EXPERIMENT-1

AIM:

BASIC TOOLS/TECHNIQUES NEEDED IN ML/DL

In [8]:
```
print(7)
```

```
7
```

In [9]:
```
a=3
b="VRAJ"
c=7.07
```

In [10]:
```
print(a,b,c)
```

```
3 VRAJ 7.07
```

In [11]:
```
print(type(a),type(b),type(c))
```

```
<class 'int'> <class 'str'> <class 'float'>
```

In [12]:
```
type(a),type(b),type(c)
```

Out[12]:
```
(int, str, float)
```

In [13]:
```
a,b,c
```

Out[13]:
```
(3, 'VRAJ', 7.07)
```

In [ ]:
```

```

operator

In [14]:
```
n1=5
n2=3
print(n1/n2)
print(n1//n2)
print(n1+n2)
print(n1-n2)
print(n1*n2)
print(n1**n2)
```

```
1.6666666666666667
1
8
2
15
125
```

condition

In [15]:
```
if(n1>n2):
    print(n1,":n1 is big")
elif(n2>n1):
    print(n2,":n2 is big")
elif(n1 is n2):
    print("both are same")
```

```
5 :n1 is big
```

loop

In [16]:
```
for i in range(1,10):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
```

In [17]:
```
i=0
while(i<10):
    print(i)
    i+=1
```

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

composite data type

In [19]:
```
#mutable : list
arr=[1,3,4,'vraj',3.5,3]
print(arr)
for i in range(0,len(arr)):
    print(arr[i])
```

```
[1, 3, 4, 'vraj', 3.5, 3]
1
3
4
vraj
3.5
3
```

In [20]:
```
#immutable :tuple
tp=(1,2,5,4.5,'bcd',2)
print(tp)
for i in tp:
    print(i)
```

```
(1, 2, 5, 4.5, 'bcd', 2)
1
2
5
4.5
bcd
2
```

In [22]:
```
#key value pair :dictionary
kp={'name':'vraj','rno':21}
print(kp)
```

```
{'name': 'vraj', 'rno': 21}
```

In [23]:
```
arr[0:-1:2]
```

Out[23]:
```
[1, 4, 3.5]
```

function

In [25]:
```
def summation(a,b):
    return a+b;

summation(3,5)
```

Out[25]:
```
8
```

In [28]:
```
def Fibonacci(n):
    if n < 0:
        print("Incorrect input")

    elif n == 0:
        return 0

    elif n == 1 or n == 2:
        return 1

    else:
        return Fibonacci(n-1) + Fibonacci(n-2)

print(Fibonacci(7))
```

```
13
```

In [29]:
```
def rec(n):
    if n < 0:
        print("Incorrect input")

    elif n == 0 or n == 1:
        return 1

    else:
```

```
        return n*rec(n-1)

print(rec(7))
```

```
5040
```

pip command

Numpy

In [30]:
```
pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\vrajc\miniconda3\envs\py312\lib
\site-packages (2.0.1)Note: you may need to restart the kernel to use updated pac
kages.
```

In [32]:
```
import numpy as np
```

In [33]:
```
arr=np.array([11,21,32,41,51,61,71,81,91])
arr
```

Out[33]:
```
array([11, 21, 32, 41, 51, 61, 71, 81, 91])
```

In [37]:
```
arr[1]
```

Out[37]:
```
np.int64(21)
```

In [35]:
```
arr[-1]
```

Out[35]:
```
np.int64(91)
```

In [36]:
```
arr[1:-1]
```

Out[36]:
```
array([21, 32, 41, 51, 61, 71, 81])
```

In [38]:
```
arr[-3:]
```

Out[38]:
```
array([71, 81, 91])
```

In [39]:
```
b=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(b)
```

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

In [40]:
```
b[0,2]
```

Out[40]:
```
np.int64(3)
```

In [41]:
```
b[0:2,2]
```

Out[41]:
```
array([3, 6])
```

In [42]:
```
arr[0]
```

Out[42]:
```
np.int64(11)
```

In [43]:
```
arr[2:]
```

Out[43]:
```
array([32, 41, 51, 61, 71, 81, 91])
```

In [44]:
```
arr[:3]
```

Out[44]:
```
array([11, 21, 32])
```

In [45]:
```
arr[-1]
```

Out[45]:
```
np.int64(91)
```

In [46]:
```
arr[-3:]
```

Out[46]:
```
array([71, 81, 91])
```

In [47]:
```
arr[1:2]
```

Out[47]:
```
array([21])
```

In [48]:
```
b[0,2]
```

Out[48]:
```
np.int64(3)
```

In [49]:
```
b[0,:]
```

Out[49]:
```
array([1, 2, 3])
```

In [50]:
```
b[1,:]
```

Out[50]:
```
array([4, 5, 6])
```

In [51]:
```
b[1:2,0:1]
```

Out[51]:
```
array([[4]])
```

In [52]:
```
b[1:3,0:2]
```

Out[52]:
```
array([[4, 5],
       [7, 8]])
```

In [43]:
```
b.shape
```

Out[43]:
```
(3, 3)
```

matrix operations

In [53]:
```
c=np.array([[1,2,4],[4,5,6],[7,8,9]])
b+c
```

Out[53]:
```
array([[ 2,  4,  7],
       [ 8, 10, 12],
       [14, 16, 18]])
```

In [54]:
```
b-c
```

Out[54]:
```
array([[ 0,  0, -1],
       [ 0,  0,  0],
       [ 0,  0,  0]])
```

In [55]:
```
b*c
```

Out[55]:
```
array([[ 1,  4, 12],
       [16, 25, 36],
       [49, 64, 81]])
```

In [56]:
```
b/c
```

Out[56]:
```
array([[1.  , 1.  , 0.75],
       [1.  , 1.  , 1.  ],
       [1.  , 1.  , 1.  ]])
```

In [57]:
```
b//c
```

Out[57]:
```
array([[1, 1, 0],
       [1, 1, 1],
       [1, 1, 1]])
```

In [58]:
```
b**c
```

Out[58]:
```
array([[        1,         4,        81],
       [      256,      3125,     46656],
       [   823543,  16777216, 387420489]])
```

In [59]:
```
np.dot(b,c)
```

Out[59]:
```
array([[ 30,  36,  43],
       [ 66,  81, 100],
       [102, 126, 157]])
```

In [60]:
```
np.transpose(b)
```

Out[60]:
```
array([[1, 4, 7],
       [2, 5, 8],
       [3, 6, 9]])
```

In [61]:
```
np.cross(b,c)
```

Out[61]:
```
array([[ 2, -1,  0],
       [ 0,  0,  0],
       [ 0,  0,  0]])
```

In [62]:
```
np.sin(b)
```

Out[62]:
```
array([[ 0.84147098,  0.90929743,  0.14112001],
       [-0.7568025 , -0.95892427, -0.2794155 ],
       [ 0.6569866 ,  0.98935825,  0.41211849]])
```

In [63]:
```
np.exp(b)
```

Out[63]:
```
array([[2.71828183e+00, 7.38905610e+00, 2.00855369e+01],
       [5.45981500e+01, 1.48413159e+02, 4.03428793e+02],
       [1.09663316e+03, 2.98095799e+03, 8.10308393e+03]])
```

In [64]:
```
np.log(b)
```

Out[64]:
```
array([[0.        , 0.69314718, 1.09861229],
       [1.38629436, 1.60943791, 1.79175947],
       [1.94591015, 2.07944154, 2.19722458]])
```

In [65]:
```
np.abs(b)
```

Out[65]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [66]:
```
d=np.random.uniform(size=4).reshape(2,2)
np.floor(d*4)
```

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

DATASERIES AND DATAFRAME

In [67]:
```
import pandas as pd
ds = pd.Series([3,5,3]);
ds
```

Out[67]:
```
0    3
1    5
2    3
dtype: int64
```

In [68]:
```
ds[0]
```

Out[68]:
```
np.int64(3)
```

In [69]:
```
ds = pd.Series([3,5,3],index= ["A","B","C"]);
ds
```

Out[69]:
```
A    3
B    5
C    3
dtype: int64
```

In [70]:
```
ds["B"]
```

Out[70]:
```
np.int64(5)
```

In [71]:
```
ds[0:2]
```

Out[71]:
```
A    3
B    5
dtype: int64
```

In [72]:
```
ds["A"],ds["B"]
```

Out[72]:
```
(np.int64(3), np.int64(5))
```

In [73]:
```
ds[ds>10]
```

Out[73]:
```
Series([], dtype: int64)
```

In [74]:
```
ds[ds==10]
```

Out[74]:
```
Series([], dtype: int64)
```

In [75]:
```
df = pd.DataFrame(([3,5,3],[3,11,90]),columns=['A','B','C'],dtype=int)
df
```

Out[75]:

	A	B	C
0	3	5	3
1	3	11	90

In [76]:
```
df.A
```

Out[76]:
```
0    3
1    3
Name: A, dtype: int64
```

In [77]:
```
dict = { "name":['','B','C'] , "age" : [35,45,55]}
dict
```

Out[77]:
```
{'name': ['', 'B', 'C'], 'age': [35, 45, 55]}
```

In [78]:
```
df = pd.DataFrame(dict)
df
```

Out[78]:

	name	age
0		35
1	B	45
2	C	55

In [79]:
```
df['weight']=[60,90,100]
df
```

Out[79]:

	name	age	weight
0		35	60
1	B	45	90
2	C	55	100

In [80]:
```
student = {"id":[1,2,3,4,5] , "name" : ["A","B","C","D","E"] ,"address":["A11","
```

In [81]:
```
df = pd.DataFrame(student)
df
```

Out[81]:

	id	name	address	marks1	marks2
0	1	A	A11	30	40
1	2	B	B11	40	50
2	3	C	C11	50	60
3	4	D	D11	60	70
4	5	E	E11	70	80

In [82]:
```
df['total']=[70,90,110,130,150]
```

In [83]:
```
df
```

Out[83]:

	id	name	address	marks1	marks2	total
0	1	A	A11	30	40	70
1	2	B	B11	40	50	90
2	3	C	C11	50	60	110
3	4	D	D11	60	70	130
4	5	E	E11	70	80	150

In [84]:
```
df.iloc[:2,0:1]
```

Out[84]:

	id
0	1
1	2

In [85]:
```
df.loc[:2,'id']
```

Out[85]:
```
0    1
1    2
2    3
Name: id, dtype: int64
```

In [ ]:
```

```