In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
x = np.array([[1, 2, 3], [4, 5, 6]])
x
```

Out[2]:
```
array([[1, 2, 3],
       [4, 5, 6]])
```

In [3]:
```
x = np.array([[1, 2, 3], [4, 5, 6]]).ndim
x
```

Out[3]:
```
2
```

In [4]:
```
x = np.array([[[1, 2, 3], [4, 5, 6]]]).ndim
x
```

Out[4]:
```
3
```

Slicing

In [5]:
```
#list
lst = [[1,2,3], [4,5,6], [7,8,9]]
lst[2:3]
```

Out[5]:
```
[[7, 8, 9]]
```

In [6]:
```
#acessing element
lst[0][1]
```

Out[6]:
```
2
```

In [7]:
```
lst[1][2]
```

Out[7]:
```
6
```

In [8]:
```
lst[::-1] # reverse list
```

Out[8]:
```
[[7, 8, 9], [4, 5, 6], [1, 2, 3]]
```

Numpy Array

In [9]:
```
a = np.array([1, 2, 3, 4, 5, 6])
```

In [10]:
```
print(a[0])
```

```
1
```

In [11]:
```
print(a[0:4])
```

```
[1 2 3 4]
```

In [12]:
```
print("Slicing 0 to 2  = ", a[0:2])

print("Slicing 1 to last = ",a[1:])

print("Slicing 1 to last =",a[-2:])
```

```
Slicing 0 to 2  =  [1 2]
Slicing 1 to last =  [2 3 4 5 6]
Slicing 1 to last = [5 6]
```

In [13]:
```
a1 = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]).reshape(3,4)
a1
```

Out[13]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

In [14]:
```
a1[0][1]=99
a1
```

Out[14]:
```
array([[ 1, 99,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

In [15]:
```
# Create an empty array with 2 elements
np.empty(2 , dtype = int)
```

Out[15]:
```
array([         0, 1075970048])
```

In [16]:
```
np.arange(9).reshape(3,3)
```

Out[16]:
```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

In [17]:
```
np.arange(2,20,3)
```

Out[17]:
```
array([ 2,  5,  8, 11, 14, 17])
```

In [18]:
```
o = np.ones(9, dtype=int)
o
```

Out[18]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1])
```

In [19]:
```
z = np.zeros(9, dtype=int)
z
```

Out[19]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0])
```

In [20]:
```
#use reshape()
o.reshape(3,3)
```

Out[20]:
```
array([[1, 1, 1],
       [1, 1, 1],
       [1, 1, 1]])
```

In [21]:
```
#Sorting elements
arr = np.array([2, 1, 5, 3, 7, 4, 6, 8])
np.sort(arr)
```

Out[21]:
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

In [22]:
```
# merging two array
j = np.array([1, 2, 3, 4])
k = np.array([5, 6, 7, 8])
np.concatenate((j,k))
```

Out[22]:
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

In [23]:
```
j1 = np.array([[1, 2], [3, 4]])
k1 = np.array([[5, 6]])
np.concatenate((j1, k1))
```

Out[23]:
```
array([[1, 2],
       [3, 4],
       [5, 6]])
```

In [24]:
```
s = np.array([1, 2, 3, 4])
print("Adding all-",s.sum())
print("Give max-",s.max())
print("give min -",s.min())
```

```
Adding all- 10
Give max- 4
give min - 1
```

In [25]:
```
list1 = [1, 2, 3, 4 ,5, 6]
list2 = [10, 9, 8, 7, 6, 5]

# Multiplying both lists directly would give an error.
print(list1*list2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[25], line 5
      2 list2 = [10, 9, 8, 7, 6, 5]
      4 # Multiplying both lists directly would give an error.
----> 5 print(list1*list2)

TypeError: can't multiply sequence by non-int of type 'list'
```

In [26]:
```
list1 = np.array([1, 2, 3, 4 ,5, 6])
list2 =np.array([10, 9, 8, 7, 6, 5])
print(list1*list2)
```

```
[10 18 24 28 30 30]
```

In [27]:
```
#  Multipling two matrix
data = np.array([[1, 2], [3, 4]])
print("a=",data)
ones = np.array([[1, 2], [1, 2]])
print("b=",ones)
np.multiply(data, ones)
```

```
a= [[1 2]
 [3 4]]
b= [[1 2]
 [1 2]]
```

Out[27]:
```
array([[1, 4],
       [3, 8]])
```

In [28]:
```
arr = np.array([11, 11, 12, 13, 14, 15, 16, 17, 12, 13, 11, 14, 18, 19, 20])
unique_value=np.unique(arr)
unique_value
```

Out[28]:
```
array([11, 12, 13, 14, 15, 16, 17, 18, 19, 20])
```

Transposing a matrix

In [29]:
```
arr2 = np.arange(6).reshape((2, 3))
print("before",arr2)
print("after", arr2.transpose())
```

```
before [[0 1 2]
 [3 4 5]]
after [[0 3]
 [1 4]
 [2 5]]
```

In [30]:
```
arr2.T
```

Out[30]:
```
array([[0, 3],
       [1, 4],
       [2, 5]])
```

In [31]:
```
l = np.array([[1 , 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
l.flatten()
```

Out[31]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

In [32]:
```
d = l.flatten()
l[1][2]=44
print(l)
```

```
[[ 1  2  3  4]
 [ 5  6 44  8]
 [ 9 10 11 12]]
```

In [33]:
```
np.random.random(9).reshape(3,3)
```

Out[33]:
```
array([[0.99147812, 0.96305222, 0.63017234],
       [0.3650373 , 0.36879708, 0.36925226],
       [0.86600639, 0.29484728, 0.09379608]])
```

In [34]:
```
np.random.randint(10, size=(3,4))
```

Out[34]:
```
array([[5, 5, 1, 6],
       [5, 0, 9, 3],
       [5, 4, 1, 9]])
```

In [35]:
```
Z = np.ones((3,3))
Z = np.pad(Z, pad_width=1)# change pad = 4
print(Z)
```

```
[[0. 0. 0. 0. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 0. 0. 0. 0.]]
```

In [36]:
```
# Convert tuple into list
inputTuple = (12, 1, 3, 18, 5)
print(inputTuple)
outputArray = np.asarray(inputTuple)
print("Array = ",outputArray)
```

```
(12, 1, 3, 18, 5)
Array =  [12  1  3 18  5]
```

Pandas

In [37]:
```
import pandas as pd
```

In [38]:
```
import numpy as np
import pandas as pd
d = {"A" :[1,2,4], "B" : [4,5,4], "C" : [7, 8, 9], "D" : [12, 23,5]}
df = pd.DataFrame(d, index = ["a", "b", "c"])
df
```

Out[38]:

	A	B	C	D
a	1	4	7	12
b	2	5	8	23
c	4	4	9	5

In [46]:
```
#Dataframe using dictionary and series
d = {"one": pd.Series([1, 2, 3 ],  index=["a", "b", "c"]),
    "two": pd.Series([1, 2, 3, 4] ,  index=["a", "b", "c" ,"d"]),
    }
df = pd.DataFrame(d , index=["a", "b", "c", "d"])
df
```

Out[46]:

	one	two
a	1.0	1
b	2.0	2
c	3.0	3
d	NaN	4

In [40]:
```
# Create a list of dictionaries
list_of_dictionaries = [
    {'Name': 'Emma', 'Age': 29, 'Department': 'HR'},
    {'Name': 'Oliver', 'Age': 25, 'Department': 'Finance'},
    {'Name': 'Harry', 'Age': 33, 'Department': 'Marketing'},
    {'Name': 'Sophia', 'Age': 24, 'Department': 'IT'}]
# Create the DataFrame
df4 = pd.DataFrame(list_of_dictionaries)
df4
```

Out[40]:

	Name	Age	Department
0	Emma	29	HR
1	Oliver	25	Finance
2	Harry	33	Marketing
3	Sophia	24	IT

In [41]:
```
# Create a list of lists
list_of_lists = [
    ['Emma', 29, 'HR'],
    ['Oliver', 25, 'Finance'],
    ['Harry', 33, 'Marketing'],
    ['Sophia', 24, 'IT']]
# Create the DataFrame
df3 = pd.DataFrame(list_of_lists, columns = ['Name', 'Age', 'Department'])
df3
```

Out[41]:

	Name	Age	Department
0	Emma	29	HR
1	Oliver	25	Finance
2	Harry	33	Marketing
3	Sophia	24	IT

In [42]:
```
# Create a numpy array
nparray = np.array(
    [['Emma', 'Oliver', 'Harry', 'Sophia'],
     [29, 25, 33, 24],
     ['HR', 'Finance', 'Marketing', 'IT']])
# Create a dictionary of nparray
dictionary_of_nparray = { 'Name': nparray[0], 'Age': nparray[1], 'Department':
# Create the DataFrame
df2 = pd.DataFrame(dictionary_of_nparray)
df2
```

Out[42]:

	Name	Age	Department
0	Emma	29	HR
1	Oliver	25	Finance
2	Harry	33	Marketing
3	Sophia	24	IT

In [43]:
```
# Create a dictionary of list
dictionary_of_lists = {
    'Name': ['Emma', 'Oliver', 'Harry', 'Sophia'],
    'Age': [29, 25, 33, 24],
    'Department': ['HR', 'Finance', 'Marketing', 'IT']}
# Create the DataFrame
df1 = pd.DataFrame(dictionary_of_lists)
df1
```

Out[43]:

	Name	Age	Department
0	Emma	29	HR
1	Oliver	25	Finance
2	Harry	33	Marketing
3	Sophia	24	IT

In [44]:
```
# Create Series
series1 = pd.Series(['Emma', 'Oliver', 'Harry', 'Sophia'])
series2 = pd.Series([29, 25, 33, 24])
series3 = pd.Series(['HR', 'Finance', 'Marketing', 'IT'])
# Create a dictionary of Series
dictionary_of_nparray = {'Name': series1, 'Age': series2, 'Department':series3
# Create the DataFrame
df5 = pd.DataFrame(dictionary_of_nparray)
df5
```

Out[44]:

	Name	Age	Department
0	Emma	29	HR
1	Oliver	25	Finance
2	Harry	33	Marketing
3	Sophia	24	IT

In [68]:
```
df5[['Age','Name']]
```

Out[68]:

	Age	Name
0	29	Emma
1	25	Oliver
2	33	Harry
3	24	Sophia

In [52]:
```
# df[col] Returns column with label col as Series
# df[[col1, col2]] Returns columns as a new DataFrame
# s.iloc[0] Selection by position
# s.loc['index_one'] Selection by index
# df.iloc[0,:] First row
# df.iloc[0,0] First element of first column
```

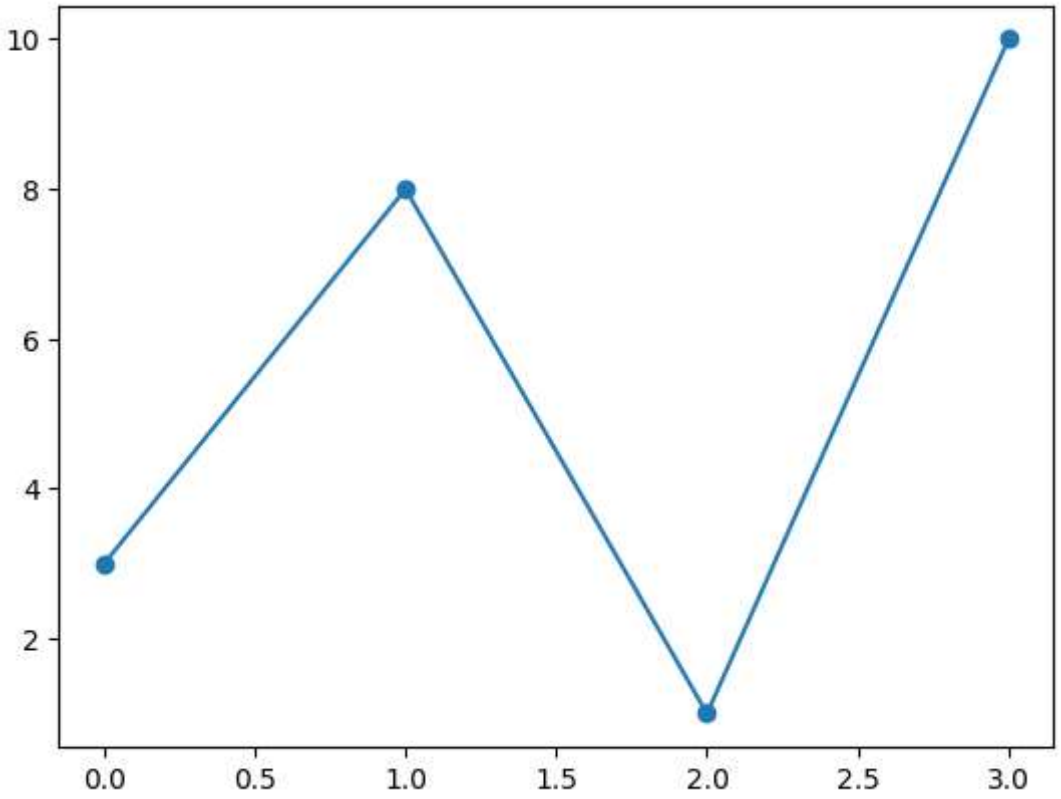
In [51]:
```
# data in the form of list of tuples
data = [('Peter', 18, 7),
        ('Riff', 15, 6),
        ('John', 17, 8),
        ('Michel', 18, 7),
        ('Sheli', 17, 5) ]


#create DataFrame using data
df = pd.DataFrame(data, columns =['Name', 'Age', 'Score'], index =["a","s","sa
print(df)
```

```
      Name  Age  Score
a    Peter   18      7
s     Riff   15      6
sa    John   17      8
as  Michel   18      7
sw   Sheli   17      5
```

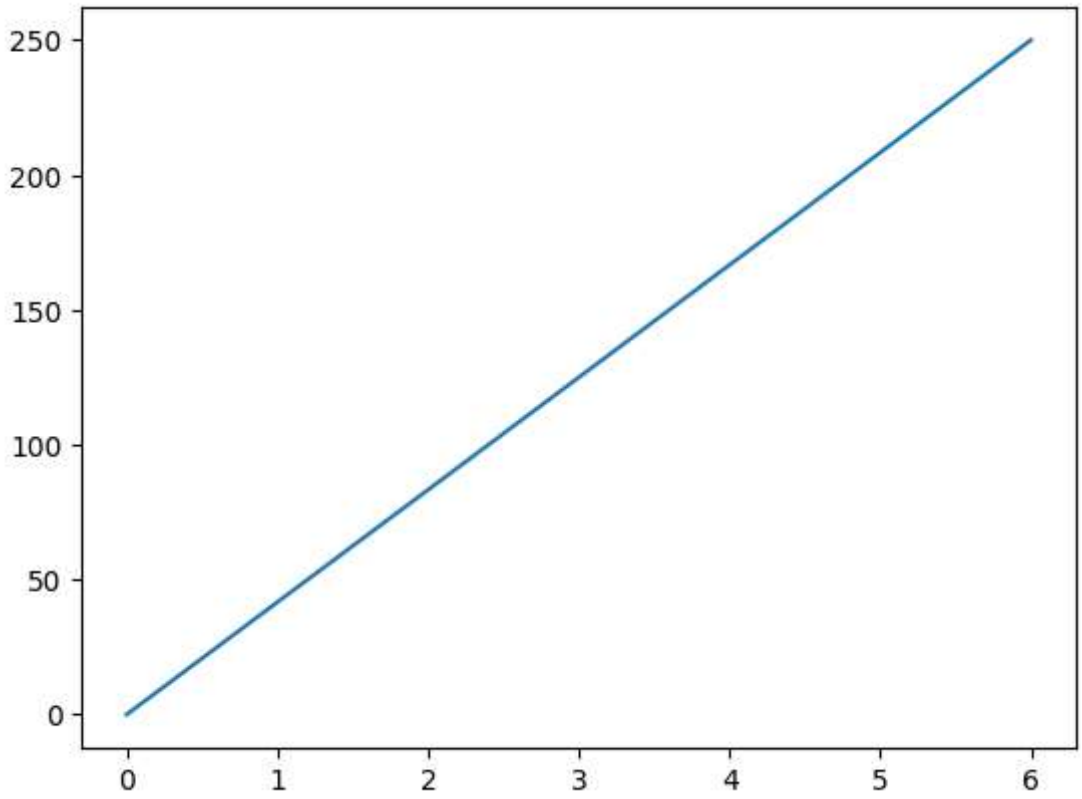
Matplotlib

In [69]:
```
import matplotlib.pyplot as plt
```

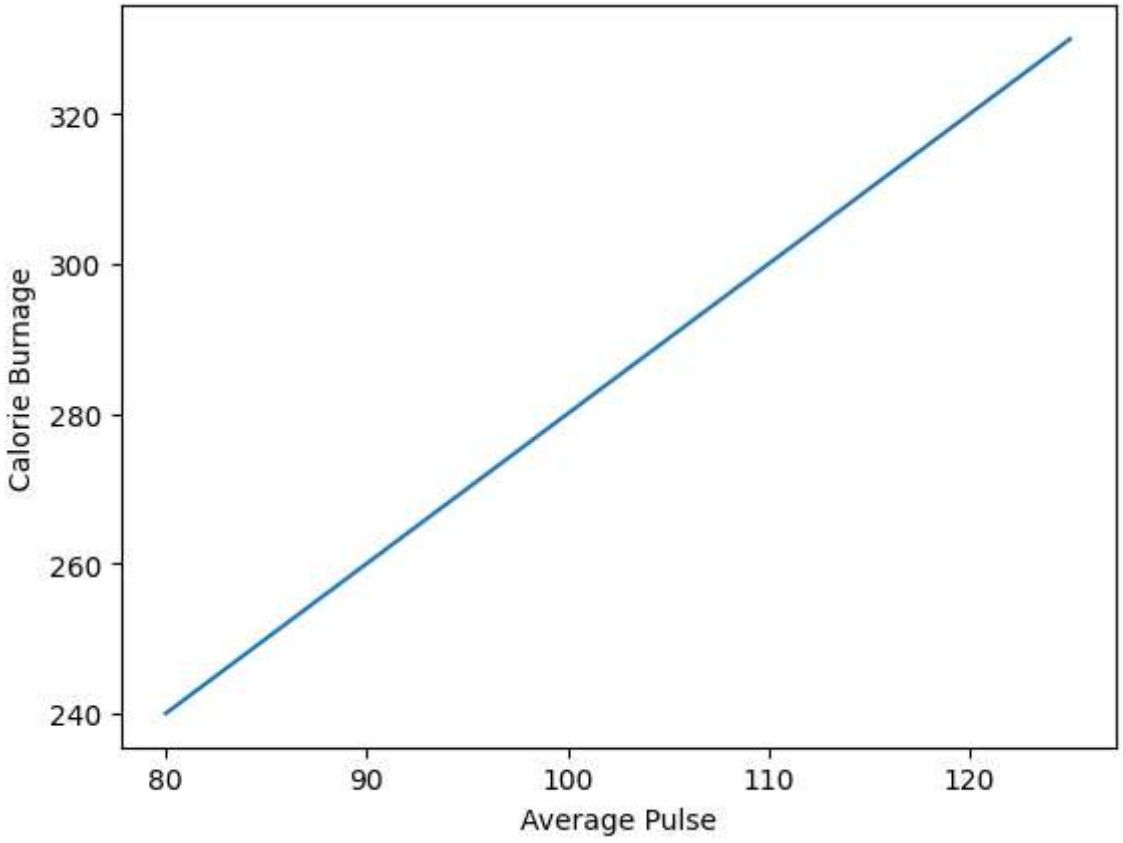
In [71]:
```
ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker = 'o')
plt.show()
```



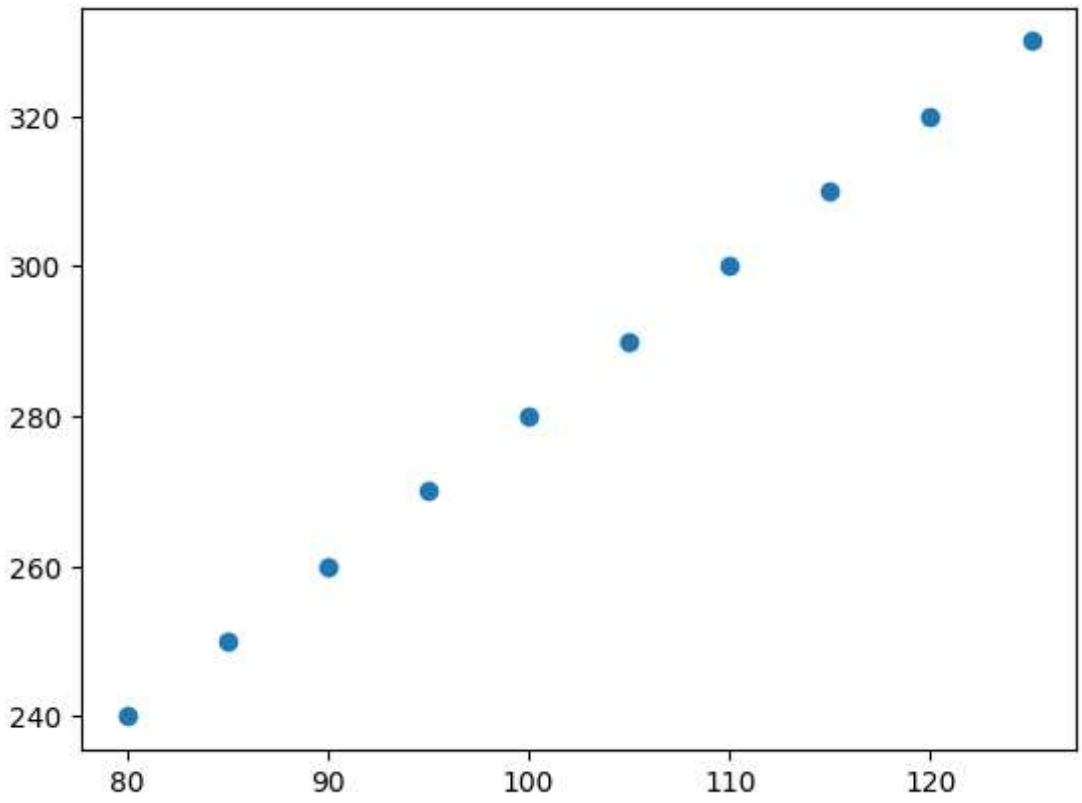
In [70]:
```
import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([0, 6])
ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)
plt.show()
```



In [72]:
```
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.plot(x, y)

plt.xlabel("Average Pulse")
plt.ylabel("Calorie Burnage")

plt.show()
```

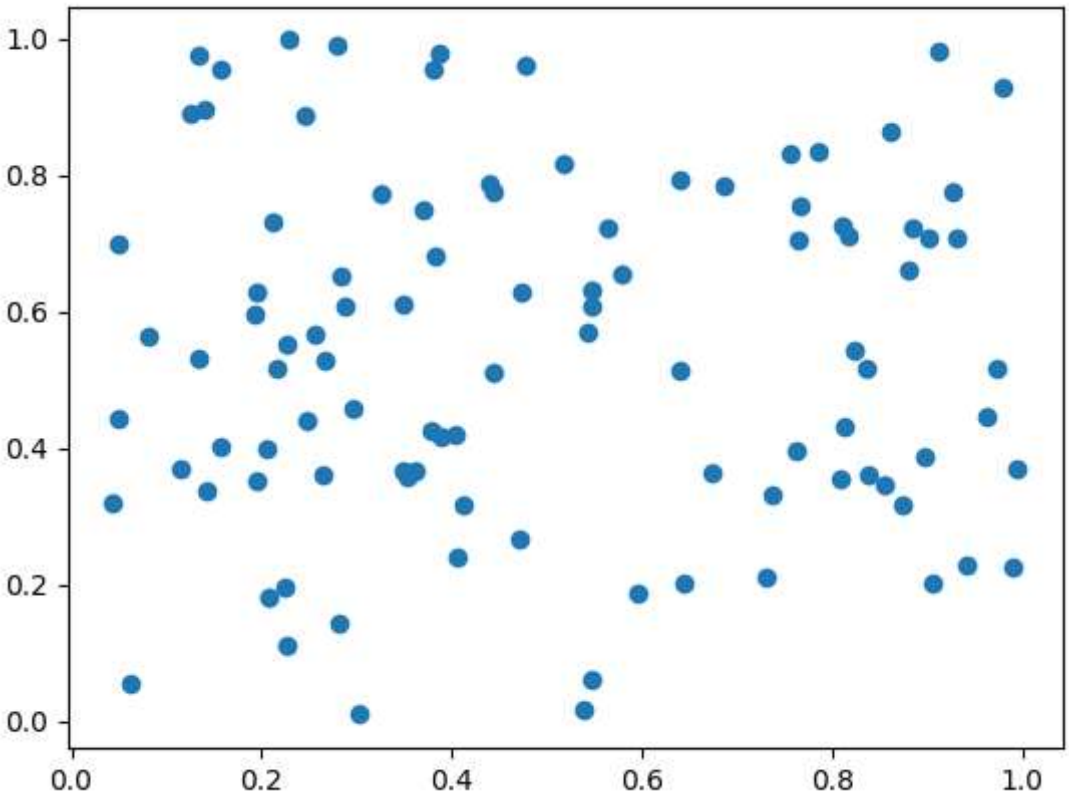


In [73]:
```
plt.scatter(x, y)
plt.show()
```

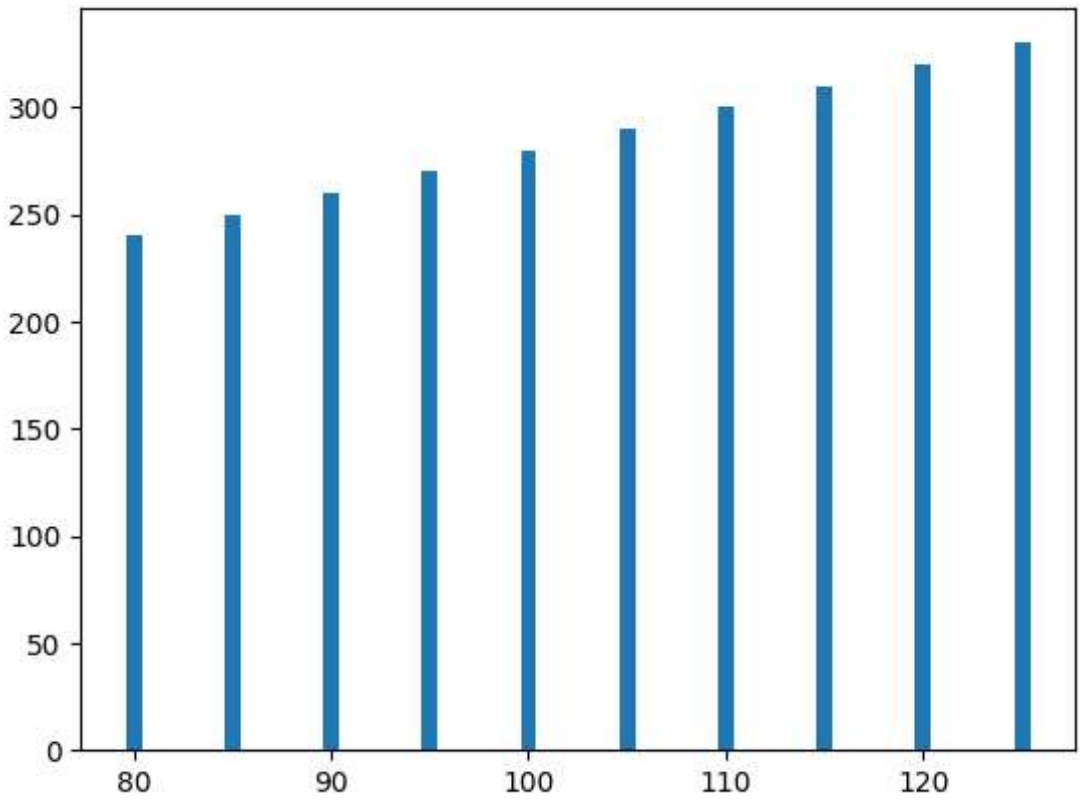


In [89]:
```
X = np.random.uniform(0, 1, 100)
Y = np.random.uniform(0, 1, 100)
plt.scatter(X, Y)
```

Out[89]: <matplotlib.collections.PathCollection at 0x2082bfca210>

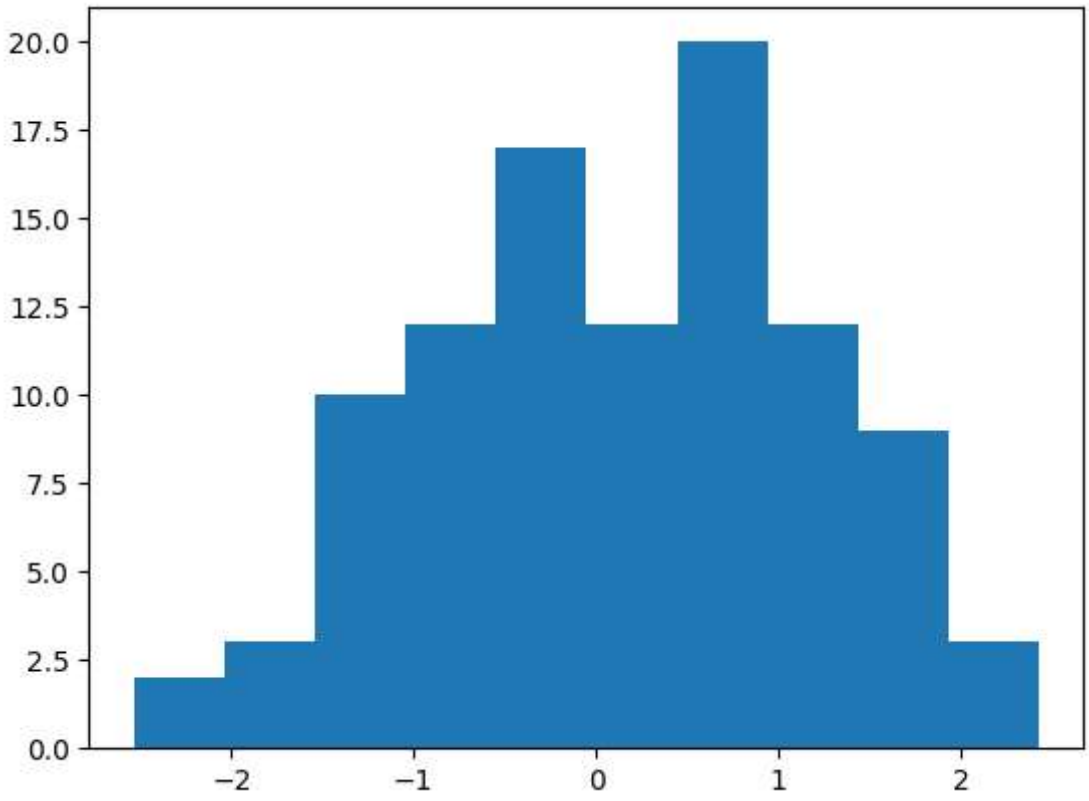


In [74]:
```
plt.bar(x,y)
plt.show()
```

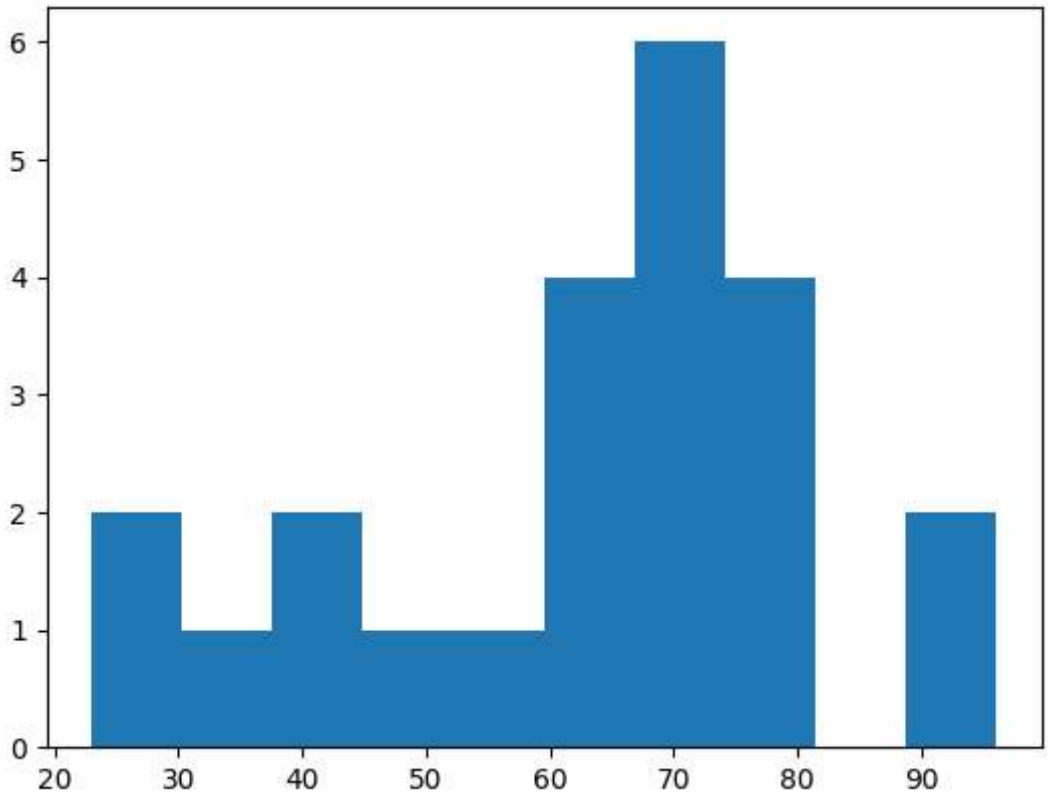


In [88]:
```
Z= np.random.normal(0, 1, 100)
plt.hist(Z)
```

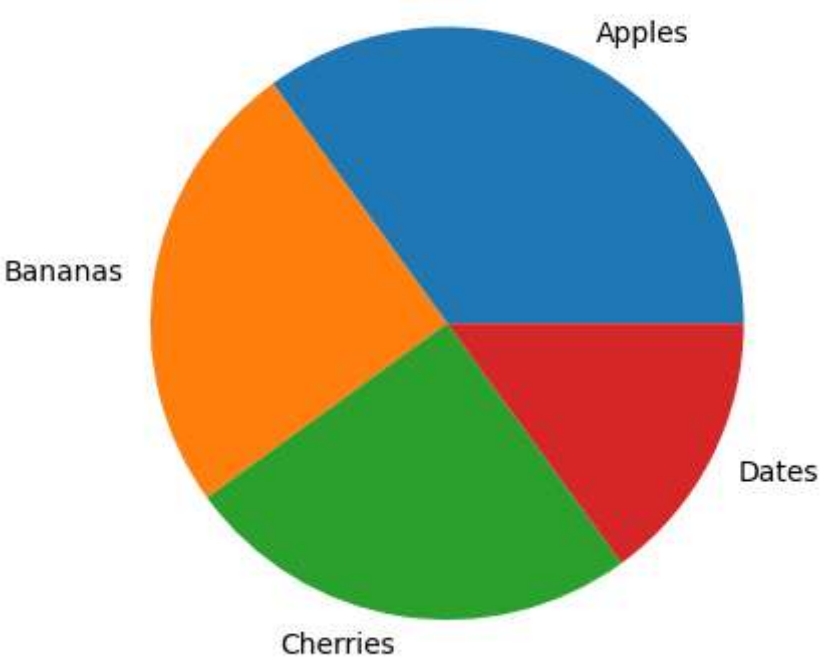
Out[88]:
```
(array([ 2.,  3., 10., 12., 17., 12., 20., 12.,  9.,  3.]),
 array([-2.52984851, -2.0345052 , -1.5391619 , -1.04381859, -0.54847528,
        -0.05313197,  0.44221134,  0.93755464,  1.43289795,  1.92824126,
         2.42358457]),
 <BarContainer object of 10 artists>)
```



In [86]:
```
# import module
import matplotlib.pyplot as plt

# create data
data = [32, 96, 45, 67, 76, 28, 79, 62, 43, 81, 70,61, 95, 44, 60, 69, 71, 23,

# create histogram
plt.hist(data)
# display histogram
plt.show()
```



In [87]:
```
y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
# myexplode = [0.2, 0, 0, 0]
plt.pie(y, labels = mylabels)#use explode = myexplode
plt.show()
```



Data Cleaning

In [54]:
```
# df.columns = ['a','b','c']  Rename columns
# pd.isnull() Checks for null Values, Returns Boolean Arrray
# pd.notnull() Opposite of pd.isnull()
# df.dropna() Drop all rows that contain null values
# df.dropna(axis=1)Drop all columns that contain null values
# df.dropna(axis=1,thresh=n) Drop all rows have have less than n non null valu
# df.fillna(x) Replace all null values with x
# s.fillna(s.mean()) Replace all null values with the mean (mean can bereplace
# s.astype(float) Convert the datatype of the series to float
# s.replace(1,'one') Replace all values equal to 1 with 'one
#df.groupby([col1,col2])Returns groupby object for values from multiplecolumns
```