P1 (4 **puntos**). Una compañía vende dos tipos de producto: normal y premium. La compañía tiene 200 toneladas de materia prima para fabricar los productos y tiene disponibles 300 horas de tiempo de fabricación. Cada tonelada de producto normal requiere una tonelada de materia prima y una hora de fabricación, y produce una ganancia de 3.000 euros. Cada tonelada de producto premium requiere también una tonelada de materia prima, pero necesita dos horas de fabricación, y produce una ganancia de 5.000 euros.

a) (1 *punto*) Bajo este contexto, formula el problema (formulación general) de programación lineal que determine la cantidad de toneladas de cada tipo de producto que debe producir la compañía, para que el beneficio total que obtenga sea máximo.

b) (0,5 *puntos*) A partir de la formulación general, escribe la formulación estándar del problema.

c) (1,5 puntos) Implementa el modelo en PuLP o Pyomo (sólo uno de los dos) y resuélvelo.

d) (1 punto) Escribe el problema dual del problema planteado en a).

P2 (**4 puntos**). Ana es la gerente de una fábrica y quiere minimizar los costes de mano de obra y, a la vez, asegurar que en cada uno de los turnos (m turnos) hay n trabajadores para satisfacer la demanda de producción.

Disponemos de los siguientes datos:

c_i ≡ coste de mano de obra del trabajador i.

t_j ≡ demanda de personal en el turno j.

d_{ij} ≡ disponibilidad del trabajador i en el turno j.

a) (1 *puntos*) Bajo este contexto, formula el problema (formulación general) de minimización de coste de mano de obra mediante la asignación óptima de personal en los distintos turnos. Asume que la demanda de cada turno se debe satisfacer con igualdad ("trabajadores del turno j" = t_j).

b) (*1 puntos*) Implementa el modelo en PuLP o Pyomo (sólo uno de los dos) y resuélvelo considerando 8 trabajadores y 4 turnos de trabajo, para valores de los parámetros incluidos en el archivo adjunto "fábrica.dat".

c) (2 *puntos*) Supongamos que, por motivos personales, Enrique no puede trabajar en turnos junto con David ni con Juan.

Implementa el modelo en PuLP o Pyomo (sólo uno de los dos).
Pista: puede ser recomendable el uso de variables binarias.

P3 **(2 puntos)**. Genera datos aleatorios, al menos $m = 1000$ observaciones, de un modelo de regresión lineal predefinido con al menos $n = 100$ variables.

Considera que los coeficientes de la regresión $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_i, \ldots, \beta_n)$ son enteros donde: $-5 \leq \beta_i \leq 5\ \forall \boldsymbol{i}$. Considera también que los errores (residuos) son normales ($e \sim N(0, \sigma)$) e independientes.

$$Y = \boldsymbol{\beta}'X + s$$

a) (*0.4 punto*) Estimar el valor de los parámetros de la regresión aplicando la solución analítica del problema de mínimos cuadrados.

b) (0.4 *punto*) Estimar el valor de los coeficientes de la regresión por mínimos cuadrados usando la herramienta **minimize** del paquete de Python **Scipy.optimize**. Prueba al menos cuatro solvers diferentes y compara su eficiencia en términos de: número de iteraciones totales, número de evaluaciones de la función objetivo, gradiente y hesiano, así como el tiempo de cómputo total.

c) Estimar el valor de los coeficientes de la regresión por mínimos cuadrados **implementando manualmente**:

   i. (*0.4 puntos*) Método del **Gradiente**.
   ii. (*0.4 puntos*) Método de **Newton**.
   iii. (*0.4 puntos*) Método de **Quasi-Newton**.

IMPORTANTE: Presentar los códigos en dos notebooks de Jupyter, uno por cada problema, con el nombre en formato: Apellidos-Nombre-P1.ipynb, Apellidos-Nombre-P2.ipynb y Apellidos-Nombre-P3.ipynb. Indicar las formulaciones y respuestas a cada apartado en los propios notebooks de Jupyter mediante celdas “Markdown”.