A red sign with white text

Description automatically generated

**P8. Matemáticas Avanzadas en Python: NumPy y SymPy**

Alumna: Daniela Mendez Ramirez

Número de Cuenta: 258331-9

**Aplicaciones de Redes**

Profesor: Omar Vázquez González

Fecha de Entrega: 16 de febrero de 2025

Ejercicios para entender el funcionamiento de simPy.

|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir la variable simbólica  x = sp.Symbol('x')  # Definir la función a derivar  f = x\*\*3 + 2\*x\*\*2 + x + 5  # Calcular la derivada  dfdx = sp.diff(f, x)  # Mostrar el resultado  print(dfdx)  # Salida: 3\*x\*\*2 + 4\*x + 1 |  |

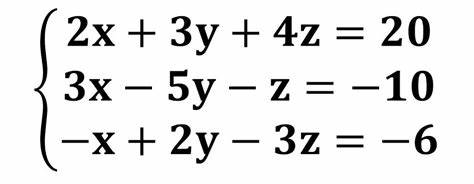
|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir las variables simbólicas  x, y, z = sp.symbols('x y z')  # Definir las ecuaciones  eq1 = sp.Eq(2\*x + y - z, 3)  eq2 = sp.Eq(x + 3\*y + 2\*z, 12)  eq3 = sp.Eq(3\*x - y + 4\*z, 10)  # Resolver el sistema  solucion = sp.solve((eq1, eq2, eq3), (x, y, z))  # Mostrar el resultado  print(solucion)  # Devuelve un diccionario con los valores de x, y, z |  |

|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir la variable  x = sp.Symbol('x')  # Definir la ecuación  ecuacion = sp.Eq(x\*\*2 - 5\*x + 6, 0)  # Resolver la ecuación  solucion = sp.solve(ecuacion, x)  # Mostrar el resultado  print("Solución de la ecuación:", solucion) |  |

|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir la variable  x = sp.Symbol('x')  # Definir la ecuación exponencial  ecuacion = sp.Eq(2\*\*x, 16)  # Resolver la ecuación  solucion = sp.solve(ecuacion, x)  # Mostrar el resultado  print("Solución de la ecuación exponencial:", solucion) |  |

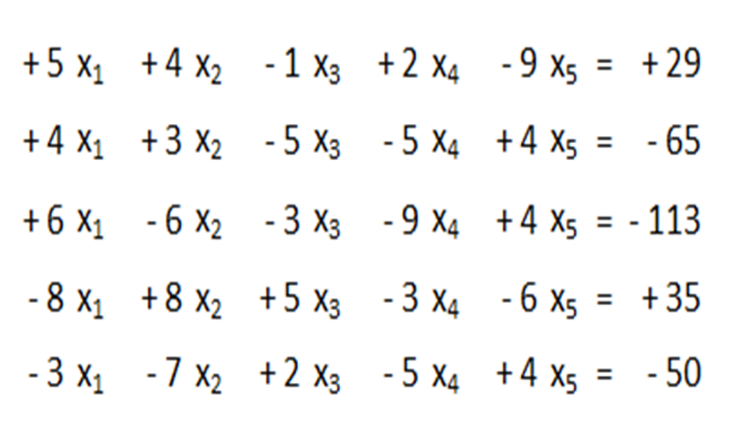
|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir la variable y la función  x = sp.Symbol('x')  f = x\*\*3 + 2\*x\*\*2 - 3\*x + 5  # Calcular la primera derivada  primera\_derivada = sp.diff(f, x)  # Calcular la segunda derivada  segunda\_derivada = sp.diff(primera\_derivada, x)  # Mostrar los resultados  print("Primera derivada:", primera\_derivada)  print("Segunda derivada:", segunda\_derivada) |  |

1. Resolver operación.



|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir las variables simbólicas  x, y, z = sp.symbols('x y z')  # Definir las ecuaciones  eq1 = sp.Eq(2\*x + 3\*y + 4\*z, 20)  eq2 = sp.Eq(3\*x - 5\*y - z, -10)  eq3 = sp.Eq(-x + 2\*y - 3\*z, -6)  # Resolver el sistema  solucion = sp.solve((eq1, eq2, eq3), (x, y, z))  # Mostrar el resultado  print(solucion)  # Devuelve un diccionario con los valores de x, y, z |  |

2. Resolver operación.



|  |  |
| --- | --- |
| Código | Resultado |
| import sympy as sp  # Definir las variables  x1, x2, x3, x4, x5 = sp.symbols('x1 x2 x3 x4 x5')  # Definir las ecuaciones  eq1 = sp.Eq(5\*x1 + 4\*x2 - x3 + 2\*x4 - 9\*x5, 29)  eq2 = sp.Eq(4\*x1 + 3\*x2 - 5\*x3 - 5\*x4 + 4\*x5, -65)  eq3 = sp.Eq(6\*x1 - 6\*x2 - 3\*x3 - 9\*x4 + 4\*x5, -113)  eq4 = sp.Eq(-8\*x1 + 8\*x2 + 5\*x3 - 3\*x4 - 6\*x5, 35)  eq5 = sp.Eq(-3\*x1 - 7\*x2 + 2\*x3 - 5\*x4 + 4\*x5, -50)  # Resolver el sistema de ecuaciones  solucion = sp.solve((eq1, eq2, eq3, eq4, eq5), (x1, x2, x3, x4, x5))  # Mostrar la solución  print("Solución del sistema:")  for var, val in solucion.items():      print(f"{var} = {val}") |  |

Conclusión

La librería SymPy es una herramienta poderosa para el cálculo simbólico en Python, que permite manipular expresiones algebraicas, resolver ecuaciones y simplificar expresiones de manera eficiente. Es especialmente útil en áreas como matemáticas, física y química. Una de sus principales ventajas sobre MATLAB es que es completamente gratuita y de código abierto, lo que la hace más accesible. Además, al estar integrada con Python, ofrece la flexibilidad de combinar cálculos simbólicos con otras herramientas y bibliotecas, como NumPy y Pandas, lo que facilita tareas multidisciplinarias. Si bien MATLAB es más eficiente para cálculos numéricos complejos y tiene funciones especializadas en áreas como ingeniería o procesamiento de señales, SymPy es más práctica para trabajos que requieren álgebra simbólica y es ideal para proyectos pequeños o de código abierto.