EJEMPLO DE EJERCICIO – ASIGNACIÓN DE TAREAS A PROGRAMADORES

⭐ 1. Identificación del problema (realista)

Una empresa de desarrollo de software necesita asignar **4 tareas** del sprint a **4 programadores**.
Cada programador tiene diferentes habilidades y por lo tanto se demora distinto tiempo según la tarea.

El objetivo es **minimizar el tiempo total de desarrollo** asignando cada tarea a un programador distinto (modelo de asignación).

⭐ 2. Modelo de Investigación de Operaciones usado

Se utiliza el **Modelo de Asignación** (un caso especial de Programación Lineal).

¿Por qué es adecuado?
✓ Cada recurso (programador) solo puede hacer una tarea.
✓ Cada tarea debe ser hecha por exactamente un programador.
✓ Queremos optimizar tiempo → función objetivo lineal.

⭐ 3. Datos del problema

Tabla de tiempos estimados (en horas):

Programador	Tarea 1	Tarea 2	Tarea 3	Tarea 4
Ana	6	8	7	9
Luis	9	6	8	7
Marta	7	5	9	6
Carlos	8	7	6	5

⭐ 4. Formulación del Modelo

✓ Variables de decisión

$$x_{ij} = \begin{cases} 1 & \text{si el programador } i \text{ realiza la tarea } j \\ 0 & \text{de lo contrario} \end{cases}$$

Con $i = \{Ana, Luis, Marta, Carlos\}$
y $j = \{1,2,3,4\}$

✓ **Función Objetivo**

Minimizar el tiempo total:

$$\min Z = 6x_{11} + 8x_{12} + 7x_{13} + 9x_{14} + 9x_{21} + 6x_{22} + 8x_{23} + 7x_{24} + 7x_{31} + 5x_{32} + 9x_{33} + 6x_{34} + 8x_{41} + 7x_{42} + 6x_{43} + 5x_{44}$$

✓ **Restricciones**

1 **Cada tarea debe ser asignada a un programador:**

$$x_{1j} + x_{2j} + x_{3j} + x_{4j} = 1 \forall j$$

2 **Cada programador solo puede recibir 1 tarea:**

$$x_{i1} + x_{i2} + x_{i3} + x_{i4} = 1 \forall i$$

3 **Variables binarias:**

$$x_{ij} \in \{0,1\}$$

Solucion en Python

```python
# MODELO DE ASIGNACIÓN DE TAREAS A PROGRAMADORES
# ------------------------------------------------------------
# Este programa busca la asignación que minimiza el tiempo total,
# probando todas las combinaciones posibles (método exhaustivo).
# ------------------------------------------------------------

import itertools

# Matriz de tiempos (horas) para cada programador según la tarea
tiempos = {
    "Ana":    [6, 8, 7, 9],
    "Luis":   [9, 6, 8, 7],
    "Marta":  [7, 5, 9, 6],
    "Carlos": [8, 7, 6, 5]
}

# Lista de programadores
programadores = list(tiempos.keys())

# Número de tareas (4)
n = 4

# Todas las permutaciones posibles de asignar 4 tareas a 4 programadores
# Por ejemplo: (0,1,2,3) significa:
# Ana -> Tarea 1
# Luis -> Tarea 2
# Marta -> Tarea 3
# Carlos -> Tarea 4
```

```python
permutaciones = list(itertools.permutations(range(n)))

mejor_asignacion = None
mejor_tiempo = float("inf")

# ------------------------------------------------------------
# Evaluación de todas las asignaciones posibles
# ------------------------------------------------------------
for asignacion in permutaciones:
    tiempo_total = 0

    # Sumar los tiempos según la permutación actual
    for i, programador in enumerate(programadores):
        tarea_asignada = asignacion[i]   # índice de la tarea (0-3)
        tiempo_total += tiempos[programador][tarea_asignada]

    # Guardar la mejor asignación (tiempo mínimo)
    if tiempo_total < mejor_tiempo:
        mejor_tiempo = tiempo_total
        mejor_asignacion = asignacion

# ------------------------------------------------------------
# Resultado final
# ------------------------------------------------------------
print("MEJOR ASIGNACIÓN ENCONTRADA:\n")

for i, programador in enumerate(programadores):
    print(f"{programador} → Tarea {mejor_asignacion[i] + 1} "
```

```
            f"(Tiempo: {tiempos[programador][mejor_asignacion[i]]} horas)")

print(f"\nTiempo total mínimo: {mejor_tiempo} horas")
```

✅ Salida esperada del programa

less

```
MEJOR ASIGNACIÓN ENCONTRADA:

Ana → Tarea 1 (Tiempo: 6 horas)
Luis → Tarea 2 (Tiempo: 6 horas)
Marta → Tarea 4 (Tiempo: 6 horas)
Carlos → Tarea 3 (Tiempo: 6 horas)

Tiempo total mínimo: 24 horas
```