In [1]:

```python
#taylor series
from sympy import*
var("x,x1")
y=Function("y")(x)
f=x**2+y**2
x0,y0=0,1
fi=f
der=[f]
for i in range(3):
    der.append(diff(f,x).subs(diff(y,x),fi))
    f=der[i+1]
yx1=y0
for i in range(3):
    f1=lambdify([x,y],der[i])
    yx1=yx1+(((x1-x0)**(i+1))/factorial(i+1))*f1(x0,y0)
print("The taylor series expansion upto 4th degree is:")
display(yx1)
print(f"y({x1})=%0.4f"%yx1.subs(x1,float(input("enter the value of x:"))))
```

```
The taylor series expansion upto 4th degree is:
```

$\displaystyle \frac{4 x_{1}^{3}}{3} + x_{1}^{2} + x_{1} + 1$

```
enter the value of x:0.1
y(x1)=1.1113
```

In [3]:

```python
#modified eulers y'=e^-x  x=0.2
f=lambda x,y:exp(-x)
x0=0
y0=-1
h=0.2
n=int(input("Enter max number of iterations need to be performed: "))
x1=x0+h
y1E=y0+h*f(x0,y0)
print("\nInitial guess by eulers method is x=%0.2f y=%0.4f"%(x1,y1E))
print("By modified eulers method:")
print(f"Iteration\t\ty1({x1})")
for i in range(n):
    y1=y0+(h/2)*(f(x0,y0)+f(x1,y1E))
    print(i+1,"\t\t\t%.4f"%y1)
    if abs(y1-y1E)<0.0001:
        break
    else:
        y1E=y1
print(f"\ny({x1})=%.4f"%y1)
```

```
Enter max number of iterations need to be performed: 2

Initial guess by eulers method is x=0.20 y=-0.8000
By modified eulers method:
Iteration               y1(0.2)
1                       -0.8181
2                       -0.8181

y(0.2)=-0.8181
```

In [4]:

```python
#runge kutta
f=lambda x,y: x-y**2
x0,y0=0,1
h=0.1
x1=x0+h
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)
print("\nk1=%0.4f\tk2=%0.4f\tk3=%0.4f\tk4=%0.4f"%(k1,k2,k3,k4))
print("y(%0.2f)=%.4f"%(x1,y1))
```

```
k1=-0.1000      k2=-0.0852      k3=-0.0867      k4=-0.0734
y(0.10)=0.9138
```

In [5]:

```
#milnes predictor and corrector
func=lambda x,y:x**2+(y/2)
x=[1,1.1,1.2,1.3]
y=[2,2.2156,2.4649,2.7514]
h=0.1
f=[]
for i in range(4):
    f.append(func(x[i],y[i]))
    print(f"x{i}=%.4f\ty{i}=%.4f\tf{i}=%0.4f"%(x[i],y[i],f[i]))
#predict y4=y(x4)
x4=x[3]+h
y4p=y[0]+(4*h/3)*(2*f[1]-f[2]+2*f[3])
print("\nThe predicted value at x4=%.4f is y4p=%.5f\n"%(x4,y4p))
#correction
for i in range(3):
    f4p=func(x4,y4p)
    y4c=y[2]+(h/3)*(f[2]+4*f[3]+f4p)
    y4p=y4c
    print(f"the corrected value at x4=%.4f in iteration {i+1} is y4c{i+1}=%.5f"
          %(x4,y4c))
```

```
x0=1.0000       y0=2.0000       f0=2.0000
x1=1.1000       y1=2.2156       f1=2.3178
x2=1.2000       y2=2.4649       f2=2.6724
x3=1.3000       y3=2.7514       f3=3.0657

The predicted value at x4=1.4000 is y4p=3.07927

the corrected value at x4=1.4000 in iteration 1 is y4c1=3.07940
the corrected value at x4=1.4000 in iteration 2 is y4c2=3.07940
the corrected value at x4=1.4000 in iteration 3 is y4c3=3.07940
```

In [ ]: