**Learning Scientific Computing**

Follow all the instructions below and Have Fun! 😊

1. Given the **equations** below, make a **program** that can **print all** the **process** of determining the **equations** is **Diagonally** **Dominant** or not.

Graphical user interface, text, application

Description automatically generated

**Figure 1 Diagonally Dominant Check Result**

From those **Equations**, here is a **snippet code** you can **use** for the **program**!

mX = [

        [

            [10, 2, -6],

            [-1, 4, 2],

            [2, 3, 6]

        ],

        [

            [-5, 2, -8],

            [4, -3, -9],

            [-8, -10, 9]

        ],

        [

            [10, -5, 3, 0],

            [3, -9, 2, -2],

            [-8, 0, 17, -4],

            [3, -3, -8, -15]

        ],

]

mY = [

    [17, -4, -12],

    [13, -7, -2],

    [9, 10, -10, -18]

]

1. From the previous number’s **equations**, you will need to **find** each **value** of the **variables** x1, x2, … (maksudnya **a, b, c** dan sebagainya). Use **Gauss** **Seidel** method to **find** the **value** of the **variables** with **maximum** **iteration** of **5** and **threshold** **456.** Here are some requirements that you must include in your program:
   1. Make sure you only **search** the **value** from the **equation** that are **Diagonally** **Dominant**. If there is an **equation** that **is not** **Diagonally** **Dominant**, show an **error** **message**.
   2. You will **need** to **print** the formula of each **variable** and each iteration result.
   3. If in between the **maximum iteration**, the program **found** **out** the **result**, print **“Converged!”** and **show** **all** the **result** with the **format** like the **example** **result** in **figure** **2**. If **not** **found** in **between** of the **maximum** **iteration**, then print **“Not Converged, need more iteration!”.**

Text

Description automatically generated

**Figure 2 Gauss Seidel Example Result**

1. You are given **10 point** in **(x,y)** **format** below. You have a duty is to make a **program** that can **calculate** the **Least Square Regression** of **those** **points** and **show** a **graphic** of **result** using **matplotlib**. The **graphic** must have **several** **requirements**:
   1. Add **title**, **xLabel**, **yLabel**, and **legend** to the **graphic**.
   2. The **figure** **size** must be **12 x 8**.
   3. The **point** and the **line** of **Least** **Square** **Regression** must have **different** **colors**.

x = [1, 5, 7, 2, 3, 9, 10, 4, 6, 8]

y = [34, 51, 80, 25, 90, 76, 100, 11, 45, 62]

Chart, scatter chart

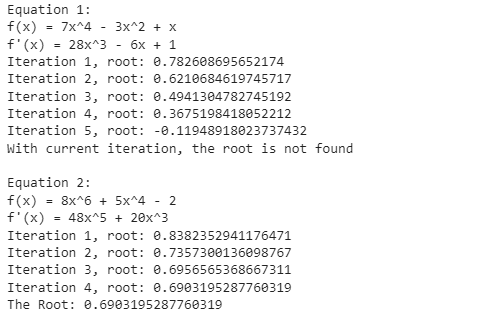
Description automatically generated

**Figure 3 Least Square Regression Graphic Result**

1. You have given some equations f(x) below:

You need to implement Newton Raphson in your program to find the root of each equation. You must include some requirements for your program:

* 1. The  **formula must** be automatically **generated** by your program (ini **bonus** aja, kalau mau langsung diturunin boleh)
  2. **First guess** **value** is **1**.
  3. The **tolerance** **value** is **0.006541**.
  4. Set **max** **iterations** to **5**.
  5. Print the and **formula** that you **generated** (ini dilakuin kalau kerjain **point a** aja).
  6. If the **iteration** is **over** the **max** **iterations**, print an **error** **message**.
  7. Print the **root** if the **program** **finds** it in **between** the **maximum** **iteration**.



**Figure 4 Root Finding Result**

1. Find the **result** of using **Left Riemann**, **Right Riemann**, **Mid-Point Riemann**, and **Trapezoid** with **30 evenly spaced grid points** over the **whole interval** and then **print** the **result**.