output:

$1/3$

aluate the integral $\int_0^3 \int_0^{3-x} \int_0^{3-x-y} (xyz)\, dz\, dy\, dx$

```
rom sympy import *
= Symbol('x')
= symbol('y')
= symbol('z')
2= integrate((x*y*z),(z,0, 3-x-y)), (y,0,3-x), (x,0,3))
int (w2)
```

put:

$81/80$

Find Beta (3,5) , Gamma (5)

```
from sympy import *
m= input ('m: ');
n= input ('n: ');
m= float (m);
n= float (n);
s= beta (m,n);
t= gamma (n);
print ("gamma (", n, ") is %3.3f" %t)
print (" Beta (", m, n, ") is %3.3f" %s)
```

output:

```
m: 3
n: 5
gamma (5.0) is 24.000
Beta (3.0 5.0) is 0.010
```

geometrical Interpretation

Name: Shivaraja.T

USN: 4SN23EE011

Date of submission: 11/5/24

To find gradient of $\phi = x^2yz$

1) from sympy.physics vector import*

```
from sympy import var, pprint
var('x, y, z')
v= Reference Frame ('v')
F= v[0]**2*v[1]*v[2]
G= gradient (F,v)
F= F.subs([(v[0],x),(v[1],y),(v[2],z)])
Print(" Given scalar function F= ")
display (F)
G= G.subs([(v[0],x),(v[1],y),(v[2],z)])
Print("\n Gradient of F= ")
display (G)
```

output:

given scalar function:

x^2yz

gradient of f=

$2xyz\,\hat{v}_x + x^2z\,\hat{v}_y + x^2y\,\hat{v}_z$

2) To find divergence of $\vec{F} = x^2y\hat{i} + yz^2\hat{j} + x^2z\hat{k}$

```
from sympy.physics.vector import *
from sympy import var
var('x,y,z')
v= Reference Frame ('v')
```

```
F= v[0]**2*v[1] [illegible] v[1]*v[2]**2*[illegible].y+v[0]**2*[illegible]
G= divergence(F,v)
F = F.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Given vector point function is ")
display(F)
G= G.subs([(v[0],x),(v[1],y),(v[2],z)])
print("Divergence of F=")
display(G)
```

output:

Given vector point function is

$x^2y\,\hat{v}_x + yz^2\,\hat{v}_y + x^2z\,\hat{v}_z$

Divergence of F=

$x^2 + 2xy + z^2$

Regula - Falsi method to solve a transcendental equation

```
from sympy import *
x= Symbol('x')
g= input('Enter the function')
f= lambdify (x,g)
a= float( input ('Enter a values : '))
b= float ( input('Enter b values : '))
N= int (input('Enter number of iterations : '))

for i in Range (1,N+1) :
    c=(a*f(b)-b*f(a))/ f(b)-f(a))
    if ((f(a)* f(c)<0)):
        b=c
    else:
        a=c
    print('iteration %d \t the root %0.3f \t function value %0.3f \n',%(i,c,
    f(c)));
```

output:

```
Enters the function: x**3-2*x-5
Enter a values: 2
Enter b values: 3
Enter number of iterations: 5
```

Iteration 1	the root 2.059	function values -0.391
Iteration 2	the root 2.081	function values -0.147
Iteration 3	the root 2.090	function values -0.055
Iteration 4	the root 2.093	function values -0.020
Iteration 5	the root 2.094	function values -0.007

```
from sympy import
x= symbol('x')
g= input('Enter the function')
f= lambdify (x, g)
dg= diff (g);
df= lambdify (x, dg)
x0= float (input("Enter the initial approximation"));
n= int (input("Enter the number of iterations"));
for i in range (1, n+1):
    x1= (x0-(f(x0)/df(x0)))
    print ("iteration %d \t the root %0.3f \t function value %0.3f \t"
% (i, x1, f(x1)));
    x0=x1
```

Output:

Enter the function: 3*x - cos(x) - 1

Enter the initial approximation: 1

Enter the number of iterations: 5

itration 1	the root 0.620	function value 0.01
itration 2	the root 0.607	function value 0.0
itration 3	the root 0.607	function value 0.0
itration 4	the root 0.607	function value 0.0
itration 5	the root 0.607	function value -0.

LAB 3: Computation of area under the curve using Trapezoidal, Simpson's $(1/3)^{rd}$ & Simpson's $(3/8)^{th}$ rule.

name: Shivaraja T
USN: 4CN23EE0[illegible]
Date of submission:

Evaluate $\int_0^5 1/(1+x^2)$

```
def my_func(x):
        return 1/(1+x**2)

def trapezoidal(x0,xn,n):
        h=(xn-x0)/n

integration= my_func(x0)+ my_func(xn)

for i in range(1,n):
    k= x0+ i*h
    integration= integration +2* my_func(k)

integration = integration *h/2
    return integration

lower_limit= float(input(" Enter lower limit of integration"))
upper_limit= float(input(" Enter upper limit of integration ")
sub_interval = int(input (" Enter number of sub interval")

result= trapezoidal(lower_limit, upper_limit, sub_interval)
print(" Integration result by trapezoidal method is", result)
```

Output:
Enter lower limit of integration: 0
Enter upper limit of integration: 5
Enter number of sub interval: 100
Integration result by simpson's 1/3 method is: 1.4041

Evaluate $\int_0^6 1/1+x^2 \, dx$ using simpson's 3/8th rule taking 6 sub interval.

```
def simpson's_3_8_rule(f, a, b, n):
    h = (b-a)/n
    s = f(a) + f(b)

for i in range(1, n, 3):
    s+= 3*f(a+i*h):

for i in range(3, n-1, 3):
    s+= 3*f(a+i*h).

for i in range(2, n-2, 3):
    s+= 2*f(a+i*h)

return s*3*h/8

def f(x):
    return 1/(1+x**2)

a=0
b=6
n=6

result = simpsons_3_8_rule(f, a, b, n)
print("%3.5f" % result)
```

Output:

1.27631

Apply the Runge [illegible] of dy/dx = 1+(y/x) at y(2) taking h=0.2 Given that y(1)=2

4)
```
from sympy import *
import numpy as np
def Runge kutta (g, x0, h, y0, xn):
    x, y = symbols('x, y')
    f = lambdify([x, y], g)

    xt = x0+h
    y = [y0]

  while xt =< xn:
      k1 = h*f(x0, y0)
      k2 = h*f(x0+h/2, y0+k1/2)
      k3 = h*f(x0+h/2, y0+k2/2)
      k4 = h*f(x0+h, y0+k3)
      y1 = y0+1/6*(k1+2*k2+2*k3+k4)
       y.append(y1)
        x0 = xt
        y0 = y1
        xt = xt+h
    return np.round(y, 2)
Runge kutta ('1+(y/x)', 1, 0.2, 2, 2)
```

output:

array[(2, 2.62, 3.27, 3.95, 4.66, 5.39)]

Apply milne's predictor & corrector method to solve $dy/dx = x^2 + (y/2)$ at $y(1.4)$ Given that $y(1) = 2$, $y(1.1) = 2.2156$, $y(1.2) = 2.4649$, $y(1.3) = 2.7514$

```
x0=1
y0=2
y1= 2.2156
y2= 2.4649
y3= 2.7514
h=0.1
x1= x0+h
x2= x1+h
x3= x2+h
x4= x3+h
def f(x,y)
    return x**2+(y/2)
y10= f(x0,y0)
y11 = f(x1,y1)
y12= f(x2,y2)
y13= f(x3,y3)
y4p= y0+(4*h/3)*(2*y11-y12+2*y13)
print("predicted value of y4 is %3.3f')%y4p)
y14= f(x4, y4p):
for i in range(1,4):
        y4+y2(h/3)*(y14+h*y13+y12):
  print("corrected value of y4 after iteration %d is \t %3.5f \t')%(i,y4))
  y14= f(x4,y4);
```

output:

Predicted value of y4 is 3.079

Corrected value of y4 after iteration 1 is 3.07940

Corrected value of y4 after iteration 2 is 3.07940

Corrected value of y4 after iteration 3 is 3.07940