Slip 1 to 5

Powered by Akhil Bharatiya
Bereozgar Parishad

Slip 1

1. Use Python code to evaluate each of the following expression.

a. 20 modulus 2 + 7 - (3 +7) × 20 ÷ 2

In [5]:
```
20%2+7-(3+7)*20/2
```

Out[5]: -93.0

b.30 × 10 floor division 3 + 10 modulus 3

In [8]:
```
30*10//3+10%3
```

Out[8]: 101

c.2^5 - 2^4 + 4 floor division 4

In [11]:
```
2**5-2**4+4//4
```

Out[11]: 17

2. Write Python code to repeat the following string 9 times using the string operator

'*'.

In [14]:
```
print(("PYTHON\n")*9)
```

```
PYTHON
PYTHON
PYTHON
PYTHON
PYTHON
PYTHON
PYTHON
PYTHON
PYTHON
```

In [16]:
```
print(("Mathematics\n")*9)
```

```
Mathematics
Mathematics
Mathematics
Mathematics
Mathematics
Mathematics
Mathematics
Mathematics
Mathematics
```

3. Write Python program to generate the square of numbers from 1 to 10.

In [19]:
```
for i in range(1,11):
    print(i**2)
```

```
1
4
9
16
25
36
49
64
81
100
```

Q.2. Attempt any two of the following.

1. Using Python code construct the following matrices.

1. An identity matrix of order 10 × 10.

In [24]:
```
from sympy import *
eye(10)
```

Out[24]:
$$\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix}$$

2. Zero matrix of order 7 × 3

In [27]:
```
zeros(7,3)
```

Out[27]:
$$\begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 0\end{bmatrix}$$

3. Ones matrix of order 5 × 4.

In [30]:
```
ones(5,4)
```

Out[30]:
$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

2. Write Python program to find the 10 term of the sequence of function f(x) = x^2+x.

In [33]:
```
for x in range(1,11):
    y=x**2+x
    print(y)
```

```
2
6
12
20
30
42
56
72
90
110
```

3. Generate all the prime numbers between 1 to 100 using Python code.

In [36]:
```
for i in range(2,101):
    for j in range(2,101):
        if i%j==0:
            break
    if i==j:
        print(i,end=",")
```

```
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,
```

Q.3. a. Attempt any oneof the following.

1. Write Python program to estimate the value of the integral $\int_0^{\pi}$ sin(x)dx using Simpson's $\left(\frac{1}{3}\right)^{rd}$ rule (n=6).

In [40]:
```
from  math import *
```

In [42]:
```
def Simpsons13(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
    for i in range (1,n):
       k=a+i*h
       if i%2==0:
        I=I+2*f(k)
       else:
        I=I+4*f(k)
    I=I*(h/3)
    return I
```

In [44]:
```
def f(x):
    return sin(x)
```

In [46]:
```
Simpsons13(0,pi,6)
```

Out[46]:
```
2.0008631896735363
```

2. Write Python program to evaluate interpolate value f(3) of the given data by Langranges method.

x	0	1	2	5
Y = f(x)	5	13	22	129

In [49]:
```
def lagrange_interpolation(x, y, x_new):
    n = len(x)
    result = 0.0
    for i in range(n):
        term = y[i]
        for j in range(n):
            if j != i:
                term *= (x_new - x[j]) / (x[i] - x[j])
        result += term
    return result

def interpolate(x_values, y_values, x_new):
    if len(x_values) != len(y_values) or not x_values:
        raise ValueError("x_values and y_values must have the same non-zero length.")
    result = lagrange_interpolation(x_values, y_values, x_new)
    print(f"Interpolated value at {x_new}: {result}")
```

In [51]:
```
interpolate([0,1, 2, 5], [5,13,22,129], 3)
```

```
Interpolated value at 3: 39.4
```

b. Attempt any one of the following.

Write Python program to obtained the approximate real root of $x^3-4x-9=0$ by using Regula-falsi method.

In [55]:
```
def falsePosition(a,b,e):
    i = 1
    print('\n** FALSE POSTION METHOD IMPLEMENTATION **')
    condition = True
    while condition:
        c = (a*f(b)-b*f(a))/(f(b) - f(a))
        print('Iteration-%d, c = %0.6f and f(c) = %0.6f' % (i,c,f(c)))

        if f(a) * f(c) <0:
            b = c
        else:
            a = c
        i = i + 1
        condition = abs(f(c)) > e
    print('\n required root is: %0.8f' % c)
```

In [57]:
```
def f(x):
    return x**3-4*x-9
```

In [59]:
```
falsePosition(2,3,0.00001)
```

```
** FALSE POSTION METHOD IMPLEMENTATION **
Iteration-1, c = 2.600000 and f(c) = -1.824000
Iteration-2, c = 2.693252 and f(c) = -0.237227
Iteration-3, c = 2.704918 and f(c) = -0.028912
Iteration-4, c = 2.706333 and f(c) = -0.003495
Iteration-5, c = 2.706504 and f(c) = -0.000422
Iteration-6, c = 2.706525 and f(c) = -0.000051
Iteration-7, c = 2.706528 and f(c) = -0.000006

 required root is: 2.70652761
```

2. Write Python program to estimate the value of the integral $\int_2^{10} \frac{1}{(1+x)}$ dx using Trapezoidal rule (n=8).

In [62]:
```
def trapezoidal(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
    for i in  range(1,n):
        k=a+i*h
        I=I+2*f(k)
    I=I*(h/2)
    return I
```

In [64]:
```
def f(x):
    return 1/(1+x)
```

In [66]:
```
trapezoidal(2,10,8)
```

Out[66]: 1.307756132756133

Slip 2

Padhke kya ukhad loge sab to ram
bharose hai tumhare liye

1.Write Python code to calculate the volume of a sphere with radius r=7($V=\frac{4}{3}\pi r^3$)

In [70]:
```
from math import *
```

In [72]:
```
def f(r):
    v=(4/3)*pi*r**3
    print('Volume of the sphere =',v)
```

In [74]:
```
f(7)
```

```
Volume of the sphere = 1436.7550402417319
```

2. Use Python code to construct string operation ‘+’ below string.

a. string1=Hello,string2=World!

In [78]:
```
string1="Hello"
string2="World!"
```

In [80]:
```
print(string1+" "+string2)
```

```
Hello World!
```

b. string1=Good,string2=Morning

In [83]:
```
print("Good"+" "+"Morning")
```

```
Good Morning
```

3.Write Python code to generate the square of numbers from 20 to 30

In [86]:
```
for i in range(20,31):
    print(i**2)
```

```
400
441
484
529
576
625
676
729
784
841
900
```

Q.2.Attempt any two of the following.

1.Use python code find value of f(−2),f(0),f(2) where f(x)=x^2−5x+6.

In [90]:
```
def f(x):
    a=x**2-5*x+6
    print(a)
```

In [92]:
```
f(-2)
```

```
20
```

In [94]:
```
f(0)
```

```
6
```

In [96]:
```
f(2)
```

```
0
```

2.Write Python program to find the 10 term of the sequence of function f(x)=x^3+5x

In [99]:
```
for x in range (1,11):
    a=x**3+5*x
    print(a)
```

```
6
18
42
84
150
246
378
552
774
1050
```

3.Using sympy module of python,find the eigenvalues and corresponding eigenvectors of the matrix A=

$$\begin{bmatrix} 4 & 2 & 2 \\ 2 & 4 & 2 \\ 2 & 4 & 4 \end{bmatrix}$$

In [102…
```
from sympy import *
```

In [104…
```
A=Matrix([[4,2,2],[2,4,2],[2,2,4]])
```

In [106…
```
A
```

Out[106…
$$\begin{bmatrix} 4 & 2 & 2 \\ 2 & 4 & 2 \\ 2 & 2 & 4 \end{bmatrix}$$

In [108…
```
A.eigenvals()
```

Out[108…
```
{8: 1, 2: 2}
```

In [110…
```
A.eigenvects()
```

Out[110…
```
[(2,
  2,
  [Matrix([
   [-1],
   [ 1],
   [ 0]]),
   Matrix([
   [-1],
   [ 0],
   [ 1]])]),
 (8,
  1,
  [Matrix([
   [1],
   [1],
   [1]])])]
```

Q.3. a.Attempt any one of the following

1. Write Python program to estimate the value of the integral $\int_0^1 \frac{1}{(1+x^2)} dx$ using Simpson's $\left(\frac{1}{3}\right)^{rd}$ rule (n=4).

In [114…
```
from math import *
```

In [116…
```
def Simpsons13(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
```

```
    for i in range (1,n):
       k=a+i*h
       if i%2==0:
        I=I+2*f(k)
       else:
        I=I+4*f(k)
    I=I*(h/3)
    return I
```

In [118…
```
def f(x):
    return 1/(1+x**2)
```

In [120…
```
Simpsons13(0,1,4)
```

Out[120…
```
0.7853921568627451
```

2.Write Python program to obtained a real root of f(x)=x^3−8x−4=0 by using Newton–Raphson method

In [123…
```
def newtonRaphson(x0,e):
    print('\n** NEWTON RAPHSON METHOD IMPLEMENTATION **')
    i=1
    condition = True
    while condition:
        if g(x0)== 0.0:
            print('Divide by zero error')
            break



        x1=x0 - f(x0)/g(x0)
        print('Iteration-%d, x1 = %0.6f and f(x1) = %0.6f' % (i,x1,f(x1)))
        x0=x1
        i=i+1
        condition = abs(f(x1)) > e
        print('\n required root is: %0.8f' % x1)
```

In [125…
```
def f(x):
    return x**3-8*x-4
def g(x):
    return 3*x**2-8
```

In [127…
```
newtonRaphson(3.5,0.00001)
```

```
** NEWTON RAPHSON METHOD IMPLEMENTATION **
Iteration-1, x1 = 3.121739 and f(x1) = 1.448231

 required root is: 3.12173913
Iteration-2, x1 = 3.053541 and f(x1) = 0.043240

 required root is: 3.05354138
Iteration-3, x1 = 3.051376 and f(x1) = 0.000043

 required root is: 3.05137640
Iteration-4, x1 = 3.051374 and f(x1) = 0.000000

 required root is: 3.05137424
```

b.Attempt any one of the following

1.Write Python program to obtained the approximate real roots of x^3−2x−5=0 in[2,3]using Regula-falsi method.

In [131…
```python
def falsePosition(a,b,e):
    i = 1
    print('\n** FALSE POSTION METHOD IMPLEMENTATION **')
    condition = True
    while condition:
        c = (a*f(b)-b*f(a))/(f(b) - f(a))
        print('Iteration-%d, c = %0.6f and f(c) = %0.6f' % (i,c,f(c)))

        if f(a) * f(c) <0:
            b = c
        else:
            a = c
        i = i + 1
        condition = abs(f(c)) > e
    print('\n required root is: %0.8f' % c)
```

In [133…
```python
def f(x):
    return x**3-2*x-5
```

In [135…
```python
falsePosition(2,3,0.00001)
```

```
** FALSE POSTION METHOD IMPLEMENTATION **
Iteration-1, c = 2.058824 and f(c) = -0.390800
Iteration-2, c = 2.081264 and f(c) = -0.147204
Iteration-3, c = 2.089639 and f(c) = -0.054677
Iteration-4, c = 2.092740 and f(c) = -0.020203
Iteration-5, c = 2.093884 and f(c) = -0.007451
Iteration-6, c = 2.094305 and f(c) = -0.002746
Iteration-7, c = 2.094461 and f(c) = -0.001012
Iteration-8, c = 2.094518 and f(c) = -0.000373
Iteration-9, c = 2.094539 and f(c) = -0.000137
Iteration-10, c = 2.094547 and f(c) = -0.000051
Iteration-11, c = 2.094550 and f(c) = -0.000019
Iteration-12, c = 2.094551 and f(c) = -0.000007

 required root is: 2.09455087
```

2. Write Python program to evaluate interpolate value f(3.5) of the given data by Lagranges method

x	0	1	2	5
Y = f(x)	2	3	12	147

In [138…
```python
def lagrange_interpolation(x, y, x_new):
    n = len(x)
    result = 0.0
    for i in range(n):
        term = y[i]
        for j in range(n):
            if j != i:
                term *= (x_new - x[j]) / (x[i] - x[j])
        result += term
    return result

def interpolate(x_values, y_values, x_new):
    if len(x_values) != len(y_values) or not x_values:
        raise ValueError("x_values and y_values must have the same non-zero length.")
    result = lagrange_interpolation(x_values, y_values, x_new)
    print(f"Interpolated value at {x_new}: {result}")
```

In [140…
```python
interpolate([0,1, 2, 5], [2,3,12,147], 3.5)
```

```
Interpolated value at 3.5: 53.625
```

Slip 3

Life is a race be
a racist.

Q.1. Attempt any two of the following.

1. Write python code to test whether given number is divisible by 2 or 3 or 5.

In [145…
```python
def f(x):
    if x % 2 == 0:
        print(x,'is divisible by 2')
    if x % 3 == 0:
        print(x,'is divisible by 3')
    if x % 5 == 0:
        print(x,'is divisible by 5')
    if x % 2 != 0 and x % 3 != 0 and x % 5 != 0:
        print(x,'is not divisible by 2,3 or 5')
```

In [147…
```python
f(4)
```

```
4 is divisible by 2
```

In [149…
```python
f(9)
```

```
9 is divisible by 3
```

In [151…
```python
f(25)
```

```
25 is divisible by 5
```

In [153…
```python
f(30)
```

```
30 is divisible by 2
30 is divisible by 3
30 is divisible by 5
```

In [155…
```python
f(6)
```

```
6 is divisible by 2
6 is divisible by 3
```

In [157…
```python
f(10)
```

```
10 is divisible by 2
10 is divisible by 5
```

In [159…
```python
f(15)
```

```
15 is divisible by 3
15 is divisible by 5
```

In [161…
```python
f(7)
```

```
7 is not divisible by 2,3 or 5
```

2. Repeat the following string 11 times using the string operator '*' on Python.

a. LATEX

In [165…
```
print(("LATEX\n")*9)
```

```
LATEX
LATEX
LATEX
LATEX
LATEX
LATEX
LATEX
LATEX
LATEX
```

In [167…
```
print(("MATLAB\n")*9)
```

```
MATLAB
MATLAB
MATLAB
MATLAB
MATLAB
MATLAB
MATLAB
MATLAB
MATLAB
```

3. Use Python code to find sum of first thirty natural numbers.

In [170…
```
# With for loop
n=0
for i in range(1,31):
    n+=i;
print(n,'is the sum fo the first 30  natural numbers')
```

```
465 is the sum fo the first 30  natural numbers
```

In [172…
```
# With formula
n=30
sum=n*(n+1)//2
print(sum,'is the sum fo the first 30  natural numbers')
```

```
465 is the sum fo the first 30  natural numbers
```

Q.2. Attempt any two of the following.

1. Using Python construct the following matrices.

1. An identity matrix of order 10 × 10.

In [177…
```
from sympy import *
```

In [179…
```
eye(10,10)
```

Out[179…

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

2. Zero matrix of order 7 × 3.

In [182…
```
zeros(7,3)
```

Out[182…

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

3. Ones matrix of order 5 × 4

In [185…
```
ones(5,4)
```

Out[185…

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

2. Using python, find the eigenvalues and corresponding eigenvectors of the matrix

$$\begin{bmatrix} 3 & -2 \\ 6 & -4 \end{bmatrix}$$

In [188…
```
A=Matrix([[3,-2],[6,-4]])
```

In [190…
```
A
```

Out[190…

$$\begin{bmatrix} 3 & -2 \\ 6 & -4 \end{bmatrix}$$

In [192…
```
A.eigenvals()
```

Out[192…
```
{-1: 1, 0: 1}
```

In [194…
```
A.eigenvects()
```

Out[194…
```
[(-1,
  1,
  [Matrix([
   [1/2],
   [  1]])]),
 (0,
  1,
  [Matrix([
   [2/3],
   [  1]])])]
```

3. Generate all the prime numbers between 1 to 100 using Python code

In [197…
```
for i in range(2,101):
    for j in range(2,101):
        if i%j==0:
            break
    if i==j:
        print(i,end=",")
```

2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,

Q.3. a. Attempt any one of the following.

1. Write Python program to estimate the value of the integral $\int_0^{\pi}$ sin(x)dx using Simpson's $\left(\frac{1}{3}\right)^{rd}$ rule (n=6).

In [201…
```
from  math import *
```

In [203…
```
def Simpsons13(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
    for i in range (1,n):
       k=a+i*h
       if i%2==0:
        I=I+2*f(k)
       else:
        I=I+4*f(k)
    I=I*(h/3)
    return I
```

In [205…
```
def f(x):
    return sin(x)
```

In [207…
```
Simpsons13(0,pi,6)
```

Out[207… 2.0008631896735363

2. Write Python program to evaluate third order forward difference of the given data.

x	0	1	2	3
Y = f(x)	2	1	2	10

In [210…
```
def forward_difference(x, y):

    delta1 = [y[i+1] - y[i] for i in range(len(y)-1)]
```

```
    delta2 = [delta1[i+1] - delta1[i] for i in range(len(delta1)-1)]


    delta3 = [delta2[i+1] - delta2[i] for i in range(len(delta2)-1)]


    print("First-order differences:", delta1)
    print("Second-order differences:", delta2)
    print("Third-order difference:", delta3)
```

In [212…
```
forward_difference([0, 1, 2, 3], [1, 0, 1, 10])
```

```
First-order differences: [-1, 1, 9]
Second-order differences: [2, 8]
Third-order difference: [6]
```

b. Attempt anyone of the following.

1. Write Python program to evaluate f(3.5) of the given data

x	1	2	3	4	5
Y = f(x)	30	50	55	40	11

In [216…
```
import math

def newtons_forward_interpolation(x, y, value):
    n = len(x)
    h = x[1] - x[0]
    u = (value - x[0]) / h

    diff_table = [y.copy()]
    for i in range(1, n):
        next_diff = []
        for j in range(n - i):
            next_diff.append(diff_table[i - 1][j + 1] - diff_table[i - 1][j])
        diff_table.append(next_diff)

    result = diff_table[0][0]

    for i in range(1, n):
        term = 1
        for j in range(i):
            term *= (u - j)
        term /= math.factorial(i)

        result += term * diff_table[i][0]

    print(round(result, 6))
```

In [218…
```
newtons_forward_interpolation([1,2,3,4,5],[30,50,55,40,11],3.5)
```

```
49.882812
```

2. Write Python program to estimate the value of the integral $\int_2^{10} \frac{1}{(1+x)}$ dx using Trapezoidal rule (n=5).

In [221…
```
def trapezoidal(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
    for i in  range(1,n):
        k=a+i*h
        I=I+2*f(k)
```

```
    I=I*(h/2)
    return I
```

In [223…
```
def f(x):
    return 1/(1+x)
```

In [225…
```
trapezoidal(2,10,5)
```

Out[225…
```
1.3206255135651455
```

Slip 4

Itni mehnat 12th mein ki hoti to koi acche college padhta abhi

1. Using python code sort the tuple in ascending and descending order 5,-3, 0, 1, 6,-6, 2.

Descending

In [230…
```
n=(5,-3,0,1,6,-6,2)
n=list(n)
n.sort(reverse=True)
n=tuple(n)
print(n)
```

```
(6, 5, 2, 1, 0, -3, -6)
```

Ascending

In [233…
```
n=(5,-3,0,1,6,-6,2)
n=list(n)
n.sort(reverse=False)
n=tuple(n)
print(n)
```

```
(-6, -3, 0, 1, 2, 5, 6)
```

2. Write python program which deals with concatenation and repetition of lists.

List1 = [15, 20, 25, 30, 35, 40]

List2 = [7, 14, 21, 28, 35, 42]

In [236…
```
List1 = [15, 20, 25, 30, 35, 40]
List2 = [7, 14, 21, 28, 35, 42]
```

(a) Find List1 + List2

In [239…
```
print(List1+List2)
```

```
[15, 20, 25, 30, 35, 40, 7, 14, 21, 28, 35, 42]
```

(b) Find 9*List1

In [242…
```
print(9*List1)
```

```
[15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40, 15, 20, 25, 30, 35, 40]
```

In [244…
```
print(7*List2)
```

```
[7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42, 7, 14, 21, 28, 35, 42]
```

3. Write Python code to find the square of odd numbers from 1 to 20 using while loop

In [247…
```
for i in range(1,21):
    if i % 2 !=0:
        print(i**2)
```

```
1
9
25
49
81
121
169
225
289
361
```

Q.2. Attempt any twoof the following

1. Using Python construct the following matrices.

1. An identity matrix of order 10 × 10

In [252…
```
from sympy import *
```

In [254…
```
eye(10,10)
```

Out[254…
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

2. Zero matrix of order 7 × 3

In [257…
```
zeros(7,3)
```

Out[257…

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

3. Ones matrix of order 5 ×4

In [260…
```
ones(5,4)
```

Out[260…

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

2. Find the data type of the following data by using Python code

In [263…
```
a='number'
b=31.25
c=8 + 4j
d='Mathematics'
e=49
```

In [265…
```
print((type(a)))
print((type(b)))
print((type(c)))
print((type(d)))
print((type(e)))
```

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

3. Write Python program to find the determinant of matrices A=

$$\begin{bmatrix} 1 & 0 & 5 \\ 2 & 1 & 6 \\ 3 & 4 & 0 \end{bmatrix}$$

and B=

$$\begin{bmatrix} 2 & 5 \\ -1 & 4 \end{bmatrix}$$

In [268…
```
A=Matrix([[1,0,5],[2,1,6],[3,4,0]])
B=Matrix([[2,5],[-1,4]])
```

In [270…
```
A
```

Out[270…
$$\begin{bmatrix}1 & 0 & 5\\2 & 1 & 6\\3 & 4 & 0\end{bmatrix}$$

In [272…
```
B
```

Out[272…
$$\begin{bmatrix}2 & 5\\-1 & 4\end{bmatrix}$$

In [274…
```
A.det()
```

Out[274…
$$1$$

In [276…
```
B.det()
```

Out[276…
$$13$$

Q.3. a.Attempt any one of the following.

1. Write Python program to estimate the value of the integral $\int_0^{\pi} x\sin(x)dx$ using Simpson's $\left(\frac{1}{3}\right)^{rd}$ rule (n=6).

In [280…
```
from math import *
```

In [282…
```
def Simpsons13(a,b,n):
    h=(b-a)/n
    I=f(a)+f(b)
    for i in range (1,n):
       k=a+i*h
       if i%2==0:
        I=I+2*f(k)
       else:
        I=I+4*f(k)
    I=I*(h/3)
    return I
```

In [284…
```
def f(x):
    return x*sin(x)
```

In [286…
```
Simpsons13(0,pi,6)
```

Out[286…
```
3.1429485487583113
```

2.Write Python program to estimate a root of anequation f(x)=3x−cos(x)−1 using Newton–Raphson method correct up to four decimal places

In [289…
```
from math import *
```

In [291…
```
def newtonRaphson(x0,e):
    print('\n** NEWTON RAPHSON METHOD IMPLEMENTATION **')
    i=1
    condition = True
    while condition:
```

```
        if g(x0)== 0.0:
            print('Divide by zero error')
            break



        x1=x0 - f(x0)/g(x0)
        print('Iteration-%d, x1 = %0.6f and f(x1) = %0.6f' % (i,x1,f(x1)))
        x0=x1
        i=i+1
        condition = abs(f(x1)) > e
        print('\n required root is: %0.8f' % x1)
```

In [293…
```
def f(x):
    return 3*x-cos(x)-1
def g(x):
    return 3+sin(x)
```

In [295…
```
newtonRaphson(0.5,0.00001)
```

```
** NEWTON RAPHSON METHOD IMPLEMENTATION **
Iteration-1, x1 = 0.608519 and f(x1) = 0.005060

 required root is: 0.60851865
Iteration-2, x1 = 0.607102 and f(x1) = 0.000001

 required root is: 0.60710188
```

b.Attempt any one of the following

1.Write Python program to find all positive prime numbers less then given number n.

In [299…
```
def f(n):
    for i in range(2, n + 1):

        for j in range(2, int(i**0.5) + 1):
            if i % j == 0:
                break
        else:
            print(i, end=",")
```

In [301…
```
f(69)
```

```
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,
```

2.Write Python program to evaluate f(2.5)by forward difference formula of the given data

x	0	1	2	3
Y = f(x)	2	1	2	10

In [304…
```
import math

def newtons_forward_interpolation(x, y, value):
    n = len(x)
    h = x[1] - x[0]
    u = (value - x[0]) / h

    diff_table = [y.copy()]
    for i in range(1, n):
        next_diff = []
        for j in range(n - i):
            next_diff.append(diff_table[i - 1][j + 1] - diff_table[i - 1][j])
```

```
        diff_table.append(next_diff)

    result = diff_table[0][0]

    for i in range(1, n):
        term = 1
        for j in range(i):
            term *= (u - j)
        term /= math.factorial(i)

        result += term * diff_table[i][0]

    print(round(result, 6))
```

In [306…
```
newtons_forward_interpolation([0,1,2,3],[2,1,2,10],2.5)
```

```
4.8125
```

Slip 5

1.Using sympy module of python find the following for the matrices A=

$$\begin{bmatrix} -1 & 1 & 0 \\ 8 & 5 & 2 \\ 2 & -6 & 2 \end{bmatrix}$$

and B=

$$\begin{bmatrix} 9 & 0 & 3 \\ 1 & 4 & 1 \\ 1 & 0 & -4 \end{bmatrix}$$

In [310…
```
from sympy import *
```

In [312…
```
A=Matrix([[-1,1,0],[8,5,2],[2,-6,2]])
B=Matrix([[9,0,3],[1,4,1],[1,0,-4]])
```

In [314…
```
A
```

Out[314…
$$\begin{bmatrix} -1 & 1 & 0 \\ 8 & 5 & 2 \\ 2 & -6 & 2 \end{bmatrix}$$

In [316…
```
B
```

Out[316…
$$\begin{bmatrix} 9 & 0 & 3 \\ 1 & 4 & 1 \\ 1 & 0 & -4 \end{bmatrix}$$

(a) 2A+B.

In [319…
```
2*A+B
```

Out[319…
$$\begin{bmatrix} 7 & 2 & 3 \\ 17 & 14 & 5 \\ 5 & -12 & 0 \end{bmatrix}$$

(b) 3A–5B.

In [322…
```
3*A-5*B
```

Out[322…
$$\begin{bmatrix} -48 & 3 & -15 \\ 19 & -5 & 1 \\ 1 & -18 & 26 \end{bmatrix}$$

(c)A^{-1}.

In [325…
```
A.inv()
```

Out[325…
$$\begin{bmatrix} -\frac{11}{17} & \frac{1}{17} & -\frac{1}{17} \\ \frac{6}{17} & \frac{1}{17} & -\frac{1}{17} \\ \frac{29}{17} & \frac{2}{17} & \frac{13}{34} \end{bmatrix}$$

(d)B^3.

In [328…
```
B**3
```

Out[328…
$$\begin{bmatrix} 771 & 0 & 192 \\ 145 & 64 & 46 \\ 64 & 0 & -61 \end{bmatrix}$$

(e)$A^T + B^T$.

In [331…
```
A.T+B.T
```

Out[331…
$$\begin{bmatrix} 8 & 9 & 3 \\ 1 & 9 & -6 \\ 3 & 3 & -2 \end{bmatrix}$$

2. Evaluate following expression on Python

(a)M=[1,2,3,4],Find length M.

In [335…
```
M=[1,2,3,4]
```

In [337…
```
print(len(M))
```
```
4
```

(b)L="XYZ"+"pqr",Find L

In [340…
```
L="XYZ"+"pqr"
```

In [342…
```
print(L)
```
```
XYZpqr
```

(c) s='MakeInIndia',Find (s[:7] & (s[:9]).

In [345…
```
s='MakeinIndia'
```

In [347…
```
s[:7]
```
Out[347…
```
'MakeinI'
```

In [349…
```
s[:79]
```
Out[349…
```
'MakeinIndia'
```

3.Use Python code to generate the square root of numbers from 21 to 49.

In [352…
```
from math import *
```

In [354…
```
for i in range(21,49):
    print('square root of',i,"is=",sqrt(i))
```
```
square root of 21 is= 4.58257569495584
square root of 22 is= 4.69041575982343
square root of 23 is= 4.795831523312719
square root of 24 is= 4.898979485566356
square root of 25 is= 5.0
square root of 26 is= 5.0990195135927845
square root of 27 is= 5.196152422706632
square root of 28 is= 5.291502622129181
square root of 29 is= 5.385164807134504
square root of 30 is= 5.477225575051661
square root of 31 is= 5.5677643628300215
square root of 32 is= 5.656854249492381
square root of 33 is= 5.744562646538029
square root of 34 is= 5.830951894845301
square root of 35 is= 5.916079783099616
square root of 36 is= 6.0
square root of 37 is= 6.082762530298219
square root of 38 is= 6.164414002968976
square root of 39 is= 6.244997998398398
square root of 40 is= 6.324555320336759
square root of 41 is= 6.4031242374328485
square root of 42 is= 6.48074069840786
square root of 43 is= 6.557438524302
square root of 44 is= 6.6332495807108
square root of 45 is= 6.708203932499369
square root of 46 is= 6.782329983125268
square root of 47 is= 6.855654600401044
square root of 48 is= 6.928203230275509
```

Q.2.Attempt any two of the following

1.Using Python construct the following matrices.

1.An identity matrix of order 10×10

In [359…
```
eye(10,10)
```

Out[359…

$$\begin{bmatrix}1&0&0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0&0\\0&0&0&1&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0&0\\0&0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&1&0&0&0\\0&0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&1\end{bmatrix}$$

2. Zero matrix of order 7×3.

In [362…
```
zeros(7,3)
```

Out[362…

$$\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\\0&0&0\\0&0&0\\0&0&0\\0&0&0\end{bmatrix}$$

3.Ones matrix of order 5×4.

In [365…
```
ones(5,4)
```

Out[365…

$$\begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}$$

2.Usinglin solvec ommand in python,solve the following system of linear equations.

$$x-2y+3z=7$$

$$2x+y+z=4$$

$$-3x+2y-2z=-10$$

In [368…
```
from sympy import *
```

In [370…
```
x,y,z=symbols("x,y,z")
A=Matrix([[1,-2,3],[2,1,1],[-3,2,-2]])
B=Matrix([[7,4,-10]])
linsolve((A,B),[x,y,z])
```

Out[370…

$$\{(2,\ -1,\ 1)\}$$

3.Generate all relatively prime numbers to 111 which are less than 150 using Python code

In [373…
```python
import math

def relatively_prime_numbers(x, n):
    relatively_prime_list = []

    for i in range(1, n):
        if math.gcd(x, i) == 1:
            relatively_prime_list.append(i)

    print(f"Numbers less than {n} that are relatively prime to {x}:")
    print(relatively_prime_list)
```

In [375…
```python
relatively_prime_numbers(111,150)
```

```
Numbers less than 150 that are relatively prime to 111:
[1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26, 28, 29, 31, 32, 34, 35, 38, 40, 41, 43,
44, 46, 47, 49, 50, 52, 53, 55, 56, 58, 59, 61, 62, 64, 65, 67, 68, 70, 71, 73, 76, 77, 79, 80, 82, 83, 8
5, 86, 88, 89, 91, 92, 94, 95, 97, 98, 100, 101, 103, 104, 106, 107, 109, 110, 112, 113, 115, 116, 118, 11
9, 121, 122, 124, 125, 127, 128, 130, 131, 133, 134, 136, 137, 139, 140, 142, 143, 145, 146, 149]
```

Q.3. a.Attempt any one of the following.

1.Write Python code to find eigenvalues and corresponding eigen vectors of the matrix A=

$$\begin{bmatrix} 1 & 3 & 3 \\ 2 & 2 & 3 \\ 4 & 2 & 1 \end{bmatrix}$$

and hence find matrix P with diagonalize to A.

In [379…
```python
from sympy import *
```

In [381…
```python
A=Matrix([[1,3,3],[2,2,3],[4,2,1]])
```

In [383…
```python
A
```

Out[383…
$$\begin{bmatrix} 1 & 3 & 3 \\ 2 & 2 & 3 \\ 4 & 2 & 1 \end{bmatrix}$$

In [385…
```python
A.eigenvals()
```

Out[385…
```
{7: 1, -1: 1, -2: 1}
```

In [387…
```python
A.eigenvects()
```

Out[387…
```
[(-2,
  1,
  [Matrix([
   [-1/2],
   [-1/2],
   [   1]])]),
 (-1,
  1,
  [Matrix([
   [ 0],
   [-1],
   [ 1]])]),
 (7,
  1,
  [Matrix([
   [1],
   [1],
   [1]])])]
```

In [389…
```
P,D=A.diagonalize()
```

In [391…
```
P
```

Out[391…
$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & -1 & 1 \\ 2 & 1 & 1 \end{bmatrix}$$

In [393…
```
D
```

Out[393…
$$\begin{bmatrix} -2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 7 \end{bmatrix}$$

2.Write Python program to estimate a root of an equation f(x)=$3x^2$+4x−10 using Newton–Raphson method correct up to four decimal places.

In [396…
```
def newtonRaphson(x0,e):
    print('\n** NEWTON RAPHSON METHOD IMPLEMENTATION **')
    i=1
    condition = True
    while condition:
        if g(x0)== 0.0:
            print('Divide by zero error')
            break



        x1=x0 - f(x0)/g(x0)
        print('Iteration-%d, x1 = %0.6f and f(x1) = %0.6f' % (i,x1,f(x1)))
        x0=x1
        i=i+1
        condition = abs(f(x1)) > e
        print('\n required root is: %0.8f' % x1)
```

In [398…
```
def f(x):
    return  3*x**2 +4*x-10
def g(x):
    return 6*x+4
```

In [400…
```
newtonRaphson(2.5,0.00001)
```

```
** NEWTON RAPHSON METHOD IMPLEMENTATION **
Iteration-1, x1 = 1.513158 and f(x1) = 2.921572

 required root is: 1.51315789
Iteration-2, x1 = 1.289778 and f(x1) = 0.149696

 required root is: 1.28977814
Iteration-3, x1 = 1.277026 and f(x1) = 0.000488

 required root is: 1.27702580
Iteration-4, x1 = 1.276984 and f(x1) = 0.000000

 required root is: 1.27698397
```

b.Attempt any one of the following.

1.Write Python program to obtained the approximate real root of $x^3-4x-9=0$ by using Regula-falsi method.

In [404…
```
def falsePosition(a,b,e):
    i = 1
    print('\n** FALSE POSTION METHOD IMPLEMENTATION **')
    condition = True
    while condition:
        c = (a*f(b)-b*f(a))/(f(b) - f(a))
        print('Iteration-%d, c = %0.6f and f(c) = %0.6f' % (i,c,f(c)))

        if f(a) * f(c) <0:
            b = c
        else:
            a = c
        i = i + 1
        condition = abs(f(c)) > e
    print('\n required root is: %0.8f' % c)
```

In [406…
```
def f(x):
    return x**3-4*x-9
```

In [408…
```
falsePosition(2,3,0.00001)
```

```
** FALSE POSTION METHOD IMPLEMENTATION **
Iteration-1, c = 2.600000 and f(c) = -1.824000
Iteration-2, c = 2.693252 and f(c) = -0.237227
Iteration-3, c = 2.704918 and f(c) = -0.028912
Iteration-4, c = 2.706333 and f(c) = -0.003495
Iteration-5, c = 2.706504 and f(c) = -0.000422
Iteration-6, c = 2.706525 and f(c) = -0.000051
Iteration-7, c = 2.706528 and f(c) = -0.000006

 required root is: 2.70652761
```

b.Attempt any one of the following.

1.Write Python program to find all positive prime numbers less then given number n.

In [412…
```
def f(n):
    for i in range(2, n + 1):

        for j in range(2, int(i**0.5) + 1):
            if i % j == 0:
                break
        else:
            print(i, end=",")
```

In [414…
```
f(420)
```

```
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,13
9,149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,233,239,241,251,257,263,269,271,277,281,
283,293,307,311,313,317,331,337,347,349,353,359,367,373,379,383,389,397,401,409,419,
```

2.Write Python program to evaluate f(3.5) by forward difference formula of the given data.

x	1	2	3	4	5
Y = f(x)	41	62	65	50	17

In [417…
```
import math

def newtons_forward_interpolation(x, y, value):
    n = len(x)
    h = x[1] - x[0]
    u = (value - x[0]) / h

    diff_table = [y.copy()]
    for i in range(1, n):
        next_diff = []
        for j in range(n - i):
            next_diff.append(diff_table[i - 1][j + 1] - diff_table[i - 1][j])
        diff_table.append(next_diff)

    result = diff_table[0][0]

    for i in range(1, n):
        term = 1
        for j in range(i):
            term *= (u - j)
        term /= math.factorial(i)

        result += term * diff_table[i][0]

    print(round(result, 6))
```

In [419…
```
newtons_forward_interpolation([1,2,3,4,5],[41,62,65,50,17],3.5)
```

```
59.75
```