In [1]:
```
# txt file read
f = open("file1.txt", "r")
print(f.read())
```

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
```

In [3]:
```
#txt file write
f = open("file1.txt", "a")
f.write("Now the file has more content!")
f.close()

#open and read the file after the appending:
f = open("file1.txt", "r")
print(f.read())
```

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!Now the file has more content!
```

In [9]:
```
#csv file read
import csv
f = open("file2.csv")
print(f.read())
```

```
player,runs
virat,56
rohit,45
yuvraj,23
```

In [10]:
```
# csv write
import csv
f = open("file2.csv","a",newline="")
tup1=("dhoni",33,)
writer=csv.writer(f)
writer.writerow(tup1)
f.close()

f = open("file2.csv")
print(f.read())
```

```
player,runs
virat,56
rohit,45
yuvraj,23
dhoni,33
```

In [18]:
```
#json read
f = open("file3.json")
print(f.read())
```

```
{
    "fruite":"apple",
    "size":"large",
    "color":"red"
}
```

In [19]:
```
import json

# Data to be written
dictionary = {"quantity": 5}

with open("file3.json", "a") as outfile:
    json.dump(dictionary, outfile)

f = open("file3.json")
print(f.read())
```

```
{
    "fruite":"apple",
    "size":"large",
    "color":"red"
}{"quantity": 5}
```

In [ ]:
```

```

In [16]:
```
#2. Python program to import libraries for loading & read a
#dataset. (Use head(), tail(), shape, info() describe() columns)
import seaborn as sns
dt=sns.load_dataset("iris")
print(dt)
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

In [17]:
```
dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [18]:
```
dt.head(10)
```

Out[18]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa

In [19]:
```
dt.tail(8)
```

Out[19]:

	sepal_length	sepal_width	petal_length	petal_width	species
142	5.8	2.7	5.1	1.9	virginica
143	6.8	3.2	5.9	2.3	virginica
144	6.7	3.3	5.7	2.5	virginica
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

In [20]:
```
dt.describe()
```

Out[20]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [21]:
```
dt.columns
```

Out[21]:
```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [22]:
```
dt.shape
```

Out[22]:
```
(150, 5)
```

In [24]:
```
dt.size
```

Out[24]:
```
750
```

In [25]:
```
#3. Write a python program to reshaping data-
#Convert categorical data into numerical value using dataset.
import seaborn as sns
dt=sns.load_dataset("iris")
dt["species"]=dt["species"].replace({'setosa':1,'versicolor':2,'virginica':
dt
```

Out[25]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	1
1	4.9	3.0	1.4	0.2	1
2	4.7	3.2	1.3	0.2	1
3	4.6	3.1	1.5	0.2	1
4	5.0	3.6	1.4	0.2	1
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	3
146	6.3	2.5	5.0	1.9	3
147	6.5	3.0	5.2	2.0	3
148	6.2	3.4	5.4	2.3	3
149	5.9	3.0	5.1	1.8	3

150 rows × 5 columns

In [26]:
```
# 4. Implementation of data cleaning-finding missing data removing
# and filling missing data.
import pandas as pd
df=pd.read_excel(r"C:\Users\PURVA\Desktop\data.xlsx")
df
```

Out[26]:

	Duration	Pulse	Maxpulse	Calories
0	60.0	110.0	130	409.1
1	60.0	117.0	145	479.0
2	60.0	103.0	135	340.0
3	45.0	NaN	175	NaN
4	NaN	117.0	148	406.0
5	60.0	102.0	127	NaN
6	60.0	110.0	136	374.0
7	45.0	110.0	134	253.3
8	30.0	109.0	133	195.1
9	60.0	98.0	124	269.0
10	60.0	103.0	147	329.3

In [27]:
```
df.isnull()
```

Out[27]:

	Duration	Pulse	Maxpulse	Calories
0	False	False	False	False
1	False	False	False	False
2	False	False	False	False
3	False	True	False	True
4	True	False	False	False
5	False	False	False	True
6	False	False	False	False
7	False	False	False	False
8	False	False	False	False
9	False	False	False	False
10	False	False	False	False

In [28]:
```
df.isnull().sum()
```

Out[28]:
```
Duration    1
Pulse       1
Maxpulse    0
Calories    2
dtype: int64
```

In [29]:
```
df.dropna()
```

Out[29]:

	Duration	Pulse	Maxpulse	Calories
0	60.0	110.0	130	409.1
1	60.0	117.0	145	479.0
2	60.0	103.0	135	340.0
6	60.0	110.0	136	374.0
7	45.0	110.0	134	253.3
8	30.0	109.0	133	195.1
9	60.0	98.0	124	269.0
10	60.0	103.0	147	329.3

In [30]:
```
df.fillna({"Duration":30,"Pulse":97,"Maxpulse":100,"Calories":200})
```

Out[30]:

	Duration	Pulse	Maxpulse	Calories
0	60.0	110.0	130	409.1
1	60.0	117.0	145	479.0
2	60.0	103.0	135	340.0
3	45.0	97.0	175	200.0
4	30.0	117.0	148	406.0
5	60.0	102.0	127	200.0
6	60.0	110.0	136	374.0
7	45.0	110.0	134	253.3
8	30.0	109.0	133	195.1
9	60.0	98.0	124	269.0
10	60.0	103.0	147	329.3

In [31]:
```
#5. Write a python program implement data wrangling operations-
# filtering and removing duplication of data.
import pandas as pd
data=pd.DataFrame({"Name":["Mary","Amey","John","raj","John"],
 "Age":[20,19,18,20,18]})
print(data)
var=data.drop_duplicates()
var
```

```
   Name  Age
0  Mary   20
1  Amey   19
2  John   18
3   raj   20
4  John   18
```

Out[31]:

	Name	Age
0	Mary	20
1	Amey	19
2	John	18
3	raj	20

In [32]:
```
# 6. Python program to Implement data transformation -
#Combine data frames/datasets using join() merge(),concat() etc.
dt1={'id':[1,2,3,4,5,6],
'name': ['John Deo','Max Ruin','Arnold','Krish Star','John Mike',
         'Alex John'],
 'class':['Four','Three','Three','Four','Four','Four']}
dt2={'id':[1,2,3,4,5,6],
 'gender':['female','male','male','female','female','male'],
 'city':['pune','mumbai','junnar','pune','mumbai','pune']}
df1=pd.DataFrame(dt1)
df2=pd.DataFrame(dt2)
merge_data=pd.merge(df1,df2,on="id")
merge_data
```

Out[32]:

	id	name	class	gender	city
0	1	John Deo	Four	female	pune
1	2	Max Ruin	Three	male	mumbai
2	3	Arnold	Three	male	junnar
3	4	Krish Star	Four	female	pune
4	5	John Mike	Four	female	mumbai
5	6	Alex John	Four	male	pune

In [33]:
```
dt1={'id':[1,2,3,4,5,6],
'name': ['John Deo','Max Ruin','Arnold','Krish Star','John Mike',
         'Alex John'],
 'class':['Four','Three','Three','Four','Four','Four']}
dt2={'gender':['female','male','male','female','female','male'],
 'city':['pune','mumbai','junnar','pune','mumbai','pune']}
df1=pd.DataFrame(dt1)
df2=pd.DataFrame(dt2)
df1.join(df2)
```

Out[33]:

	id	name	class	gender	city
0	1	John Deo	Four	female	pune
1	2	Max Ruin	Three	male	mumbai
2	3	Arnold	Three	male	junnar
3	4	Krish Star	Four	female	pune
4	5	John Mike	Four	female	mumbai
5	6	Alex John	Four	male	pune

In [34]:
```
dt1={'name':['a','b','c','d'],
 'age':[20,21,18,20]}
dt2={'name':['E','F','G','H'],
 'age':[18,21,20,19]}
df1=pd.DataFrame(dt1)
df2=pd.DataFrame(dt2)
var=pd.concat([df1,df2],axis=1)
var
```

Out[34]:

	name	age	name	age
0	a	20	E	18
1	b	21	F	21
2	c	18	G	20
3	d	20	H	19

In [35]:
```
#7. Using the inbuilt mtcar dataset perform the following
import pandas as pd
data = pd.read_excel(r'C:\Users\PURVA\Downloads\mtcar.xlsx')
data
```

Out[35]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
11	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
12	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
13	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
14	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
15	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
16	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
20	Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
23	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
26	Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
27	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
28	Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
30	Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

In [36]:
```
#a. Display all the cars having 4 gears
data[data['gear']==4]
```

Out[36]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

In [37]:
```
#b. Display all the cars having 3 gears and 2 carburetor.
data[(data['gear']==3)&(data['carb']==2)]
```

Out[37]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2

In [38]:
```
#8. Using inbuilt dataset women perform the following
import pandas as pd
data2 = pd.read_excel(r'C:\Users\PURVA\Downloads\women.xlsx')
data2
```

Out[38]:

	id	weight	height
0	1	58	115
1	2	59	117
2	3	60	120
3	4	61	123
4	5	62	126
5	6	63	129
6	7	64	132
7	8	65	135
8	9	66	139
9	10	67	142
10	11	68	146
11	12	69	150
12	13	70	154
13	14	71	159
14	15	72	164

In [39]:
```
#a. display all rows of dataset having height greater than 120.
data2[data2['height']>120]
```

Out[39]:

	id	weight	height
3	4	61	123
4	5	62	126
5	6	63	129
6	7	64	132
7	8	65	135
8	9	66	139
9	10	67	142
10	11	68	146
11	12	69	150
12	13	70	154
13	14	71	159
14	15	72	164

In [40]:
```
# b. display all rows of dataset in ascending order of weight
data2.sort_values("weight")
```

Out[40]:

	id	weight	height
0	1	58	115
1	2	59	117
2	3	60	120
3	4	61	123
4	5	62	126
5	6	63	129
6	7	64	132
7	8	65	135
8	9	66	139
9	10	67	142
10	11	68	146
11	12	69	150
12	13	70	154
13	14	71	159
14	15	72	164

In [41]:
```
#9. Using the inbuilt airquality dataset perform the following
import pandas as pd
data3 = pd.read_excel(r'C:\Users\PURVA\Downloads\airquality.xlsx')
data3
```

Out[41]:

	Ozone	ozone	solar.R	wind	temp	month	day
0	1	41.0	190.0	7.4	67	5	1
1	2	36.0	118.0	8.0	72	5	2
2	3	12.0	149.0	12.6	74	5	3
3	4	18.0	313.0	11.5	62	5	4
4	5	NaN	NaN	14.3	56	5	5
...	...	...	...	...	...	...	...
148	149	30.0	193.0	6.9	70	9	26
149	150	NaN	145.0	13.2	77	9	27
150	151	14.0	191.0	14.3	75	9	28
151	152	18.0	131.0	8.0	76	9	29
152	153	20.0	223.0	11.5	68	9	30

153 rows × 7 columns

In [42]:
```
#a. Find the temperature of day 30 of month 8
data3[(data3["day"]==30)& (data3["month"]==8)]
```

Out[42]:

	Ozone	ozone	solar.R	wind	temp	month	day
121	122	84.0	237.0	6.3	96	8	30

In [43]:
```
#b. Display the details of all the days if the
# temperature is greater than 90
data3[data3['temp']>90]
```

Out[43]:

	Ozone	ozone	solar.R	wind	temp	month	day
41	42	NaN	259.0	10.9	93	6	11
42	43	NaN	250.0	9.2	92	6	12
68	69	97.0	267.0	6.3	92	7	8
69	70	97.0	272.0	5.7	92	7	9
74	75	NaN	291.0	14.9	91	7	14
101	102	NaN	222.0	8.6	92	8	10
119	120	76.0	203.0	9.7	97	8	28
120	121	118.0	225.0	2.3	94	8	29
121	122	84.0	237.0	6.3	96	8	30
122	123	85.0	188.0	6.3	94	8	31
123	124	96.0	167.0	6.9	91	9	1
124	125	78.0	197.0	5.1	92	9	2
125	126	73.0	183.0	2.8	93	9	3
126	127	91.0	189.0	4.6	93	9	4

In [44]:
```
#10. Using iris inbuilt dataset perform the following
import seaborn as sns
df=sns.load_dataset("iris")
df
```

Out[44]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

In [45]:
```
#a. Display details of all flowers of type virginica in ascending
# order of petal length.
dt1=df[df["species"]=="virginica"]
dt1.sort_values("petal_length")
```

Out[45]:

	sepal_length	sepal_width	petal_length	petal_width	species
106	4.9	2.5	4.5	1.7	virginica
126	6.2	2.8	4.8	1.8	virginica
138	6.0	3.0	4.8	1.8	virginica
127	6.1	3.0	4.9	1.8	virginica
121	5.6	2.8	4.9	2.0	virginica
123	6.3	2.7	4.9	1.8	virginica
119	6.0	2.2	5.0	1.5	virginica
146	6.3	2.5	5.0	1.9	virginica
113	5.7	2.5	5.0	2.0	virginica
133	6.3	2.8	5.1	1.5	virginica
114	5.8	2.8	5.1	2.4	virginica
149	5.9	3.0	5.1	1.8	virginica
141	6.9	3.1	5.1	2.3	virginica
101	5.8	2.7	5.1	1.9	virginica
110	6.5	3.2	5.1	2.0	virginica
142	5.8	2.7	5.1	1.9	virginica
145	6.7	3.0	5.2	2.3	virginica
147	6.5	3.0	5.2	2.0	virginica
115	6.4	3.2	5.3	2.3	virginica
111	6.4	2.7	5.3	1.9	virginica
148	6.2	3.4	5.4	2.3	virginica
139	6.9	3.1	5.4	2.1	virginica
116	6.5	3.0	5.5	1.8	virginica
112	6.8	3.0	5.5	2.1	virginica
137	6.4	3.1	5.5	1.8	virginica
140	6.7	3.1	5.6	2.4	virginica
136	6.3	3.4	5.6	2.4	virginica
103	6.3	2.9	5.6	1.8	virginica
128	6.4	2.8	5.6	2.1	virginica
134	6.1	2.6	5.6	1.4	virginica
132	6.4	2.8	5.6	2.2	virginica
144	6.7	3.3	5.7	2.5	virginica
124	6.7	3.3	5.7	2.1	virginica
120	6.9	3.2	5.7	2.3	virginica
129	7.2	3.0	5.8	1.6	virginica
104	6.5	3.0	5.8	2.2	virginica
108	6.7	2.5	5.8	1.8	virginica
102	7.1	3.0	5.9	2.1	virginica
143	6.8	3.2	5.9	2.3	virginica

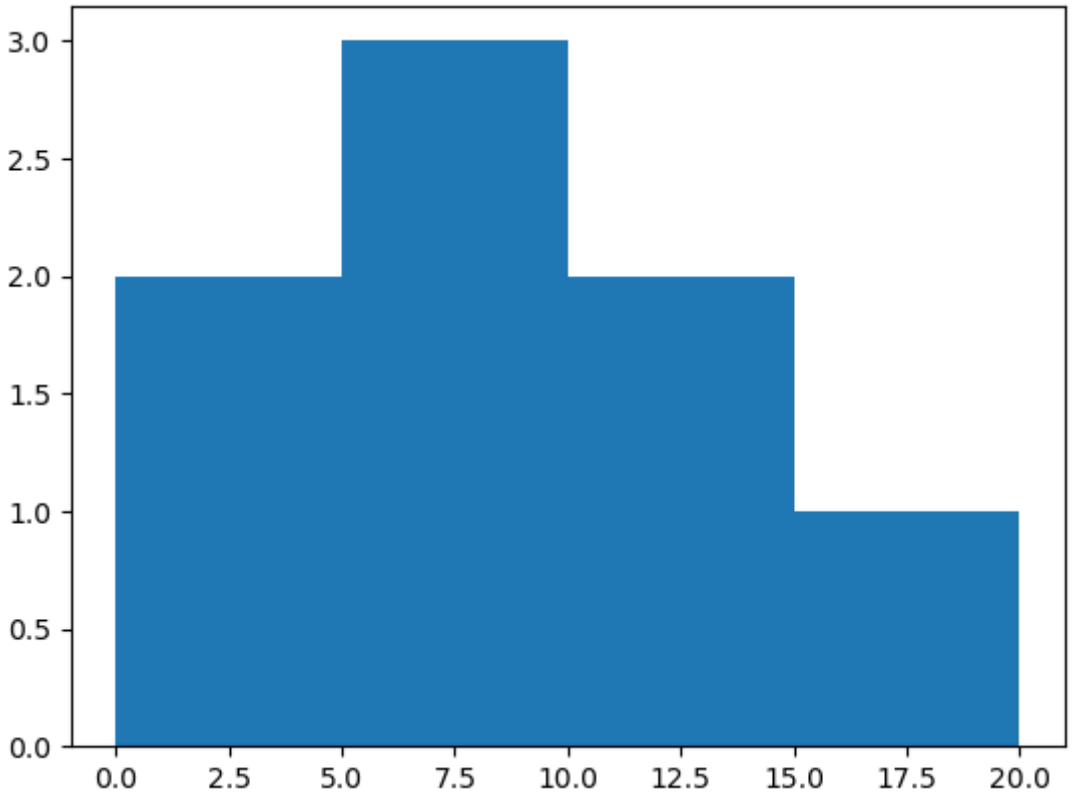
	sepal_length	sepal_width	petal_length	petal_width	species
100	6.3	3.3	6.0	2.5	virginica
125	7.2	3.2	6.0	1.8	virginica
130	7.4	2.8	6.1	1.9	virginica
135	7.7	3.0	6.1	2.3	virginica
109	7.2	3.6	6.1	2.5	virginica
107	7.3	2.9	6.3	1.8	virginica
131	7.9	3.8	6.4	2.0	virginica
105	7.6	3.0	6.6	2.1	virginica
117	7.7	3.8	6.7	2.2	virginica
122	7.7	2.8	6.7	2.0	virginica
118	7.7	2.6	6.9	2.3	virginica

In [46]:
```
#b. Display details of first five flowers of type setosa
# having maximum petal length..
dt2=df[df["species"]=="setosa"]
dt2.sort_values("petal_length",ascending=False).head()
```

Out[46]:

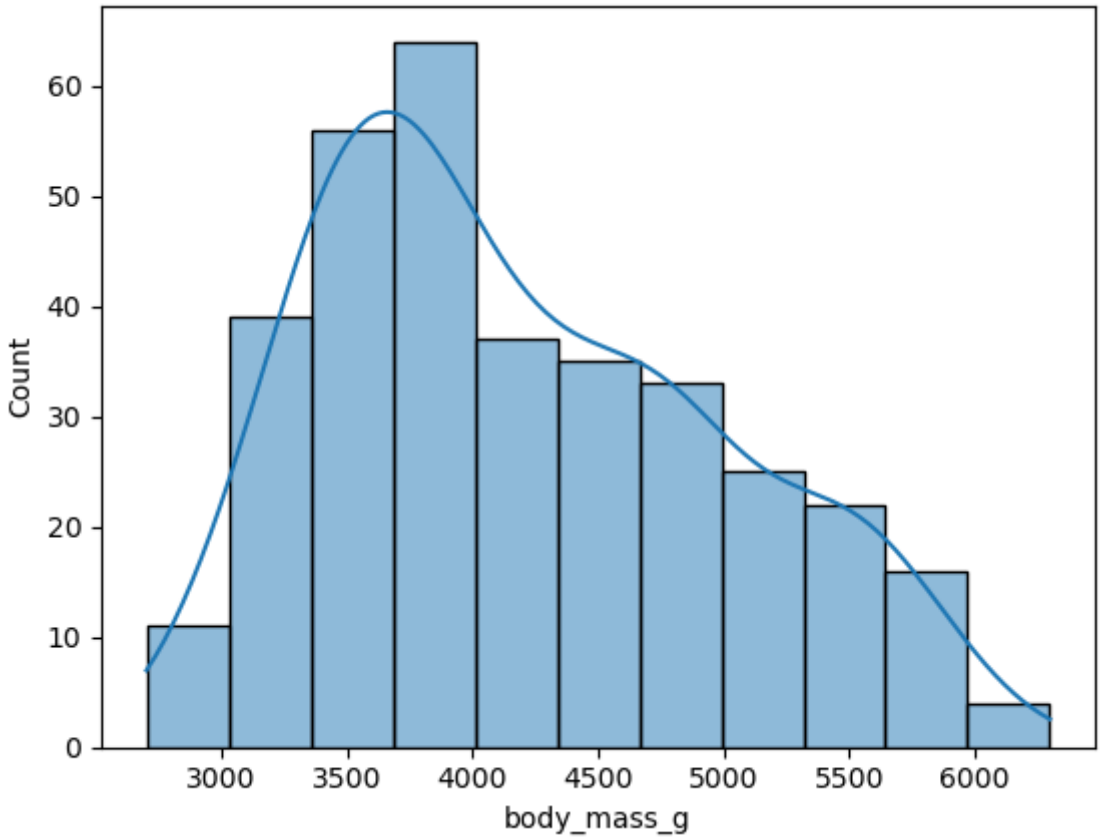
	sepal_length	sepal_width	petal_length	petal_width	species
24	4.8	3.4	1.9	0.2	setosa
44	5.1	3.8	1.9	0.4	setosa
23	5.1	3.3	1.7	0.5	setosa
5	5.4	3.9	1.7	0.4	setosa
20	5.4	3.4	1.7	0.2	setosa

In [47]:
```
#11. Write a python program to representation of data using Histogram.
import matplotlib.pyplot as plt
import numpy as np
y=np.array([1,3,5,7,6,14,12,15])
plt.hist(y,bins=[0,5,10,15,20])
plt.show()
```



In [48]:
```
import seaborn as sns
penguins=sns.load_dataset("penguins")
sns.histplot(data=penguins,x="body_mass_g",kde=True)
```

Out[48]:
```
<Axes: xlabel='body_mass_g', ylabel='Count'>
```



In [50]:
```
#12. Using airquality dataset
import pandas as pd
df = pd.read_excel(r'C:\Users\PURVA\Downloads\airquality.xlsx')
df
```

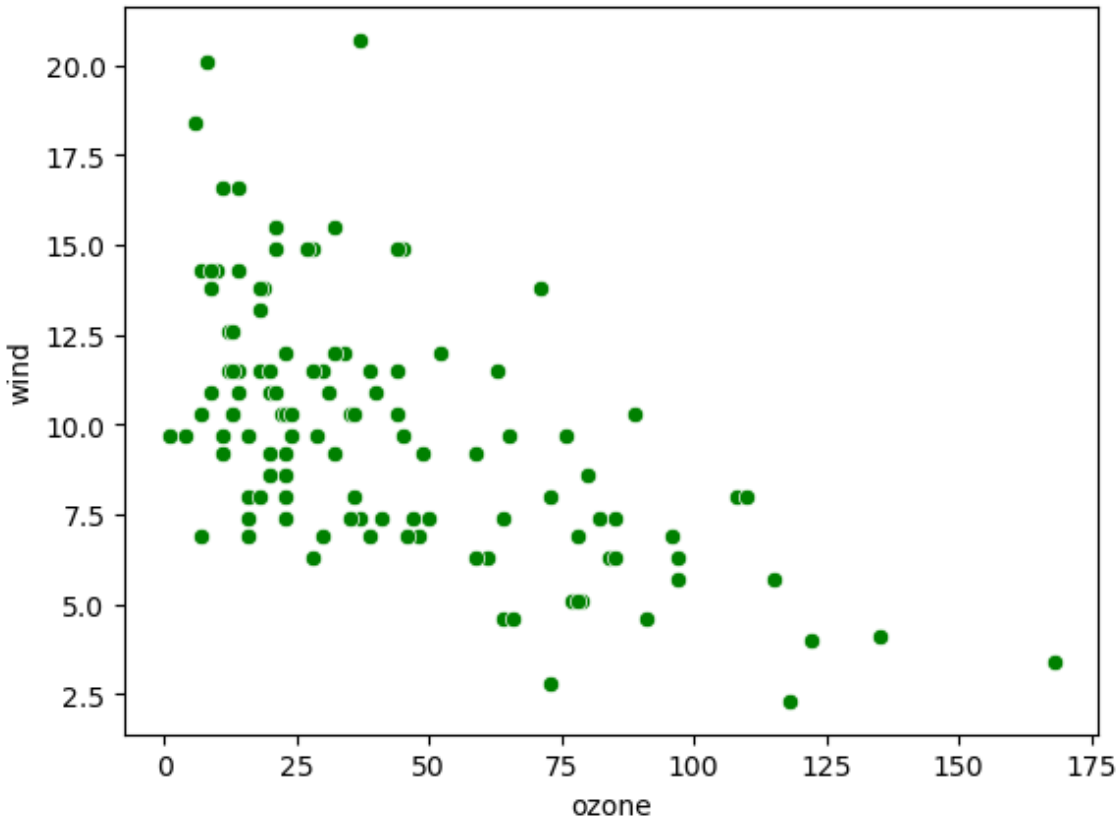
Out[50]:

	Ozone	ozone	solar.R	wind	temp	month	day
0	1	41.0	190.0	7.4	67	5	1
1	2	36.0	118.0	8.0	72	5	2
2	3	12.0	149.0	12.6	74	5	3
3	4	18.0	313.0	11.5	62	5	4
4	5	NaN	NaN	14.3	56	5	5
...	...	...	...	...	...	...	...
148	149	30.0	193.0	6.9	70	9	26
149	150	NaN	145.0	13.2	77	9	27
150	151	14.0	191.0	14.3	75	9	28
151	152	18.0	131.0	8.0	76	9	29
152	153	20.0	223.0	11.5	68	9	30

153 rows × 7 columns

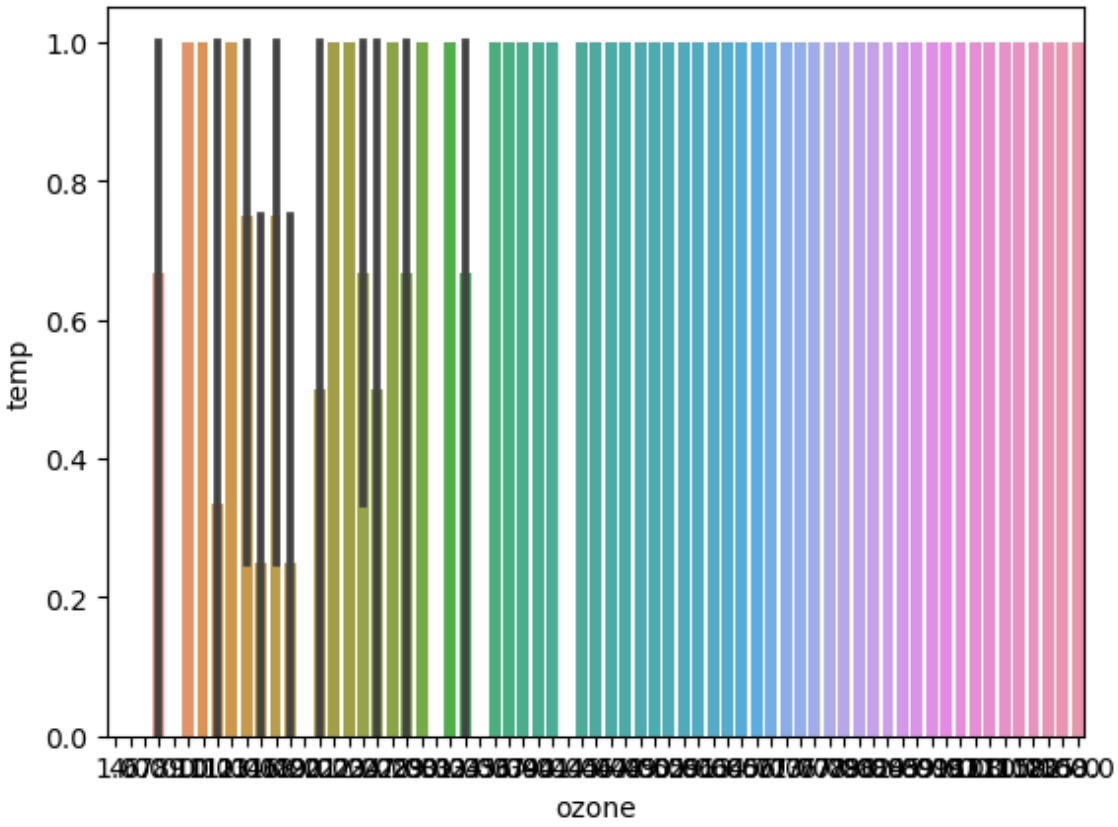
In [51]:
```
#a. Create a scatter plot to show the relationship between
# ozone and wind values by giving appropriate value to color argument
sns.scatterplot(x="ozone",y="wind",data=df,color='g')
```

Out[51]:
```
<Axes: xlabel='ozone', ylabel='wind'>
```



In [52]:
```
# b. Create a bar plot to show the ozone level for all the days
# having temperature greater than 70
x1=df['ozone']
y1=df['temp']>70
sns.barplot(x=x1,y=y1)
```

Out[52]: <Axes: xlabel='ozone', ylabel='temp'>



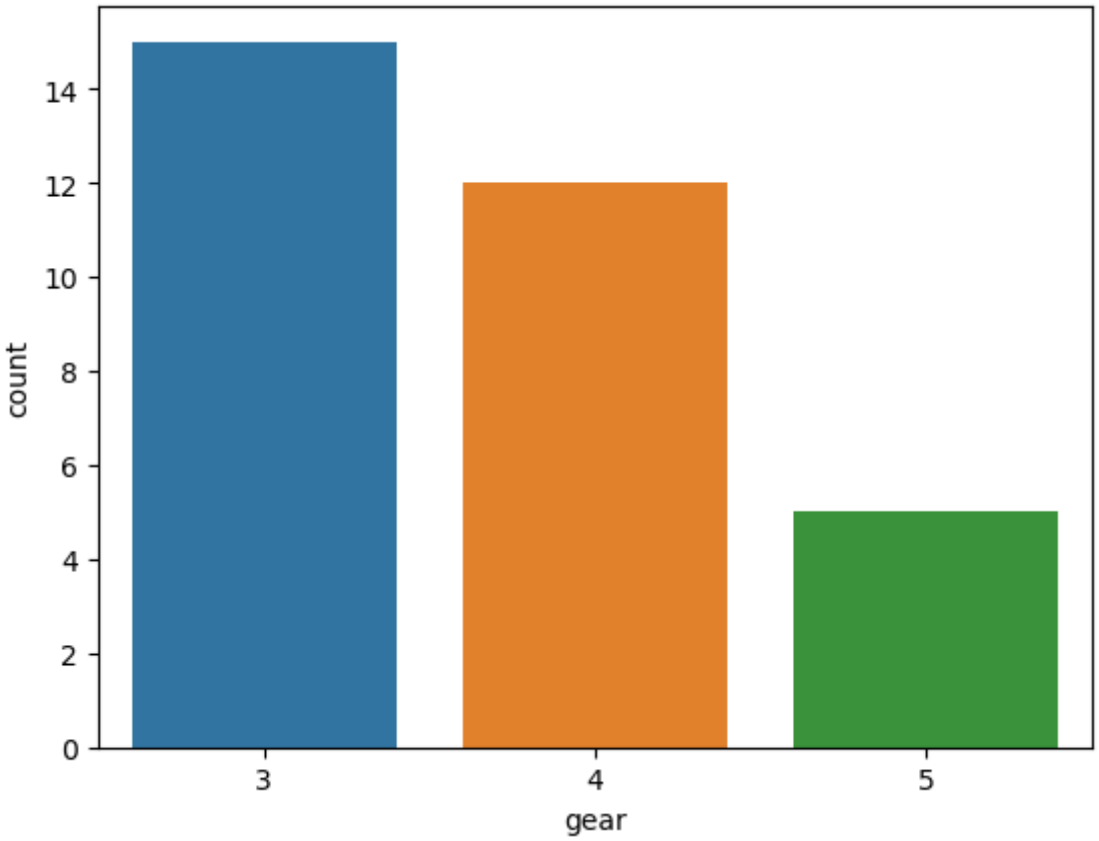
In [53]:
```
#13. Using inbuilt mtcars dataset
import pandas as pd
data = pd.read_excel(r'C:\Users\PURVA\Downloads\mtcar.xlsx')
data
```

Out[53]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
10	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
11	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
12	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
13	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
14	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
15	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
16	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
17	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
18	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
19	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
20	Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
21	Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
22	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
23	Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
24	Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
25	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
26	Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
27	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
28	Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
29	Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
30	Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
31	Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

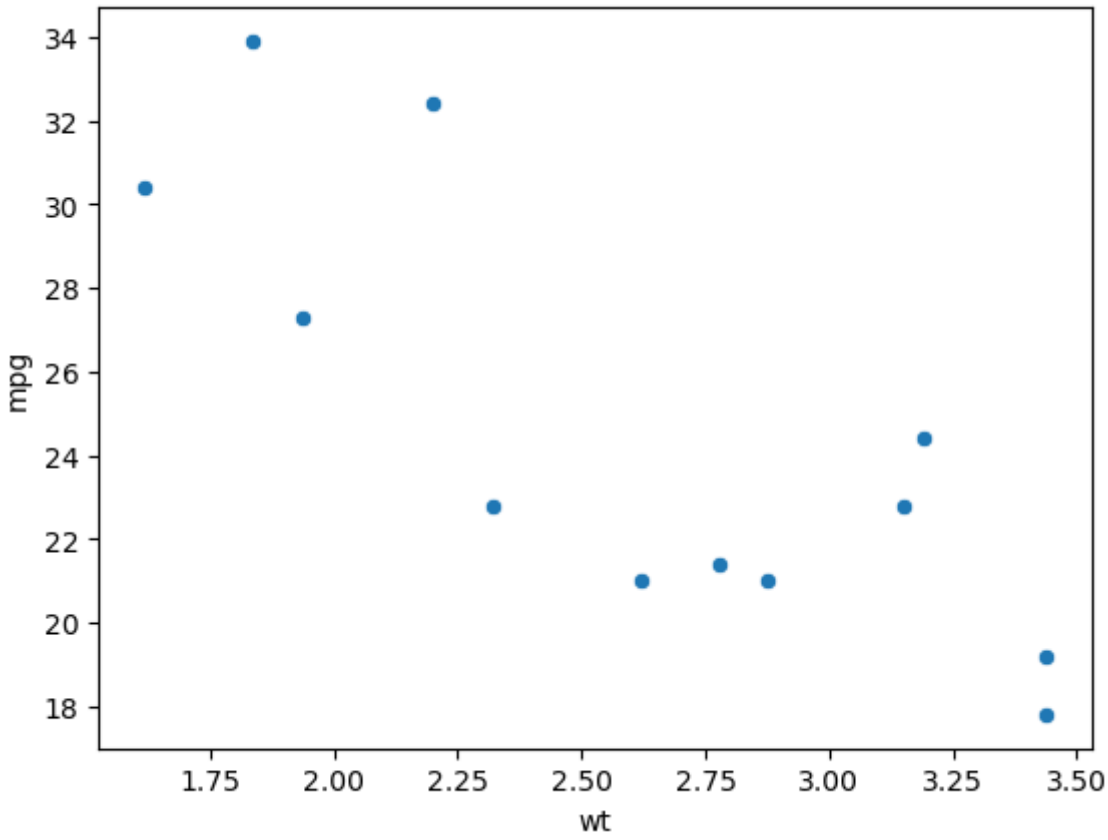
In [54]:
```
#a. Create a bar plot that shows the number of cars of each gear type.
sns.countplot(data=data,x="gear")
```

Out[54]:
```
<Axes: xlabel='gear', ylabel='count'>
```

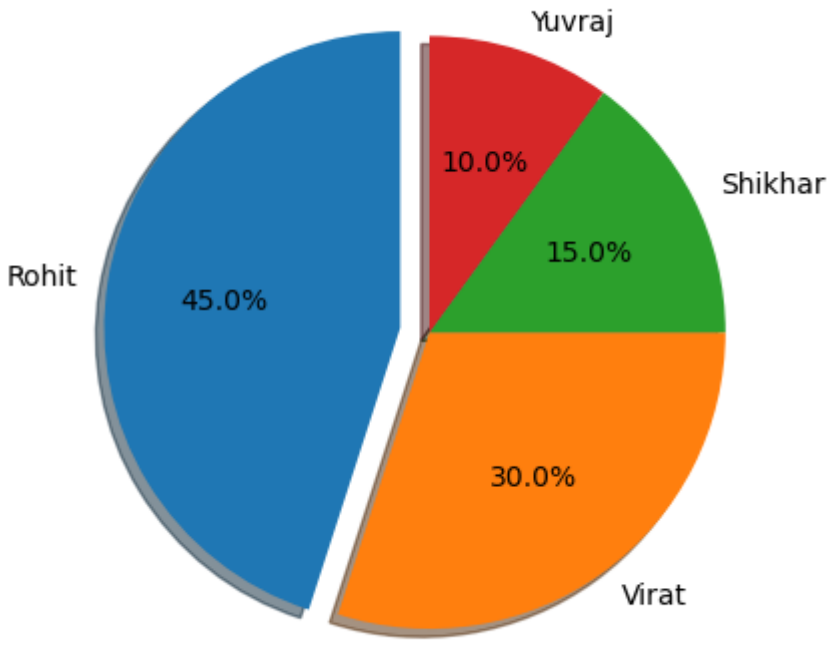


In [55]:
```
#b. Draw a scatter plot showing the relationship between
# wt and mpg for all the cars having 4 gear------
import seaborn as sns
x=data[data['gear']==4]
sns.scatterplot(x='wt',y='mpg',data=x)
```

Out[55]: <Axes: xlabel='wt', ylabel='mpg'>



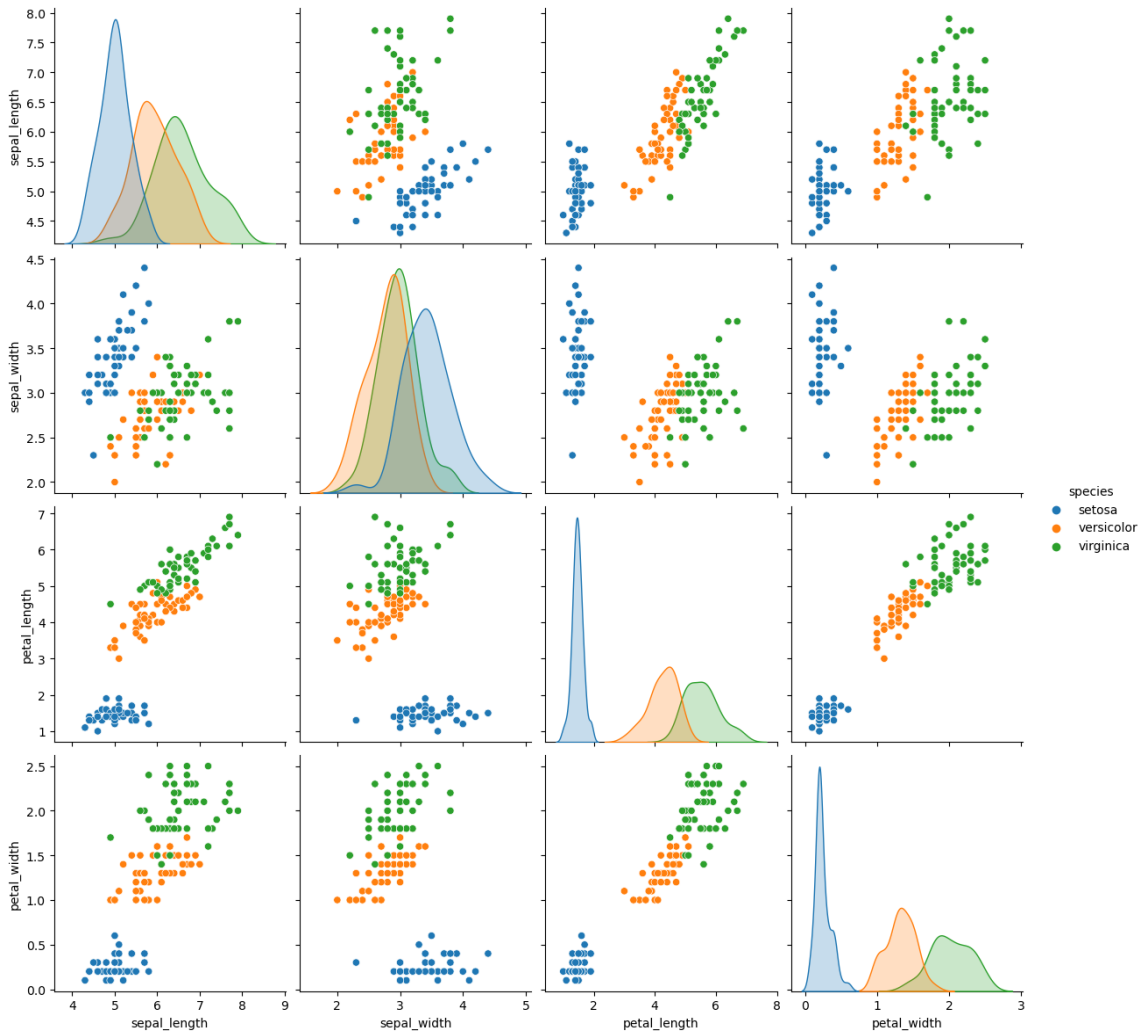
In [56]:
```
#14. Write a python program to representation of data using Pie chart.
from matplotlib import pyplot as plt
Players=["Rohit","Virat","Shikhar","Yuvraj"]
Runs=[45,30,15,10]
plt.pie(Runs,explode=(0.1,0,0,0),labels=Players,shadow=True,
        autopct='%1.1f%%',startangle=90)
plt.show()
```



In [57]:
```
#15.Write python program to representation of data using Pair plot/chart.
import seaborn as sns
df=sns.load_dataset("iris")
sns.pairplot(df,hue="species",height=3)
```

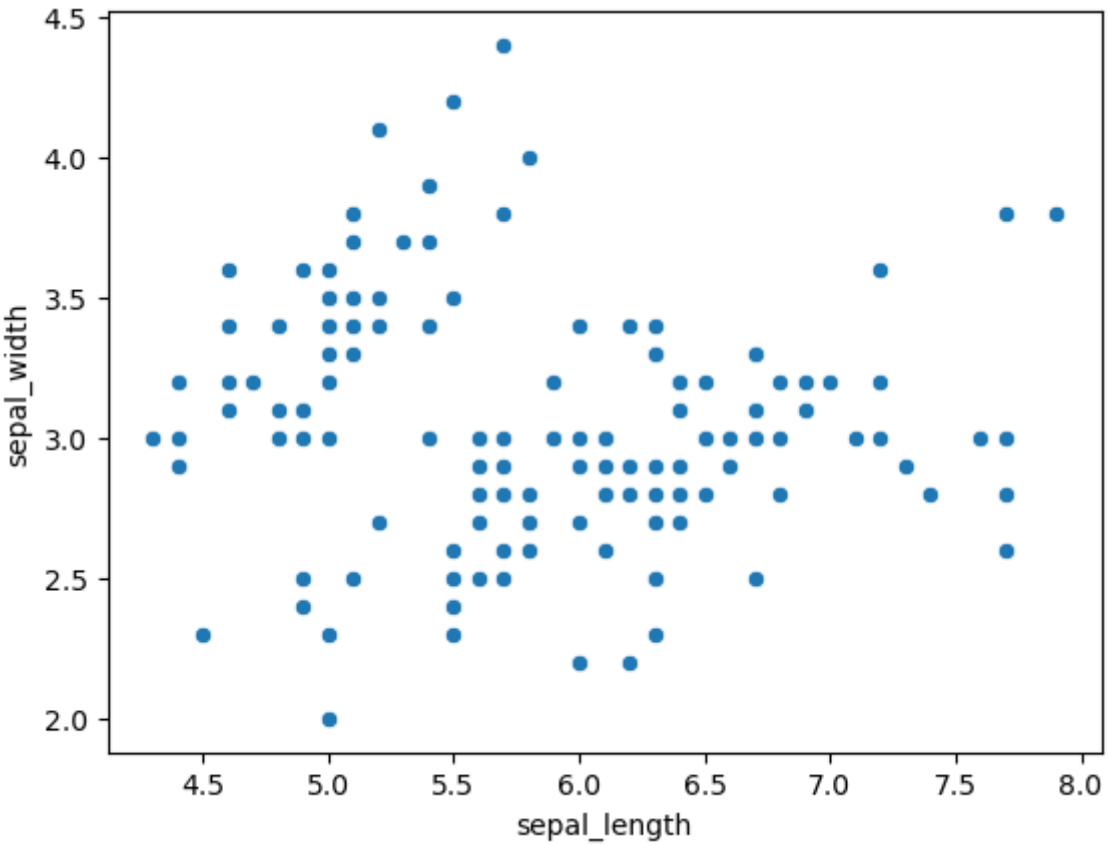
```
C:\Users\PURVA\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWa
rning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[57]: <seaborn.axisgrid.PairGrid at 0x1f0b29c2610>



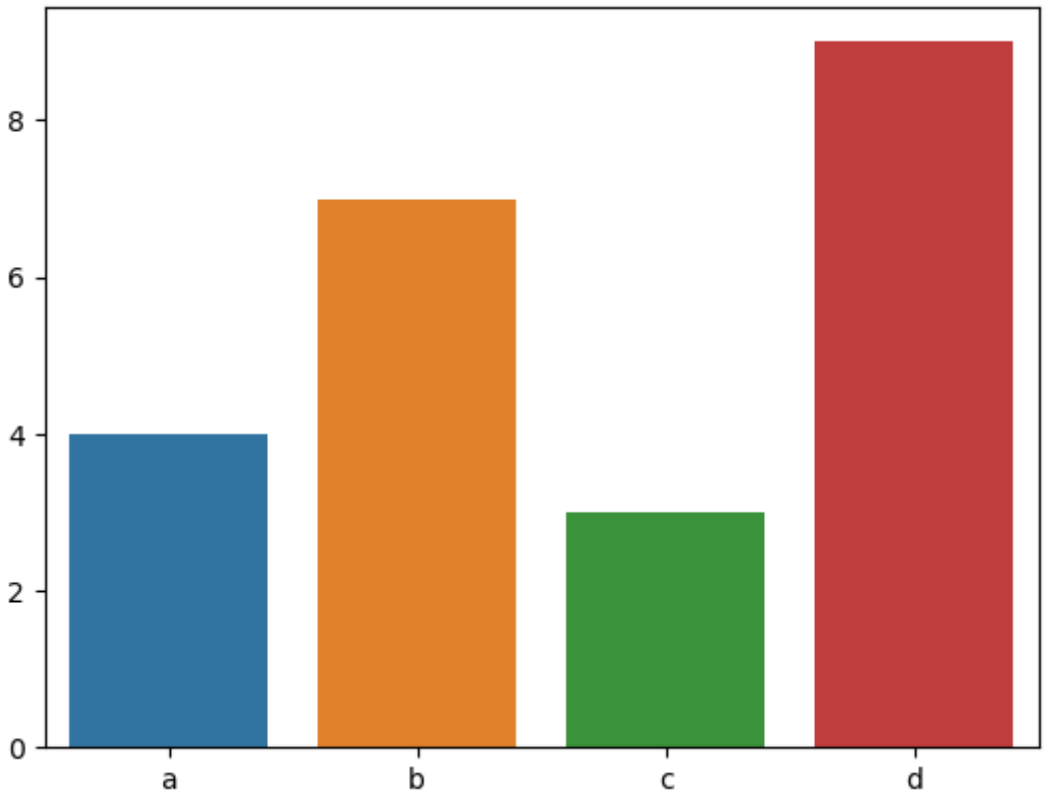
In [58]:
```
#16. Write a python program for analysis of data through Scatter plot.
import seaborn as sns
data=sns.load_dataset("iris")
sns.scatterplot(x="sepal_length",y="sepal_width",data=data)
```

Out[58]:
```
<Axes: xlabel='sepal_length', ylabel='sepal_width'>
```



In [59]:
```
#17. Write a python program to representation of data using Bar plot.
import seaborn as sns
a=["a","b","c","d"]
b=[4,7,3,9]
sns.barplot(x=a,y=b)
```

Out[59]: <Axes: >

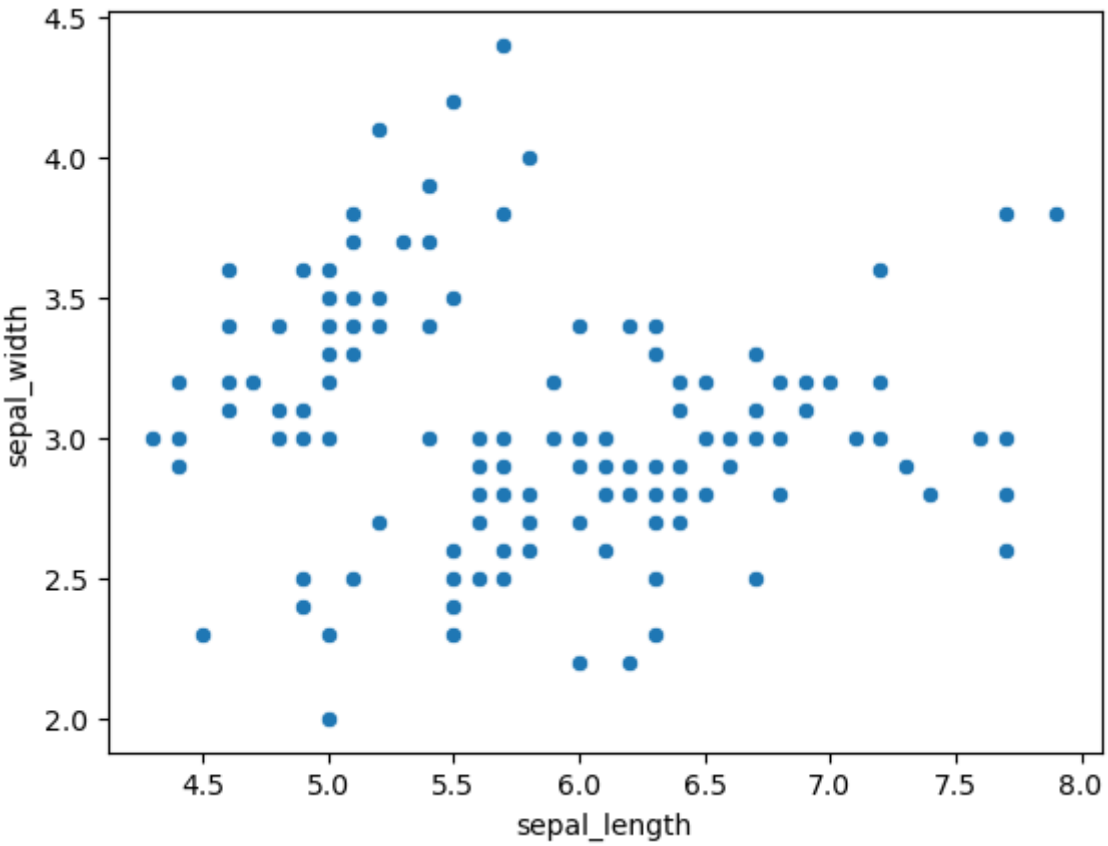


In [60]:
```
#18. Write a python program to implement Univariate analysis.
import pandas as pd
data=pd.read_excel(r"C:\Users\PURVA\OneDrive\Documents\Book1.xlsx")
print(data)
Marks=data["marks"]
x=Marks.mean()
y=Marks.median()
z=Marks.std()
print("Mean=",x)
print("Median=",y)
print("Std=",z)
```

```
   sr no name  marks
0      1    A     98
1      2    B     95
2      3    C     98
3      4    D     97
4      5    E    100
5      6    F     96
Mean= 97.33333333333333
Median= 97.5
Std= 1.7511900715418263
```

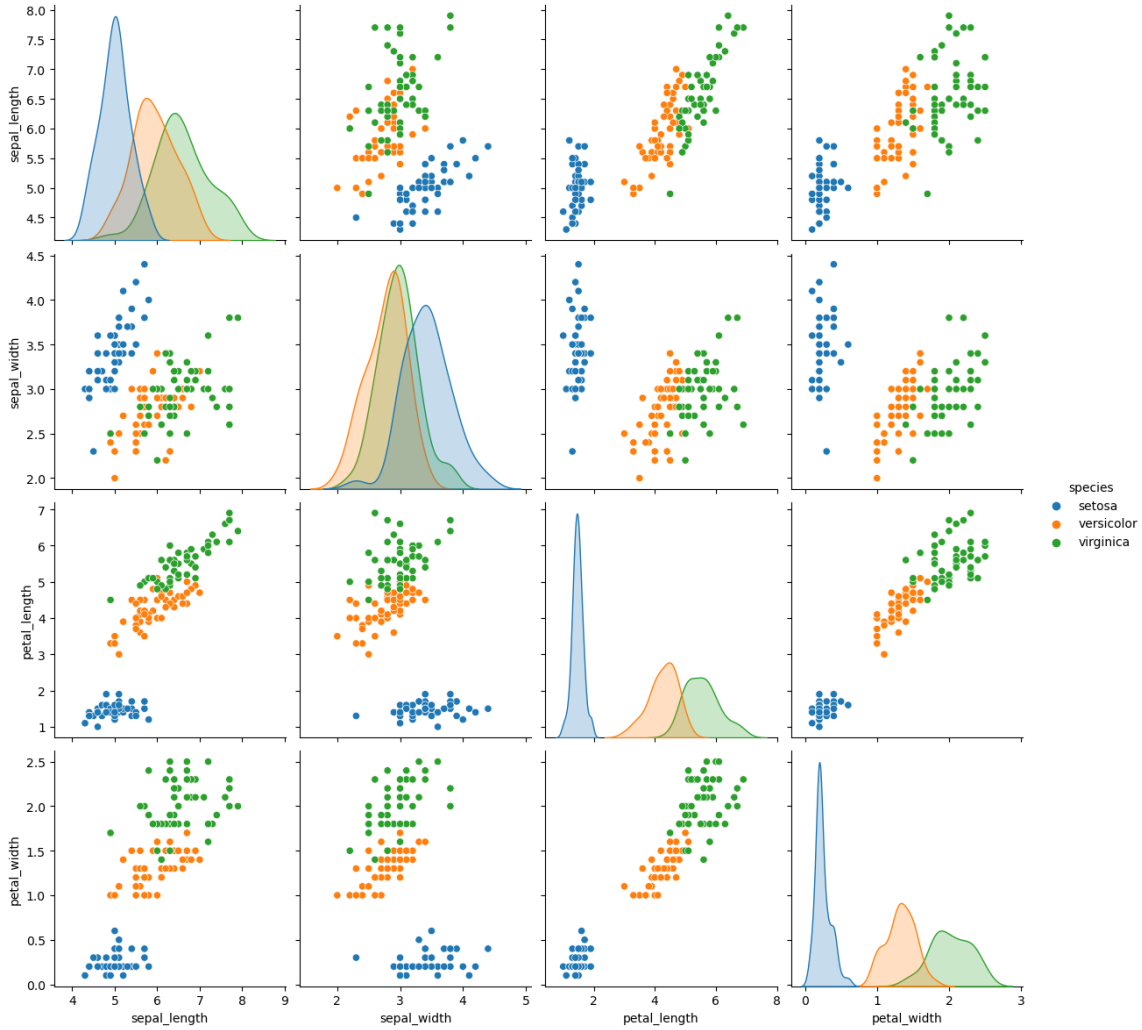
In [61]:
```
# 19 bivariate analysis
import seaborn as sns
data=sns.load_dataset("iris")
sns.scatterplot(x="sepal_length",y="sepal_width",data=data)
```

Out[61]:
```
<Axes: xlabel='sepal_length', ylabel='sepal_width'>
```



In [62]:
```
# 20 multivariate analysis
import seaborn as sns
import matplotlib.pyplot as plt
df=sns.load_dataset("iris")
sns.pairplot(df,hue="species",height=3)
plt.show()
```

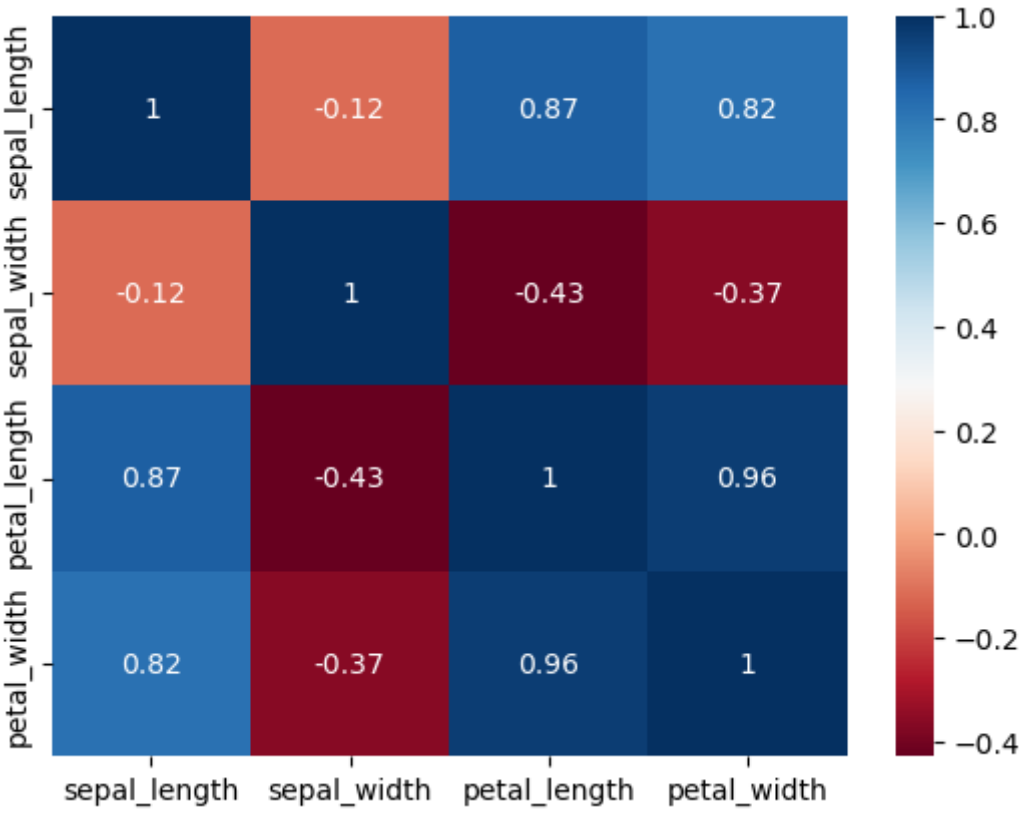
```
C:\Users\PURVA\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWa
rning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [63]:
```
#21 correlation matrix and correlation graph
import seaborn as sns
df=sns.load_dataset("iris")
df.head()
cor_mat=df.corr(numeric_only=True)
print(cor_mat)
sns.heatmap(cor_mat,cmap='RdBu',annot=True)
```

```
              sepal_length  sepal_width  petal_length  petal_width
sepal_length      1.000000    -0.117570      0.871754     0.817941
sepal_width      -0.117570     1.000000     -0.428440    -0.366126
petal_length      0.871754    -0.428440      1.000000     0.962865
petal_width       0.817941    -0.366126      0.962865     1.000000
```

Out[63]: <Axes: >



In [67]:
```
# 22 . cross tabulation using crosstab() function.
import pandas as pd
df=pd.read_excel(r"C:\Users\PURVA\OneDrive\Documents\Book2.xlsx")
df
```

Out[67]:

	Subject	Age	Gender
0	GW	young	F
1	JA	middle	F
2	TJ	young	M
3	JMA	young	M
4	AJ	middle	F
5	JQA	old	F
6	AJ	old	F
7	MVB	young	M
8	WHH	old	F
9	JT	young	F
10	JKP	middle	M

In [68]:
```
pd.crosstab(index=df["Age"],columns=df["Gender"])
```

Out[68]:

Gender	F	M
Age		
middle	2	1
old	3	0
young	2	3

In [69]:
```
# 23  grouping data using group by.
import pandas as pd
data = {
  'co2': [95, 90, 99, 104, 105, 94, 99, 104],
  'model': ['Citigo', 'Fabia', 'Fiesta', 'Rapid',
            'Focus', 'Mondeo', 'Octavia', 'B-Max'],
  'car': ['Skoda', 'Skoda', 'Ford', 'Skoda', 'Ford',
          'Ford', 'Skoda', 'Ford']
}
df = pd.DataFrame(data)
df.groupby(["car"],group_keys=True).apply(lambda x:x)
```

Out[69]:

		co2	model	car
car				
Ford	**2**	99	Fiesta	Ford
	4	105	Focus	Ford
	5	94	Mondeo	Ford
	7	104	B-Max	Ford
Skoda	**0**	95	Citigo	Skoda
	1	90	Fabia	Skoda
	3	104	Rapid	Skoda
	6	99	Octavia	Skoda

In [70]:
```
#24  Implementation measures of central tendency (mean, median and mode)
import pandas as pd
data=pd.read_excel(r"C:\Users\PURVA\OneDrive\Documents\Book1.xlsx")
print(data)
Marks=data["marks"]
x=Marks.mean()
y=Marks.median()
z=Marks.mode()
print("Mean=",x)
print("Median=",y)
print("Mode=",z)
```

```
   sr no name  marks
0      1    A     98
1      2    B     95
2      3    C     98
3      4    D     97
4      5    E    100
5      6    F     96
Mean= 97.33333333333333
Median= 97.5
Mode= 0    98
Name: marks, dtype: int64
```

In [71]:
```
#25 Implementation of measures of dispersion (range, variance)
import numpy as np
def _range(series):
    return series.max() - series.min()
speed = np.array([32,111,138,28,59,77,97])
x = np.var(speed)
r= _range(speed)
print("varience=",x)
print("range=",r)
```

```
varience= 1432.2448979591834
range= 110
```

In [1]:
```
#26 statistical characteristics of dataset
import seaborn as sns
dt=sns.load_dataset("iris")
print(dt)
dt.describe()
```

```
     sepal_length  sepal_width  petal_length  petal_width    species
0             5.1          3.5           1.4          0.2     setosa
1             4.9          3.0           1.4          0.2     setosa
2             4.7          3.2           1.3          0.2     setosa
3             4.6          3.1           1.5          0.2     setosa
4             5.0          3.6           1.4          0.2     setosa
..            ...          ...           ...          ...        ...
145           6.7          3.0           5.2          2.3  virginica
146           6.3          2.5           5.0          1.9  virginica
147           6.5          3.0           5.2          2.0  virginica
148           6.2          3.4           5.4          2.3  virginica
149           5.9          3.0           5.1          1.8  virginica

[150 rows x 5 columns]
```

Out[1]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [1]:
```
#27 simple regression
from scipy import stats
x=[6,7,8,7,2,17]
y=[99,86,87,88,111,86]

slope,intercept,r,p,std_err=stats.linregress(x,y)
def myfunc(x):
    return slope * x + intercept

speed=myfunc(10)
print(speed)
```

```
89.81411126187245
```

In [2]:
```
#29 t test using python
from scipy import stats

# Example data for two groups (replace with your own data)
group1 = [25, 30, 28, 35, 34]
group2 = [27, 32, 30, 31, 29]

# Perform independent samples T-test
t_statistic, p_value = stats.ttest_ind(group1, group2)

# Display results
print(f"T-statistic: {t_statistic}")
print(f"P-value: {p_value}")

# Check for statistical significance (alpha value usually 0.05)
alpha = 0.05
if p_value < alpha:
    print("Reject null hypothesis: There is a significant difference betwee
else:
    print("Cannot reject null hypothesis: No significant difference between
```

```
T-statistic: 0.29277002188455886
P-value: 0.7771479736234914
Cannot reject null hypothesis: No significant difference between the group
s.
```

In [3]:
```
#30 weighted average
import numpy as np
data=np.array([10,15,20,25,30])
weights=np.array([0.1,0.2,0.3,0.2,0.2])
weighted_mean=np.average(data,weights=weights)
print("weighted mean=",weighted_mean)
```

```
weighted mean= 21.0
```

In [ ]: