

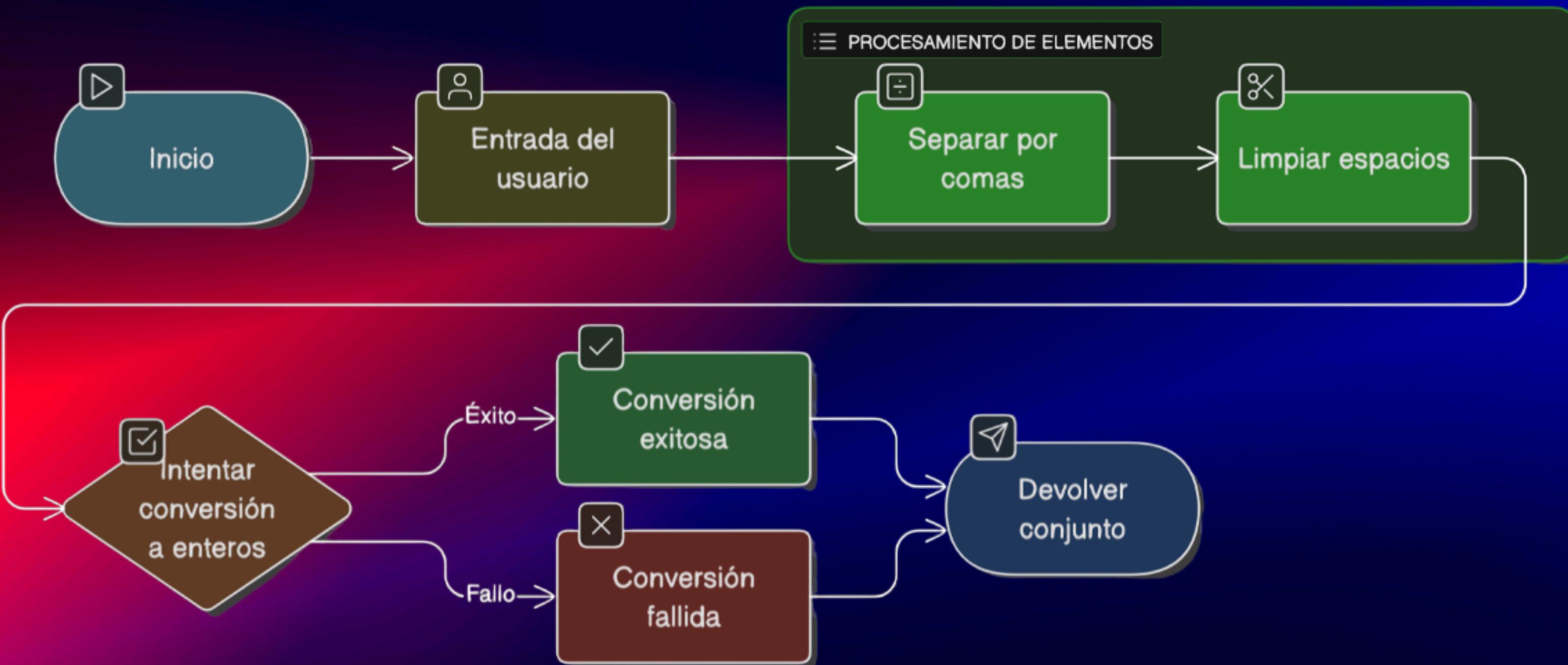
Programa: Propiedades de relación de conjuntos

Obtención del conjunto y la relación.

La relación se pide en un bucle.

```
def pedir_relacion(conjunto):
    print("Introduce los pares de la relación uno por uno. Ejemplo para (1,2): 1,2")
    print("Escribe 'fin' para terminar.")
    relacion = set()
    while True:
        entrada = input("Par: ")
        if entrada.lower() == "fin":
            break
    try:
        a, b = [e.strip() for e in entrada.split(",")]
    try:
        a, b = int(a), int(b)
    except ValueError:
        pass
    if a in conjunto and b in conjunto:
        relacion.add((a, b))
    else:
        print("Ambos elementos deben estar en el conjunto.")
    except Exception:
        print("Formato incorrecto. Usa: elemento1,elemento2")
    return relacion
```

Flujo: Obtener los datos conjuntos



Inicialización

`__init__`

Crea el analizador con un conjunto y una relación

`_validar_relacion`

Verifica que todos los pares de la relación pertenezcan al conjunto

Propiedades Individuales

`es_reflexiva`

Determina si cada elemento se relaciona consigo mismo

`es_irreflexiva`

Determina si ningún elemento se relaciona consigo mismo

`es_simetrica`

Verifica si la relación es bidireccional
($a \rightarrow b$ implica $b \rightarrow a$)

`es_asimetrica`

Verifica si la relación es unidireccional estricta

`es_antisimetrica`

Verifica si la bidireccionalidad solo ocurre cuando $a=b$

`es_transitiva`

Verifica si las cadenas de relación se completan

Clase: AnalizadorRelaciones



Evaluación Global

`es_orden_parcial`

Determina si la relación forma una jerarquía ordenada

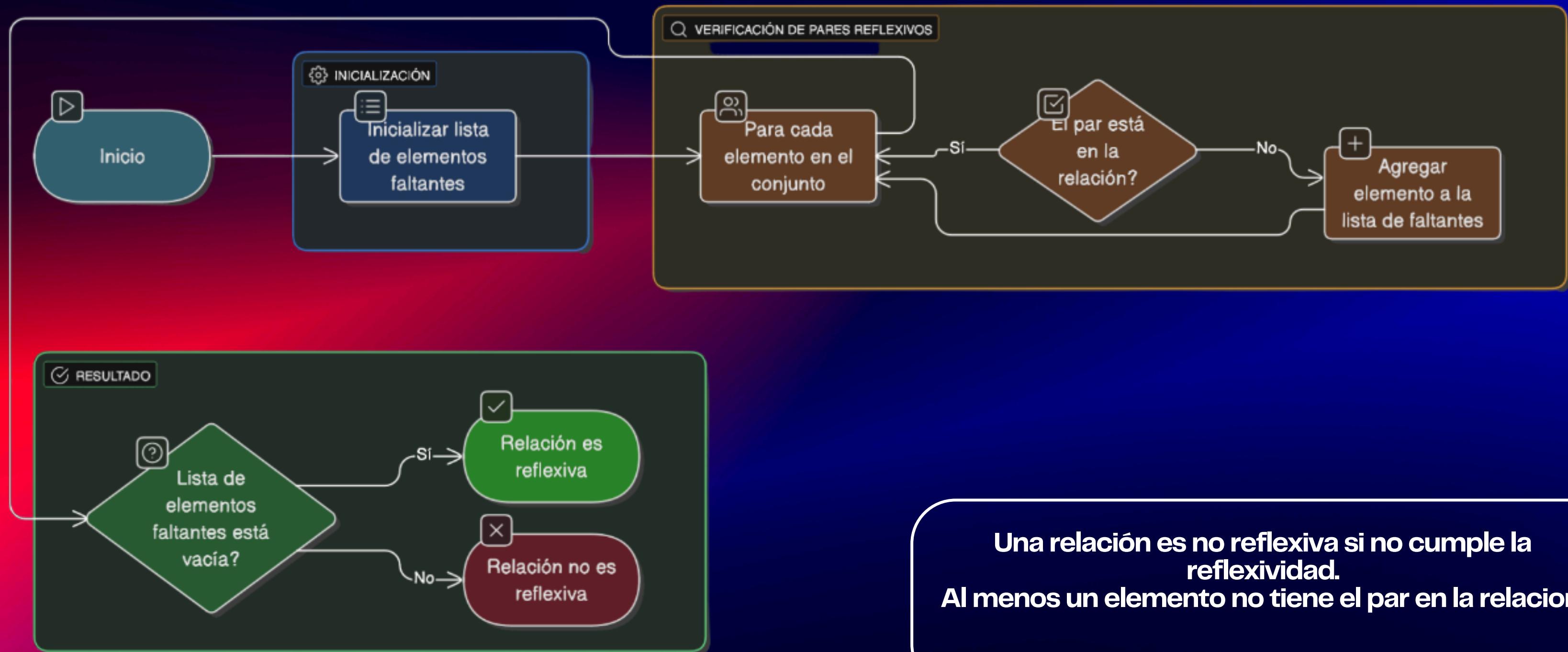
`analizar_propiedades`

Ejecuta todas las verificaciones y retorna un reporte completo

**Diagramas de
las funciones de
la clase para
obtener
propiedades.**



Reflexiva

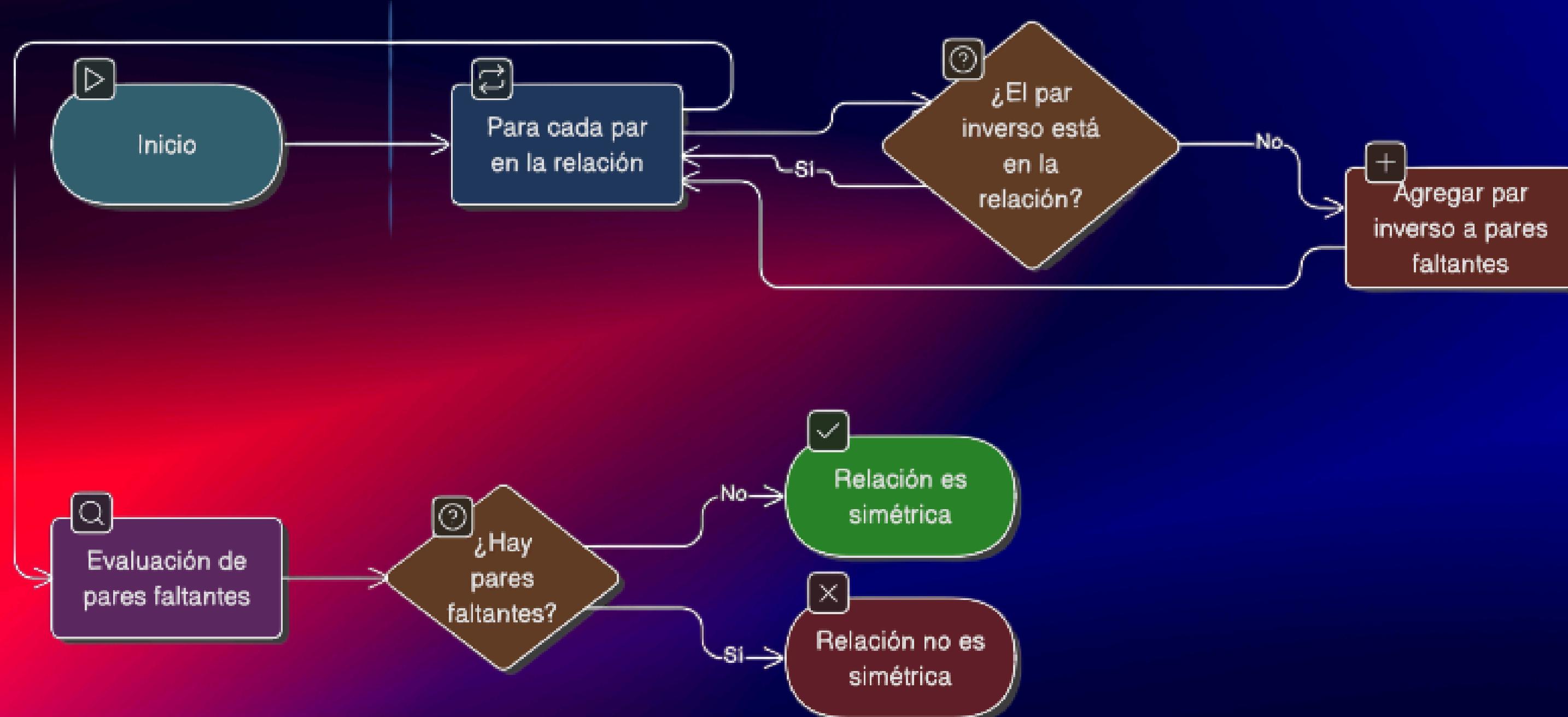


Irreflexiva

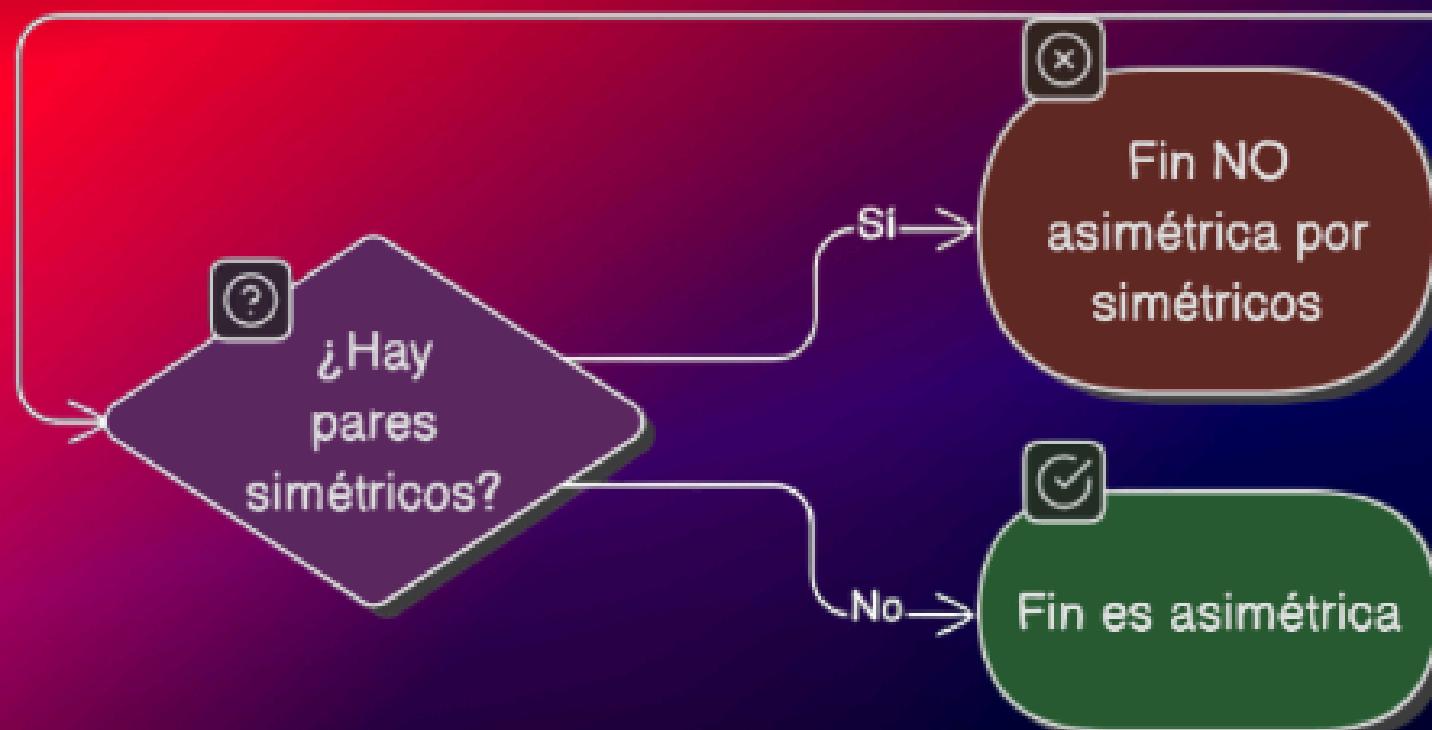
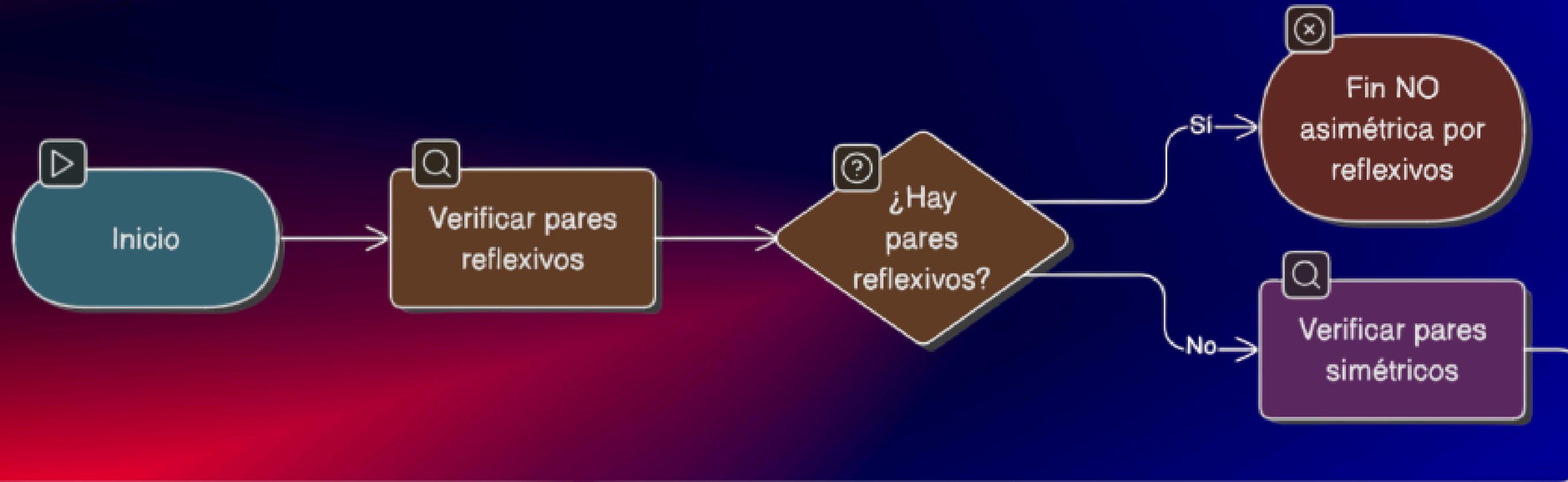
Ningún elemento se relaciona consigo mismo.



Simetrica

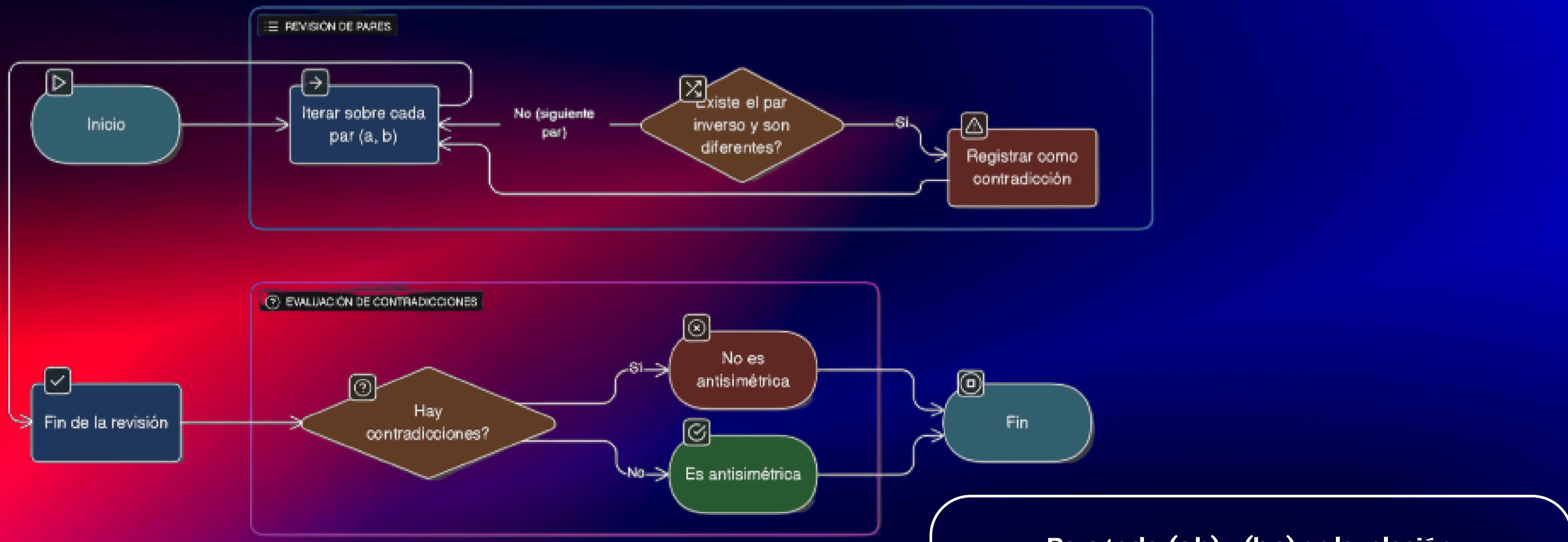


Asimetrica



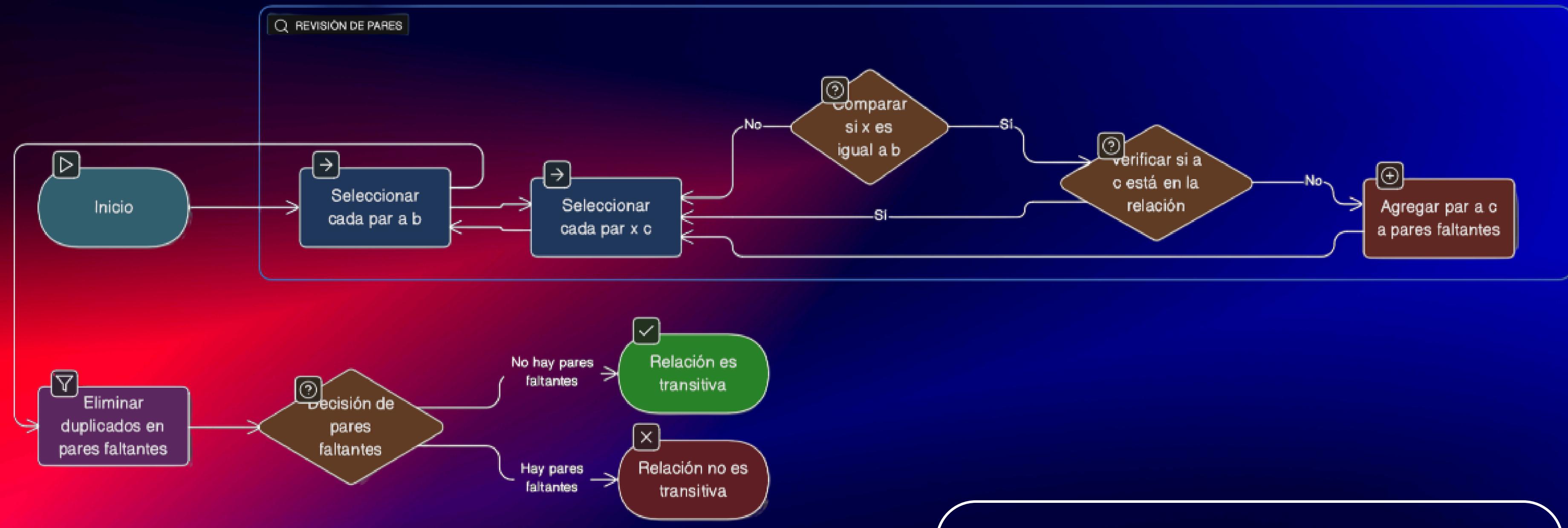
- 1- No tener pares reflexivos
- 2- No tener pares simétricos

Antisimetrica



Para todo: (a,b) y (b,a) en la relación,
forzosamente $a = b$.

Transitiva



$$(a,b) \in R \text{ y } (b,c) \in R \Rightarrow (a,c) \in R$$

Orden Parcial

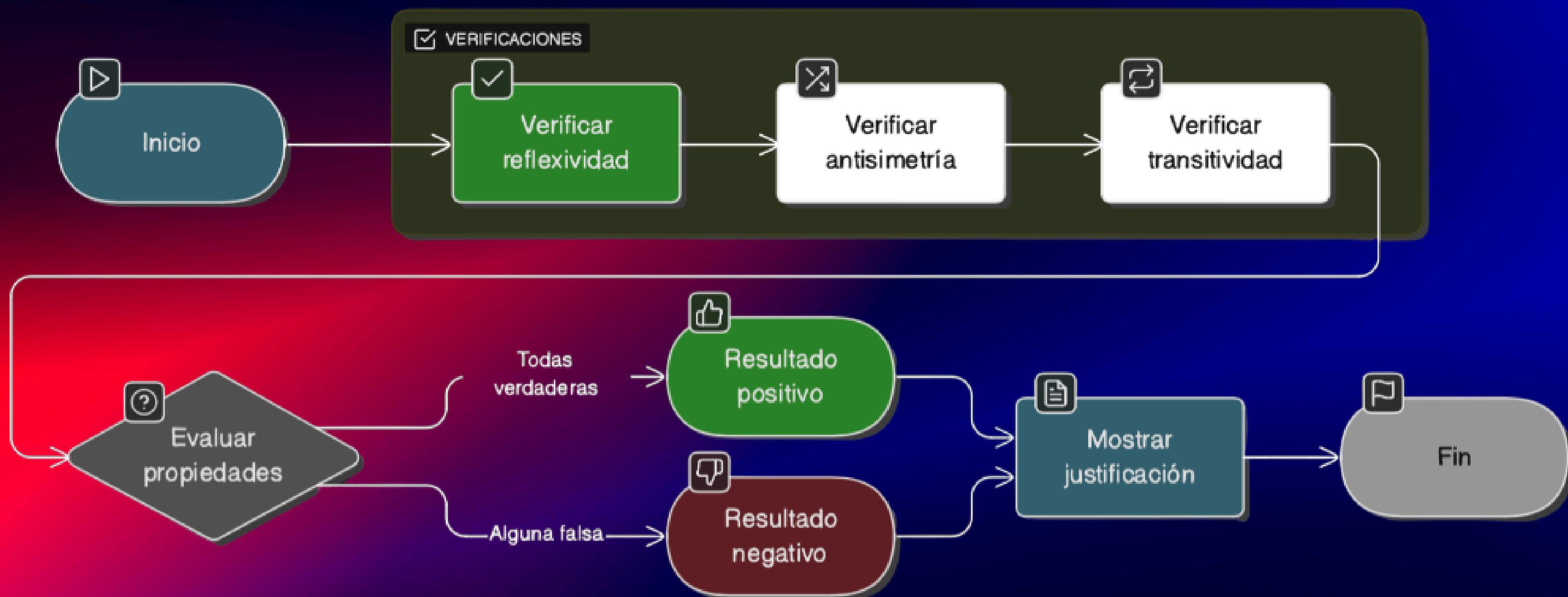
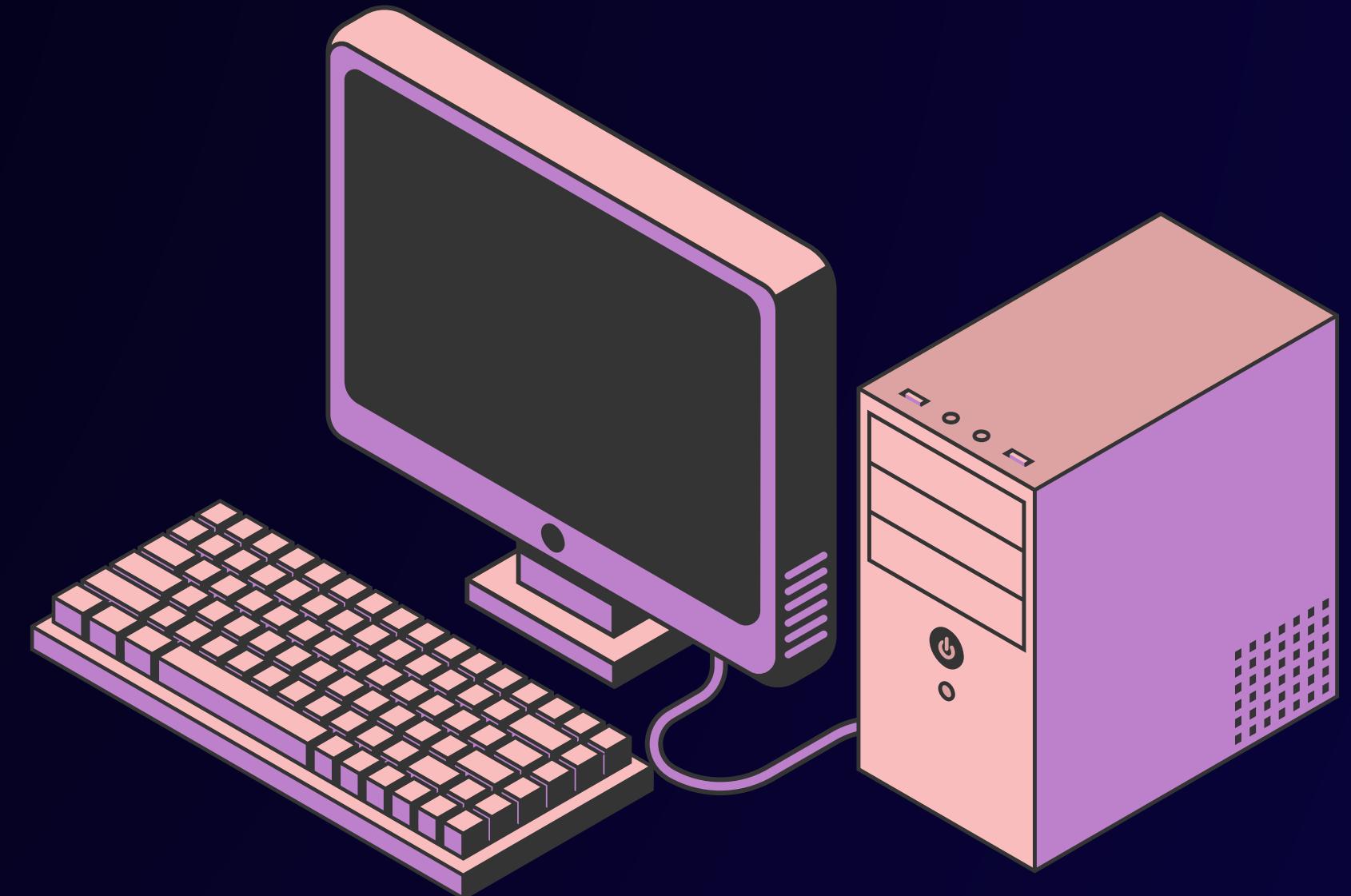
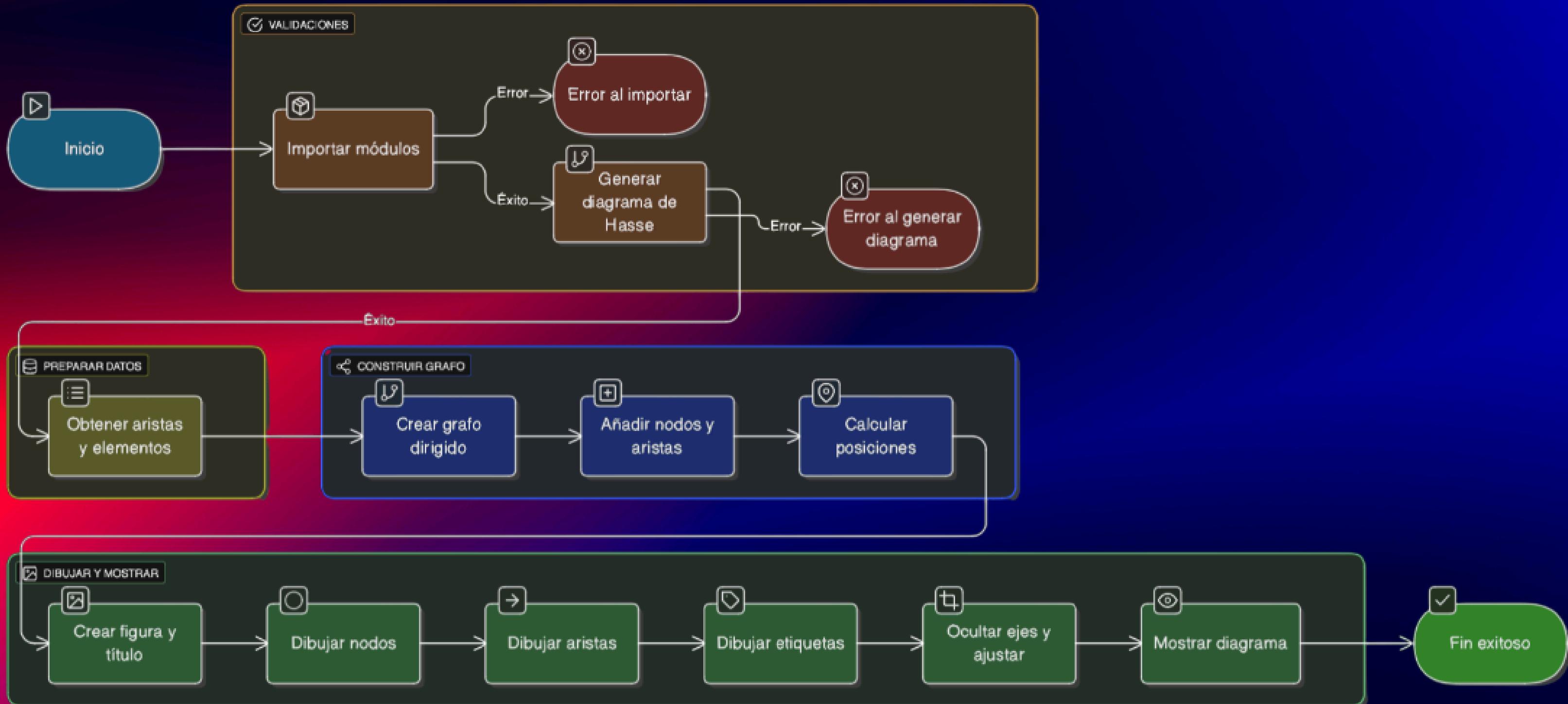


Diagrama de Hasse.

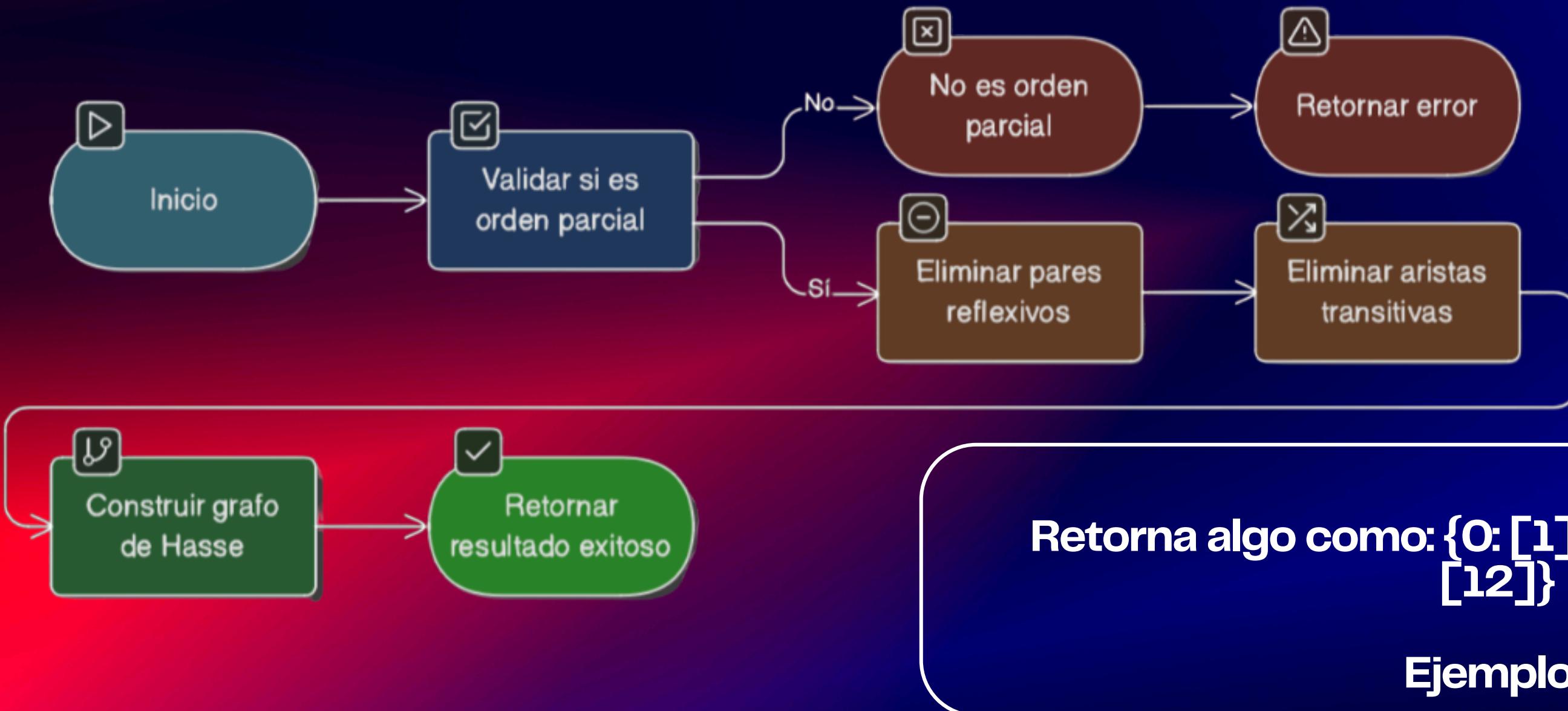
*Para graficar se usaron
las libreria de NetworkX y
Matplotlib de Python..



Función: mostrar_diagrama_hasse



Función: generar_diagrama_hasse

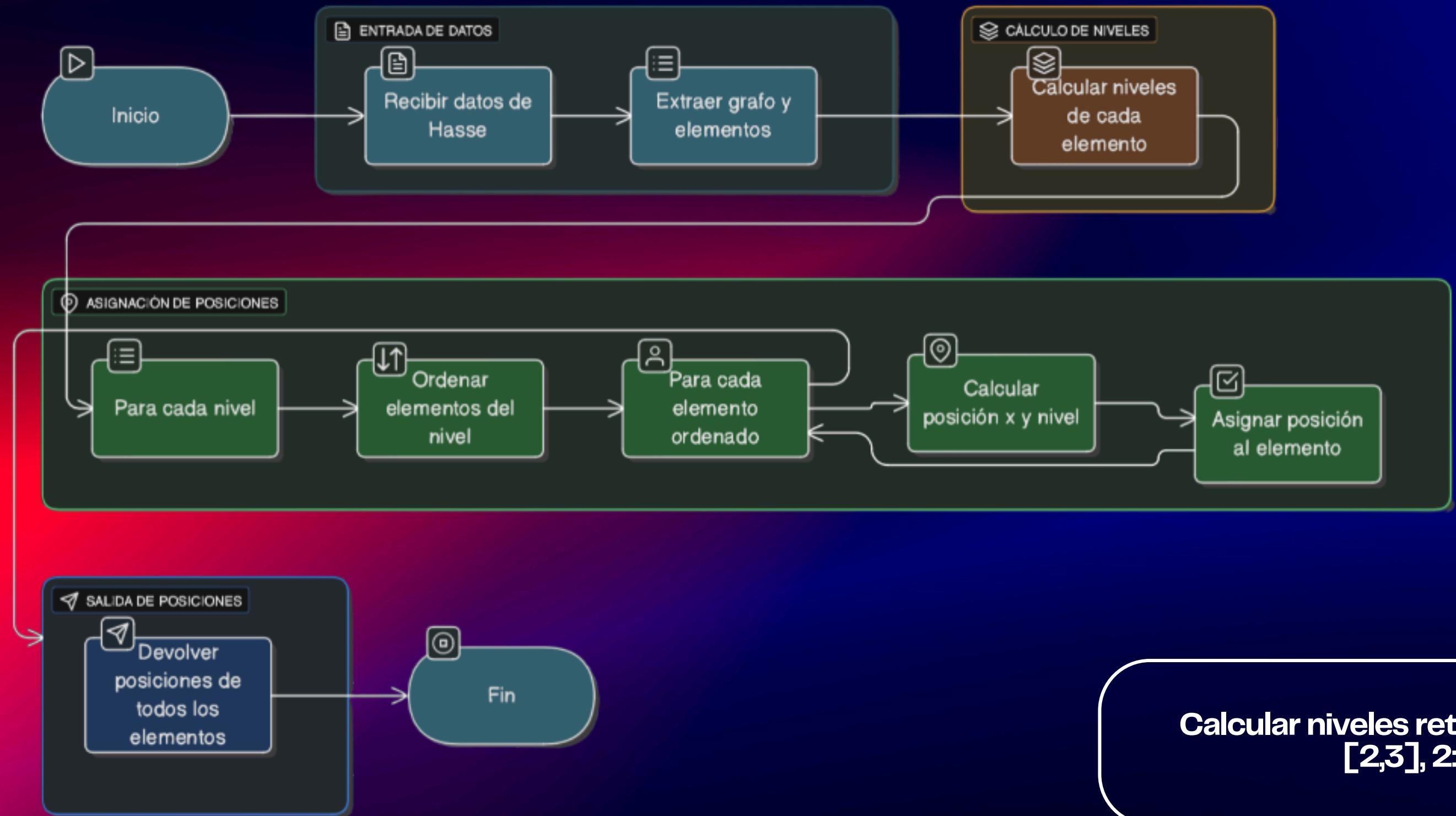


Retorna algo como: {0: [1], 1: [2,3], 2: [4,6], 3: [12]}

Ejemplo:

```
(True, {
    'grafo': {1: [2,3], 2: [4,6], 3: [6], 4: [12], 6: [12], 12: []},
    'aristas': {(1,2), (1,3), (2,4), (2,6), (3,6), (4,12), (6,12)},
    'elementos': {1, 2, 3, 4, 6, 12}
})
```

Función: _calcular_posiciones



Calcular niveles retorna algo como: {0: [1], 1: [2,3], 2: [4,6], 3: [12]}

Algoritmo de distribución.

Para 1 elemento:

- `len(elementos_ordenados) = 1`
- Condición: `1 > 1 → False`
- Resultado: `x = 0` (centro)

Para 2 elementos:

- `len(elementos_ordenados) = 2`
- Elemento 0: $x = 0 - 2/2 + 0.5 = 0 - 1 + 0.5 = -0.5$
- Elemento 1: $x = 1 - 2/2 + 0.5 = 1 - 1 + 0.5 = 0.5$
- Resultado: `[-0.5, 0.5]` (distribuidos simétricamente)

Para 3 elementos:

- `len(elementos_ordenados) = 3`
- Elemento 0: $x = 0 - 3/2 + 0.5 = 0 - 1.5 + 0.5 = -1.0$
- Elemento 1: $x = 1 - 3/2 + 0.5 = 1 - 1.5 + 0.5 = 0.0$
- Elemento 2: $x = 2 - 3/2 + 0.5 = 2 - 1.5 + 0.5 = 1.0$
- Resultado: `[-1.0, 0.0, 1.0]` (distribuidos simétricamente)

Componentes:

- `i`: Posición del elemento (`0, 1, 2...`)
- `len(elementos_ordenados)/2`: Punto medio del grupo
- `+ 0.5`: Ajuste para centrar la distribución

Asignación de coordenadas.

- X: Posición horizontal (calculada con la fórmula)
- Y: Nivel jerárquico (altura en el diagrama)

Ejemplo divisores de 12

```
{0: [1], 1: [2, 3], 2: [4, 6], 3: [12]}
```

Procesamiento:

Nivel 0: [1]

- 1 elemento → $x = 0$
- $\text{pos}[1] = (0, 0)$

Nivel 1: [2, 3]

- 2 elementos → $x = [-0.5, 0.5]$
- $\text{pos}[2] = (-0.5, 1)$
- $\text{pos}[3] = (0.5, 1)$

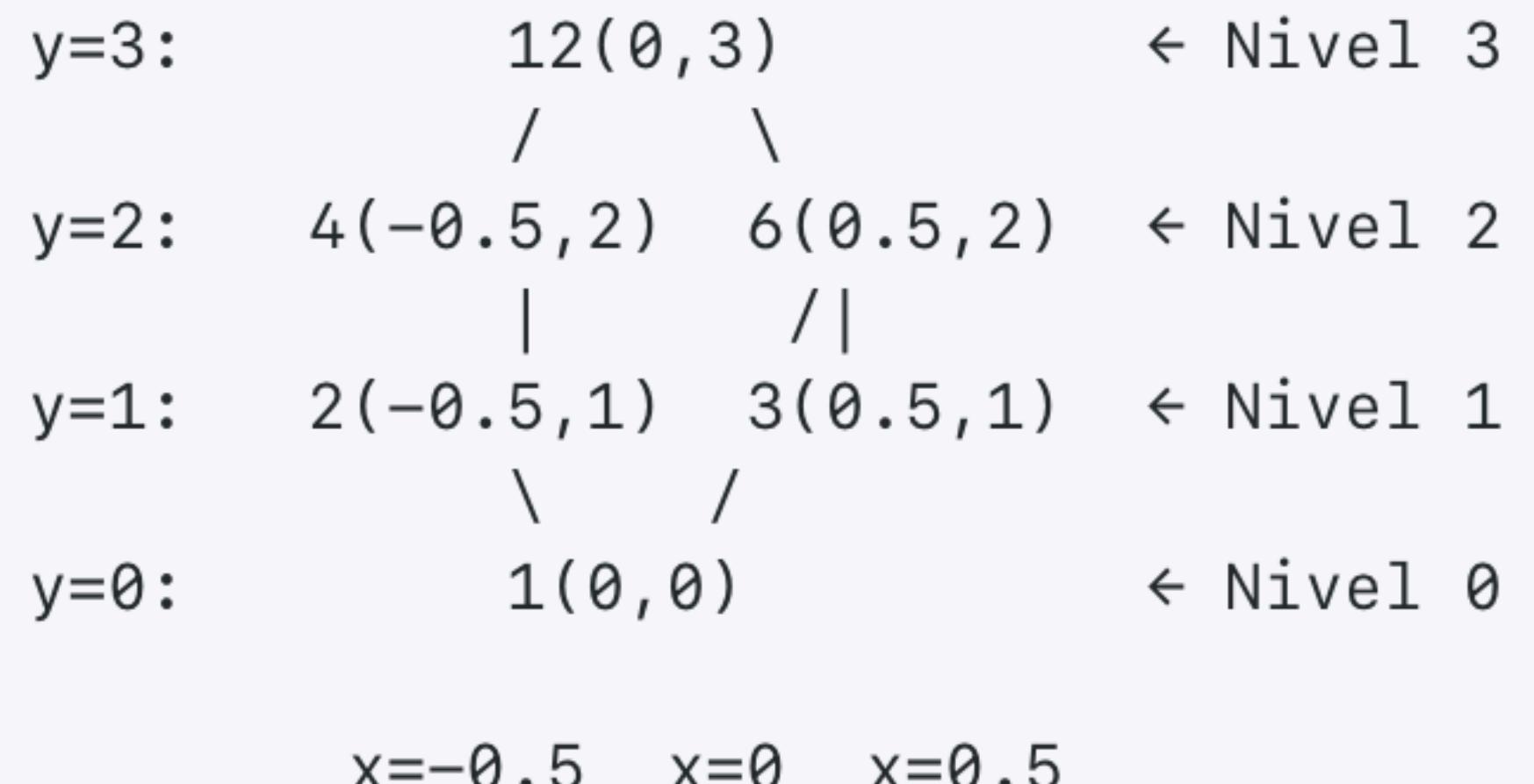
Nivel 2: [4, 6]

- 2 elementos → $x = [-0.5, 0.5]$
- $\text{pos}[4] = (-0.5, 2)$
- $\text{pos}[6] = (0.5, 2)$

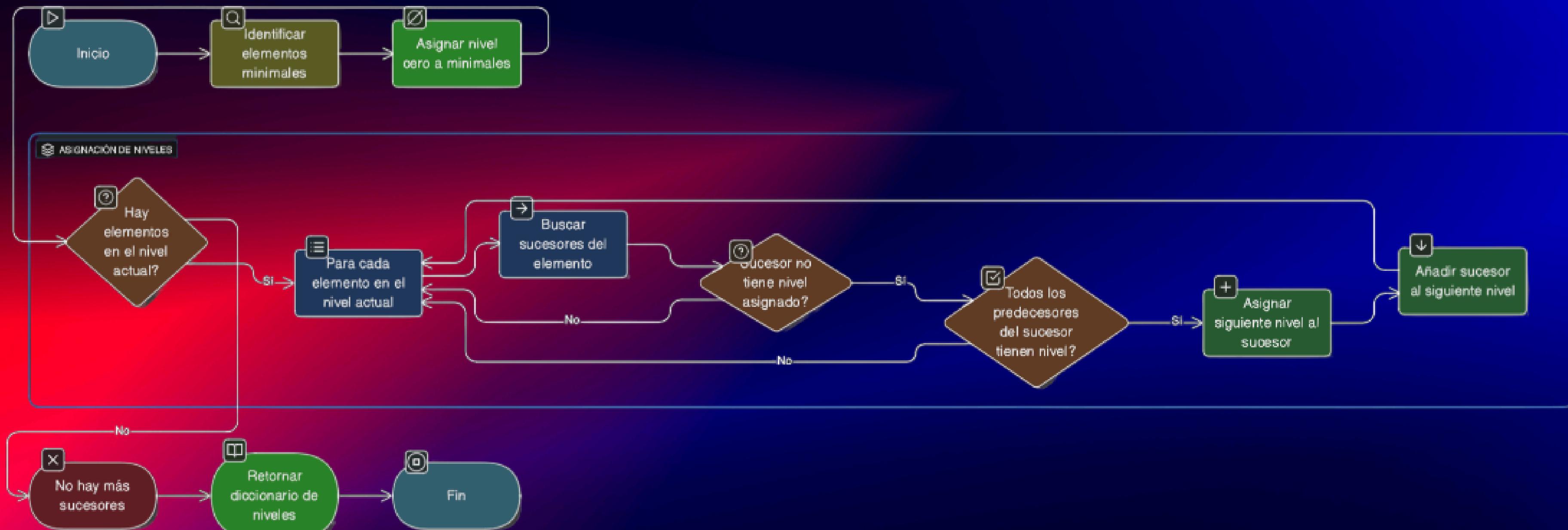
Nivel 3: [12]

- 1 elemento → $x = 0$
- $\text{pos}[12] = (0, 3)$

Coordenadas en el plano.



Función: _calcular_niveles_hasse



Prueba de escritorio.

Ejemplo: Divisores de 12

Inicialización:

Nivel 0: [1] (minimales)

Iteración 1 (desde nivel 0):

Desde 1: examinar sucesores [2, 3]
- 2: predecesores [1] → 1 tiene nivel ✓ → 2 va a nivel 1
- 3: predecesores [1] → 1 tiene nivel ✓ → 3 va a nivel 1

Resultado: Nivel 1: [2, 3]

Iteración 2 (desde nivel 1):

Desde 2: examinar sucesores [4, 6]
- 4: predecesores [2] → 2 tiene nivel ✓ → 4 va a nivel 2
- 6: predecesores [2, 3] → ambos tienen nivel ✓ → 6 va a nivel 2

Desde 3: examinar sucesores [6]
- 6: ya procesado

Resultado: Nivel 2: [4, 6]

Iteración 3 (desde nivel 2):

Desde 4: examinar sucesores [12]
- 12: predecesores [4, 6] → ambos tienen nivel ✓ → 12 va a nivel 3

Desde 6: examinar sucesores [12]
- 12: ya procesado

Resultado: Nivel 3: [12]

Iteración 4 (desde nivel 3):

Desde 12: examinar sucesores []
No hay sucesores → TERMINA

```
return niveles = {
    0: [1],      # Base (minimales)
    1: [2, 3],   # Primer nivel
    2: [4, 6],   # Segundo nivel
    3: [12]      # Tope (maximales)
}
```

Programa: Propiedades de relación de conjuntos

FIN