In [1]:

**from**

matplotlib

**import**

pyplot

**as**

plt

In [2]:

*# 1. scatter plot*

*#use- to find relation between 2 variable(ex.age & sal)*

*# continuous variable*

In [ ]:

*# relation in that are 3 types*

*# 1.Positive relation -> x increases y increses(+1 strongly corelated / directely propor*

*# 2.Negative relation -> if x increses then y decreses(-ve corelatinal -1/universely prop*

*# 3.Polynomial->no relation between x & y data*

In [3]:

age **=** [12,22,32,42,52,62,13,23,43,53,56,66] salary **=**[112235,112545,556213,112256,112254,124565,123654,145632,147850,123698,125874,12 In [4]:

age

Out[4]: [12, 22, 32, 42, 52, 62, 13, 23, 43, 53, 56, 66] In [6]:

print

(

salary

)

[112235, 112545, 556213, 112256, 112254, 124565, 123654, 145632, 147850, 1

23698, 125874, 125689] In [9]:

len

(

age

)

Out[9]:

12 In [10]:

len

(

salary

)

Out[10]:

12

[11]: *# To know relation between salary & age* In [12]:

*#general statement ->plt.scatter(x,y)*

In [13]:

plt

.

scatter

(

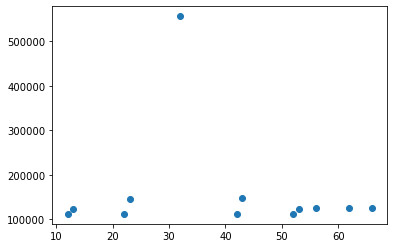
age

,

salary

)

Out[13]: <matplotlib.collections.PathCollection at 0x2399866b550>



In [14]:

ls

Volume in drive C is Windows-SSD

Volume Serial Number is 94E5-B974

Directory of C:\Users\prafu\ML\MATPLOT-LIB

19-06-2023 19:13 <DIR> .

19-06-2023 18:58 <DIR> ..

19-06-2023 18:58 <DIR> .ipynb\_checkpoints

19-06-2023 19:11 150 1-Copy1.csv

19-06-2023 19:11 938,020 Bengaluru\_House\_Data-Copy1.csv

19-06-2023 19:11 7,460 CardioGoodFitness-Copy1.csv

19-06-2023 19:11 14,784 country\_wise\_latest-Copy1.csv

19-06-2023 19:11 15,743 covid-Copy1.xlsx

19-06-2023 19:10 11,065 MATPLOTLIB.ipynb

19-06-2023 19:11 313,956 property-Copy1.xlsx

10-11-2022 10:36 279 test-1.csv

8 File(s) 1,301,457 bytes

3 Dir(s) 189,659,889,664 bytes free

[15]:

**import** pandas **as** pd In [16]:

df

**=**

pd

.

read\_csv

(

'CardioGoodFitness-Copy1.csv'

)

In [17]:

df

Out[17]:

**Product Age Gender Education MaritalStatus Usage Fitness Income Miles**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | TM195 | 18 | Male | 14 | Single | 3 | 4 | 29562 | 112 |
| **1** | TM195 | 19 | Male | 15 | Single | 2 | 3 | 31836 | 75 |
| **2** | TM195 | 19 | Female | 14 | Partnered | 4 | 3 | 30699 | 66 |
| **3** | TM195 | 19 | Male | 12 | Single | 3 | 3 | 32973 | 85 |
| **4** | TM195 | 20 | Male | 13 | Partnered | 4 | 2 | 35247 | 47 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **175** | TM798 | 40 | Male | 21 | Single | 6 | 5 | 83416 | 200 |
| **176** | TM798 | 42 | Male | 18 | Single | 5 | 4 | 89641 | 200 |
| **177** | TM798 | 45 | Male | 16 | Single | 5 | 5 | 90886 | 160 |
| **178** | TM798 | 47 | Male | 18 | Partnered | 4 | 5 | 104581 | 120 |
| **179** | TM798 | 48 | Male | 18 | Partnered | 4 | 5 | 95508 | 180 |

180 rows × 9 columns

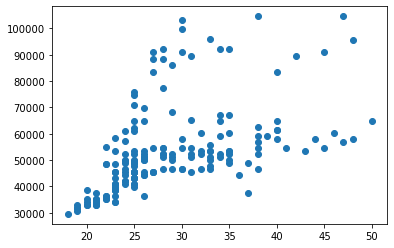
In [18]:

*# Age \* Income*

[19]:

plt.scatter(df['Age'],df['Income'])

Out[19]: <matplotlib.collections.PathCollection at 0x2399b615550>



In [20]:

plt

.

bar

(

df

[

'Age'

]

,

df

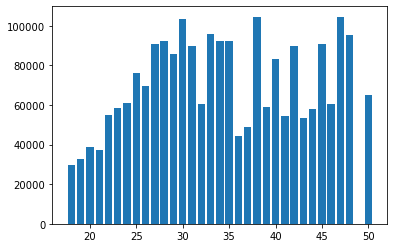
[

'Income'

])

Out[20]:

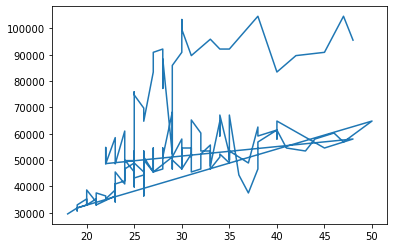
<BarContainer object of 180 artists>



[21]:

plt.plot(df['Age'],df['Income'])

Out[21]: [<matplotlib.lines.Line2D at 0x2399be58370>]



In [25]:

plt

.

plot

(

df

[

'Age'

]

,

df

[

'Income'

]

,

marker

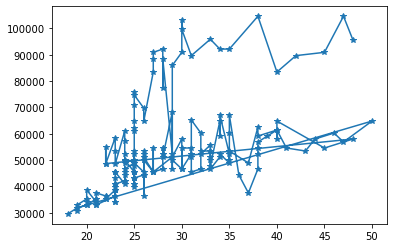
**=**

'\*'

)

Out[25]:

[<matplotlib.lines.Line2D at 0x2399bc55ee0>]

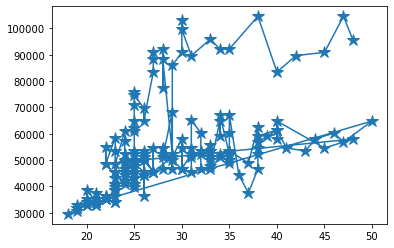


[26]:

plt.plot(df['Age'],df['Income'],marker**=**'\*',markersize**=**12)

Out[26]:

[<matplotlib.lines.Line2D at 0x2399c0a94f0>]



In

[27]:

Out[27]:

plt

.

plot

(

df

[

'Age'

]

,

df

[

'Income'

]

,

marker

**=**

'\*'

,

markersize

**=**

6

)

plt

.

title

(

"Test"

)

*#title of graph*

plt

.

xlabel

(

'Age'

)

*#x-axis name*

plt

.

ylabel

(

'Income'

)

*#y-axis name*

plt

.

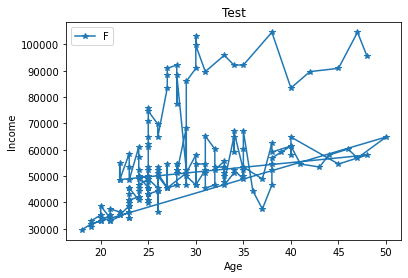
legend

(

"Fitness"

)

<matplotlib.legend.Legend at 0x2399c111fa0>



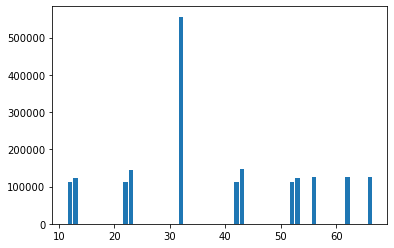
In [28]:

age **=** [12,22,32,42,52,62,13,23,43,53,56,66]

salary **=**[112235,112545,556213,112256,112254,124565,123654,145632,147850,123698,125874,12 [30]:

plt.bar(age,salary)

Out[30]: <BarContainer object of 12 artists>



In [32]:

x

**=**

[

"Python32"

,

"Python33"

,

"Python34"

,

"Python35"

]

girls

**=**

[

35

,

67

,

35

,

78

]

boys

**=**

[

24

,

75

,

24

,

87

]

In [33]:

**import**

numpy

**as**

np

In [34]:

x\_axis

**=**

np

.

arange

(

len

(

x

))

In [35]:

x\_axis

Out[35]:

array([0, 1, 2, 3])

In [ ]:

*# 0.4 is width*

[41]:

plt

.

bar

(

x\_axis

,

boys

,

0.4

,

label

**=**

"Boys"

)

plt

.

bar

(

x\_axis

,

girls

,

0.4

,

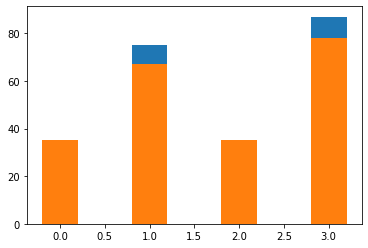
label

**=**

"girls"

)

Out[41]: <BarContainer object of 4 artists>



In [40]:

*#x\_axis-0.2 = > -0.2 is used just change their position from origin so we get the diff g*

*#x\_axis+0.2 => +0.2 is used just change their position from origin so we get the diff gra # otherwise we get merge result as above* In [37]:

plt

.

bar

(

x\_axis

**-**

0.2

,

boys

,

0.4

,

label

**=**

"Boys"

)

plt

.

bar

(

x\_axis

**+**

0.2

,

girls

,

0.4

,

label

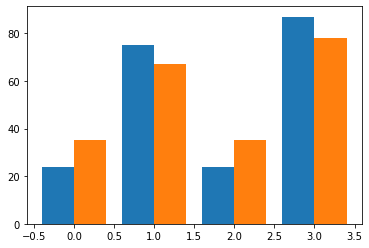
**=**

"girls"

)

Out[37]:

<BarContainer object of 4 artists>



[39]:

Out[39]:

plt

.

bar

(

x\_axis

**-**

0.2

,

boys

,

0.4

,

label

**=**

"Boys"

)

plt

.

bar

(

x\_axis

**+**

0.2

,

girls

,

0.4

,

label

**=**

"girls"

)

plt

.

xticks

(

x\_axis

,

x

)

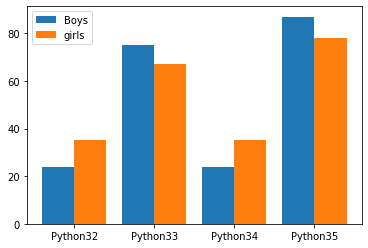
plt

.

legend

()

<matplotlib.legend.Legend at 0x2399c2b1520>



In [42]:

*#change in width*

In

[43]:

Out[43]:

plt

.

bar

(

x\_axis

**-**

0.2

,

boys

,

0.2

,

label

**=**

"Boys"

)

plt

.

bar

(

x\_axis

**+**

0.2

,

girls

,

0.4

,

label

**=**

"girls"

)

plt

.

xticks

(

x\_axis

,

x

)

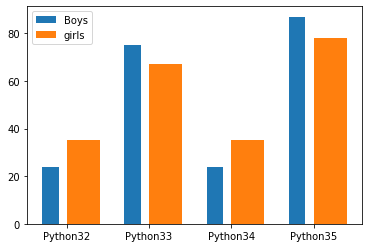
plt

.

legend

()

<matplotlib.legend.Legend at 0x2399c29fd90>



[31]:

x\_course**=**["python","java","c"] In [1]:

*# histogram or density graph*

In [ ]:

*#interview que*

1.

univariate

analysis

**-**

**if**

we

analyse

on

one

function

2.

bivariate

analysis

**-**

In [ ]:

1. Univariate data –

This type of data consists of only one variable.

The analysis of univariate data **is** thus the simplest form of analysis since the information deals **with** only one quantity that changes. It does **not** deal **with** causes **or** relationships **and** the main purpose of the analysis **is** to describe the data **and** find patterns that exist within it. The example of a univariate data can be height.

Suppose that the heights of seven students of a **class** **is** recorded(figure 1), there **is** only one variable that **is** height **and** it **is** **not** dealing **with** any cause **or** relati The description of patterns found **in** this type of data can be made by drawing conclusion using central tendency measures (mean, median **and** mode), dispersion **or** spread of data (range, minimum, maximum, quartiles, variance **and** standard deviation) **and** by using frequency distribution tables, histograms, pie charts, frequency polygon **and** bar charts.

1. Bivariate data

This type of data involves two different variables.

The analysis of this type of data deals **with** causes **and** relationships **and** the analysis **i** to find out the relationship among the two variables. Example of bivariate data can be temperature **and** ice cream sales **in** summer season.

Suppose the temperature **and** ice cream sales are the two variables of a bivariate data(fi Here, the relationship **is** visible **from** the table that temperature **and** sales are directly each other **and** thus related because **as** the temperature increases, the sales also increas Thus bivariate data analysis involves comparisons, relationships, causes **and** explanations.

These variables are often plotted on X **and** Y axis on the graph **for** better understanding of data **and** one of these variables **is** independent **while** the other **is** dependent.

In [ ]:

*# Outliers :- the data point which is far away from mean point/centre point is called ou*

[ ]:

*# ex. a=[1,2,1,1,2,2,56,1,2,69,2,1,2]*

outliers

are

56

**&**

69

In [ ]:

*# bar graph is used to categerical/compare values*

In [12]:

age\_data **=** [16,16,20,19,15,15,14,16,14,15,16,16,15,16,15,25,15,25,15,16,26,20,25,15,16,1 In [13]:

plt

.

hist

(

age\_data

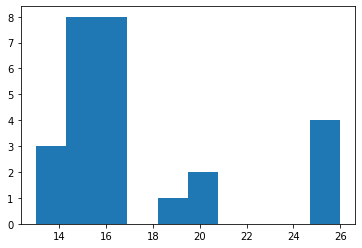
)

Out[13]:

(array([3., 8., 8., 0., 1., 2., 0., 0., 0., 4.]), array([13. , 14.3, 15.6, 16.9, 18.2, 19.5, 20.8, 22.1, 23.4, 24.7, 26.

]),

<BarContainer object of 10 artists>)



[14]:

plt.hist(age\_data,bins**=**20)

Out[14]:

(array([1., 2., 0., 8., 8., 0., 0., 0., 0., 1., 2., 0., 0., 0., 0., 0.,

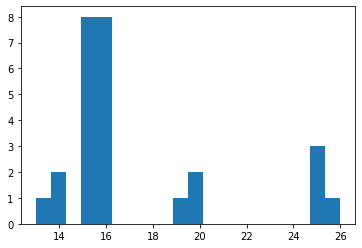
0.,

0., 3., 1.]),

array([13. , 13.65, 14.3 , 14.95, 15.6 , 16.25, 16.9 , 17.55, 18.2 , 18.85, 19.5 , 20.15, 20.8 , 21.45, 22.1 , 22.75, 23.4 , 24.05,

24.7 , 25.35, 26. ]),

<BarContainer object of 20 artists>)



In [20]:

plt

.

hist

(

age\_data

,

bins

**=**

10

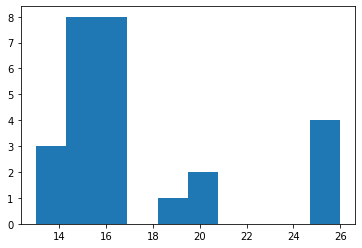
)

Out[20]:

(array([3., 8., 8., 0., 1., 2., 0., 0., 0., 4.]), array([13. , 14.3, 15.6, 16.9, 18.2, 19.5, 20.8, 22.1, 23.4, 24.7, 26.

]),

<BarContainer object of 10 artists>)



[21]:

*# pie chart* In [22]:

courses

**=**

[

'python'

,

'java'

,

'testing'

,

'aws'

]

In [23]:

count

**=**

[

150

,

25

,

200

,

45

]

In [24]:

plt

.

pie

(

count

)

Out[24]:

([<matplotlib.patches.Wedge at 0x1ea37492ee0>, <matplotlib.patches.Wedge at 0x1ea374a0400>,

<matplotlib.patches.Wedge at 0x1ea374a08e0>,

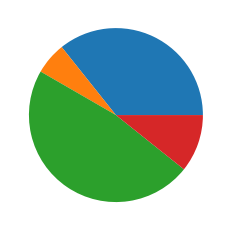
<matplotlib.patches.Wedge at 0x1ea374a0dc0>],

[Text(0.4772720865178074, 0.9910657674599294, ''),

Text(-0.8337689342477049, 0.7175161073337982, ''),

Text(-0.6196520101798082, -0.9088626883529344, ''),

Text(1.0382716876358113, -0.363306898714963, '')])



[25]:

plt.pie(count,labels**=**courses)

Out[25]:

([<matplotlib.patches.Wedge at 0x1ea374ea610>, <matplotlib.patches.Wedge at 0x1ea374eaaf0>,

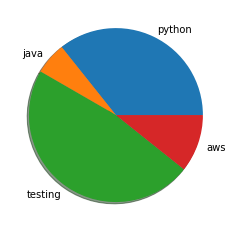
<matplotlib.patches.Wedge at 0x1ea374eafd0>,

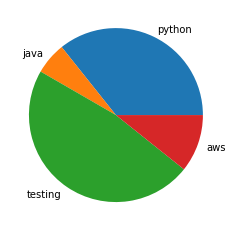
<matplotlib.patches.Wedge at 0x1ea374f84f0>],

[Text(0.4772720865178074, 0.9910657674599294, 'python'),

Text(-0.8337689342477049, 0.7175161073337982, 'java'),

Text(-0.6196520101798082, -0.9088626883529344, 'testing'),

 Text(1.0382716876358113, -0.363306898714963, 'aws')])



In [26]:

plt

.

pie

(

count

,

labels

**=**

courses

,

shadow

**=**

**True**

)

Out[26]:

([<matplotlib.patches.Wedge at 0x1ea37536f40>, <matplotlib.patches.Wedge at 0x1ea37543610>,

<matplotlib.patches.Wedge at 0x1ea37450d60>,

<matplotlib.patches.Wedge at 0x1ea3754f430>],

[Text(0.4772720865178074, 0.9910657674599294, 'python'), Text(-0.8337689342477049, 0.7175161073337982, 'java'),

Text(-0.6196520101798082, -0.9088626883529344, 'testing'),

Text(1.0382716876358113, -0.363306898714963, 'aws')])

[28]:

plt.pie(count,labels**=**courses,shadow**=True**,explode**=**(0.2,0,0,0))

Out[28]:

([<matplotlib.patches.Wedge at 0x1ea3b834040>, <matplotlib.patches.Wedge at 0x1ea3b83b670>,

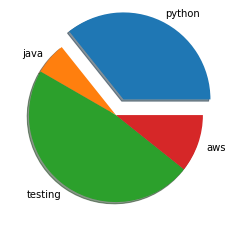
<matplotlib.patches.Wedge at 0x1ea3b83bdc0>,

<matplotlib.patches.Wedge at 0x1ea3b847550>],

[Text(0.5640488295210451, 1.1712595433617348, 'python'), Text(-0.8337689342477049, 0.7175161073337982, 'java'),

Text(-0.6196520101798082, -0.9088626883529344, 'testing'),

Text(1.0382716876358113, -0.363306898714963, 'aws')])



[37]:

plt.pie(count,labels**=**courses,shadow**=True**,explode**=**(0.2,0,0,0),autopct**=**"%1.2f")

Out[37]:

([<matplotlib.patches.Wedge at 0x1ea3bb261c0>, <matplotlib.patches.Wedge at 0x1ea3bb26b80>,

<matplotlib.patches.Wedge at 0x1ea3bb33550>,

<matplotlib.patches.Wedge at 0x1ea3bb33ee0>],

[Text(0.5640488295210451, 1.1712595433617348, 'python'), Text(-0.8337689342477049, 0.7175161073337982, 'java'),

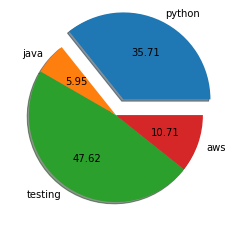
Text(-0.6196520101798082, -0.9088626883529344, 'testing'),

Text(1.0382716876358113, -0.363306898714963, 'aws')],

[Text(0.3471069720129508, 0.7207751036072212, '35.71'), Text(-0.4547830550442026, 0.3913724221820717, '5.95'),

Text(-0.33799200555262265, -0.4957432845561459, '47.62'),

Text(0.5663300114377152, -0.1981673992990707, '10.71')])



[34]:

plt.pie(count,labels**=**courses,shadow**=True**,explode**=**(0.2,0,0,0),autopct**=**"%1.2f%%")

Out[34]:

([<matplotlib.patches.Wedge at 0x1ea3ba0d310>, <matplotlib.patches.Wedge at 0x1ea3ba0dca0>,

<matplotlib.patches.Wedge at 0x1ea3ba1a640>,

<matplotlib.patches.Wedge at 0x1ea3ba1afd0>],

[Text(0.5640488295210451, 1.1712595433617348, 'python'),

Text(-0.8337689342477049, 0.7175161073337982, 'java'),

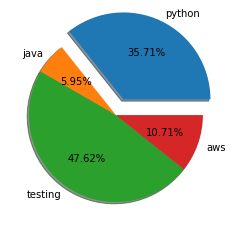
Text(-0.6196520101798082, -0.9088626883529344, 'testing'),

Text(1.0382716876358113, -0.363306898714963, 'aws')],

[Text(0.3471069720129508, 0.7207751036072212, '35.71%'), Text(-0.4547830550442026, 0.3913724221820717, '5.95%'),

Text(-0.33799200555262265, -0.4957432845561459, '47.62%'),

Text(0.5663300114377152, -0.1981673992990707, '10.71%')])



In [ ]: