```
import sympy as sp

x,y = sp.symbols ('x, y')
f= sp.exp(x)*sp.sin(y)+(y**3)
df_dx= sp.diff(f, x)
df_dy= sp.diff(f, y)

print ("df_dx=", df_dx,"     ", "df_dx=",df_dy)
```

```
df_dx= exp(x)*sin(y)       df_dx= 3*y**2 + exp(x)*cos(y)
```

```
x,y = sp.symbols ('x, y')
f= sp.ln((x**2)+(y**2))
df_dx= sp.diff(f, x)
df_dy= sp.diff(f, y)


df_dxx= sp.diff(df_dx, x)
df_dyy= sp.diff(df_dy, y)
df_dxy= sp.diff(df_dx, y)
df_dyx= sp.diff(df_dy, x)

print ("df_dxx=", df_dxx,"    ", "df_dyy",df_dyy, "     ", "df_dxy=", df_dxy, "     ", "df_dyx", df_dyx)
```

```
df_dxx= -4*x**2/(x**2 + y**2)**2 + 2/(x**2 + y**2)      df_dyy -4*y**2/(x**2 + y**2)**2 + 2/(x**2 + y**2)      df_dxy= -4*x*y/(x**2 + y**2)**2      df_dyx -4*x*y//(x**2 + y**2)**2
```

```
x0= 1
y0= -1
alpha= 0.01
num_iterations= 100

def gradient_descent(x0, y0,f, grad_f, alpha, num_iterations):
    x = x0
    y = y0

    for i in range(num_iterations):
        grad_x, grad_y =  grad_f(x,y)
        x = x- (alpha*grad_x)
        y = y- (alpha*grad_y)
    return x, y

def f(x,y):
    f= ((x**2)*y)+(x*(y**2))

def grad_f(x,y):
    grad_x = 2*x*y + y**2
    grad_y = x**2 + 2*x*y

    return grad_x, grad_y

values= gradient_descent(x0, y0,f, grad_f, alpha, num_iterations)
print("x and y:", values)
```

```
x and y: (3.8979383461689086, -1.4938261773981383)
```

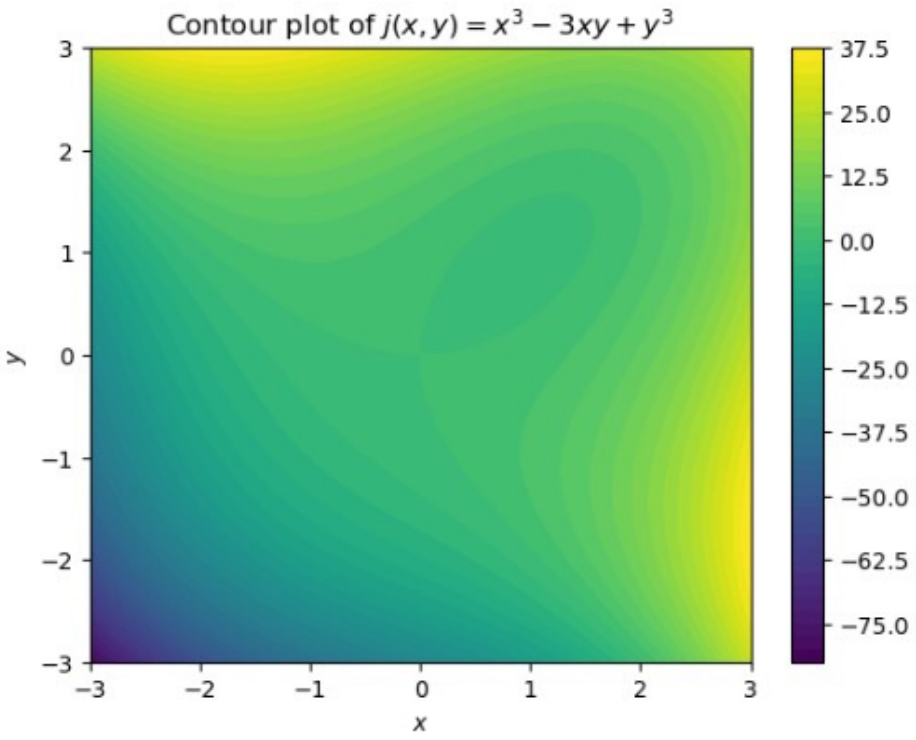
```python
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt

x, y = sp.symbols('x y')
j = x**3 - 3*x*y + y**3

j_func = sp.lambdify((x, y), j, 'numpy')

x_vals = np.linspace(-3, 3, 400)
y_vals = np.linspace(-3, 3, 400)
X, Y = np.meshgrid(x_vals, y_vals)
Z = j_func(X, Y)

plt.contourf(X, Y, Z, levels=50, cmap='viridis')
plt.colorbar()
plt.title('Contour plot of $j(x, y) = x^3 - 3xy + y^3$')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.show()
```



```
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt

x, y = sp.symbols('x y') #y^2 + x^2
j = y**2 + x**2

j_func = sp.lambdify((x, y), j, 'numpy')

x_vals = np.linspace(-3, 3, 400)
y_vals = np.linspace(-3, 3, 400)
X, Y = np.meshgrid(x_vals, y_vals)
Z = j_func(X, Y)

plt.contourf(X, Y, Z, levels=50, cmap='viridis')
plt.colorbar()
plt.title('Contour plot of $j(x, y) = y^2 + x^2 $')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.show()
```