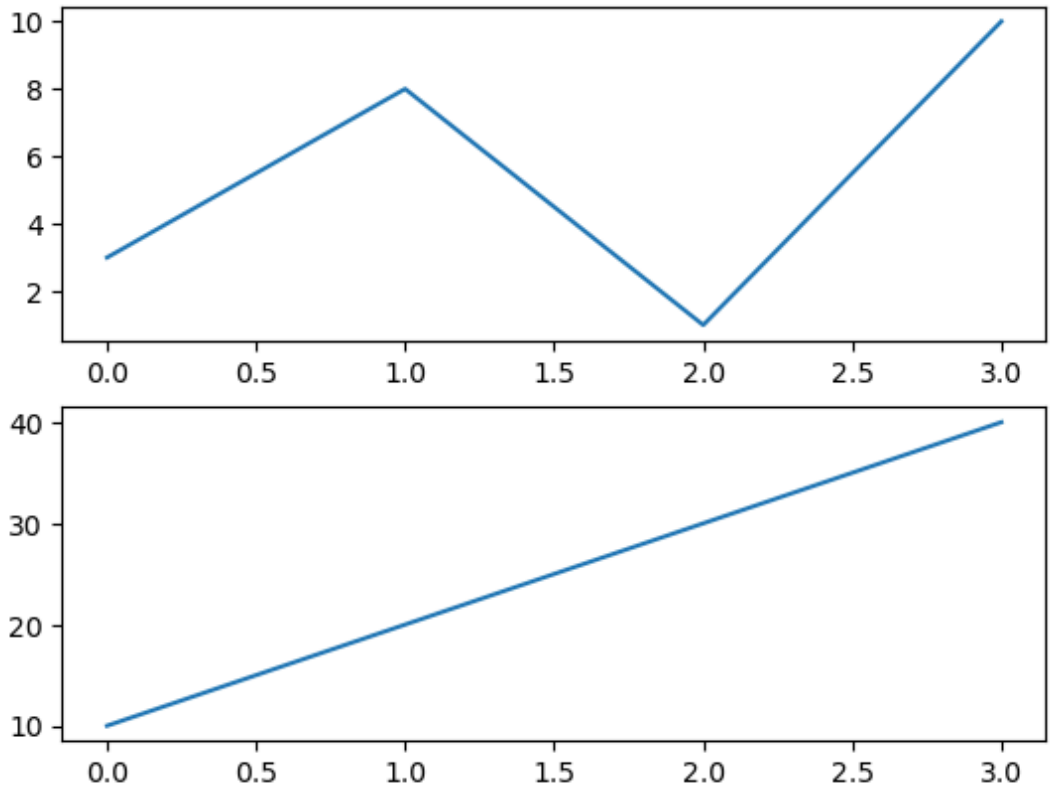
In [9]:

```
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([80,85,90,95,100,105,110,115,120,125])
ypoints = np.array([240,250,260,270,280,290,300,310,320,330])
font1={'family':'serif','color':'darkviolet','size':15}
font2={'family':'serif','color':'darkred','size':15}

plt.title("Sports Watch Data",fontdict=font1,loc='left')
plt.xlabel("AVERAGE PULSE",fontdict=font2)
plt.ylabel("CALORIE BURNAGE",fontdict=font2)
plt.plot(xpoints,ypoints,marker='o',mfc='r')
plt.grid(color='green',linestyle='--',linewidth=1)
plt.show()
```



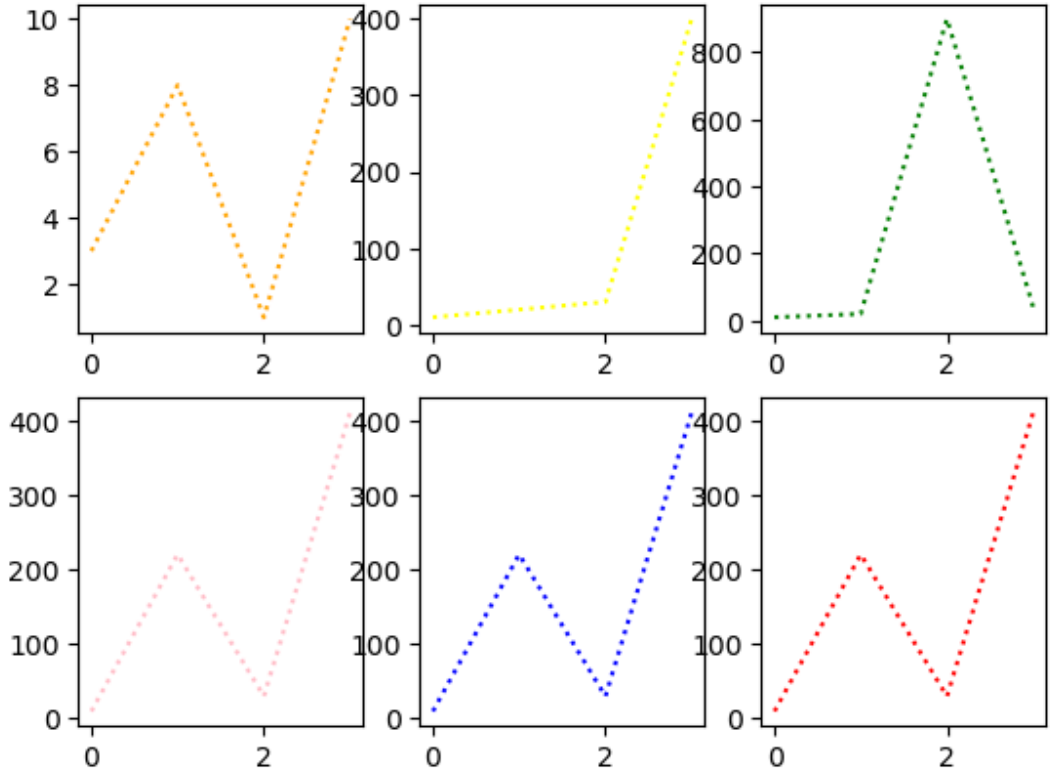
In [10]:
```
import matplotlib.pyplot as plt
import numpy as np
x=np.array([0,1,2,3])
y=np.array([3,8,1,10])

plt.subplot(2,1,1)
plt.plot(x,y)

x=np.array([0,1,2,3])
y=np.array([10,20,30,40])

plt.subplot(2,1,2)
plt.plot(x,y)
```

Out[10]: [<matplotlib.lines.Line2D at 0x2960215df50>]



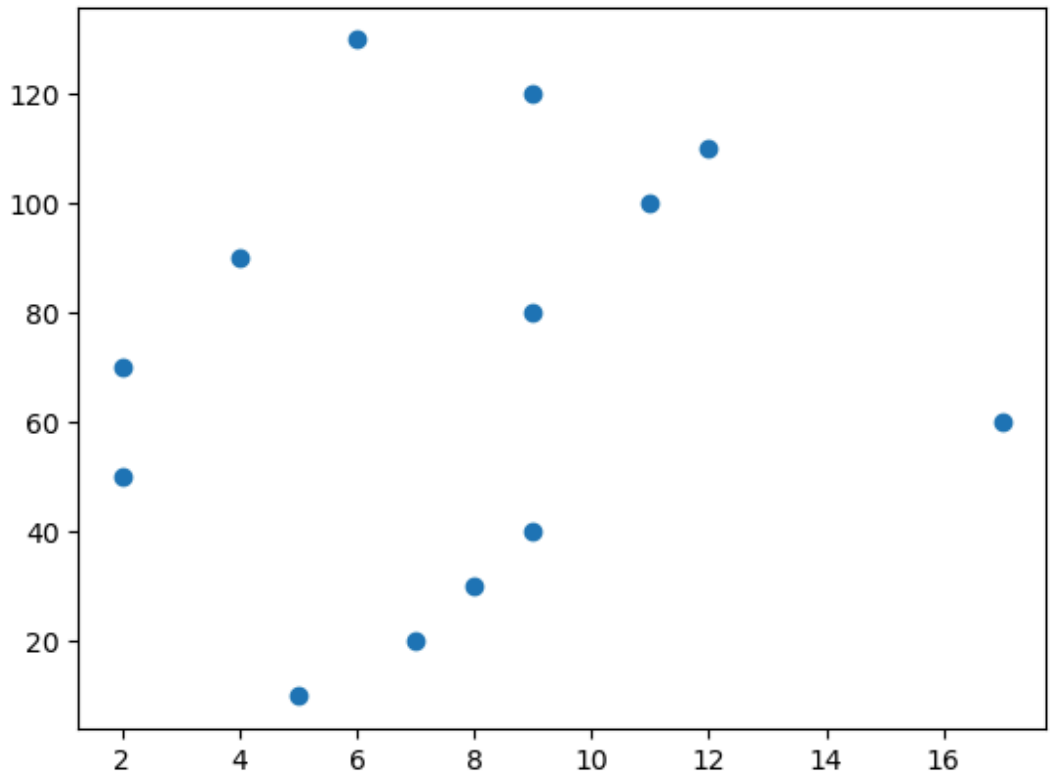
In [11]:

```
import numpy as np

#plot 1:
x = np.array([0,1,2,3])
y=np.array([3,8,1,10])

plt.subplot(2,3,1)
plt.plot(x,y,linestyle='dotted',color='orange')

#plot 2:
x = np.array([0,1,2,3])
y=np.array([10,20,30,400])

plt.subplot(2,3,2)
plt.plot(x,y,linestyle='dotted',color='yellow')
#plot 3:
x = np.array([0,1,2,3])
y=np.array([10,20,900,40])

plt.subplot(2,3,3)
plt.plot(x,y,linestyle='dotted',color='green')

#plot 3:
x = np.array([0,1,2,3])
y=np.array([10,220,30,410])

plt.subplot(2,3,4)
plt.plot(x,y,linestyle='dotted',color='pink')

#plot 3:
x = np.array([0,1,2,3])
y=np.array([10,220,30,410])
plt.subplot(2,3,5)
plt.plot(x,y,linestyle='dotted',color='blue')

#plot 3:
x = np.array([0,1,2,3])
y=np.array([10,220,30,410])

plt.subplot(2,3,6)
plt.plot(x,y,linestyle='dotted',color='red')



plt.show()
```

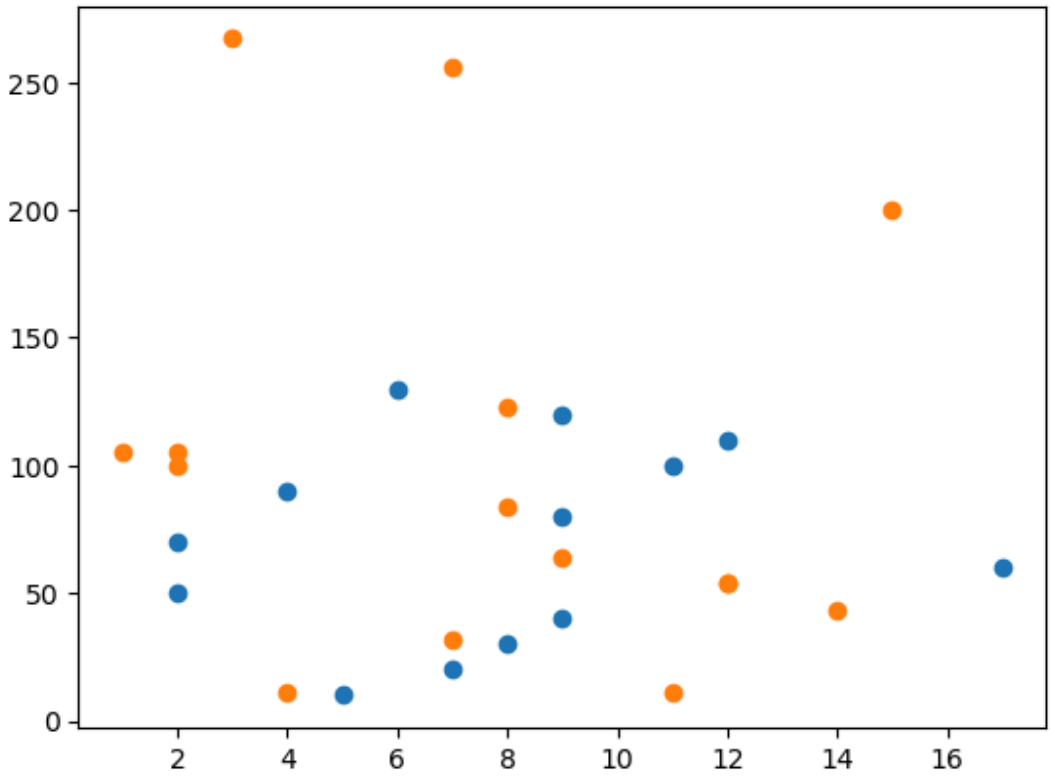


In [12]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.array([5,7,8,9,2,17,2,9,4,11,12,9,6])
y=np.array([10,20,30,40,50,60,70,80,90,100,110,120,130])

plt.scatter(x,y)
plt.show()
```

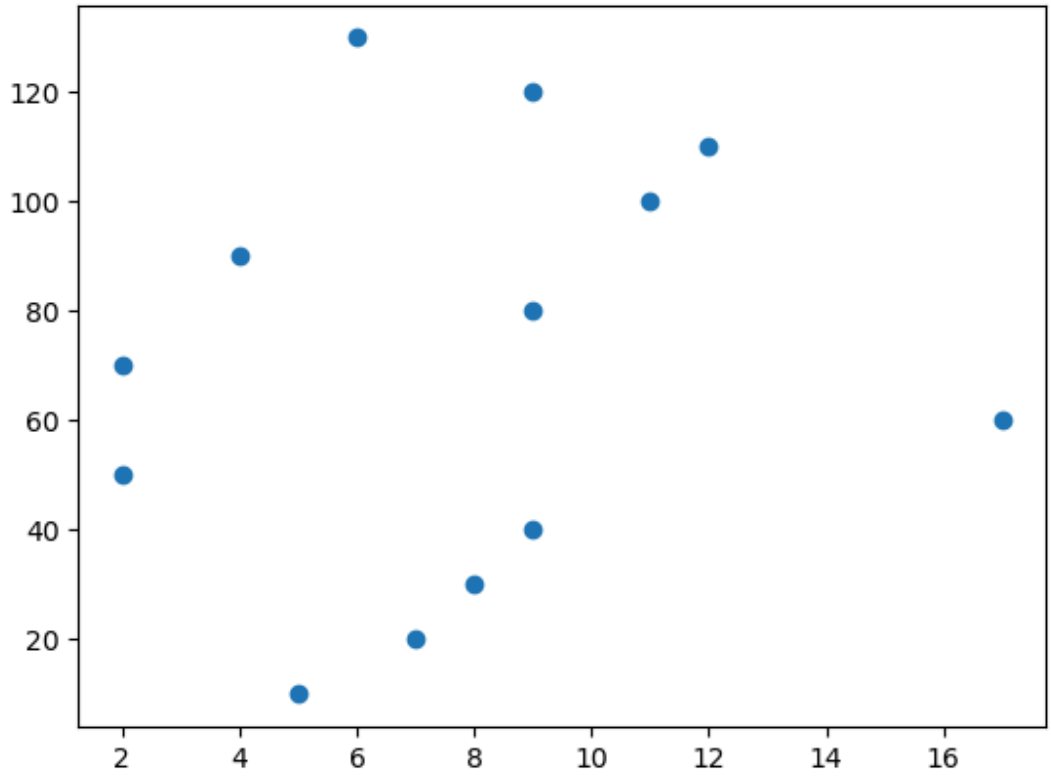


In [13]:
```
import matplotlib.pyplot as plt
import numpy as np

x=np.array([5,7,8,9,2,17,2,9,4,11,12,9,6])
y=np.array([10,20,30,40,50,60,70,80,90,100,110,120,130])

plt.scatter(x,y)
x=np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])
y=np.array([100,105,84,105,200,123,54,64,256,267,11,11,32,43,54])

plt.scatter(x,y)
plt.show()
```



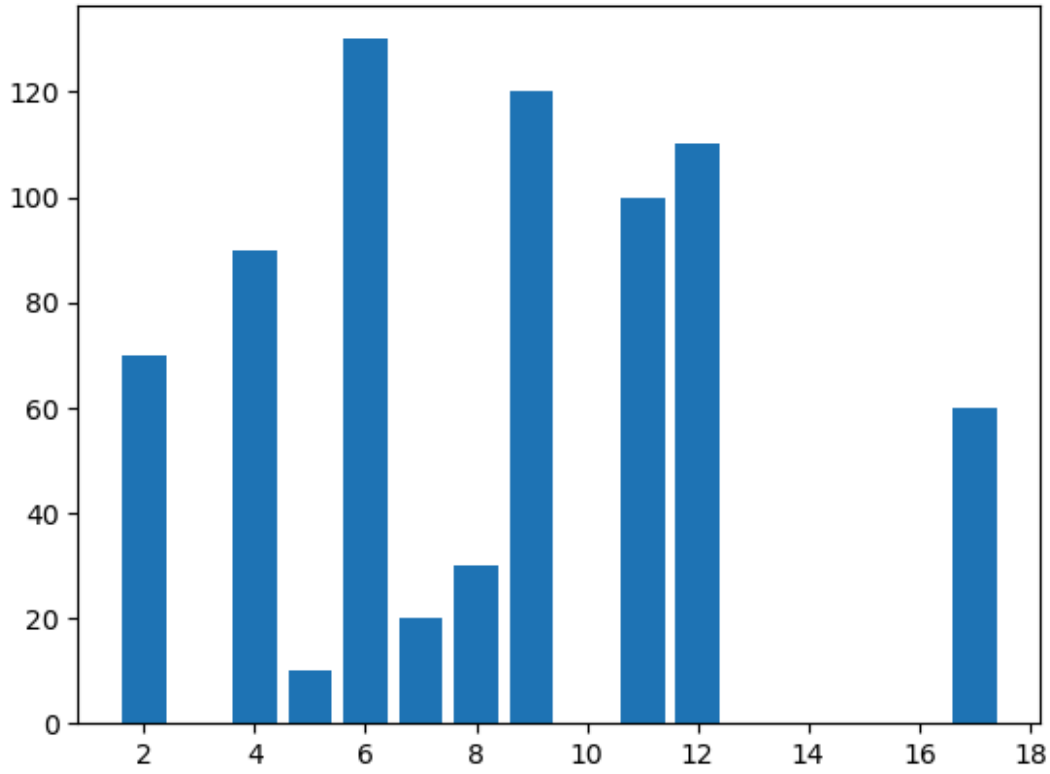
In [14]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.array([5,7,8,9,2,17,2,9,4,11,12,9,6])
y=np.array([10,20,30,40,50,60,70,80,90,100,110,120,130])
sizes=np.array([20,30,402,413,43,32,43,15,76,54,100,21,34])

plt.scatter(x,y)
plt.show()
```

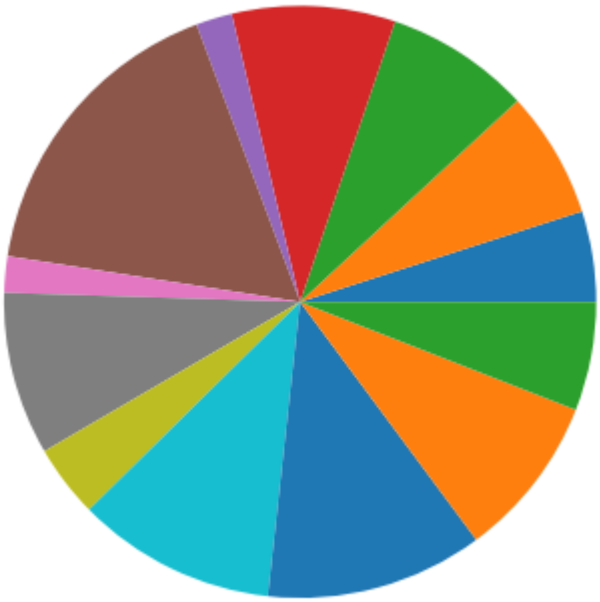


In [15]:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.array([5,7,8,9,2,17,2,9,4,11,12,9,6])
y=np.array([10,20,30,40,50,60,70,80,90,100,110,120,130])

plt.bar(x,y)
plt.show()
```



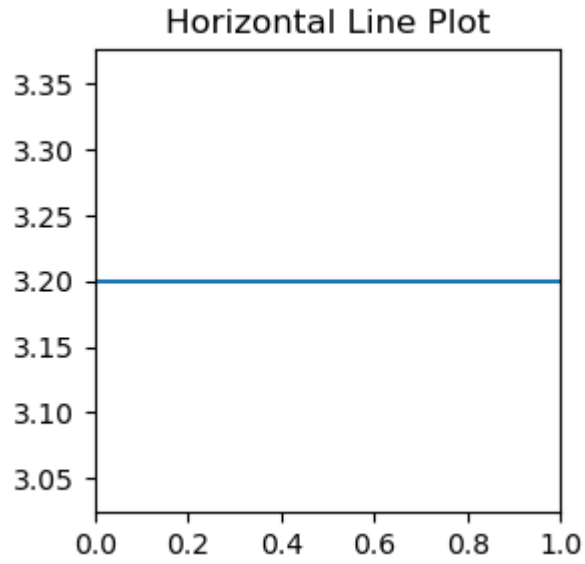
In [16]:
```
import matplotlib.pyplot as plt
import numpy as np

x=np.array([5,7,8,9,2,17,2,9,4,11,12,9,6])

plt.pie(x)
plt.show()
```



Questions

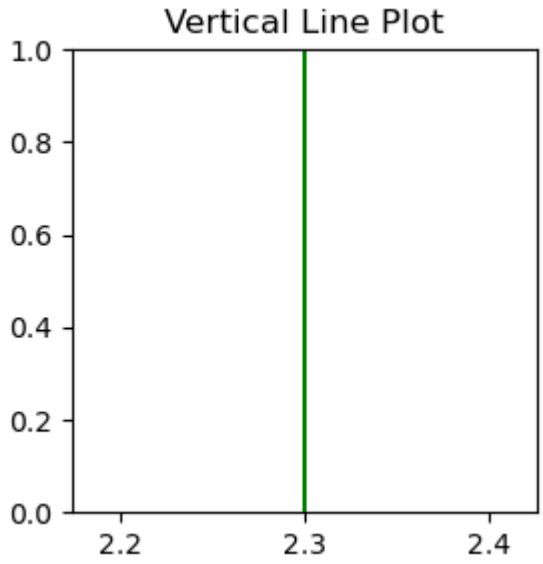
In [17]:

```
#Plot a Horizontal line in Matplotlib taking some value

import matplotlib.pyplot as plt
horizontal_value = 3.2
plt.figure(figsize=(3, 3))
plt.axhline(y=horizontal_value)
# plt.xlabel('X-axis')
# plt.ylabel('Y-axis')
plt.title('Horizontal Line Plot')
# plt.grid(True)
plt.show()
```



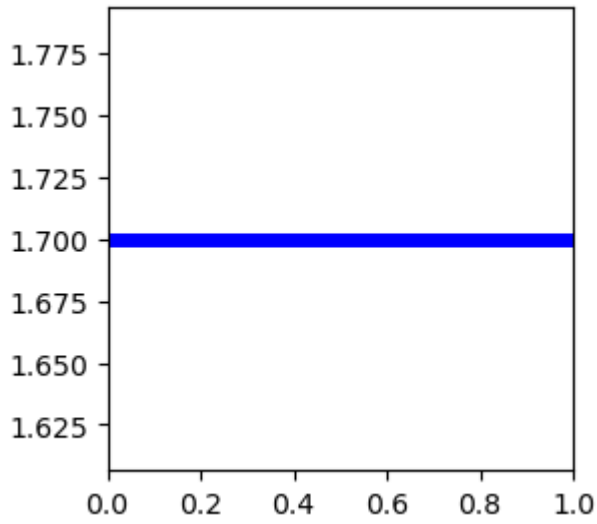
In [18]:

```
#Plot a Vertical line in Matplotlib
import matplotlib.pyplot as plt

vertical_value = 2.3

plt.figure(figsize=(3, 3))

plt.axvline(x=vertical_value, color='g', linestyle='-')

plt.title('Vertical Line Plot')

plt.show()
```



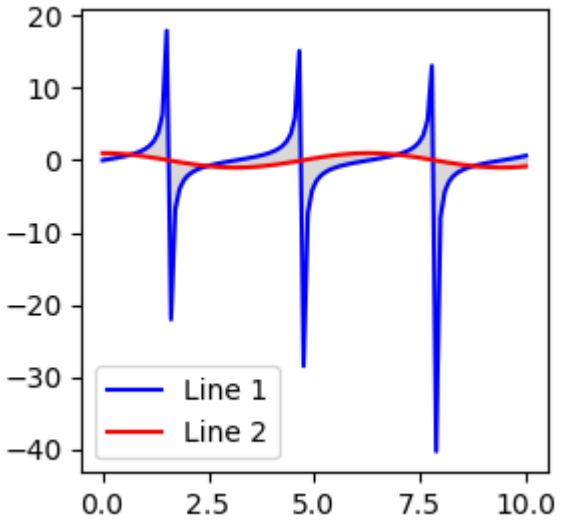
In [19]:

```
#Increase the thickness of a line with Matplotlib
import matplotlib.pyplot as plt

line_value = 1.7

plt.figure(figsize=(3, 3))

plt.axhline(y=line_value, color='b', linestyle='-', linewidth=5)

plt.show()
```

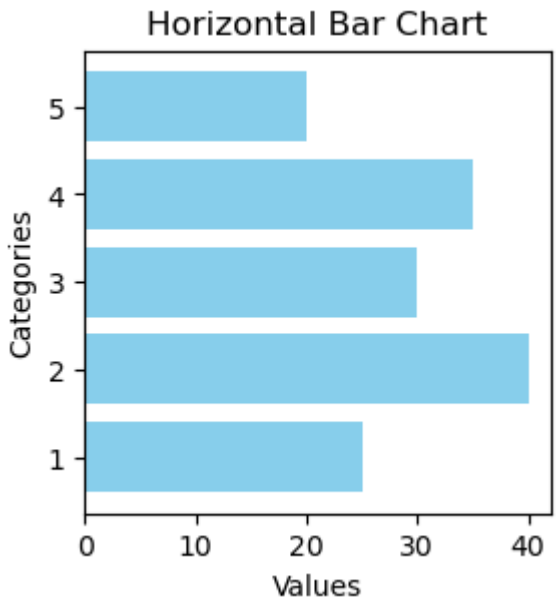


In [20]:

```
#How to Fill Between Multiple Lines in Matplotlib?
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)
y1 = np.tan(x)
y2 = np.cos(x)
plt.figure(figsize=(3, 3))
plt.plot(x, y1, color='blue', label='Line 1')
plt.plot(x, y2, color='red', label='Line 2')
plt.fill_between(x, y1, y2, color='gray', alpha=0.3)
plt.legend()
plt.show()
```

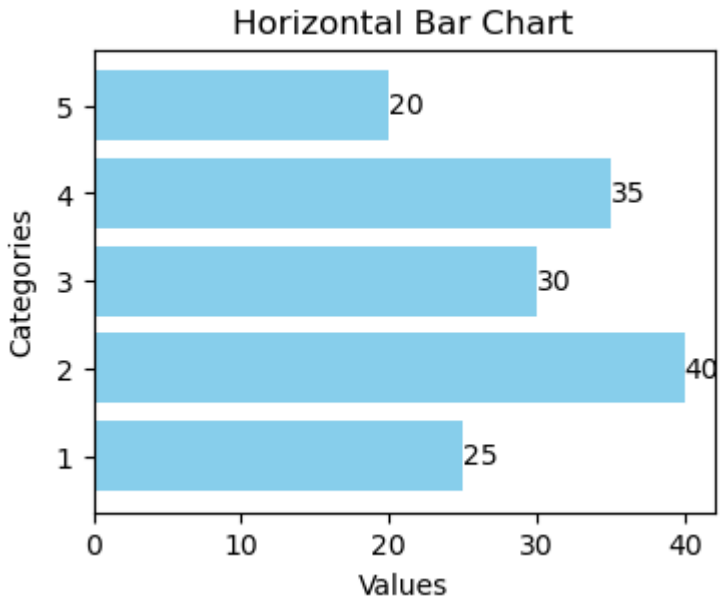


In [21]:
```
#Draw a horizontal bar chart with Matplotlib
categories = ['1', '2', '3', '4', '5']
values = [25, 40, 30, 35, 20]
plt.figure(figsize=(3, 3))
plt.barh(categories, values, color='skyblue')
plt.xlabel('Values')
plt.ylabel('Categories')
plt.title('Horizontal Bar Chart')
plt.show()
```



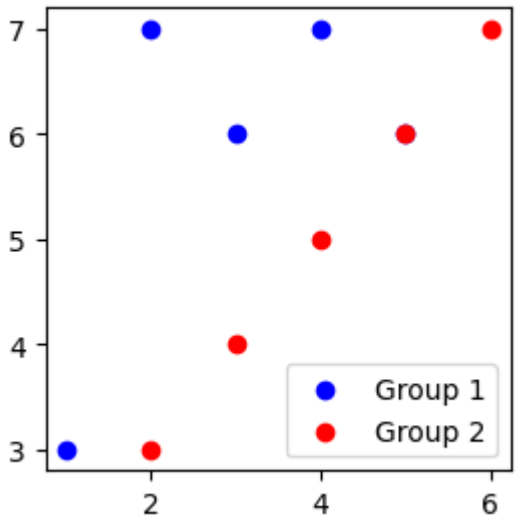
In [22]:

```
#Display the value of each bar in a bar chart using Matplotlib?
import matplotlib.pyplot as plt
categories = ['1', '2', '3', ' 4', '5']
values = [25, 40, 30, 35, 20]
plt.figure(figsize=(4, 3))
bars = plt.barh(categories, values, color='skyblue')
plt.xlabel('Values')
plt.ylabel('Categories')
plt.title('Horizontal Bar Chart')
for bar, value in zip(bars, values):
    plt.text(bar.get_width(), bar.get_y() + bar.get_height()/2, f'{valu
             va='center', ha='left')
plt.show()
```

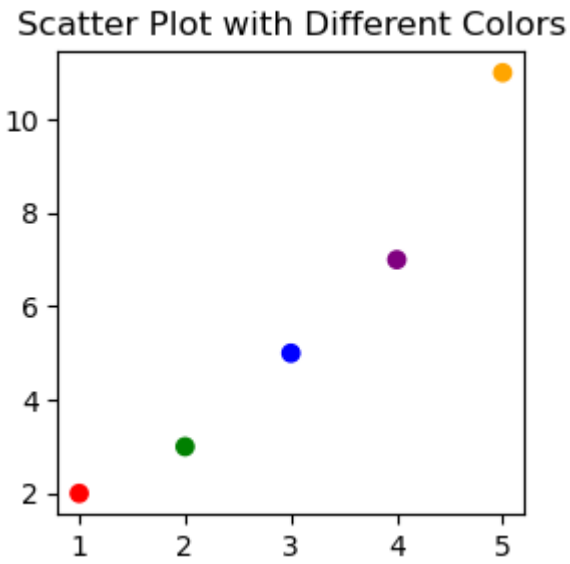


In [23]:

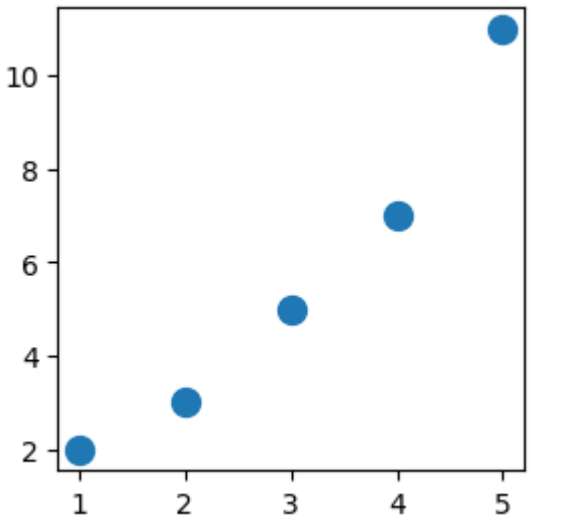
```
#Add a legend to a scatter plot in Matplotlib ?
import matplotlib.pyplot as plt
x1 = [1, 2, 3, 4, 5]
y1 = [3, 7, 6, 7, 6]
x2 = [2, 3, 4, 5, 6]
y2 = [3, 4, 5, 6, 7]
plt.figure(figsize=(3, 3))
plt.scatter(x1, y1, label='Group 1', color='blue')
plt.scatter(x2, y2, label='Group 2', color='red')
plt.legend()
plt.show()
```



In [24]:
```
#Create a Scatter Plot with several colors in Matplotlib?
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]
colors = ['red', 'green', 'blue', 'purple', 'orange']
plt.figure(figsize=(3, 3))
plt.scatter(x, y, color=colors)
plt.title('Scatter Plot with Different Colors')

plt.show()
```

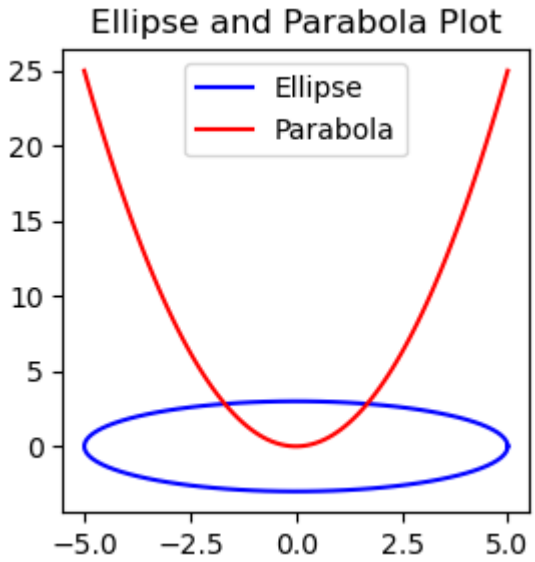


In [25]:
```
#Increase the size of scatter points in Matplotlib ?
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]
plt.figure(figsize=(3, 3))
plt.scatter(x, y, s=100)

plt.show()
```



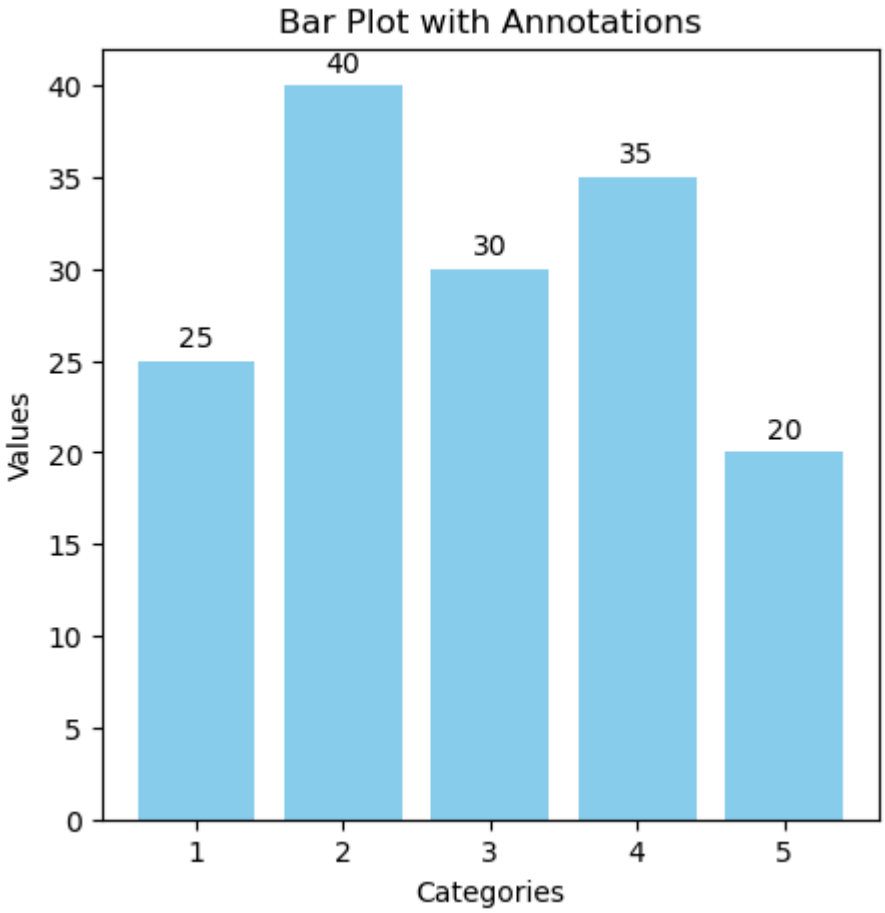
In [26]:

```
#Draw an Ellipse, Parabola by taking some value
import matplotlib.pyplot as plt
import numpy as np
a = 5
b = 3
theta = np.linspace(0, 2*np.pi, 100)
x_ellipse = a * np.cos(theta)
y_ellipse = b * np.sin(theta)
plt.figure(figsize=(3, 3))
plt.plot(x_ellipse, y_ellipse, label='Ellipse', color='blue')
x_parabola = np.linspace(-5, 5, 100)
y_parabola = x_parabola ** 2
plt.plot(x_parabola, y_parabola, label='Parabola', color='red')
plt.legend()

plt.title('Ellipse and Parabola Plot')

plt.show()
```



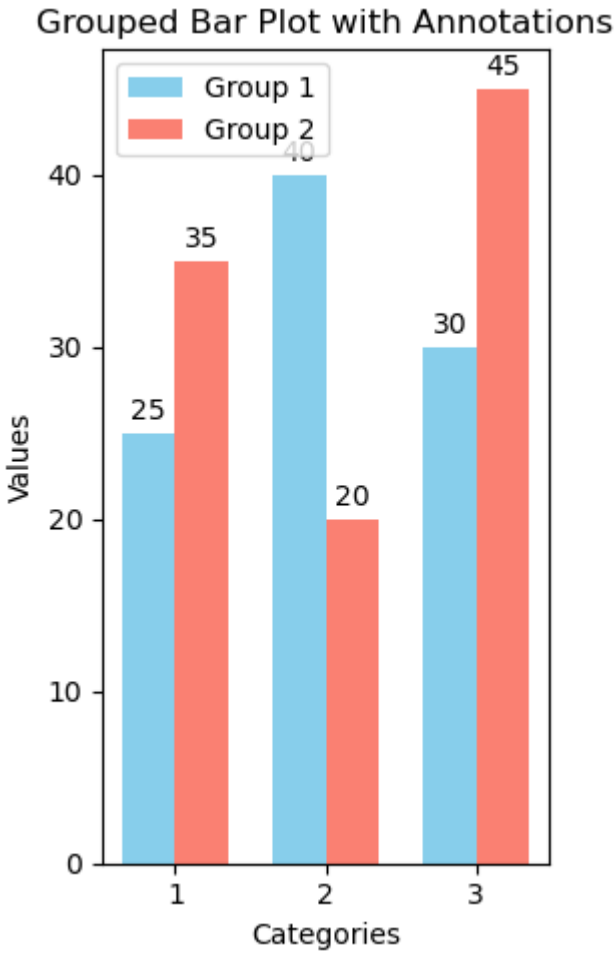
In [27]:

```
#How To Annotate Bars in Barplot wAith Matplotlib in Python?
import matplotlib.pyplot as plt
categories = ['1', '2', '3', '4', '5']
values = [25, 40, 30, 35, 20]
plt.figure(figsize=(5, 5))
bars = plt.bar(categories, values, color='skyblue')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Plot with Annotations')
for bar in bars:
    height = bar.get_height()
    plt.annotate('{}'.format(height),
                 xy=(bar.get_x() + bar.get_width() / 2, height),
                 xytext=(0, 3),
                 textcoords="offset points",
                 ha='center', va='bottom')

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



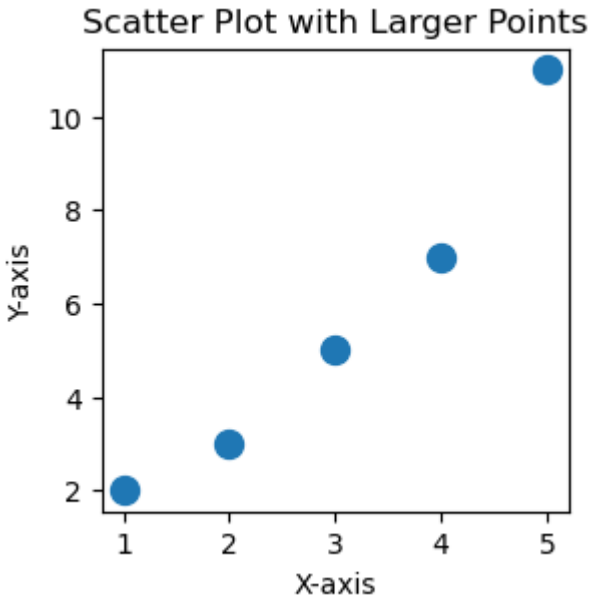
In [44]:

```
#How to Annotate Bars in Grouped Barplot in Python?
import matplotlib.pyplot as plt
import numpy as np
categories = ['1', '2', '3']
values1 = [25, 40, 30]
values2 = [35, 20, 45]
bar_width = 0.35
plt.figure(figsize=(3, 5))
bars1 = plt.bar(np.arange(len(categories)), values1, bar_width, label='
bars2 = plt.bar(np.arange(len(categories)) + bar_width, values2, bar_wi
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Grouped Bar Plot with Annotations')
plt.xticks(np.arange(len(categories)) + bar_width / 2, categories)
for bars in [bars1, bars2]:
    for bar in bars:
        height = bar.get_height()
        plt.annotate('{}'.format(height),
                     xy=(bar.get_x() + bar.get_width() / 2, height),
                     xytext=(0, 3),
                     textcoords="offset points",
                     ha='center', va='bottom')
plt.legend()

plt.tight_layout()
plt.show()
```

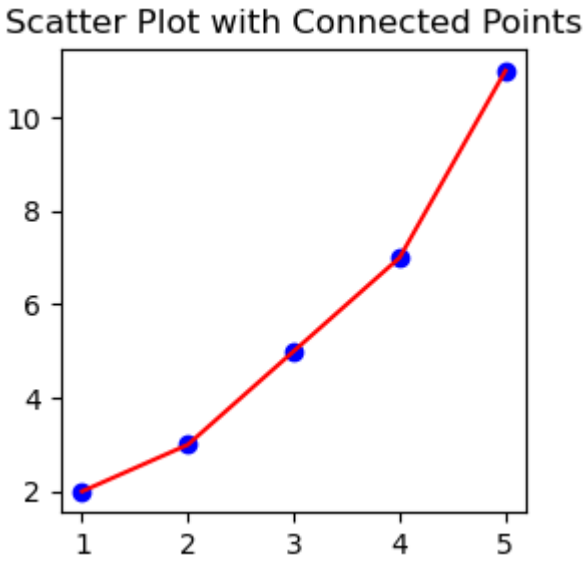


In [29]:

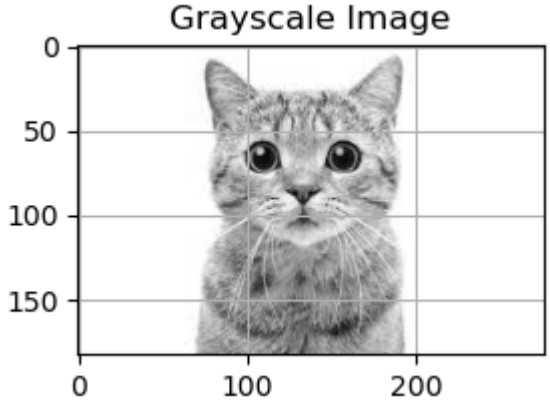
```
#How to increase the size of scatter points in Matplotlib ?
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]
plt.figure(figsize=(3, 3))
plt.scatter(x, y, s=100)
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Scatter Plot with Larger Points')

plt.show()
```



In [45]:
```
#How to Connect Scatterplot Points With Line in Matplotlib?
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]
plt.figure(figsize=(3, 3))
plt.scatter(x, y, color='blue')
plt.plot(x, y, color='red')
plt.title('Scatter Plot with Connected Points')

plt.show()
```

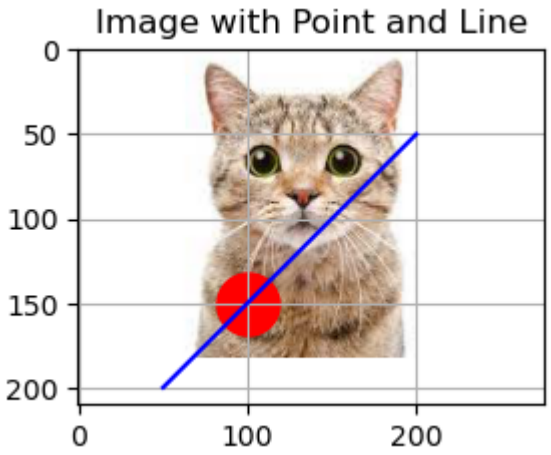


In [31]:
```
#How to Display an Image in Grayscale in Matplotlib?
import matplotlib.pyplot as plt
from PIL import Image
image = Image.open("download.jpeg")
image_gray = image.convert("L")
plt.figure(figsize=(3, 3))
plt.imshow(image_gray, cmap='gray')
plt.axis('on')
plt.grid(True)
plt.title('Grayscale Image')

plt.show()
```

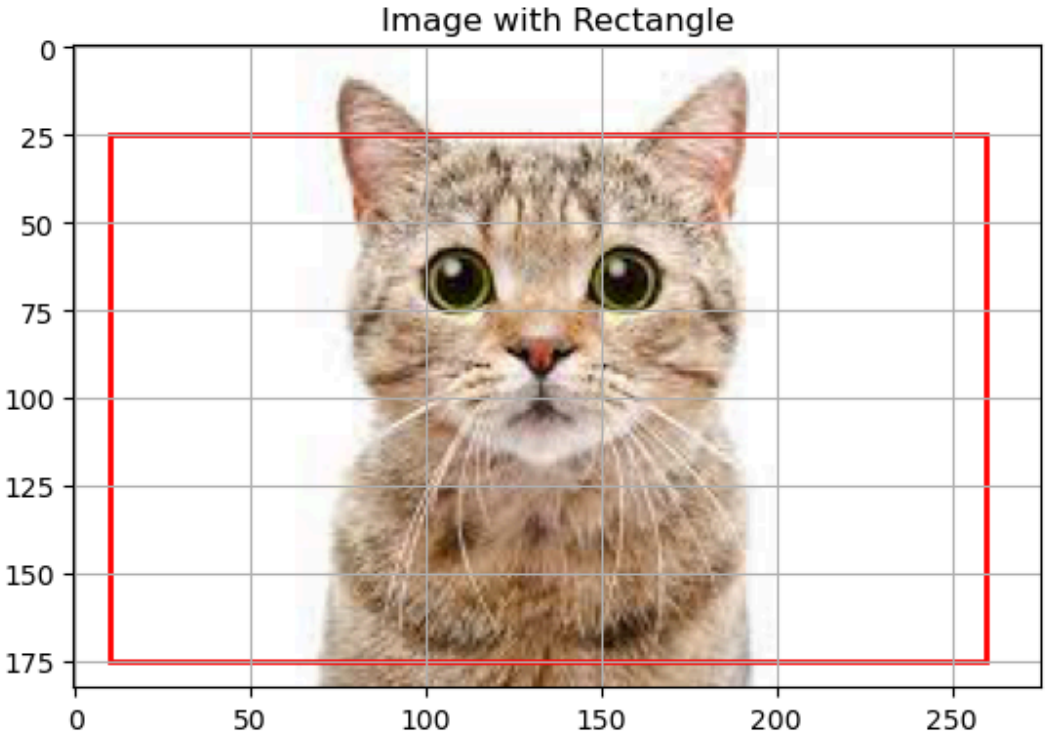


In [46]:

```
#Plot a Point or a Line on an Image with Matplotlib
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
image_path = "download.jpeg"
image = Image.open(image_path)
image_array = np.array(image)
plt.figure(figsize=(3, 3))
plt.imshow(image_array)
plt.scatter(x=100, y=150, color='red', s=500)
# Plot a line on the image
plt.plot([50, 200], [200, 50], color='blue')
plt.axis('on')
plt.title('Image with Point and Line')
plt.grid(True)

plt.show()
```



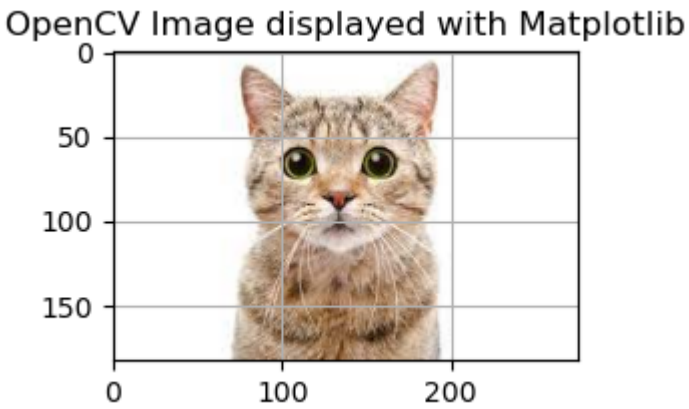
In [43]:
```
#How to Draw Rectangle on Image in Matplotlib?
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image

image_path = "download.jpeg"
image = Image.open(image_path)
fig, ax = plt.subplots()

ax.imshow(image)
x = 10
y = 25
width = 250
height = 150

rectangle = patches.Rectangle((x, y), width, height, linewidth=2, edgec

ax.add_patch(rectangle)
plt.axis('on')
plt.grid(True)

plt.title('Image with Rectangle')
plt.show()
```



In [34]:
```
pip install opencv-python
```

```
Requirement already satisfied: opencv-python in c:\users\kiit\anaconda
3\lib\site-packages (4.9.0.80)
Requirement already satisfied: numpy>=1.21.2 in c:\users\kiit\anaconda
3\lib\site-packages (from opencv-python) (1.24.3)
Note: you may need to restart the kernel to use updated packages.
```

In [37]:
```
import cv2
import matplotlib.pyplot as plt

# Load the image
image_path = "download.jpeg"
opencv_image = cv2.imread(image_path)

# Convert BGR image to RGB
opencv_image_rgb = cv2.cvtColor(opencv_image, cv2.COLOR_BGR2RGB)

# Display the image using Matplotlib
plt.figure(figsize=(3, 3))
plt.imshow(opencv_image_rgb)
plt.axis('on')  # Enable axis
plt.grid(True)  # Enable grid
plt.title('OpenCV Image displayed with Matplotlib')
plt.show()
```



In [36]:
```
#Calculate the area of an image using Matplotlib
from PIL import Image
image_path = "download.jpeg"
image = Image.open(image_path)
width, height = image.size
area = width * height

print("Area of the image:", area)
```

```
Area of the image: 50508
```

In [ ]:
```

```