control statements

In [1]:
```
n=input(input("enetr a number:"))
print("n")
```

```
enetr a number:5
56
n
```

In [2]:
```
n=int(input("enetr a number:"))#if statement
if n>0:
    print("given number is positive")
```

```
enetr a number:4
given number is positive
```

In [3]:
```
n=int(input("enter a number:"))#if else condition
if n>0:
    print("given number is positive")
else:
    print("given number is negative")
```

```
enter a number:5
given number is positive
```

In [4]:
```
n=int(input("enter a number:"))#if el ifelse condition
if n>0:
    print("given number is positive")
elif n<0:
    print("given number is negative")
else:
    print("given number is zero")
```

```
enter a number:6
given number is positive
```

In [5]:
```
for i in range (0,5,1):
    print(i)
```

```
0
1
2
3
4
```

In [6]:
```
n=int(input("enter a number:"))
for i in range(0,(n+1)):
    print(i)
```

```
enter a number:7
0
1
2
3
4
5
6
7
```

In [7]:
```
n=int(input("enter a number:"))# while comdition
i=0
while(i<=n):
    print(i)
    i+=1
```

```
enter a number:7
0
1
2
3
4
5
6
7
```

In [8]:
```
for i in range(5):#pass
    if i==3:
        pass
    print(i)
```

```
0
1
2
3
4
```

In [9]:
```
for i in range(5):#continue
    if i==3:
        continue
    print(i)
```

```
0
1
2
4
```

In [10]:
```
for i in range(5):#break
    if i==3:
        break
    print(i)
```

```
0
1
2
```

In [11]:
```
l=[]
print(l)
```

```
[]
```

In [12]:
```
l=[1,2,3,5,4]
print(l)
```

```
[1, 2, 3, 5, 4]
```

In [13]:
```
l.append(6)
print(l)
```

```
[1, 2, 3, 5, 4, 6]
```

In [14]:
```
l.pop()
print(l)
```

```
[1, 2, 3, 5, 4]
```

In [15]:
```
l.sort()
print(l)
```

```
[1, 2, 3, 4, 5]
```

In [16]:
```
l.count(5)
```

Out[16]:
```
1
```

numpy library

In [17]:
```
import numpy as np
```

In [18]:
```
a=np.array([1,2,3,4,5])
a
```

Out[18]:
```
array([1, 2, 3, 4, 5])
```

In [19]:
```
a=np.arange(10)
a
```

Out[19]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [21]:
```
l=range(100)
%timeit [i**2 for i in l]
```

```
14.9 µs ± 1.23 µs per loop (mean ± std. dev. of 7 runs, 100,000 loops each)
```

In [ ]:
```

```

In [22]:
```
l=np.arange(100)
%timeit a**2
```

```
1.26 µs ± 66.5 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)
```

creating arrys

In [23]:
```
a=np.array([1,2,3,4,5]) #one dimension
```

In [24]:
```
a.ndim #says which dimension is it
```

Out[24]:
```
1
```

In [25]:
```
b=np.array([[1,3,2],[4,5,6]])
```

In [26]:
```
b.ndim
```

Out[26]:
```
2
```

In [30]:
```
c=np.array([[[1,2,3],[4,5,6],[7,8,9]]])
c
```

Out[30]:
```
array([[[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]]])
```

In [31]:
```
c.ndim
```

Out[31]:
```
3
```

In [32]:
```
a.shape #dispalys no.of columns
```

Out[32]:
```
(5,)
```

In [33]:
```
b.shape #dispalys no.of rows and columns
```

Out[33]:
```
(2, 3)
```

In [34]:
```
c.shape
```

Out[34]:
```
(1, 3, 3)
```

funtions for creating array

In [35]:
```
a=np.arange(1,10,2)
a
```

Out[35]:
```
array([1, 3, 5, 7, 9])
```

In [36]:
```
a=np.linspace(0,1,6)
a
```

Out[36]:
```
array([0. , 0.2, 0.4, 0.6, 0.8, 1. ])
```

In [37]:
```
a=np.ones((3,3))
a
```

Out[37]:
```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

In [38]:
```
a=np.zeros((3,3))
a
```

Out[38]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

In [39]:
```
a=np.eye(3)
a
```

Out[39]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

In [40]:
```
a=np.eye(3,2)
a
```

Out[40]:
```
array([[1., 0.],
       [0., 1.],
       [0., 0.]])
```

In [41]:
```
a=np.diag([1,2,3,4])
a
```

Out[41]:
```
array([[1, 0, 0, 0],
       [0, 2, 0, 0],
       [0, 0, 3, 0],
       [0, 0, 0, 4]])
```

In [42]:
```
np.diag(a)
```

Out[42]:
```
array([1, 2, 3, 4])
```

basic datatypes

In [43]:
```
a=np.arange(10)
a.dtype
```

Out[43]:
```
dtype('int32')
```

In [44]:
```
a=np.arange(10,dtype="float")
a.dtype
```

Out[44]:
```
dtype('float64')
```

In [45]:
```
a=np.zeros((3,3))
print(a)
a.dtype
```

Out[45]:
```
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
dtype('float64')
```

other datatypes

In [46]:
```
a=np.array([1+2j,2+4j])
a.dtype
```

Out[46]:
```
dtype('complex128')
```

In [47]:
```
a=np.array([True, False])
a.dtype
```

Out[47]:
```
dtype('bool')
```

In [48]:
```
a=np.array(["Rams", "sri"])
a.dtype                              #unicode string code
```

Out[48]:
```
dtype('<U4')
```

In [49]:
```
a=np.array(["RAMYASRI", "False"])   #Maximum length of the given string
a.dtype
```

Out[49]:
```
dtype('<U8')
```

indexing and slicing

In [50]:
```
a=np.arange(10)            #range from 0
a[5]                  #index start from 0
```

Out[50]:
```
5
```

In [51]:
```
a= np.diag([1,2,3])
a[2,2]
```

Out[51]:
```
3
```

In [52]:
```
a[2,1]=5
a
```

Out[52]:
```
array([[1, 0, 0],
       [0, 2, 0],
       [0, 5, 3]])
```

slicing

In [53]:
```
a=np.arange(10)
a
```

Out[53]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [54]:
```
a[1:8:2]
```

Out[54]:
```
array([1, 3, 5, 7])
```

In [55]:
```
a=np.arange(10)
a[5:]=10
a
```

Out[55]:
```
array([ 0,  1,  2,  3,  4, 10, 10, 10, 10, 10])
```

In [56]:
```
a=np.arange(10)
b=a[::2]
print(a)
print(b)
```

```
[0 1 2 3 4 5 6 7 8 9]
[0 2 4 6 8]
```

In [57]:
```
np.shares_memory(a,b)
```

Out[57]:
```
True
```

In [58]:
```
b[0]=10
print(a)          #viewing
print(b)
```

```
[10  1  2  3  4  5  6  7  8  9]
[10  2  4  6  8]
```

In [59]:
```
a=np.arange(10)
b=a[::2].copy()           #copying
print(a)
print(b)
```

```
[0 1 2 3 4 5 6 7 8 9]
[0 2 4 6 8]
```

In [60]:
```
np.shares_memory(a,b)
```

Out[60]:
```
False
```

In [61]:
```
b[0]=10
print(a)
print(b)
```

```
[0 1 2 3 4 5 6 7 8 9]
[10  2  4  6  8]
```

random generator

In [62]:
```
a=np.random.randint(0,20,15)
a
```

Out[62]:
```
array([16,  3,  2, 15,  1, 12, 10,  6,  9, 18,  3,  1, 11,  6,  8])
```

basic operations

In [63]:
```
import numpy as np
```

In [64]:
```
a=np.array([1,2,3,4])
a+1
```

Out[64]:
```
array([2, 3, 4, 5])
```

In [65]:
```
a*0
```

Out[65]:
```
array([0, 0, 0, 0])
```

In [66]:
```
a-1
```

Out[66]:
```
array([0, 1, 2, 3])
```

In [67]:
```
a**2
```

Out[67]:
```
array([ 1,  4,  9, 16])
```

In [68]:
```
a/2
```

Out[68]:
```
array([0.5, 1. , 1.5, 2. ])
```

In [69]:
```
b=np.ones(4)+1
print("a=",a)
print("b=",b)
```

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

In [70]:
```
print(a*b)
```

```
[2. 4. 6. 8.]
```

In [71]:
```
print(a+b)
```

```
[3. 4. 5. 6.]
```

In [72]:
```
c=np.diag([1,2,3,4])
print(c)
```

```
[[1 0 0 0]
 [0 2 0 0]
 [0 0 3 0]
 [0 0 0 4]]
```

In [73]:
```
print(c*c)
```

```
[[ 1  0  0  0]
 [ 0  4  0  0]
 [ 0  0  9  0]
 [ 0  0  0 16]]
```

In [74]:
```
print(c*c)
print("**************")
print(c)
```

```
[[ 1  0  0  0]
 [ 0  4  0  0]
 [ 0  0  9  0]
 [ 0  0  0 16]]
*************
[[1 0 0 0]
 [0 2 0 0]
 [0 0 3 0]
 [0 0 0 4]]
```

In [75]:
```
a=np.array([1,2,3,4])
b=np.array([3,2,3,6])
print(a==b)
```

```
[False  True  True False]
```

In [76]:
```
a=np.array([1,2,3,4])
b=np.array([3,2,3,6])
print(a>b)
```

```
[False False False False]
```

In [77]:
```
a=np.array([1,2,3,4])
b=np.array([5,6,7,8])
c=np.array([3,4,5,6])
print(np.array_equal(a,b))
```

```
False
```

In [78]:
```
print(np.array_equal(a,c))
```

```
False
```

In [79]:
```
a=np.array([1,0,1,0],dtype="bool")#operational
b=np.array([1,1,0,0],dtype="bool")
print(np.logical_or(a,b))
```

```
[ True  True  True False]
```

In [80]:
```
a=np.array([1,0,1,0],dtype="bool")
b=np.array([1,1,0,0],dtype="bool")
print(np.logical_and(a,b))
```

```
[ True False False False]
```

In [81]:
```
a=np.arange(5)# trignometric
print(a)
```

```
[0 1 2 3 4]
```

In [82]:
```
print(np.sin(a))
```

```
[ 0.          0.84147098  0.90929743  0.14112001 -0.7568025 ]
```

In [83]:
```
print(np.log(a))
```

```
[      -inf 0.         0.69314718 1.09861229 1.38629436]
C:\Users\Ramya\AppData\Local\Temp\ipykernel_17584\897204776.py:1: RuntimeWarning: div
ide by zero encountered in log
  print(np.log(a))
```

In [84]:
```
print(np.exp(a))
```

```
[ 1.          2.71828183  7.3890561  20.08553692 54.59815003]
```

In [85]:
```
#basic reductions
x=np.array([1,2,3,4])
print(np.sum(x))
```

```
10
```

In [86]:
```
a=np.array([[1,2],[3,4]])
print(a)
```

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

In [87]:
```
a.sum(axis=0)
```

Out[87]:
```
array([4, 6])
```

statistics

In [88]:
```
import pandas as pd
import numpy as np
r"C:\Users\Ramya\OneDrive\Documents\dataset\data.csv"
```

In [89]:
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\dataset\data.csv")
```

In [90]:
```
df
```

Out[90]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	B.Yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96
5	K.Sri Niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	P.Durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	P.Akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	P.Neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

In [91]:
```
df['MARKS'].min()
```

Out[91]:
```
85
```

In [92]:
```
df['MARKS'].mean()
```

Out[92]:
```
91.9
```

In [93]:
```
df['MARKS'].max()
```

Out[93]:
```
97
```

In [94]:
```
df['MARKS'].describe()
```

Out[94]:
```
count    10.000000
mean     91.900000
std       3.725289
min      85.000000
25%      90.000000
50%      91.500000
75%      94.750000
max      97.000000
Name: MARKS, dtype: float64
```

In [95]:
```
df[df.MARKS==df.MARKS.max()]
```

Out[95]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97

In [96]:
```
df.NAME[df.MARKS==df.MARKS.max()]
```

Out[96]:
```
1    P.Devi
Name: NAME, dtype: object
```

In [97]:
```
df.count()
```

Out[97]:
```
NAME                    10
CLASS                   10
SECTION                 10
ROLLNO                  10
                EMAIL    10
PHNO                    10
MARKS                   10
dtype: int64
```

In [98]:
```
df["MARKS"].nunique()
```

Out[98]:
```
8
```

In [99]:
```
df["MARKS"].value_counts()
```

Out[99]:
```
MARKS
90    3
95    1
97    1
85    1
96    1
94    1
89    1
93    1
Name: count, dtype: int64
```

string cleaning

In [100…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\dataset\data.csv")
df
```

Out[100]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	B.Yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96
5	K.Sri Niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	P.Durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	P.Akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	P.Neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

In [101…
```
import pandas as pd
import numpy as np
```

In [102…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\dataset\data.csv")
df
```

Out[102]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	B.Yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96
5	K.Sri Niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	P.Durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	P.Akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	P.Neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

In [103…
```
df["NAME"]=df["NAME"].str.lower()
df
```

Out[103]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	b.yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	p.devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	j.ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	k.prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	k.v. niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96
5	k.sri niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	y.kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	p.durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	p.akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	p.neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

concate Data Frames

In [104…

```
us_weather=pd.DataFrame({
    "city":["new york","chicago","orlando"],
    "Temperature":[32,45,35]
})
us_weather
```

Out[104]:

	city	Temperature
0	new york	32
1	chicago	45
2	orlando	35

In [105…

```
india_weather=pd.DataFrame({
    "city":["mumbai","ap","banglore"],
    "Temperature":[33,44,35]
})
india_weather
```

Out[105]:

	city	Temperature
0	mumbai	33
1	ap	44
2	banglore	35

In [106…

```
df=pd.concat([india_weather,us_weather])
df
```

Out[106]:

	city	Temperature
0	mumbai	33
1	ap	44
2	banglore	35
0	new york	32
1	chicago	45
2	orlando	35

In [107…

```
df=pd.concat([india_weather,us_weather],axis=1)
df
```

Out[107]:

	city	Temperature	city	Temperature
0	mumbai	33	new york	32
1	ap	44	chicago	45
2	banglore	35	orlando	35

merge data frames

In [108…

```
temp_df=pd.DataFrame({
    "city":["hyderabad","bangalore","vishakapatnam","chennai"],
    "Temperature":[32,45,30,35]
})
temp_df
```

Out[108]:

	city	Temperature
0	hyderabad	32
1	bangalore	45
2	vishakapatnam	30
3	chennai	35

In [109…

```
humidity_df=pd.DataFrame({
    "city":["hyderabad","mumbai","ap","chennai"],
    "humidity":[40,30,25,26]
})
humidity_df
```

Out[109]:

	city	humidity
0	hyderabad	40
1	mumbai	30
2	ap	25
3	chennai	26

In [110…

```
df=pd.merge(temp_df,humidity_df,on="city")
df
```

Out[110]:

	city	Temperature	humidity
0	hyderabad	32	40
1	chennai	35	26

In [111…

```
df=pd.merge(temp_df,humidity_df,on="city",how="outer")
df
```

Out[111]:

	city	Temperature	humidity
0	hyderabad	32.0	40.0
1	bangalore	45.0	NaN
2	vishakapatnam	30.0	NaN
3	chennai	35.0	26.0
4	mumbai	NaN	30.0
5	ap	NaN	25.0

In [112…

```
df.isnull() #to find the missing values
```

Out[112]:

	city	Temperature	humidity
0	False	False	False
1	False	False	True
2	False	False	True
3	False	False	False
4	False	True	False
5	False	True	False

In [113…

```
df.isnull().sum()
```

Out[113]:

```
city           0
Temperature    2
humidity       2
dtype: int64
```

In [114…
```
df['Temperature']=df['Temperature'].fillna(df.Temperature.min())
```

In [115…
```
df.isnull()
```

Out[115]:

	city	Temperature	humidity
0	False	False	False
1	False	False	True
2	False	False	True
3	False	False	False
4	False	False	False
5	False	False	False

In [118…
```
df['humidity']=df['humidity'].fillna(df.humidity.min())
```

In [119…
```
df.isnull()
```

Out[119]:

	city	Temperature	humidity
0	False	False	False
1	False	False	False
2	False	False	False
3	False	False	False
4	False	False	False
5	False	False	False

In [120…
```
df
```

Out[120]:

	city	Temperature	humidity
0	hyderabad	32.0	40.0
1	bangalore	45.0	25.0
2	vishakapatnam	30.0	25.0
3	chennai	35.0	26.0
4	mumbai	30.0	30.0
5	ap	30.0	25.0

Pandas library

In [161…
```
import pandas as pd
import numpy as np
pd.__version__ # know the pandas version
```

Out[161]:
```
'2.0.3'
```

In [162…
```
s=pd.Series([1,2,3,]) # one column
print(s)
```

```
0    1
1    2
2    3
dtype: int64
```

In [164…
```
df=pd.DataFrame({
    "name": ["niha","prash"],
    "age":[12,34]})
df
```

Out[164]:

	name	age
0	niha	12
1	prash	34

In [165…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\dataset\data.csv")
df
```

Out[165]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	B.Yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96
5	K.Sri Niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	P.Durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	P.Akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	P.Neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

In [167…
```
df.to_csv('data.csv',index=False)
```

exploring data

In [168…
```
df.head()
```

Out[168]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
0	B.Yamuna	AIR	A	37	yamunalovely9@gmail.com	8897859569	95
1	P.Devi	AIR	A	58	gundrasivesaidurga@gmail.com	9347791232	97
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96

In [169…
```
df.tail()
```

Out[169]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
5	K.Sri Niharika	AIR	A	9	karresriniharika@gmail.com	6300408089	90
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94
7	P.Durga	AIR	A	24	durgarani@gmail.com	8466845651	89
8	P.Akhila	AIR	A	29	bottaakhila@gmail.com	8397941799	90
9	P.Neelima	AIR	A	36	pinapalaramesh@gmail.com	7386803023	93

In [170…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 7 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   NAME                  10 non-null     object
 1   CLASS                 10 non-null     object
 2   SECTION               10 non-null     object
 3   ROLLNO                10 non-null     int64 
 4                  EMAIL  10 non-null     object
 5   PHNO                  10 non-null     int64 
 6   MARKS                 10 non-null     int64 
dtypes: int64(3), object(4)
memory usage: 692.0+ bytes
```

In [172…
```
df['NAME'][df['ROLLNO']=='J.Ramya']
```

Out[172]:
```
Series([], Name: NAME, dtype: object)
```

In [173…
```
df.shape
```

Out[173]:
```
(10, 7)
```

In [174…
```
df[2:5]
```

Out[174]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
2	J.Ramya	AIR	A	52	jivvireddyramyasri@gmailcom	9676976652	90
3	K.Prasanthi	AIR	A	41	kesanakurthiprasanthi@gmail.com	7382934586	85
4	K.V. Niharika	AIR	A	54	karreveeraniharika@gmail.com	9182532802	96

In [175…
```
df.columns
```

Out[175]:
```
Index(['NAME', 'CLASS', 'SECTION', 'ROLLNO', '                EMAIL', 'PHNO',
       'MARKS'],
      dtype='object')
```

In [176…
```
df.NAME
```

Out[176]:
```
0          B.Yamuna
1            P.Devi
2          J.Ramya
3        K.Prasanthi
4      K.V. Niharika
5     K.Sri Niharika
6           Y.Kaveri
7            P.Durga
8           P.Akhila
9         P.Neelima
Name: NAME, dtype: object
```

In [177…
```
df.ROLLNO
```

Out[177]:
```
0    37
1    58
2    52
3    41
4    54
5     9
6     3
7    24
8    29
9    36
Name: ROLLNO, dtype: int64
```

In [178…
```
df[['NAME','MARKS']]
```

Out[178]:

	NAME	MARKS
0	B.Yamuna	95
1	P.Devi	97
2	J.Ramya	90
3	K.Prasanthi	85
4	K.V. Niharika	96
5	K.Sri Niharika	90
6	Y.Kaveri	94
7	P.Durga	89
8	P.Akhila	90
9	P.Neelima	93

In [179…
```
df.MARKS
```

Out[179]:
```
0    95
1    97
2    90
3    85
4    96
5    90
6    94
7    89
8    90
9    93
Name: MARKS, dtype: int64
```

In [180…
```
df['NAME'].max()
```

Out[180]:
```
'Y.Kaveri'
```

In [182…
```
df['NAME'].min()
```

Out[182]:
```
'B.Yamuna'
```

In [183…
```
df['MARKS'].mean()
```

Out[183]:
```
91.9
```

In [184…
```
df['MARKS'].mode()
```

Out[184]:
```
0    90
Name: MARKS, dtype: int64
```

In [185…
```
df['NAME'].describe()
```

Out[185]:
```
count           10
unique          10
top       B.Yamuna
freq             1
Name: NAME, dtype: object
```

In [186…
```
df[df.NAME==df.NAME.max()]
```

Out[186]:

	NAME	CLASS	SECTION	ROLLNO	EMAIL	PHNO	MARKS
6	Y.Kaveri	AIR	A	3	yendamurikaveri@gmail.com	7386015549	94

In [188…
```
df.MARKS[df.NAME==df.NAME.max()]
```

Out[188]:
```
6    94
Name: MARKS, dtype: int64
```

correlation & covariance

In [122…
```
import pandas as pd
```

In [123…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\study_marks.csv")
df
```

Out[123]:

	Study_Hours	Marks
0	2.9	27.0
1	5.8	61.0
2	4.7	45.0
3	4.0	38.0
4	1.8	19.0
5	1.8	8.0
6	1.3	4.0
7	5.3	50.0
8	4.0	35.0
9	4.5	47.0

In [124…
```
df.cov()
```

Out[124]:

	Study_Hours	Marks
Study_Hours	2.481000	28.906667
Marks	28.906667	348.711111

In [125…
```
df.corr()
```

Out[125]:

	Study_Hours	Marks
Study_Hours	1.00000	0.98277
Marks	0.98277	1.00000

In [126…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\temp_icecream_sales.csv")
df
```

Out[126]:

	Temperature	IceCream_Sales
0	20	200
1	22	220
2	24	250
3	26	270
4	28	300
5	30	320
6	32	350
7	34	370
8	36	400
9	38	420

In [127…
```
df.corr()
```

Out[127]:

	Temperature	IceCream_Sales
Temperature	1.000000	0.999406
IceCream_Sales	0.999406	1.000000

In [128…
```
df.cov()
```

Out[128]:

	Temperature	IceCream_Sales
Temperature	36.666667	455.555556
IceCream_Sales	455.555556	5666.666667

In [129…
```
df=pd.read_csv(r"C:\Users\Ramya\OneDrive\Documents\TEMP.csv")
df
```

Out[129]:

	Temperature	Sunny
0	34	50
1	24	45
2	34	36
3	12	56
4	67	78
5	78	89
6	89	78

In [130…
```
df.corr()
```

Out[130]:

	Temperature	Sunny
Temperature	1.000000	0.830515
Sunny	0.830515	1.000000

In [131…
```
df.cov()
```

Out[131]:

	Temperature	Sunny
Temperature	867.571429	487.761905
Sunny	487.761905	397.571429

In [132…
```
data={
    "temp":[-20,-30,-40,-60],
    "hum":[100,150,140,130]
}
df=pd.DataFrame(data)
df
```

Out[132]:

	temp	hum
0	-20	100
1	-30	150
2	-40	140
3	-60	130

In [133…
```
df.cov()
```

Out[133]:

	temp	hum
temp	291.666667	-133.333333
hum	-133.333333	466.666667

Normalization

In [134…
```
s=pd.Series([1000,2000,3000,4000])
s
```

Out[134]:
```
0    1000
1    2000
2    3000
3    4000
dtype: int64
```

In [135…
```
s.min()
```

Out[135]:
```
1000
```

In [136…
```
s.max()
```

Out[136]:
```
4000
```

In [137…
```
sd=(s-s.min())/(s.max()-s.min())
sd
```

Out[137]:
```
0    0.000000
1    0.333333
2    0.666667
3    1.000000
dtype: float64
```

In [138…
```
from sklearn.preprocessing import MinMaxScaler,StandardScaler
```

In [139…
```
import pandas as pd
```

In [140…
```
sd=MinMaxScaler().fit_transform(s.values.reshape(-1,1))
sd
```

Out[140]:
```
array([[0.        ],
       [0.33333333],
       [0.66666667],
       [1.        ]])
```

In [141…
```
data={
    "temp":[-20,-30,-40,-60],
    "hum":[100,150,140,130]
}
df=pd.DataFrame(data)
df
```

Out[141]:

	temp	hum
0	-20	100
1	-30	150
2	-40	140
3	-60	130

In [142…
```
norm_data=MinMaxScaler().fit_transform(df)    ####normalization
norm_data
```

Out[142]:
```
array([[1.  , 0.  ],
       [0.75, 1.  ],
       [0.5 , 0.8 ],
       [0.  , 0.6 ]])
```

In [143…
```
norm_df=pd.DataFrame(norm_data,columns=df.columns)
norm_df
```

Out[143]:

	temp	hum
0	1.00	0.0
1	0.75	1.0
2	0.50	0.8
3	0.00	0.6

Standardization

In [145…
```
s=pd.Series([1000,2000,3000,4000])
s
```

Out[145]:
```
0    1000
1    2000
2    3000
3    4000
dtype: int64
```

In [146…
```
s.mean()
```

Out[146]:
```
2500.0
```

In [147…
```
s.std()
```

Out[147]:
```
1290.9944487358057
```

In [154…
```
zscore=(s-s.mean())/s.std()
zscore
```

Out[154]:
```
0   -1.161895
1   -0.387298
2    0.387298
3    1.161895
dtype: float64
```

In [155…
```
zscore=StandardScaler().fit_transform(s.values.reshape(-1,1))
zscore
```

Out[155]:
```
array([[-1.34164079],
       [-0.4472136 ],
       [ 0.4472136 ],
       [ 1.34164079]])
```

In [156…
```
zscore=(s-s.mean())/s.std(ddof=0)
zscore
```

Out[156]:
```
0   -1.341641
1   -0.447214
2    0.447214
3    1.341641
dtype: float64
```

In [157…
```
data={
    "temp":[-20,-30,-40,-60],
    "hum":[100,150,140,130]
```

```
}
df=pd.DataFrame(data)
df
```

Out[157]:

	temp	hum
0	-20	100
1	-30	150
2	-40	140
3	-60	130

In [159…

```
zscore=StandardScaler().fit_transform(df)
zscoredf=pd.DataFrame(zscore, columns=df.columns)
zscore
```

Out[159]:

```
array([[ 1.18321596, -1.60356745],
       [ 0.50709255,  1.06904497],
       [-0.16903085,  0.53452248],
       [-1.52127766,  0.        ]])
```

In [160…

```
zscoredf
```

Out[160]:

	temp	hum
0	1.183216	-1.603567
1	0.507093	1.069045
2	-0.169031	0.534522
3	-1.521278	0.000000

Feature encoding

In [1]:

```
import pandas as pd
df=pd.DataFrame({
    "color": ["Red", "Green", "Blue","Green","Red"],
    "Size": ["Small", "medium","Large","medium","Small"],
    "City": ["Hyderabad","Delhi","Mumbai","Delhi","Chennai"],
    "Target": [10,20,15,25,12]
})
df
```

Out[1]:

	color	Size	City	Target
0	Red	Small	Hyderabad	10
1	Green	medium	Delhi	20
2	Blue	Large	Mumbai	15
3	Green	medium	Delhi	25
4	Red	Small	Chennai	12

In [5]:
```
from sklearn.preprocessing import LabelEncoder
```

In [6]:
```
df["Color_Label"] =LabelEncoder().fit_transform(df["color"])
df[["color","Color_Label"]]
```

Out[6]:

	color	Color_Label
0	Red	2
1	Green	1
2	Blue	0
3	Green	1
4	Red	2

In [7]:
```
one_hot=pd.get_dummies(df["color"],prefix="color") # if the value is there then it wil
one_hot
```

Out[7]:

	color_Blue	color_Green	color_Red
0	False	False	True
1	False	True	False
2	True	False	False
3	False	True	False
4	False	False	True

In [8]:
```
oh=pd.get_dummies(df,columns=["color"])
oh
```

Out[8]:

	Size	City	Target	Color_Label	color_Blue	color_Green	color_Red
0	Small	Hyderabad	10	2	False	False	True
1	medium	Delhi	20	1	False	True	False
2	Large	Mumbai	15	0	True	False	False
3	medium	Delhi	25	1	False	True	False
4	Small	Chennai	12	2	False	False	True

In [9]:
```
from sklearn.preprocessing import OrdinalEncoder # takes the values based on order
```

In [10]:
```
order=[["Small","medium","Large"]]

df["Size_ordinal"]=OrdinalEncoder(categories=order).fit_transform(df[["Size"]])
df[["Size","Size_ordinal"]]
```

Out[10]:

	Size	Size_ordinal
0	Small	0.0
1	medium	1.0
2	Large	2.0
3	medium	1.0
4	Small	0.0

In [11]:
```
te=df.groupby("City")["Target"].mean()
df["city_Target_Enc"]=df["City"].map(te)
df[["City","city_Target_Enc"]]
```

Out[11]:

	City	city_Target_Enc
0	Hyderabad	10.0
1	Delhi	22.5
2	Mumbai	15.0
3	Delhi	22.5
4	Chennai	12.0

In [12]:
```
fe=df.groupby("City")["Target"].count()
df["city_Target_Enc"]=df["City"].map(fe)  #method 1
df[["City","city_Target_Enc"]]
```

Out[12]:

	City	city_Target_Enc
0	Hyderabad	1
1	Delhi	2
2	Mumbai	1
3	Delhi	2
4	Chennai	1

In [13]:
```
f=df["City"].value_counts()
df["city_Target_Enc"]=df["City"].map(fe) # method 2
df[["City","city_Target_Enc"]]
```

Out[13]:

	City	city_Target_Enc
0	Hyderabad	1
1	Delhi	2
2	Mumbai	1
3	Delhi	2
4	Chennai	1

In [ ]:
```
!pip install category_encoders
```

In [ ]:
```
import category_encoders as ce
```

In [ ]:
```

```

In [ ]:
```

```