Daniyal Sheikh UCID: 30144209
ENEL 102 Assignment 2
October 22, 2023

Source Code:

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

# Define the variable and the function
x = sp.Symbol('x')
f = x * sp.sin(x)

# Compute the definite integral
result = sp.integrate(f, (x, 0, sp.pi/2))

print(result)



#Question 2
a, b = sp.symbols('a b')

A = sp.Matrix([[1, 2], [a, b]])

A_inverse = A.inv()

A_times_A_inverse = A * A_inverse

identity_matrix = sp.eye(2)

print("Matrix A:")
sp.pprint(A)

print("Matrix A^-1:")
sp.pprint(A_inverse)

print("Matrix A * A^-1:")
sp.pprint(A_times_A_inverse)

print("2x2 Identity Matrix:")
sp.pprint(identity_matrix)

is_identity = A_times_A_inverse.equals(identity_matrix)

if is_identity:
    print("A * A^-1 is the 2x2 identity matrix.")
```

Daniyal Sheikh UCID: 30144209
ENEL 102 Assignment 2
October 22, 2023

```
else:
    print("A * A^-1 is NOT the 2x2 identity matrix.")



#Question 3

def coefficient_of_xn_minus_1(N):
    if N < 1:
        return 0  # Invalid input, return 0

    x = sp.symbols('x')
    expanded_expression = (x + 1)**N

    # Expand the expression and extract the coefficient of x^(N-1)
    expanded_expression = sp.expand(expanded_expression)
    coefficient_xn_minus_1 = expanded_expression.coeff(x, N-1)

    return coefficient_xn_minus_1, expanded_expression

# Test the function for N = 4
N = 4
result = coefficient_of_xn_minus_1(N)
print(f"The coefficient of x^{N-1} in (x + 1)^{N} is: {result}")

#Question 4

def nth_derivative(N, x):
    # Define the symbolic variable x
    x = sp.symbols('x')

    # Define the original function f(x)
    f_x = x**3 * sp.sin(x**2 + 1)

    # Calculate the N-th derivative of the function
    nth_deriv = sp.diff(f_x, x, N)

    # Create a numeric function from the symbolic expression
    numeric_function = sp.lambdify(x, nth_deriv, 'numpy')

    # Evaluate the numeric function at the specified x_value
    result = numeric_function(x_value)

    return result

# Test the function for N = 2 and x = 0.1
```

Daniyal Sheikh UCID: 30144209
ENEL 102 Assignment 2
October 22, 2023

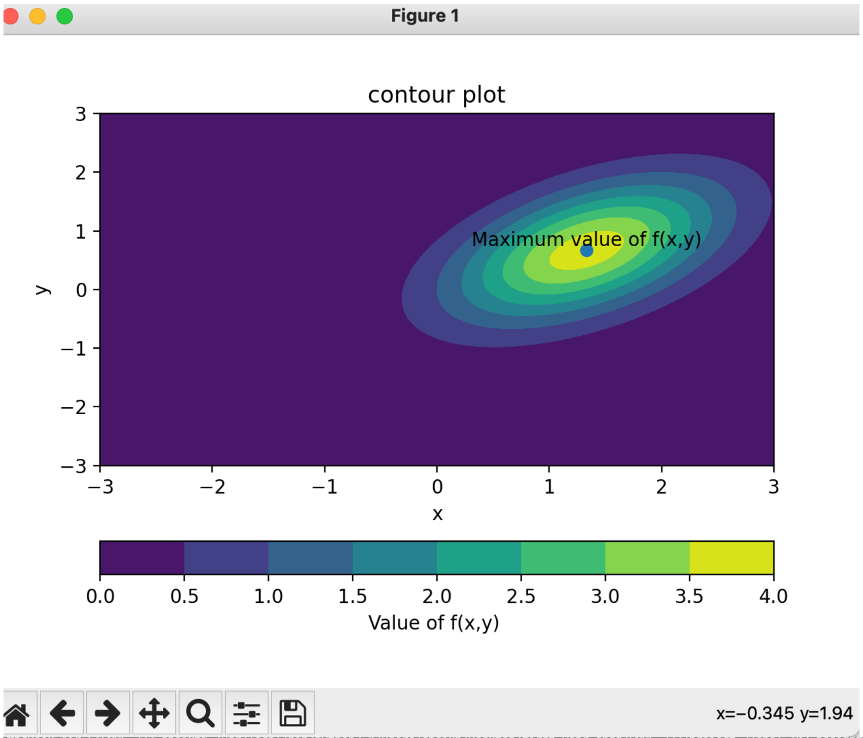
```
N = 2
x_value = 0.1
result = nth_derivative(N, x_value)

print(f"The {N}-th derivative at x = {x_value} is: {result}")


#Question 5
x = sp.symbols('x')

f = sp.cos(x**2 + sp.sqrt(x))


power_series = sp.series(f, x, 0, 6)  # The last argument '6' represents x^5 term

sympy.pprint(power_series)

#Question 6
x, alpha = sympy.symbols('x alpha', positive=True, real=True)

f = x**2 * sympy.exp(-alpha * x)

# Calculate the derivative of f(x) with respect to x
f_prime = sympy.diff(f, x)

# Find the critical points by solving f'(x) = 0
critical_points = sympy.solve(f_prime, x)

# Check the second derivative to determine if it's a maximum
maximum_points = []
for point in critical_points:
    f_double_prime = sympy.diff(f_prime, x).subs(x, point)
    if f_double_prime < 0:  # Negative second derivative indicates a maximum
        maximum_points.append(point)

maximum_values = [f.subs(x, point) for point in maximum_points]

if not maximum_values:
    print("No maximum found.")
else:
    max_x, max_value = max(zip(maximum_points, maximum_values), key=lambda x: x[1])
    print(f"The maximum of f(x) is {max_value} at x = {max_x}")
```

Daniyal Sheikh UCID: 30144209
ENEL 102 Assignment 2
October 22, 2023

```
#Question 7 A

# Define symbolic variables
x, y = sympy.symbols('x y')
fxy = sp.exp(-x**2+2*x-y**2+x*y)
#find the derivative
fxy_diffx = sp.diff(fxy, x)
fxy_diffy = sp.diff(fxy, y)
local_max = sp.solve([fxy_diffx, fxy_diffy], [x,y], dict=True)

print("The local maximums are : ", local_max)

#Question 7 B
fxy_num = sp.lambdify((x, y) , fxy) # numeric equation in numpy

x_val = np.linspace(-3,3,300)
y_val= np.linspace(-3,3,300)
x_mesh, y_mesh = np.meshgrid(x_val,y_val)

f_mesh = fxy_num(x_mesh, y_mesh)

quadcontset = plt.contourf(x_mesh, y_mesh,f_mesh)
plt.colorbar(label="Value of f(x,y) ", orientation="horizontal")
x_maxval = local_max[0][x]
y_maxval = local_max[0][y]

plt.scatter(x_maxval, y_maxval)
plt.text(x_maxval, y_maxval, "Maximum value of f(x,y)", va="bottom", ha="center")

plt.xlabel("x")
plt.ylabel("y")
plt.title("contour plot")

plt.show()
```

Outputs & Graphs

Daniyal Sheikh UCID: 30144209
ENEL 102 Assignment 2
October 22, 2023




```
MacBook-Air-5:ENDG 233 sheikh$ /usr/local/bin/python3 "/Users/sheikh/Desktop/first year engg/ENDG 233/project 2/enel_102_assignment_2.py"
1
Matrix A:
⎡1  2⎤
⎢    ⎥
⎣a  b⎦
Matrix A^-1:
⎡    b          -2    ⎤
⎢────────  ────────⎥
⎢-2·a + b  -2·a + b⎥
⎢                  ⎥
⎢   -a          1     ⎥
⎢────────  ────────⎥
⎣-2·a + b  -2·a + b⎦
Matrix A * A^-1:
⎡    2·a          b                                  ⎤
⎢- ──────── + ────────              0               ⎥
⎢  -2·a + b   -2·a + b                               ⎥
⎢                                                    ⎥
⎢                               2·a          b       ⎥
⎢          0               - ──────── + ────────    ⎥
⎣                            -2·a + b   -2·a + b    ⎦
2x2 Identity Matrix:
⎡1  0⎤
⎢    ⎥
⎣0  1⎦
A * A^-1 is the 2x2 identity matrix.
The coefficient of x^3 in (x + 1)^4 is: (4, x**4 + 4*x**3 + 6*x**2 + 4*x + 1)
The 2-th derivative at x = 0.1 is: 0.5155112835962654
          2     3          4            5            7/2    9/2    11/2
    x    x     x    20159·x     907199·x     5/2   x       x       x        ⎛ 6⎞
1 - ─ + ── - ─── - ──────── + ────────── - x    + ──── - ──── + ──── + O⎝x ⎠
    2   24   720     40320       3628800              6     120     5040
The maximum of f(x) is 4*exp(-2)/alpha**2 at x = 2/alpha
The local maximums are :  [{x: 4/3, y: 2/3}]
```