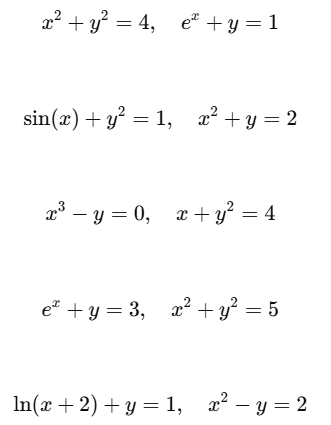
**Actividad 4: Resolución de Sistemas de Ecuaciones No Lineales.**

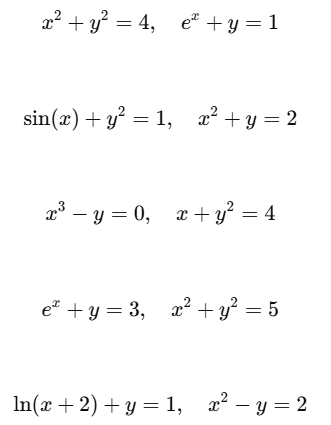
1. **Ejercicios y Desarrollo**

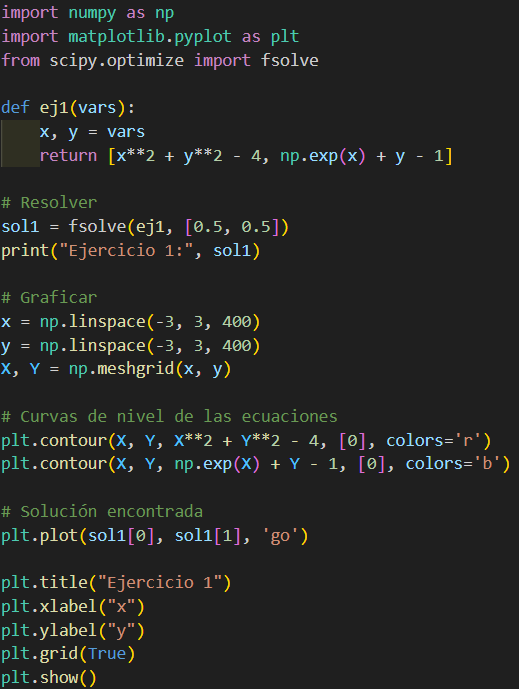
Se plantearon 10 sistemas de ecuaciones no lineales, 5 con dos variables y 5 con tres variables. Para cada sistema definir las funciones correspondientes en *Python*, utilizar la función *fsolve* de la librería *scipy*. En el caso de los sistemas con dos variables realizar gráficas de las curvas que representan las ecuaciones, de manera que las soluciones aparezcan como puntos de intersección.

1. **Sistemas de Ecuaciones No lineales de 2 variables**

** **

1. **Realizar Pseudocódigo**
2. **Codificar en Python**

****

****

**INICIO**

**FUNCIÓN ej1(x, y):**

**ecuacion1 = x^2 + y^2 - 4**

**ecuacion2 = e^x + y - 1**

**RETORNAR (ecuacion1, ecuacion2)**

**# --- Resolver ---**

**aproximacion\_inicial = (0.5, 0.5)**

**solucion ← fsolve(ej1, aproximacion\_inicial)**

**IMPRIMIR "Ejercicio 1:", solucion**

**# --- Graficar ---**

**CREAR rango\_x desde -3 hasta 3 con 400 puntos**

**CREAR rango\_y desde -3 hasta 3 con 400 puntos**

**CREAR malla (X, Y) con los rangos**

**DIBUJAR curva de nivel de (X^2 + Y^2 - 4 = 0) en color rojo**

**DIBUJAR curva de nivel de (exp(X) + Y - 1 = 0) en color azul**

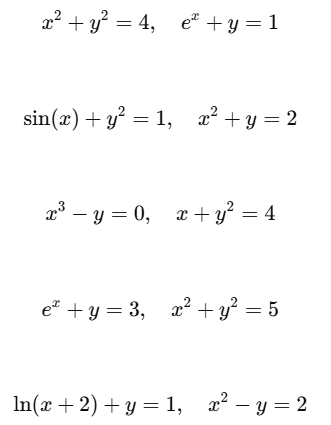
**DIBUJAR punto (solucion\_x, solucion\_y) en color verde**

**AGREGAR título "Ejercicio 1"**

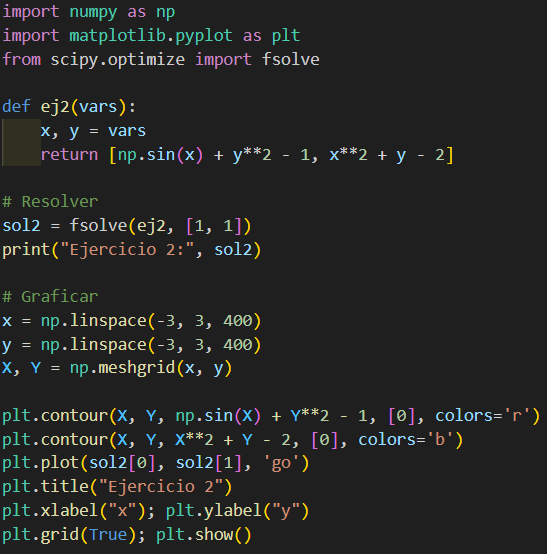
**AGREGAR etiquetas de ejes (x, y)**

**MOSTRAR la gráfica**

**FIN**

****

**INICIO**

**FUNCIÓN ej2(x, y):**

**ecuacion1 ← sen(x) + y^2 - 1**

**ecuacion2 ← x^2 + y - 2**

**RETORNAR (ecuacion1, ecuacion2)**

**# --- Resolver ---**

**aproximacion\_inicial ← (1, 1)**

**solucion ← fsolve(ej2, aproximacion\_inicial)**

**IMPRIMIR "Ejercicio 2:", solucion**

**# --- Graficar ---**

**definir rango\_x desde -3 hasta 3 con 400 puntos**

**definir rango\_y desde -3 hasta 3 con 400 puntos**

**crear malla (X, Y) con los rangos**

**trazar curva de nivel de (sen(X) + Y^2 - 1 = 0) en color rojo**

**trazar curva de nivel de (X^2 + Y - 2 = 0) en color azul**

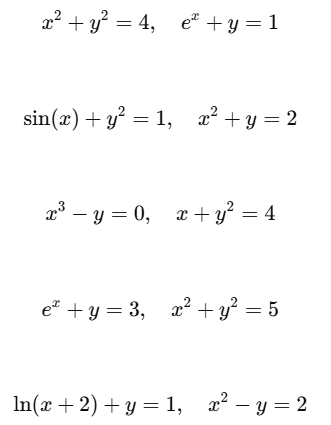
**marcar punto (solucion\_x, solucion\_y) en color verde**

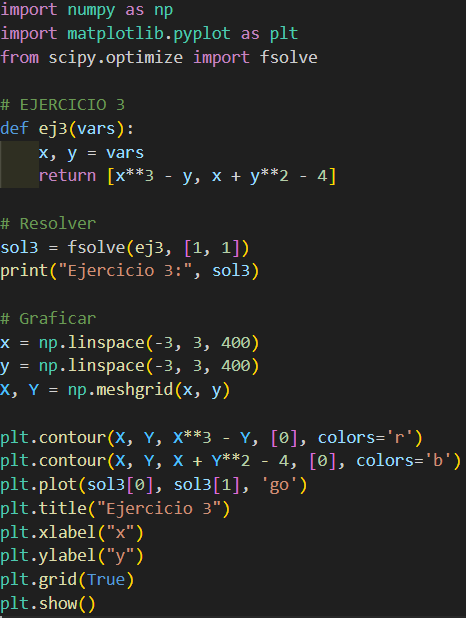
**agregar título "Ejercicio 2"**

**agregar etiquetas a ejes (x, y)**

**mostrar la gráfica con cuadrícula**

**FIN**

****

****

**INICIO**

**FUNCIÓN ej3(x, y):**

**ecuacion1 ← x^3 - y**

**ecuacion2 ← x + y^2 - 4**

**RETORNAR (ecuacion1, ecuacion2)**

**# --- Resolver ---**

**aproximacion\_inicial ← (1, 1)**

**solucion ← fsolve(ej3, aproximacion\_inicial)**

**IMPRIMIR "Ejercicio 3:", solucion**

**# --- Graficar ---**

**definir rango\_x desde -3 hasta 3 con 400 puntos**

**definir rango\_y desde -3 hasta 3 con 400 puntos**

**crear malla (X, Y) con los rangos**

**trazar curva de nivel de (X^3 - Y = 0) en color rojo**

**trazar curva de nivel de (X + Y^2 - 4 = 0) en color azul**

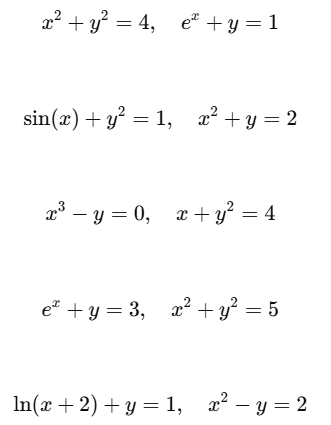
**marcar punto (solucion\_x, solucion\_y) en color verde**

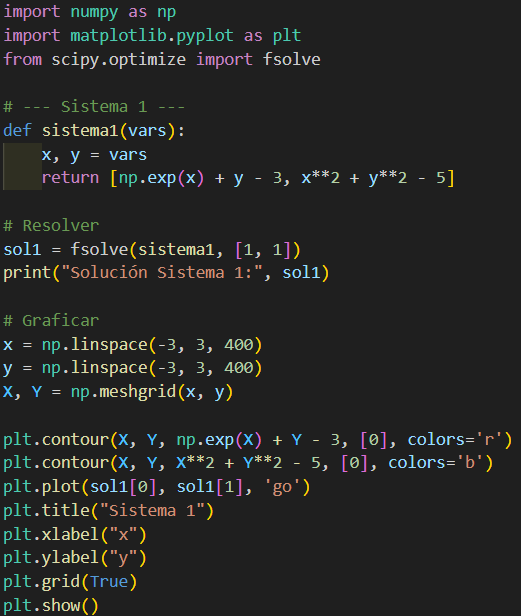
**agregar título "Ejercicio 3"**

**agregar etiquetas a ejes (x, y)**

**mostrar la gráfica con cuadrícula**

**FIN**

****

****

**INICIO**

**FUNCIÓN sistema1(x, y):**

**ecuacion1 ← e^x + y - 3**

**ecuacion2 ← x^2 + y^2 - 5**

**RETORNAR (ecuacion1, ecuacion2)**

**aproximacion\_inicial ← (1, 1)**

**solucion ← fsolve(sistema1, aproximacion\_inicial)**

**IMPRIMIR "Solución Sistema 1:", solucion**

**CREAR rango\_x desde -3 hasta 3**

**CREAR rango\_y desde -3 hasta 3**

**CREAR malla (X, Y)**

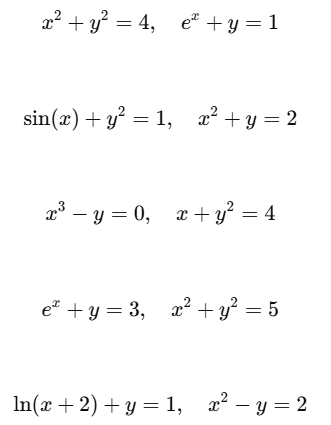
**DIBUJAR curva de nivel (e^X + Y - 3 = 0) en rojo**

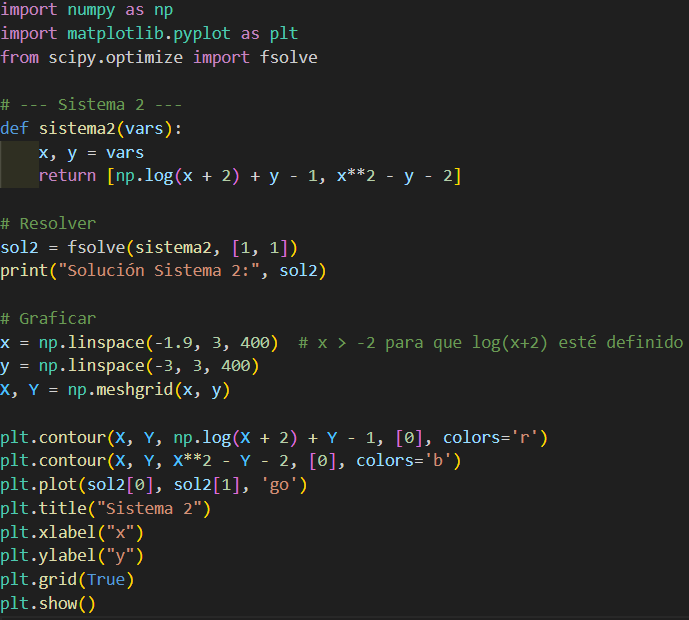
**DIBUJAR curva de nivel (X^2 + Y^2 - 5 = 0) en azul**

**MARCAR la solución en verde**

**MOSTRAR gráfica**

**FIN**

****

**INICIO**

**FUNCIÓN sistema2(x, y):**

**ecuacion1 ← ln(x+2) + y - 1**

**ecuacion2 ← x^2 - y - 2**

**RETORNAR (ecuacion1, ecuacion2)**

**aproximacion\_inicial ← (1, 1)**

**solucion ← fsolve(sistema2, aproximacion\_inicial)**

**IMPRIMIR "Solución Sistema 2:", solucion**

**CREAR rango\_x desde -1.9 hasta 3 # evitar log indefinido**

**CREAR rango\_y desde -3 hasta 3**

**CREAR malla (X, Y)**

**DIBUJAR curva de nivel (ln(X+2) + Y - 1 = 0) en rojo**

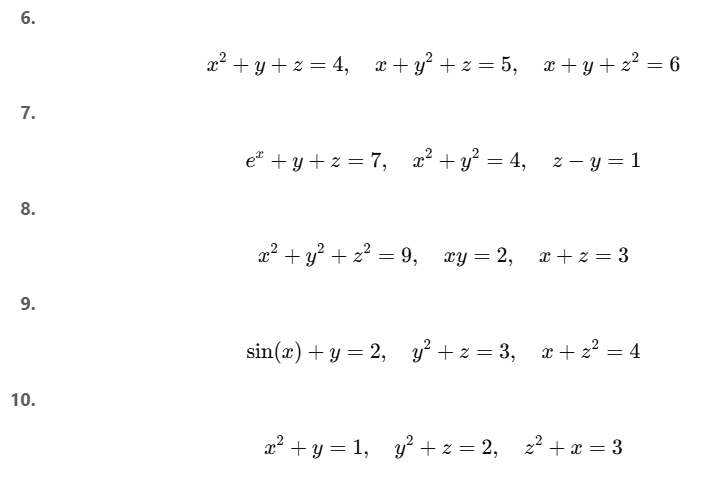
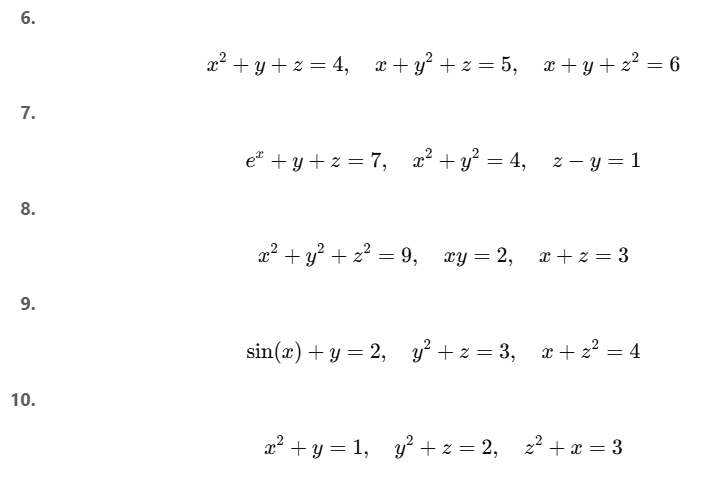
**DIBUJAR curva de nivel (X^2 - Y - 2 = 0) en azul**

**MARCAR la solución en verde**

**MOSTRAR gráfica**

**FIN**

1. **Sistemas de Ecuaciones No lineales de 3 variables**

** **

1. **Realizar Pseudocódigo**

6)

INICIO

Definir sistema(v):

ecuacion1 ← v0^2 + v1 + v2 - 4

ecuacion2 ← v1^2 + v2 + v0 - 5

ecuacion3 ← v2^2 + v0 + v1 - 6

RETORNAR (ecuacion1, ecuacion2, ecuacion3)

aproximacion\_inicial ← (1, 1, 1)

solucion ← fsolve(sistema, aproximacion\_inicial)

IMPRIMIR "Ejercicio 6 - solución aproximada (x,y,z) =", solucion

FIN

7)

INICIO

Definir sistema(v):

x ← v[0]

y ← v[1]

z ← v[2]

ecuacion1 ← exp(x) + y + z - 7

ecuacion2 ← x^2 + y^2 - 4

ecuacion3 ← z - y - 1

RETORNAR (ecuacion1, ecuacion2, ecuacion3)

aproximacion\_inicial ← (1, 1, 1)

solucion ← fsolve(sistema, aproximacion\_inicial)

IMPRIMIR "Ejercicio 7 - solución aproximada (x,y,z) =", solucion

FIN

9)

INICIO

Definir sistema2(v):

x ← v[0], y ← v[1], z ← v[2]

ec1 ← sen(x) + y - 2

ec2 ← y^2 + z - 3

ec3 ← x + z^2 - 4

RETORNAR (ec1, ec2, ec3)

aprox\_inicial ← (1,1,1)

solucion ← fsolve(sistema2, aprox\_inicial)

IMPRIMIR solucion

FIN

8)

INICIO

Definir sistema1(v):

x ← v[0], y ← v[1], z ← v[2]

ec1 ← x^2 + y^2 + z^2 - 9

ec2 ← x\*y - 2

ec3 ← x + z - 3

RETORNAR (ec1, ec2, ec3)

aprox\_inicial ← (1,1,1)

solucion ← fsolve(sistema1, aprox\_inicial)

IMPRIMIR solucion

FIN

10)

INICIO

Definir sistema3(v):

x ← v[0], y ← v[1], z ← v[2]

ec1 ← x^2 + y - 1

ec2 ← y^2 + z - 2

ec3 ← z^2 + x - 3

RETORNAR (ec1, ec2, ec3)

aprox\_inicial ← (1,1,1)

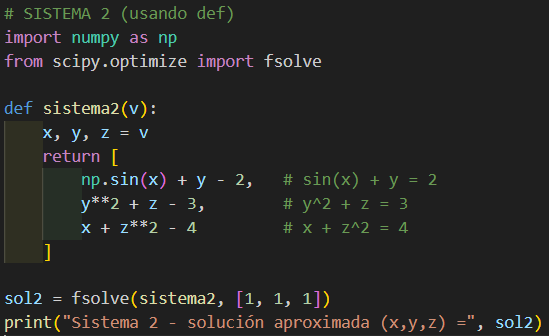
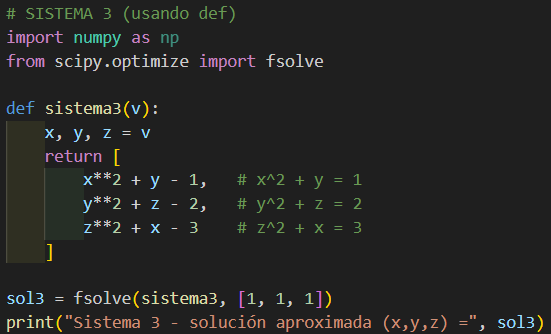
solucion ← fsolve(sistema3, aprox\_inicial)

IMPRIMIR solucion

FIN

1. **Codificar en Python (ejercicios 9 y 10 usar función DEF)**

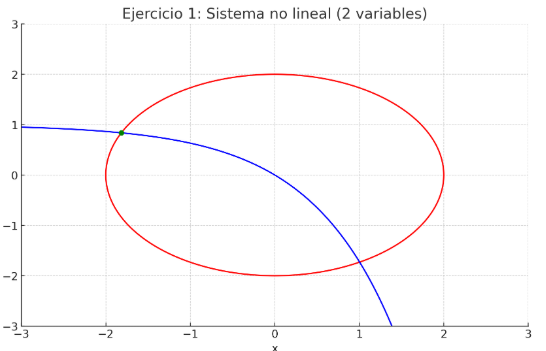
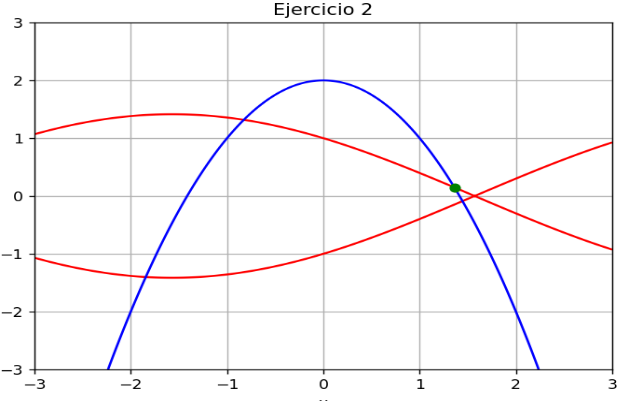
**9) 10)**

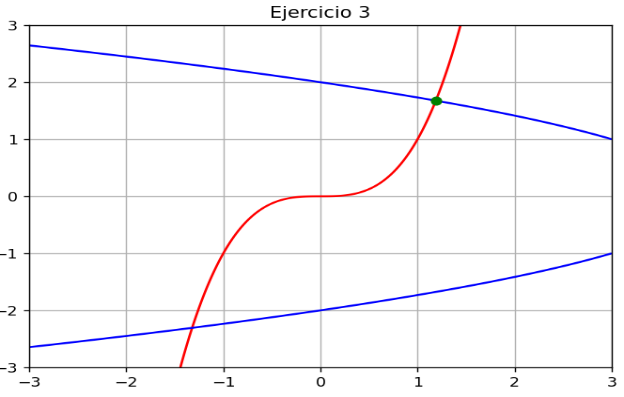
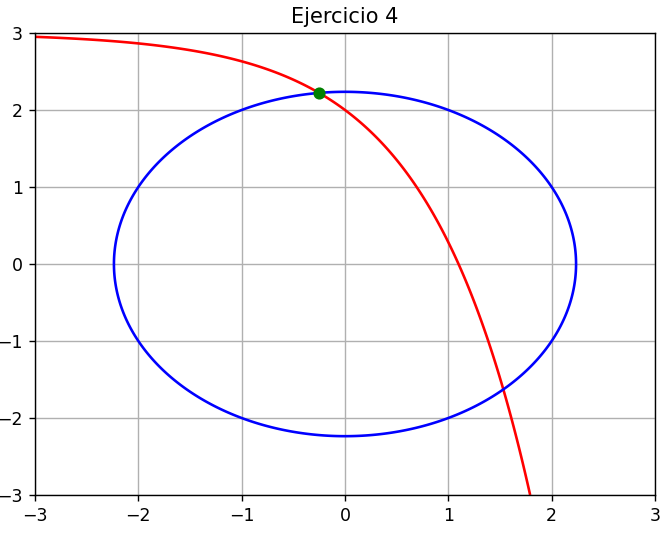
****

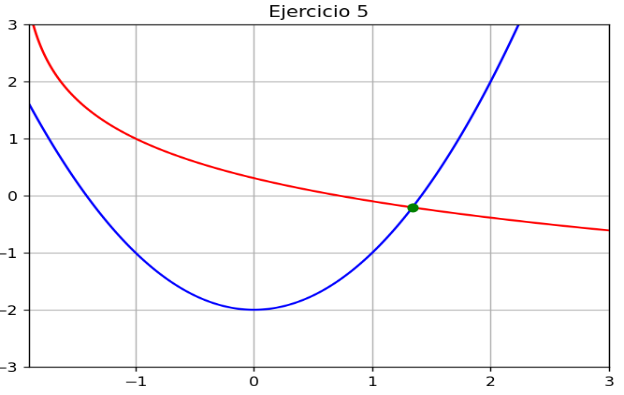
1. **Resultados (Máximo 10 líneas en texto)**

**Ejercicios:**

1. Ejercicio 1: [-1.81626407 0.8373678 ]
2. Ejercicio 2: [1.36080308 0.14821497]
3. Ejercicio 3: [1.18804969 1.67688709]
4. Ejercicio 4: [-0.25095313 2.22194116]
5. Ejercicio 5: [ 1.33947269 -0.20581292]
6. Ejercicio 6 - solución aproximada (x,y,z) = [0.73442414 1.52710987 1.93351131]
7. Ejercicio 7 - solución aproximada (x,y,z) = [0.87797832 1.79698471 2.79698471]
8. Ejercicio 8 - solución aproximada (x,y,z) = [1. 2. 2.]
9. Ejercicio 9 - solución aproximada (x,y,z) = [1.05020209 1.13247624 1.71749757]
10. Ejercicio 10 - solución aproximada (x,y,z) = [0.57785046 0.66608884 1.55632565]
11. **Pegar las 5 gráficas generadas de las ecuaciones No lineales de 2 variables (Que encajen en los recuadros.**

****

****

****

1. **Conclusiones (Máximo en 10 líneas)**

La resolución de sistemas de ecuaciones no lineales es fundamental en diferentes áreas de la ciencia e ingeniería, ya que muchos problemas reales no tienen solución analítica cerrada. A través de la práctica se comprobó que el método numérico de Newton-Raphson y la función fsolve de Scipy permiten aproximar soluciones de manera eficiente. Se observó que la elección de la aproximación inicial influye directamente en la convergencia y en la obtención de diferentes soluciones posibles. Para sistemas de dos variables, la representación gráfica facilitó la comprensión de las intersecciones como puntos solución. En los sistemas de tres variables, la interpretación fue más numérica, destacando la importancia de los métodos computacionales. En resumen, la práctica permitió reforzar el uso de Python como herramienta para modelar, resolver y analizar sistemas no lineales, valorando tanto la parte teórica como la computacional.