In [2]:
```
# Exercise 1 (1)

import numpy as np

x = np.random.rand(2,2)
print(np.min(x, axis = 1))
```

```
[0.37872846 0.24584333]
```

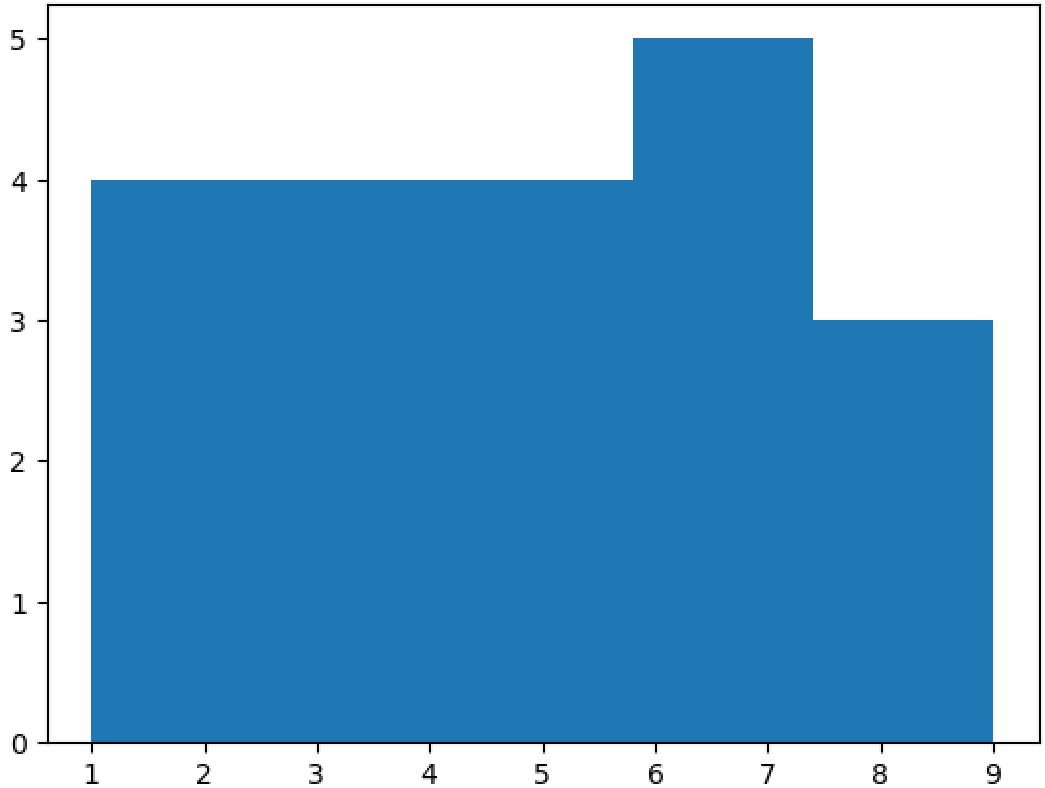
In [3]:
```
# Exercise 1 (2)

y = np.random.rand(2,5)
print(np.max(y, axis= 1) - np.min(y, axis= 1))
```

```
[0.76312496 0.93346857]
```

In [5]:
```
# Exercise 1 (3)

z = np.random.rand(2,3) * 5
values= z[z>2]
print("values are: ", values)
indices= np.argwhere(z>2)
print("indices are: ", indices)
```

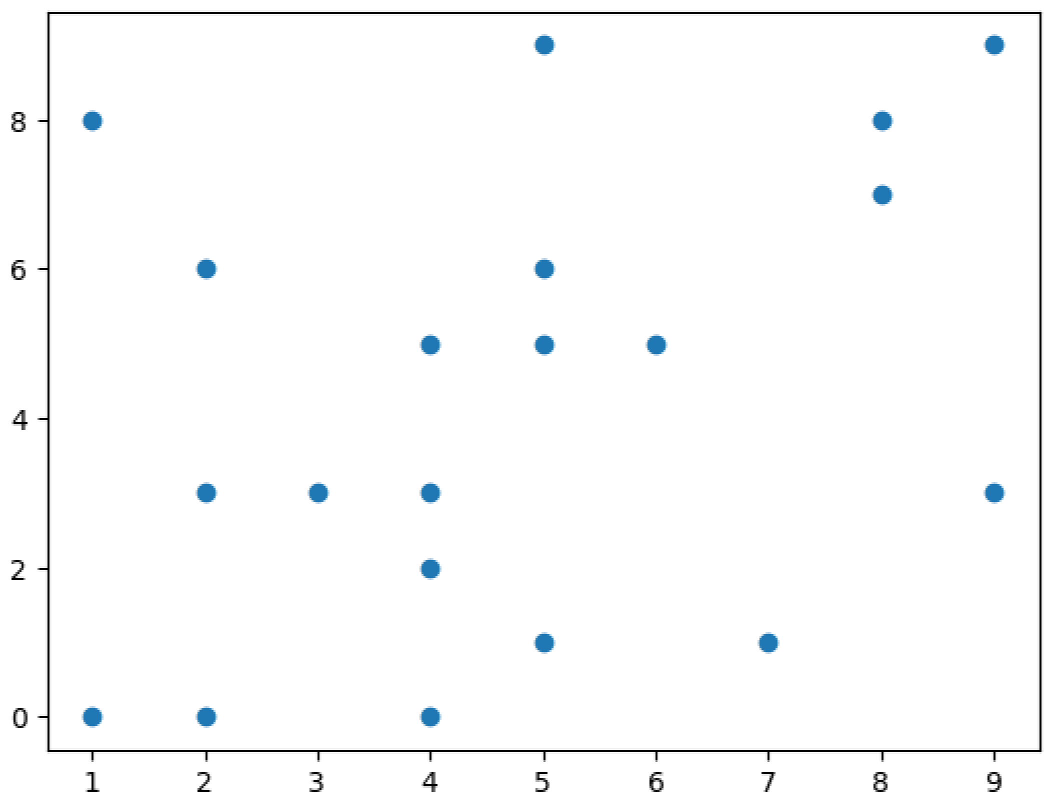
```
values are:  [4.42542429 3.88021761 4.38944541 4.77914119]
indices are:  [[0 0]
 [0 1]
 [0 2]
 [1 1]]
```

In [7]:
```
# Exercise 2 (1)

import matplotlib.pyplot as plt

x= np.random.randint(low= 0, high= 10, size= 20)
plt.hist(x, bins= 5)
plt.show()
```



In [8]:

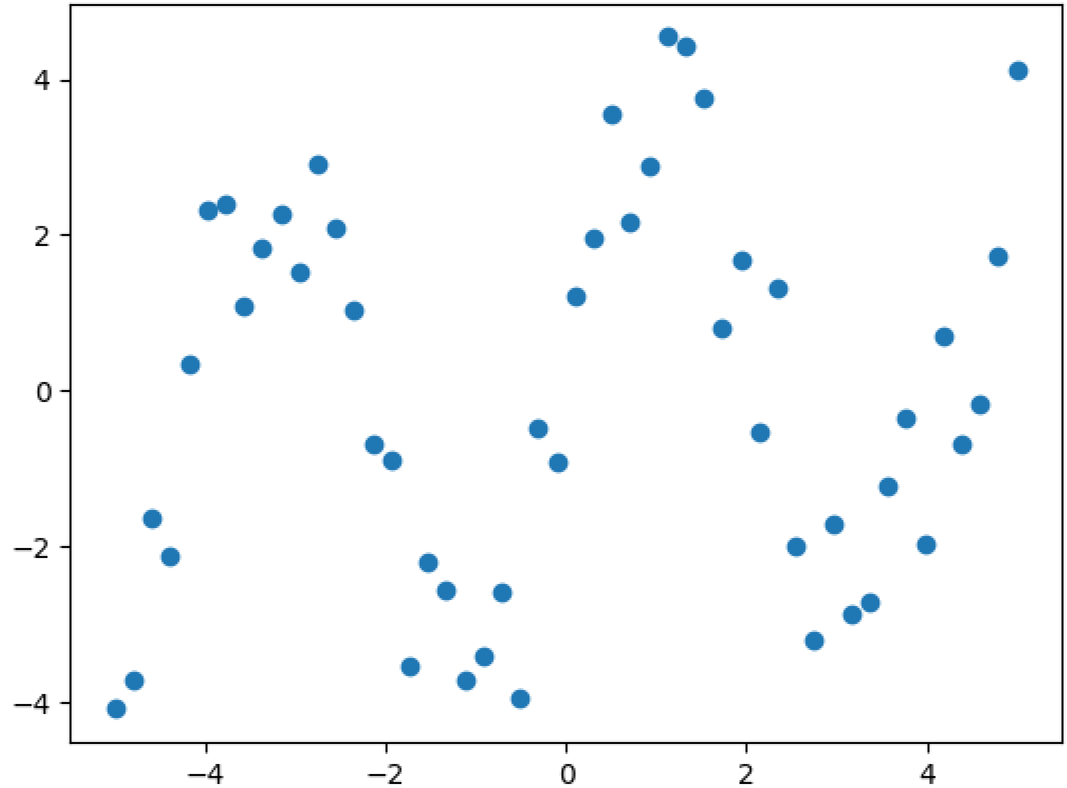
```
# Exercise 2 (2)

x= np.random.randint(low= 0, high= 10, size= 20)
y= np.random.randint(low= 0, high= 10, size= 20)

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

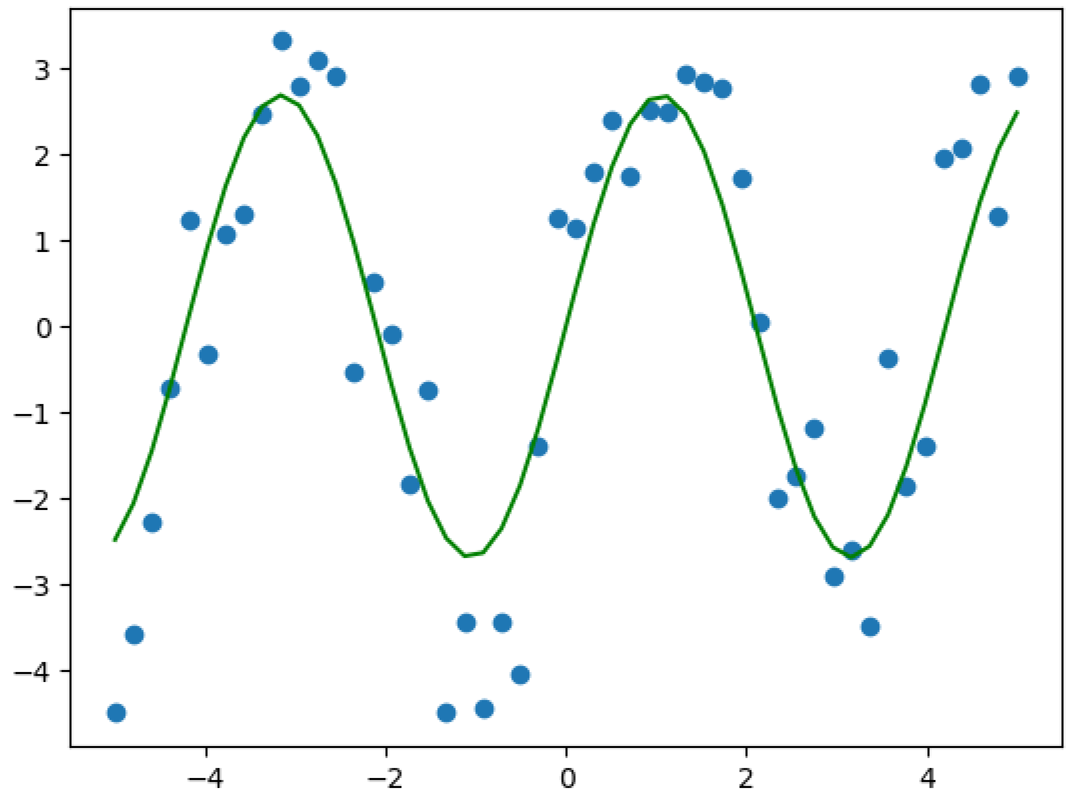


In [9]:
```
# Exercise 3

x = [1,2]
y = [[4, 1], [2, 2]]
print(np.dot(x, y))
print(np.dot(y, x))
print(np.inner(x, y))
print(np.inner(y, x))
```

```
[8 5]
[6 6]
[6 6]
[6 6]
```

In [12]:
```
# Exercise 4

from scipy.optimize import curve_fit

x_data = np.linspace(-5, 5, num=50)
y_data = 2.9 * np.sin(1.5 * x_data) + np.random.normal(size=50)
plt.scatter(x_data, y_data)
plt.show()

def sin_func(x, a, b):
    return a * np.sin(b*x)

x_data = np.linspace(-5, 5, num=50)
y_data = 2.9 * np.sin(1.5 * x_data) + np.random.normal(size=50)

attributes, variances = curve_fit(sin_func, x_data, y_data)

y_fit = sin_func(x_data, a_fit, b_fit)

plt.scatter(x_data, y_data)
plt.plot(x_data, y_fit, color='green')
plt.show()
```






```
In [ ]:
```