**b) Using Python to achieve**

**i. Differentiation**

import sympy as sp

x = sp.symbols('x')

f = x\*\*3 + x\*\*2 + x + 1

f\_prime = sp.diff(f, x)

print(f\_prime)

**ii. Numerical Integration**

import scipy.integrate as spi

import numpy as np

f = lambda x: x\*\*3 + x\*\*2 + x + 1

result, \_ = spi.quad(f, 0, 1)

print(result)

**iii. Curve Fitting**

import numpy as np

import matplotlib.pyplot as plt

from scipy.optimize import curve\_fit

# Sample data

x\_data = np.array([1, 2, 3, 4, 5])

y\_data = np.array([1, 4, 9, 16, 25])

# Model function

def model(x, a, b):

return a \* x\*\*2 + b

params, \_ = curve\_fit(model, x\_data, y\_data)

a, b = params

print(f'a: {a}, b: {b}')

**iv. Linear Regression**

import numpy as np

from sklearn.linear\_model import LinearRegression

# Sample data

x = np.array([[1], [2], [3], [4], [5]])

y = np.array([1, 2, 3, 4, 5])

model = LinearRegression().fit(x, y)

print(f'Slope: {model.coef\_[0]}, Intercept: {model.intercept\_}')

**v. Spline Interpolation**

import numpy as np

from scipy.interpolate import CubicSpline

import matplotlib.pyplot as plt

x = np.array([0, 1, 2, 3, 4, 5])

y = np.array([0, 1, 4, 9, 16, 25])

cs = CubicSpline(x, y)

x\_new = np.linspace(0, 5, 100)

y\_new = cs(x\_new)

plt.plot(x, y, 'o', label='data')

plt.plot(x\_new, y\_new, '-', label='spline')

plt.legend()

plt.show()