

ACTIVIDAD N°03

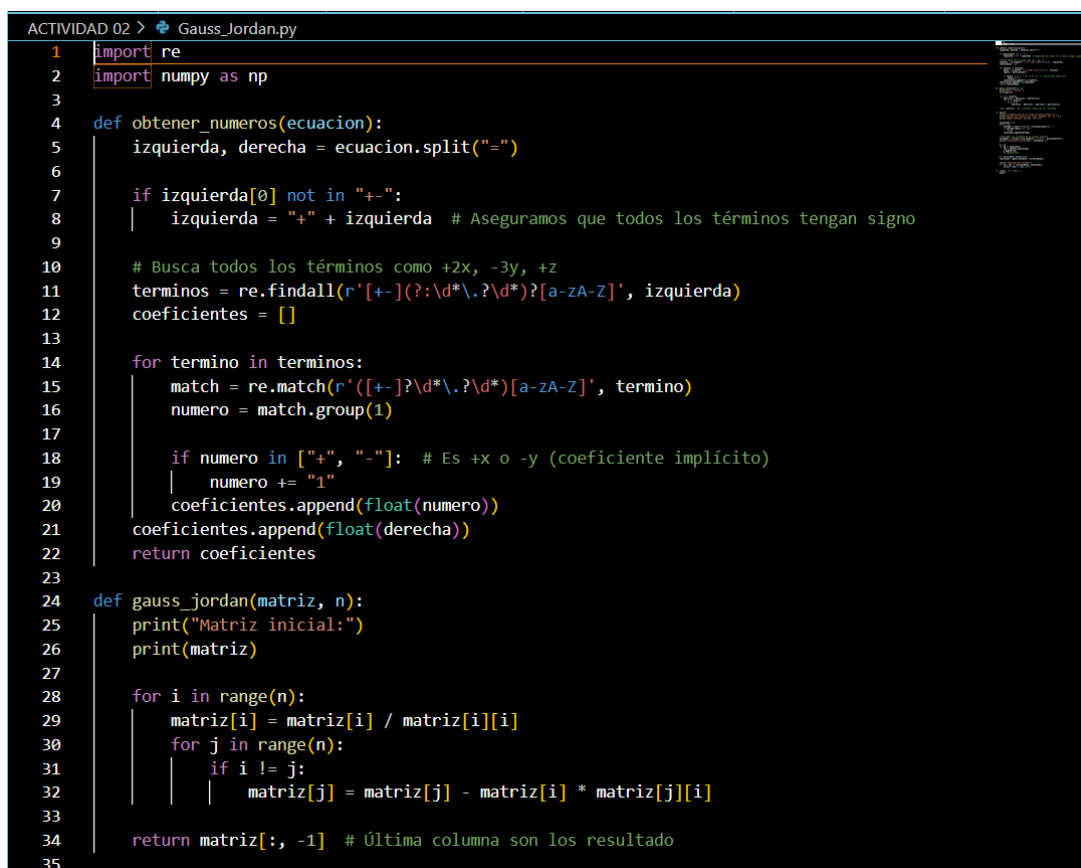
Adjuntar el Método Visual de la solución desarrollada. (FastHTML, Shiny, Flask)

IMPLEMENTACIÓN DEL MÉTODO DE GAUSS-JORDAN EN FLASK

Se presenta el desarrollo de la solución del método de **Gauss-Jordan** implementado en **Flask**, framework ligero para el desarrollo web en Python. Se adjunta tanto el código con los resultados en consola, la versión con interfaz gráfica y la visualización en el navegador.

Código con Resultados en Consola

Se muestra una captura del código fuente utilizado para resolver el método de Gauss-Jordan directamente en consola:



```
ACTIVIDAD 02 > Gauss_Jordan.py
1 import re
2 import numpy as np
3
4 def obtener_numeros(ecuacion):
5     izquierda, derecha = ecuacion.split("=")
6
7     if izquierda[0] not in "+-":
8         izquierda = "+" + izquierda # Aseguramos que todos los términos tengan signo
9
10    # Busca todos los términos como +2x, -3y, +z
11    terminos = re.findall(r'([+-]?[d*\.\d*])[a-zA-Z]', izquierda)
12    coeficientes = []
13
14    for termino in terminos:
15        match = re.match(r'([+-]?[d*\.\d*])[a-zA-Z]', termino)
16        numero = match.group(1)
17
18        if numero in ["+", "-"]: # Es +x o -y (coeficiente implícito)
19            numero += "1"
20        coeficientes.append(float(numero))
21    coeficientes.append(float(derecha))
22    return coeficientes
23
24 def gauss_jordan(matriz, n):
25     print("Matriz inicial:")
26     print(matriz)
27
28     for i in range(n):
29         matriz[i] = matriz[i] / matriz[i][i]
30         for j in range(n):
31             if i != j:
32                 matriz[j] = matriz[j] - matriz[i] * matriz[j][i]
33
34     return matriz[:, -1] # Última columna son los resultados
35
```

```

ACTIVIDAD 02 > Gauss_Jordan.py
36 def main():
37     print("=== Resolución por el método de Gauss-Jordan ===")
38     print("Ingresa una ecuación por línea. Ejemplo: 2x + 3y = 9")
39     print("Cuando termines, escribe 'fin'\n")
40
41     ecuaciones = []
42     while True:
43         entrada = input(f"Ecuación {len(ecuaciones)+1}: ")
44         if entrada.lower() == 'fin':
45             break
46         ecuaciones.append(entrada)
47
48     # Extraemos las variables de la primera ecuación
49     variables = sorted(set(re.findall(r'[a-zA-Z]', ecuaciones[0])))
50     print(f"\nVariables encontradas: {variables}\n")
51
52     A = []
53     for eq in ecuaciones:
54         res = obtener_numeros(eq)
55         A.append(res)
56         #A.append(coef)
57
58     A = np.array(A, dtype=float)
59     soluciones = gauss_jordan(A, len(variables))
60
61     print("\nSolución del sistema:")
62     for var, sol in zip(variables, soluciones):
63         print(f"{var} = {sol:.2f}")
64
65 if __name__ == "__main__":
66     main()

```

Código con Interfaz Gráfica

- Archivo: *app.py*

```

ACTIVIDAD 03 > app.py
1  import re
2  import numpy as np
3  from flask import Flask, render_template, request, redirect, url_for
4
5  app = Flask(__name__)
6
7  def obtener_numeros(ecuacion):
8      izquierda, derecha = ecuacion.split("=")
9
10     if izquierda[0] not in "+-":
11         izquierda = "+" + izquierda
12
13     terminos = re.findall(r'([+-]?[0-9]*\.?[0-9]*)[a-zA-Z]', izquierda)
14     coeficientes = []
15
16     for termino in terminos:
17         match = re.match(r'([+-]?[0-9]*\.?[0-9]*)[a-zA-Z]', termino)
18         numero = match.group(1)
19
20         if numero in ["+", "-"]:
21             numero += "1"
22         coeficientes.append(float(numero))
23     coeficientes.append(float(derecha))
24     return coeficientes
25
26 def gauss_jordan_pasos(matriz, n):
27     pasos = []
28     matriz = matriz.copy()
29
30     for i in range(n):
31         if matriz[i][i] == 0:
32             raise ValueError("División por cero: hay una columna pivotar con valor 0")
33
34         matriz[i] = matriz[i] / matriz[i][i]
35         pasos.append(f"Normalizando fila {i+1}", matriz.copy())
36

```

```

ACTIVIDAD 03 > app.py
26 def gauss_jordan_pasos(matriz, n):
27     for i in range(n):
28         for j in range(n):
29             if i != j:
30                 matriz[j] = matriz[j] - matriz[i] * matriz[j][i]
31             pasos.append((f"Eliminando columna {i+1} de fila {j+1}", matriz.copy()))
32
33     return matriz[:, -1], pasos
34
35 @app.route("/", methods=["GET", "POST"])
36 def index():
37     if request.method == "POST":
38         try:
39             num_ecuaciones = int(request.form.get("num_ecuaciones"))
40             ecuaciones = [request.form.get(f"eq_{i+1}") for i in range(num_ecuaciones)]
41
42             variables = sorted(set(re.findall(r'[a-zA-Z]', ecuaciones[0])))
43             A = [obtener_numeros(eq) for eq in ecuaciones]
44             A = np.array(A, dtype=float)
45
46             soluciones, pasos = gauss_jordan_pasos(A, len(variables))
47
48             emparejados = list(zip(variables, soluciones))
49             return render_template("resultado.html", emparejados=emparejados, pasos=pasos)
50
51         except Exception as e:
52             return render_template("index.html", error=str(e))
53
54     return render_template("index.html")
55
56 if __name__ == "__main__":
57     app.run(debug=True)
58

```

■ Archivo: *index.html*

```

ACTIVIDAD 03 > templates > index.html > html > body
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <title>Gauss-Jordan</title>
6     <script>
7         function generarCampos() {
8             const num = document.getElementById("num_ecuaciones").value;
9             const contenedor = document.getElementById("ecuaciones");
10            contenedor.innerHTML = "";
11            for (let i = 1; i <= num; i++) {
12                contenedor.innerHTML += `<label for="eq_${i}">Ecuación ${i}</label>
13                <input type="text" name="eq_${i}" required><br><br>`;
14            }
15        }
16    </script>
17 </head>
18 <body>
19     <h1>Método de Gauss-Jordan</h1>
20     <p>Resuelve un sistema de ecuaciones lineales</p>
21
22     {% if error %}
23     <p style="color: red;">Error: {{ error }}</p>
24     {% endif %}
25
26     <form method="POST">
27         <label for="num_ecuaciones">Número de ecuaciones:</label>
28         <input type="number" id="num_ecuaciones" name="num_ecuaciones" min="2" max="10" value="">
29         <div id="ecuaciones">
30             <label for="eq_1">Ecuación 1:</label>
31             <input type="text" name="eq_1" required><br><br>
32             <label for="eq_2">Ecuación 2:</label>
33             <input type="text" name="eq_2" required><br><br>
34         </div>
35         <button type="submit">Resolver</button>
36     </form>
37 </body>
38 </html>

```

■ Archivo: *resultado.html*

```

ACTIVIDAD 03 > templates > resultado.html > html > body > ul
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4  |   <meta charset="UTF-8">
5  |   <title>Resultado Gauss-Jordan</title>
6  </head>
7  <body>
8  |   <h1>Resultado del sistema</h1>
9  |
10 |   <h2>Soluciones:</h2>
11 |   <ul>
12 |       {% for var, sol in emparejados %}
13 |       <li><strong>{{ var }} = {{ "%.2f"|format(sol) }}</strong></li>
14 |       {% endfor %}
15 |   </ul>
16 |
17 |   <h2>Pasos intermedios:</h2>
18 |   {% for descripcion, paso in pasos %}
19 |   <h3>{{ descripcion }}</h3>
20 |   <table border="1">
21 |       {% for fila in paso %}
22 |       <tr>
23 |           {% for valor in fila %}
24 |           <td>{{ "%.3f"|format(valor) }}</td>
25 |           {% endfor %}
26 |       </tr>
27 |       {% endfor %}
28 |   </table>
29 |   {% endfor %}
30 |
31 |   <br><a href="{{ url_for('index') }}">Volver al inicio</a>
32 |
33 </body>
34 </html>
35

```

Visualización de la Interfaz Gráfica

Captura de la interfaz web tal como se visualiza en el navegador.



Resultado Gauss-Jordan

127.0.0.1:5000

Resultado del sistema

Soluciones:

- $x = 19.00$
- $y = -11.00$

Pasos intermedios:

Normalizando fila 1

1.000	1.500	2.500
1.000	1.000	8.000

Eliminando columna 1 de fila 2

1.000	1.500	2.500
0.000	-0.500	5.500

Normalizando fila 2

1.000	1.500	2.500
-0.000	1.000	-11.000

Eliminando columna 2 de fila 1

1.000	0.000	19.000
-0.000	1.000	-11.000

[Volver al inicio](#)