

DEIOAC-UPV

## 4. FORMULACIÓN DE MODELOS DE PROGRAMACIÓN ENTERA



# Objetivos

---

Al finalizar el tema, deberás ser capaz de:

- **Plantear y formular modelos** de programación **lineal** y **entera** de diversos tipos (mezclas óptimas, localización y cubrimiento etc.).
- **Plantear y formular modelos lineales** con uso de **variables binarias**, bien como variables de decisión, o bien como variables auxiliares.
- Utilizar **lenguajes de especificación** para la formulación de modelos de gran tamaño.

# CONTENIDOS

---

## 4.1 Introducción

## 4.2 Modelización con variables binarias

4.2.1 Producción acotada

4.2.2 Producción acotada inferiormente

4.2.3 Costes fijos

4.2.4 Variables que toman un conjunto de valores

4.2.5 Restricciones excluyentes (una u otra)

4.2.6 Varios segundos miembros de una restricción

# CONTENIDOS

---

## 4.3 Formulación de modelos

4.3.1 Un problema de mezclas (I)

4.3.2 Un problema de mezclas (II)

4.3.3 Un problema de distribución comercial

4.3.4 Un problema de cubrimiento

4.3.5 Un problema de secuenciación de tripulaciones aéreas

## 4.4 Lenguajes de Modelización

## 4.1 Introducción

---

- Muchos de los **problemas reales** exigen soluciones con valores enteros, por lo tanto las variables de dicho problema deben ser definidas como **variables enteras**

Hipótesis	Programación Lineal	Programación Entera
<b>H1:</b> Divisibilidad	<b>SI</b>	<b>NO</b>
<b>H2:</b> No negatividad	<b>SI</b>	<b>SI</b>
<b>H3:</b> Linealidad	<b>SI</b>	<b>SI</b>
<b>H4:</b> Certidumbre	<b>SI</b>	<b>SI</b>

## 4.2 Modelización con variables binarias

---

- ▶ En optimización, aspiraremos a modelizar problemas mediante un **modelo lineal** ya que son más fáciles de resolver. Sin embargo, muchos problemas presentan situaciones en que la linealidad del modelo se hace muy difícil de mantener con un conjunto de variables continuas como única herramienta de modelización.
- ▶ Es así como surgen las **variables binarias** (aquellas que sólo pueden tomar los valores 0 y 1) como un artificio que nos permite expresar situaciones no lineales como lineales. Esta definición de variables es muy útil en la práctica y existen algoritmos para resolver este tipo de problemas basándose en las técnicas de programación lineal: **Algoritmo de Bifurcación y Acotación (Branch & Bound)**

## 4.2 Modelización con variables binarias

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.1 Producción acotada

Consideremos la fabricación de un producto  $j$  ( $x_j$ ), el cual puede producirse o no, pero que **en caso de producirse** sólo puede hacerse en un nivel comprendido entre  $L_j$  y  $U_j$  (*OJO!  $U_j$  y  $L_j$  son constantes!*). Para modelizar esta restricción, además del nivel de producción  $x_j$ , definimos la siguiente variable binaria:

$y_j = (0,1)$  :  $1 \rightarrow$  Si se fabrica el producto  $j$   
 $0 \rightarrow$  Si no se fabrica el producto  $j$

Así, la restricción vendría dada por:

$$L_j \cdot y_j \leq x_j \leq U_j \cdot y_j$$

## 4.2 Modelización con variables binarias

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.2 Producción acotada inferiormente

Consideremos la fabricación de un producto  $j$  ( $x_j$ ), el cual puede producirse o no, pero que en caso de producirse sólo puede hacerse en un nivel de al menos  $L_j$  sin que exista una cota superior explícita. La táctica anterior no sirve por lo que además de la variable  $y_j$ , inventamos un nuevo parámetro  $M_j$  que sirva como una cota superior:

$y_j = (0,1)$  :       $1 \rightarrow$  Si se produce el producto  $j$   
                          $0 \rightarrow$  Si no se produce el producto  $j$

$M_j$  = Un número muy grande.

Así, la restricción vendría dada por:

$$L_j \cdot y_j \leq x_j \leq M_j \cdot y_j$$



## 4.2 Modelización con variables binarias

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.3 Costes fijos

Consideremos el caso en que debemos decidir si realizar o no una actividad ( $x_j$ ) cuyo coste tiene tanto una componente fija ( $K_j$ ) como una componente variable ( $C_j \cdot x_j$ ), es decir el coste de realizar la actividad al nivel  $x_j$  viene dado por:

Coste ( $x_j$ ):	0	si $x_j = 0$
	$K_j + C_j \cdot x_j$	si $x_j > 0$

**Modelo  
no lineal**

En este caso, nuevamente resulta de gran utilidad definir una variable binaria:

$y_j = (0,1)$ :	1 $\rightarrow$ Si se realiza la actividad j
	0 $\rightarrow$ Si no se realiza la actividad j

## 4.2 Modelización con variables binarias

---

**Situaciones frecuentes que pueden modelarse con variables binarias:**

Así, la función de coste queda como:

$$\text{Coste } (x_j) = K_j \cdot y_j + C_j \cdot x_j$$

Sin embargo, hasta el momento nada impide al modelo adoptar soluciones del tipo  $y_j = 0$  y  $x_j \neq 0$ , situación que evitamos imponiendo la siguiente restricción:

$$x_j \leq M_j \cdot y_j \quad \text{con } M_j \text{ muy grande}$$

**Observación:** Existen otras formulaciones alternativas como por ejemplo:  $\text{Coste } (x_j) = K_j \cdot y_j + C_j \cdot x_j \cdot y_j$ , pero **son NO lineales !!!**

## 4.2 Modelización con variables binarias

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.4 Variables que toman un conjunto de valores

Consideremos ahora la situación en que una variable  $x_j$  sólo puede tomar ciertos valores bien definidos:  $x_j \in \{a_1, a_2, \dots, a_n\}$ . En este caso, debemos definir:

$$y_{ij} \in (0,1) : \quad \begin{aligned} 1 &\rightarrow \text{Si } x_j = a_i \\ 0 &\rightarrow \text{Si } x_j \neq a_i \end{aligned}$$

Además,  $x_j$  vendrá dada por:

$$x_j = \sum_{i=1}^n a_i \cdot y_{ij}$$

Dado que  $x_j$  sólo puede tomar **un** valor en el conjunto, debemos definir la siguiente restricción:

$$\sum_{i=1}^n y_{ij} = 1 \quad \forall j$$

## 4.2 Modelización con variables binarias

---

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.5 Restricciones excluyentes (una u otra)

*Examinaremos esta situación a través de un ejemplo:*

Consideremos que existen 2 restricciones de las cuales se requiere que sólo una de ellas sea satisfecha:

$$3x_1 + 2x_2 \leq 18$$

ó

$$5x_1 + 4x_2 \leq 16$$

Esta restricción no está en formato de programación matemática pues en él se asume que deben cumplirse **TODAS** las restricciones.

## 4.2 Modelización con variables binarias

---

**Situaciones frecuentes que pueden modelarse con variables binarias:**

$y \in \{0,1\}$  :  $0 \rightarrow$  Si la restricción (1) es la que se cumple  
 $1 \rightarrow$  Si la restricción (2) es la que se cumple

**M** muy grande

Entonces:

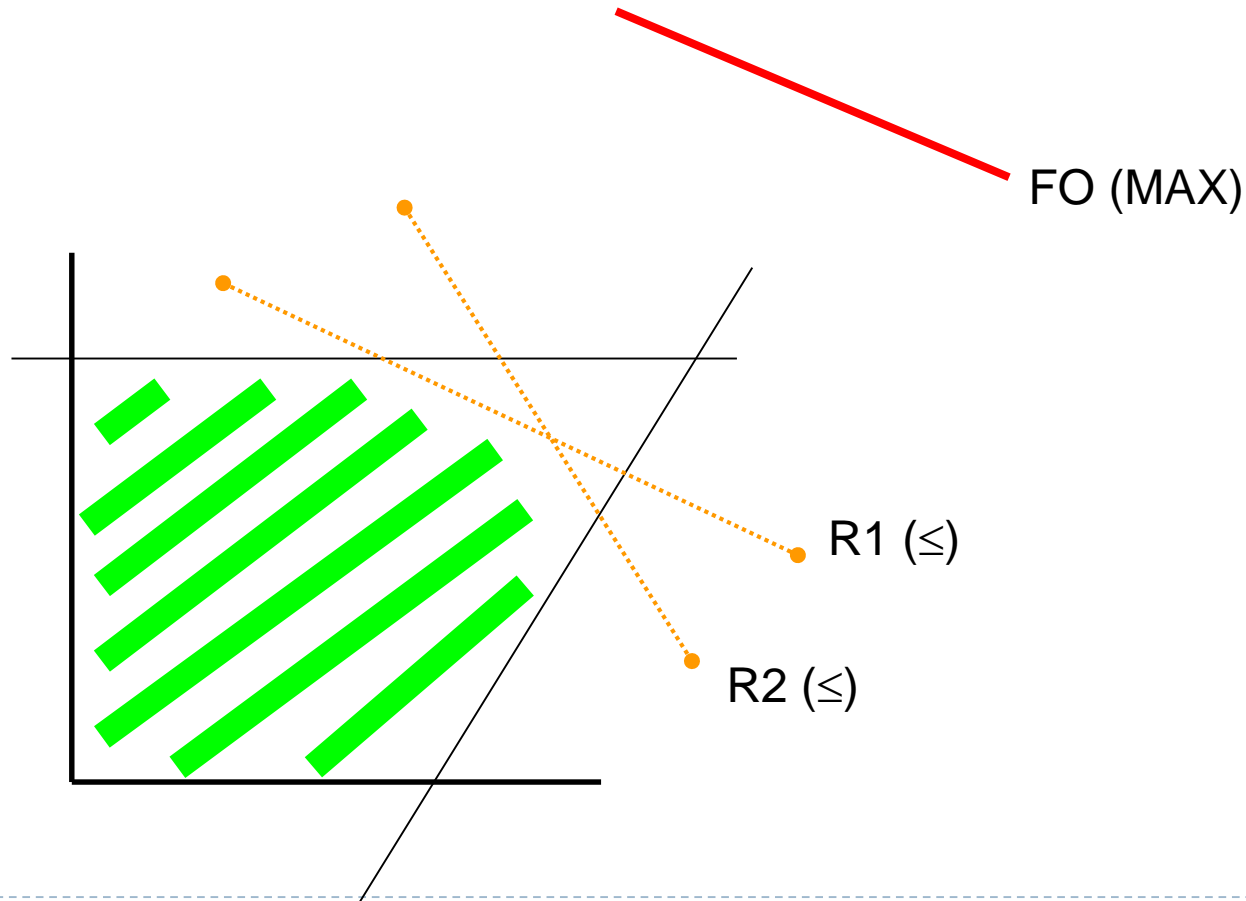
$$(R1) \ 3x_1 + 2x_2 \leq 18 + \mathbf{M} \cdot y$$

$$(R2) \ 5x_1 + 4x_2 \leq 16 + \mathbf{M} \cdot (1 - y)$$

## 4.2 Modelización con variables binarias

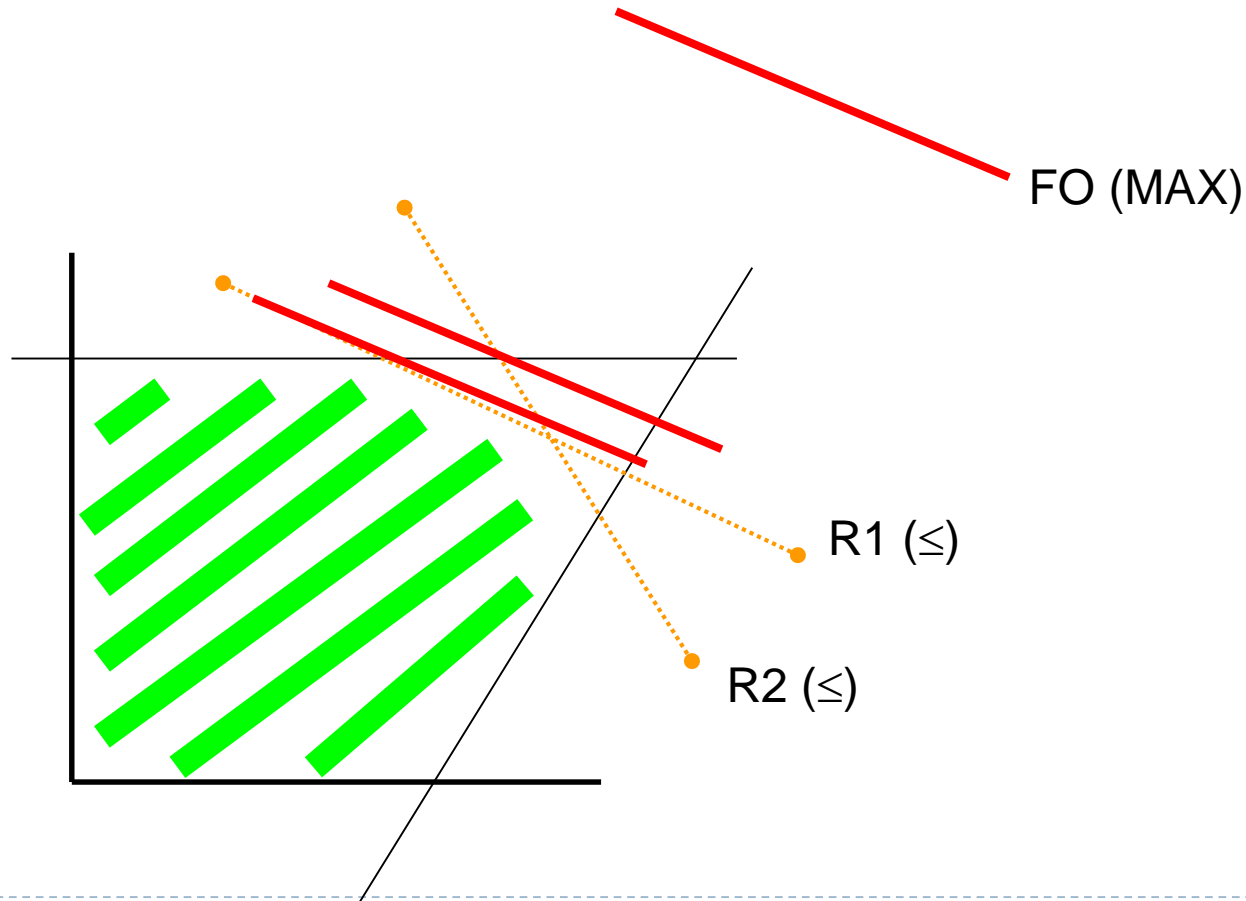
**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.5 Restricciones excluyentes (una u otra)



## 4.2 Modelización con variables binarias

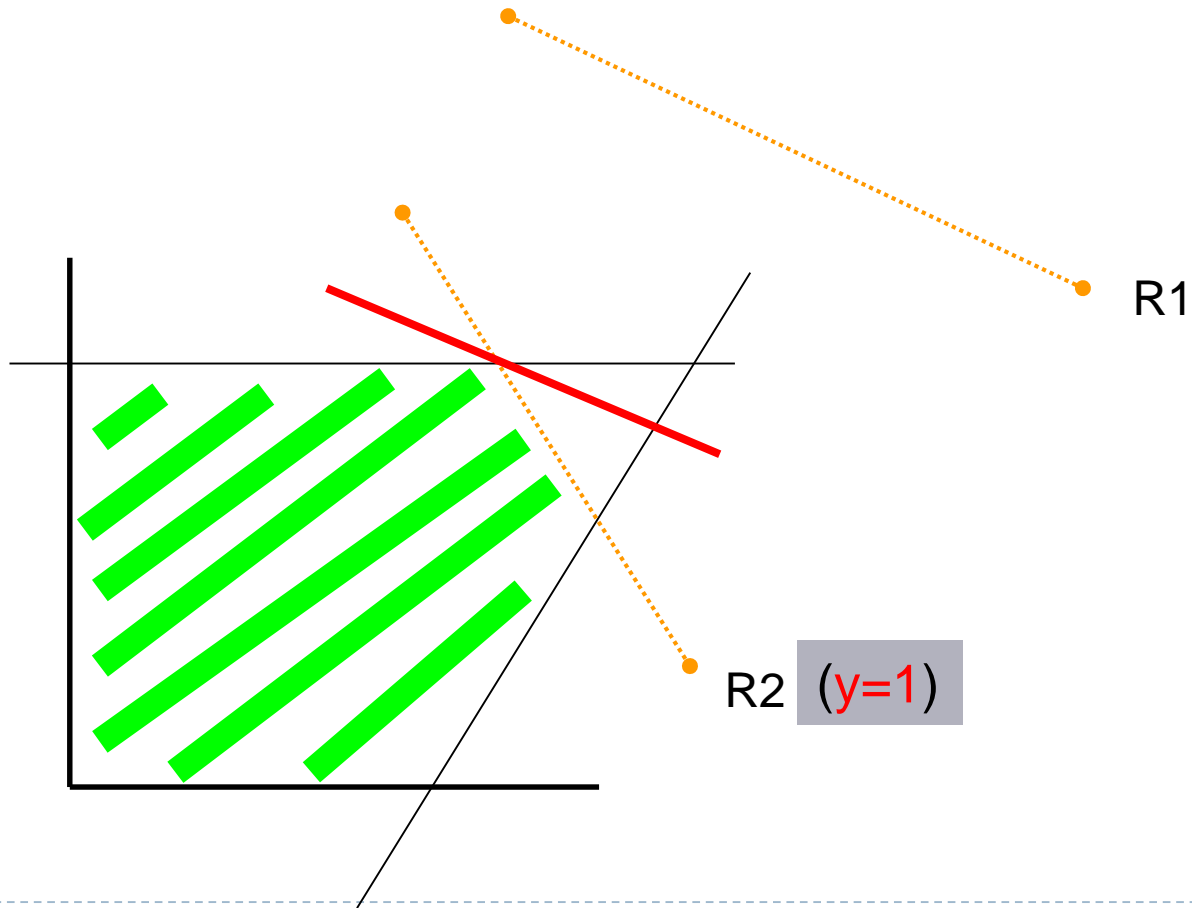
**Situaciones frecuentes que pueden modelarse con variables binarias:**  
**4.2.5 Restricciones excluyentes (una u otra)**



## 4.2 Modelización con variables binarias

Situaciones frecuentes que pueden modelarse con variables binarias:

### 4.2.5 Restricciones excluyentes (una u otra)





## 4.2 Modelización con variables binarias

### Situaciones frecuentes que pueden modelarse con variables binarias:

Cuando se desea seleccionar entre más de dos restricciones:

$$f_1(X_1, X_2, \dots, X_n) \leq b_1$$

$$f_2(X_1, X_2, \dots, X_n) \leq b_2$$

... ..

$$f_m(X_1, X_2, \dots, X_n) \leq b_m$$

Queremos exigir que:

**Se deben cumplir sólo  $k$  restricciones**

Las restricciones se reformulan como:

$$f_1(X_1, X_2, \dots, X_n) \leq b_1 + \mathbf{M} y_1$$

$$f_2(X_1, X_2, \dots, X_n) \leq b_2 + \mathbf{M} y_2$$

... ..

$$f_m(X_1, X_2, \dots, X_n) \leq b_m + \mathbf{M} y_m$$

Y además:

$$\sum_{j=1}^m Y_j = m - k$$

## 4.2 Modelización con variables binarias

---

**Situaciones frecuentes que pueden modelarse con variables binarias:**

### 4.2.6 Varios segundos miembros de una restricción

En algunos problemas puede ser necesario plantear:

$$f(x_1, x_2, \dots, x_n) \leq d_1, \text{ o } d_2, \dots \text{ o } d_N$$

Su modelización sería la siguiente:

$$f(x_1, \dots, x_N) \leq \sum_{i=1}^N d_i \cdot y_i$$

$$\sum_{i=1}^N y_i = 1 \quad \text{donde } y_i \text{ es una variable (0,1)}$$

## 4.3.2 Un problema de mezclas (I)

### » Un problema de elaboración de cerveza

Supongamos que se desea **producir un tipo de cerveza** caracterizado por un contenido en alcohol del 3,1 %, así como por otras cualidades técnicas que se recogen en el cuadro, a base de 4 tipos estándar de cerveza cuyo stock se supone ilimitado y de una cierta proporción de agua, aunque ésta no es absolutamente necesaria.

#### Características de los distintos componentes

Características	Requerimientos	Componentes				
		Agua	Cervezas			
		1	2	3	4	5
% alcohol	3,1	0	2,5	3,7	4,5	5,8
Densidad	1,034-1,040	1	1,030	1,043	1,050	1,064
Color uni. EBC	8-10	0	11	9	8	7
Isomulina mgr/Hlitro	20-25	0	30	20	28	30
Coste u.m./Hl		0	44	50	64	90

## 4.3.2 Un problema de mezclas (I)

---

A efectos de no diversificar la producción y motivado por políticas de compra se **limita a dos el número máximo de tipos estándar de cerveza que han de integrar la mezcla.**

En el caso de que el componente número 5 entre a formar parte de la mezcla se precisa la instalación de una maquinaria especial cuyo coste de instalación asciende a 5000 u.m.. La producción de la mezcla ha de ser de 100 Hl.

Formula un modelo que permita determinar qué tipos estándar de cerveza y agua y en qué cantidades se necesita que integren la mezcla para **minimizar el coste total** de los componentes, en la producción de 100 Hectolitros de cerveza.

## 4.3.1 Un problema de mezclas (II)

---

### » Un problema con costes lineales por tramos

En una fundición se desea producir al mínimo coste 1000 Kg de una aleación especial con las características químicas de un máximo del 3 % de hierro, del 5 % de cobre, del 2 % de manganeso, del 1,5 % de magnesio, de un mínimo del 75 % de aluminio y entre el 12,5 y 15 % de silicio.

Para ello se dispone de 5 tipos de chatarra, subproductos de la misma fundición y de dos metales como el aluminio y el silicio que es preciso comprar. En la siguiente tabla se recogen las características técnicas y comerciales (precio de coste unitario y disponibilidad máxima) de estos siete componentes, de tal forma que hay un límite máximo de disponibilidad en los 5 subproductos, exigiéndose además un consumo mínimo de 200 Kg. en el subproducto 3 y de 50 Kg en el subproducto 4.

## 4.3.1 Un problema de mezclas (II)

### Características técnicas y comerciales de los distintos componentes

Componente	Características técnicas						Características comerciales	
	Fe	Cu	Mn	Mg	Al	Si	Coste unitario por Kg.	Disponibilidad máxima en Kg.
Subprod.1	0,15	0,03	0,02	0,02	0,70	0,02	0,03	100
Subprod.2	0,04	0,05	0,04	0,03	0,75	0,06	0,08	1250
Subprod.3	0,02	0,08	0,01	-	0,80	0,08	0,17	400
Subprod.4	0,04	0,02	0,02	-	0,75	0,12	0,12	350
Subprod.5	0,02	0,06	0,02	0,01	0,80	0,02	0,15	750
Aluminio	0,01	0,01	-	-	0,97	0,01	0,21 (0,13)	ilimitad
Silicio	0,03	-	-	-	-	0,97	0,38	ilimitad

## 4.3.1 Un problema de mezclas (II)

---

- a) Plantea un modelo que permita determinar las cantidades de cada uno de los componentes a incorporar a la mezcla para **minimizar su coste**.
- b) En la compra de aluminio el precio unitario es de 0,21 u.m. si el número de Kg no es superior a 100, siendo de 0,13 u.m. para la cantidad que sobrepase los 100 kg. Modifica el modelo matemático de modo que se contemple esta situación.
- c) Si el precio de 0,21 u.m. por Kg debe aplicarse a una compra de hasta 100 Kg. mientras que el precio del Kg. será 0.13 u.m. si se sobrepasa esta cantidad. ¿Cómo se reflejaría esta situación?

### 4.3.3 Un problema de distribución comercial

---

Una empresa tiene **cuatro centros de producción** en **Lugo**, **Bilbao**, **Barcelona** y **Valencia**. Estas fábricas tienen una capacidad máxima de 40, 20, 40 y 30 unidades respectivamente en la producción del artículo que comercializan. Además, es necesario cubrir una producción mínima de 10 unidades en la fábrica de Lugo y de 15 en la de Bilbao, pero ésta última sólo en el caso de que se decida utilizar. Asimismo, se debe decidir la utilización o no de la fábrica de Valencia, que precisa una inversión de 150 u.m. en el caso afirmativo.

La empresa dispone de **dos centros de consumo**, localizados en **Valladolid** y **Madrid** con una capacidad máxima de recepción de 50 y 100 unidades respectivamente, siendo necesario cubrir la demanda de 70 unidades en el centro de Madrid y de 20 en Valladolid. La fábrica de Barcelona sólo puede enviar el producto a un único centro de destino.



### 4.3.3 Un problema de distribución comercial

---

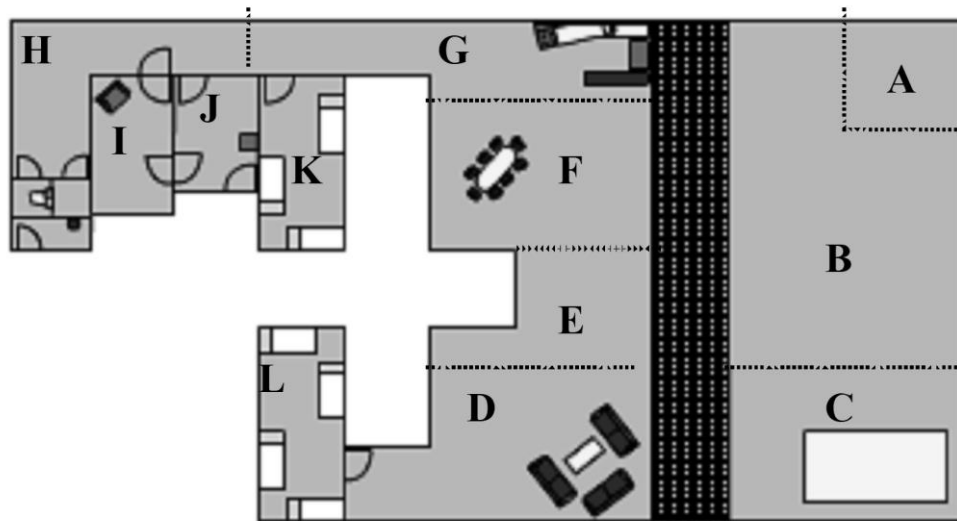
#### Costes de transporte unitarios

ORIGEN	DESTINO	
	VALLADOLID	MADRID
LUGO	30	55
BILBAO	30	35
BARCELONA	45	50
VALENCIA	40	35

Formula un modelo para determinar la cantidad de producto a transportar desde cada centro de origen a cada centro de consumo de forma que se minimicen los costes totales de la empresa.

## 4.3.4 Un problema de cubrimiento

La cadena de televisión **TVSAT** está preparando un nuevo programa de TV. El programa consiste en que los espectadores pueden ver a través de sus televisores todo lo que hacen los concursantes, “encerrados” durante tres meses en una casa. Los técnicos del programa han dividido la casa en 12 sectores, que han sido nombrados con letras (A ... L) y que se muestran en la siguiente figura:



## 4.3.4 Un problema de cubrimiento

Además se han establecido una serie de puntos estratégicos en los que sería posible situar una cámara móvil por control remoto. Estos puntos han sido numerados del 1 al 13 y en la siguiente tabla se muestra el sector o sectores que quedarían cubiertos si la cámara se instalase en ese punto y el coste de instalar dicha cámara (coste de la cámara más el coste de instalación):

<b>Cámara</b>	<b>Coste (euros)</b>	<b>Sectores cubiertos</b>
<b>1</b>	<b>350</b>	<b>A, G</b>
<b>2</b>	<b>350</b>	<b>B, C</b>
<b>3</b>	<b>350</b>	<b>C, D</b>
<b>4</b>	<b>400</b>	<b>A, B, E</b>
<b>5</b>	<b>450</b>	<b>B, E, F</b>
<b>6</b>	<b>300</b>	<b>G, F</b>
<b>7</b>	<b>300</b>	<b>G, H</b>
<b>8</b>	<b>300</b>	<b>H, I</b>
<b>9</b>	<b>250</b>	<b>H, J</b>
<b>10</b>	<b>225</b>	<b>G, K</b>
<b>11</b>	<b>225</b>	<b>D, L</b>
<b>12</b>	<b>175</b>	<b>K</b>
<b>13</b>	<b>225</b>	<b>I, J</b>

## 4.3.4 Un problema de cubrimiento

---

Por motivos técnicos, las cámaras 2 y 5 son incompatibles entre sí. Además, si se instala la cámara 6, necesariamente se debe instalar también la cámara 7.

Plantea un modelo matemático que permita decidir a los responsables del programa la instalación de mínimo coste que permita visualizar cualquiera de los sectores de la casa.

## 4.3.5 Un problema de secuenciación de tripulaciones aéreas

Una compañía aérea necesita asignar sus tripulaciones para cubrir todos sus vuelos programados. En particular, desea asignar tres tripulaciones, con base en Valencia, a los vuelos indicados en la siguiente tabla:

Vuelo	Secuencia factible de vuelos							
	1	2	3	4	5	6	7	8
Valencia a Madrid	1		1			1		
Valencia a Mallorca				1			1	1
Valencia a Lanzarote		1			1			
Madrid a Alicante			2			2		
Madrid a Valencia	2				3			5
Alicante a Mallorca			3	3				
Alicante a Lanzarote						3	3	3
Mallorca a Valencia			4	4				
Mallorca a Alicante				2			2	2
Lanzarote a Valencia		2				4	4	
Lanzarote a Madrid					2			4
Coste	1	3	5	6	4	5	7	7

### 4.3.5 Un problema de secuenciación de tripulaciones aéreas

---

- ▶ En dicha tabla, las columnas numeradas de la 1 a la 8 muestran ocho secuencias factibles de vuelo para una tripulación, siendo necesario elegir tres de esas secuencias (una por tripulación) de tal manera que se cubran la totalidad de vuelos programados (se permite tener más de una tripulación en un vuelo; en tal caso los miembros de la tripulación adicional volarán como pasajeros percibiendo su vuelo como si estuvieran trabajando). Los números que aparecen en las casillas indican a su vez el orden de los vuelos. En la última fila aparece el coste de asignar una tripulación dada a una secuencia de vuelos específica.
- ▶ Si el objetivo de la compañía consiste en **minimizar el coste total** de asignar las tres tripulaciones cubriendo la totalidad de vuelos, plantear el correspondiente **modelo de programación lineal** que permita reflejar la situación a la que se enfrenta dicha compañía aérea.

## 4.3.5 Un problema de secuenciación de tripulaciones aéreas

---

### COMENTARIOS:

- Las **secuencias factibles** de vuelos son equivalentes a las alternativas de corte en un problema de corte de materias primas.
- Se trata entonces de **determinar cuáles de las secuencias que se escogen de modo que todos los vuelos estén cubiertos.**
- Adicionalmente, en este caso y a diferencia de lo que ocurría en el problema de corte de materias primas que se vio en modelización lineal, se desea **limitar a 3 el número de secuencias utilizadas** (o lo que es equivalente, el número de tripulaciones utilizadas para garantizar el cubrimiento de todos los vuelos).

## 4.4 Lenguajes de Modelización

---

### FORMULACIÓN DE MODELOS DE PROGRAMACIÓN MATEMÁTICA DE GRAN ESCALA:

Los modelos de programación matemática pueden ser de diferentes tamaños. Aunque algunos pueden incluir un pequeño conjunto de restricciones funcionales, en la práctica los modelos requieren cientos y hasta miles de restricciones. El número de variables decisión es incluso mayor que el de **restricciones** y ocasionalmente puede ser de **cientos de miles**. La formulación de modelos de tal dimensión resulta inabordable aun cuando los parámetros puedan tomarse desde hoja de cálculo. Es imprescindible utilizar un **lenguaje de modelización**.



## 4.4 Lenguajes de Modelización

---

- ❑ Los lenguajes de modelización son sistemas software específicamente diseñados para la formulación eficiente de modelos de programación matemática de gran tamaño. Se trata de **lenguajes “no procedurales”** sino **lenguajes de especificación** de modo que cuando introducimos un problema sólo podemos expresar lo que queremos, pero no cómo hacerlo, él se ocupa de cómo hacerlo. Con estos lenguajes expresamos nuestro problema de una forma sencilla, muy parecida a la escritura matemática habitual.
- ❑ Los **modelos de gran tamaño** aunque implican miles de restricciones funcionales, éstas son habitualmente de un **número relativamente pequeño de tipos y las restricciones del mismo tipo siguen el mismo patrón**. Por tanto utilizando largos bloques de datos desde hojas de cálculo o desde bases de datos, el lenguaje de modelización formulará todas las restricciones del mismo tipo trabajando simultáneamente con las variables de cada tipo.

## 4.4 Lenguajes de Modelización

---

Además de la formulación eficiente de modelos de gran tamaño, **los lenguajes de modelización permiten agilizar otras tareas** relacionadas con la modelización tales como:

- Acceso de datos
- Transformación de datos en parámetros del modelo
- Modificación del modelo cuando sea necesario
- Análisis de la solución
- Generar informes así como documentos sobre el contenido del modelo.

## 4.4 Lenguajes de Modelización

---

En las últimas décadas se han desarrollado varios lenguajes de modelización, entre los que se encuentran:

- AMPL
- MPL
- GAMS
- LINGO

## 4.4 Lenguajes de Modelización



- ❑ **LINGO** es un lenguaje de modelización diseñado particularmente para la formulación y resolución de una amplia variedad de problemas de optimización incluyendo **programación lineal**, **programación entera** y **programación no lineal**.
- ❑ **LINGO** tiene incorporada una **librería** con **funciones matemáticas**, **estadísticas** y **financieras** que nos permiten expresar fórmulas complejas de una manera clara y sencilla. Podemos representar nuestros **datos** dentro del programa en forma natural como una lista, pero LINGO también puede leer los datos desde ficheros externos o desde hojas de cálculo.

A continuación introduciremos los principios básicos del lenguaje de modelización LINGO. Utilizaremos para ello un ejemplo de planificación de la producción. En primer lugar formularemos el modelo algebraicamente según la sintaxis de LINGO y esta formulación nos servirá de apoyo para introducir los conceptos básicos de modelización con el lenguaje LINGO.

## 4.4 Lenguajes de Modelización



» La empresa **PRODUCTSA** desea determinar la combinación de productos a fabricar semanalmente. Cada unidad de producto requiere una determinada cantidad de tiempo de producción en tres máquinas. Cada máquina tiene un número de horas de producción disponibles por semana y cada producto tiene su beneficio asociado.

En la siguiente tabla se recogen los datos anteriormente mencionados.

	TIEMPO DE PRODUCCIÓN POR UNIDAD (hrs)				Disponibilidad semanal (hrs)
	PRODUCTO				
MAQUINA	P1	P2	P3	P4	
Alisadora	1.7	2.1	1.4	2.4	28
Cortadora	1.1	2.5	1.7	2.6	34
Soldadora	1.6	1.3	1.6	0.8	21
Beneficio unitario	26	35	25	37	

El objetivo a considerar es determinar cuál es la cantidad a producir de cada producto de modo que se **maximice el beneficio** sin sobrepasar la capacidad de producción de cada máquina.

## 4.4 Lenguajes de Modelización



**Algebráicamente** el modelo anterior queda formulado de la siguiente forma:

*Sea:  $x_j$  = unidades fabricadas del producto  $P_j$*

*$c_j$  = beneficio por unidad del producto  $P_j$*

*$a_{ij}$  = tiempo de producción en la **máquina  $i$**  por unidad del **producto  $P_j$***

*$b_i$  = tiempo de producción disponible por semana en la **máquina  $i$***

La función objetivo a considerar será:

$$\text{Maximize} \quad \sum_{j=1}^4 c_j x_j$$

y las restricciones:

$$\sum_{j=1}^4 a_{ij} x_j \leq b_i \quad \text{for } i=1, 2, 3;$$

## 4.4 Lenguajes de Modelización



En general, un programa LINGO está formado por tres partes o secciones:

1. Sección **SETS**, que especifica los sets o conjuntos de objetos del modelo y sus atributos. Puede entenderse como la descripción de las estructuras de datos del problema.
2. Sección **DATA** que incluye tanto los datos a utilizar como la forma de obtenerlos.
3. Una sección que incluye el propio **modelo matemático**.

Opcionalmente, antes del modelo matemático, se puede incluir la sección **INIT**. Esta sección sirve para introducir valores aproximados de los atributos. Esto puede ayudar a LINGO en el cálculo de los valores que buscamos, ya que, en general LINGO sólo proporciona óptimos locales (programación no lineal).

## 4.4 Lenguajes de Modelización



Como ya se ha comentado, la potencia de los lenguajes de modelización se encuentra en que en un **número reducido de líneas se pueden expresar una gran cantidad de restricciones**. La clave de esta propiedad se encuentra en el concepto de **conjunto**.

Los **conjuntos** (**SETS**) son simplemente grupos de objetos relacionados. Un conjunto puede ser, por ejemplo, una lista de productos, tareas o almacenes.

Cada elemento del conjunto puede tener una o más características asociadas a él; las llamamos atributos. Estos atributos pueden ser datos o pueden ser variables. LINGO reconoce dos tipos de conjuntos:

- ❑ **Conjunto primitivo** es el que está compuesto por objetos que no pueden reducirse más. *Por ejemplo*, un conjunto de 5 almacenes o un conjunto de 200 alumnos.
- ❑ **Conjunto derivado** es el que se puede construir a partir de conjuntos primitivos. *Por ejemplo*, el conjunto formado por las **rut**as entre cinco fábricas y diez ciudades es un conjunto derivado de los conjuntos de **fábricas** y **ciudades** que son dos conjuntos primitivos. También pueden construirse conjuntos derivados usando otros conjuntos derivados.



## 4.4 Lenguajes de Modelización



La **sección SETS** se utiliza para definir conjuntos. Esta sección empieza con **SETS:** y acaba con **ENDSETS**. Entre estas dos palabras se definen los conjuntos que se van a utilizar.

Definimos un conjunto mediante su nombre, sus miembros y sus atributos. La sintaxis es:

**SETS:**

nombre: atributos;

**ENDSETS**

**SETS:**

Maquina /alisadora, cortadora, soldadora/ : HrsDisponibles;

Producto /P1, P2, P3, P4/ : Beneficio, **UProducidas**;

**ENDSETS**

Los miembros de un SET se pueden enumerar separados por comas (","), o utilizando rangos (por ejemplo 1..8). Los atributos (características que nos interesan de estos conjuntos), si los hubiera, también se separan mediante comas. La definición de un set acaba en ";"

## 4.4 Lenguajes de Modelización



Otra información clave en el ejemplo es el **número de horas de producción que cada unidad de producto utiliza en cada una de las máquinas**. Este número puede considerarse con un atributo para los miembros del SET de todas las combinaciones de máquina y producto. Dado que este *SET* se deriva de los dos sets anteriores, se hace referencia al mismo como un **SET derivado**. Se definirá del siguiente modo:

**MaPr(Maquina, Producto): HrsProdUtilizadas;**

- ❑ Las variables asociadas a conjuntos, es decir, los atributos, deben declararse antes de ser usadas.
- ❑ Los conjuntos derivados deben declararse después de los correspondientes conjuntos primitivos.

## 4.4 Lenguajes de Modelización



Utilizando el lenguaje de especificación **LINGO**, el modelo resultante es el siguiente:

**!Ejemplo de Planificación de la producción;**

**SETS:**

**Maquina / alisadora, cortadora, soldadora /: HrsDisponibles;**

**Producto / P1, P2, P3, P4/: Beneficio, UProducidas;**

**MaPr(Maquina, Producto): HrsProdUtilizadas;**

**ENDSETS**

## 4.4 Lenguajes de Modelización



En general, un programa LINGO está formado por tres partes o secciones:

1. Sección **SETS**, que especifica los sets o conjuntos de objetos del modelo y sus atributos. Puede entenderse como la descripción de las estructuras de datos del problema.
2. Sección **DATA** que incluye tanto los datos a utilizar como la forma de obtenerlos.
3. Una sección que incluye el propio **modelo matemático**.

## 4.4 Lenguajes de Modelización



La **sección DATA** nos sirve para introducir valores a miembros de los sets y a los atributos. Esta sección empieza con **DATA:** y acaba con **ENDDATA**. La sintaxis es:

**DATA:**

**Set = lista de valores;**

**atributo = lista de valores ;**

**ENDDATA**

**DATA:**

**HrsDisponibles = 28 34 21;**

**Beneficio =26 35 25 37;**

**HrsProdUtilizadas=1.7 2.1 1.4 2.4 !alisadora;**

**1.1 2.5 1.7 2.6 !cortadora;**

**1.6 1.3 1.6 0.8; !soldadora;**

**ENDDATA**

Los **valores se separan por** comas o por espacios en blanco. Es preferible separarlos por **comas** porque cuando no queramos introducir un valor podemos dejar un hueco en blanco entre comas. Si queremos no explicitar un dato hasta el momento de ejecutar el programa podemos poner el signo “?” en su lugar; en ese momento LINGO nos preguntará por el correspondiente valor.

## 4.4 Lenguajes de Modelización



Utilizando el lenguaje de especificación **LINGO**, el modelo resultante es el siguiente:

**!Ejemplo de Planificación de la producción;**

**SETS:**

**Maquina / alisadora, cortadora, soldadora /: HrsDisponibles;**

**Producto / P1, P2, P3, P4/: Beneficio, UProducidas;**

**MaPr(Maquina, Producto): HrsProdUtilizadas;**

**ENDSETS**

**DATA:**

**HrsDisponibles = 28 34 21;**

**Beneficio =26 35 25 37;**

**HrsProdUtilizadas=        1.7 2.1 1.4 2.4 !alisadora;  
                                 2.5 1.7 2.6 !cortadora;  
                                 1.3 1.6 0.8; !soldadora;**

**ENDDATA**

## 4.4 Lenguajes de Modelización



En general, un programa LINGO está formado por tres partes o secciones:

1. Sección **SETS**, que especifica los sets o conjuntos de objetos del modelo y sus atributos. Puede entenderse como la descripción de las estructuras de datos del problema.
2. Sección **DATA** que incluye tanto los datos a utilizar como la forma de obtenerlos.
3. **Una sección que incluye el propio modelo matemático.**

## 4.4 Lenguajes de Modelización



Para **especificar el modelo matemático** asociado al problema se utilizan **funciones** que permiten aplicar operaciones a los miembros de un set:

**function(set | condition: expresion);**

La parte “| condición” es optativa y sirve para restringir el “set”.

**@SUM(set: expresion)** Devuelve la suma de la expresión especificada sobre **set**.

**@SUM (Producto(j): Beneficio(j)\* UProducidas(j))**

Esta función suma la expresión que aparece a continuación del símbolo “:” sobre todos los miembros del set que aparece antes de “:”. Por tanto, esta función @SUM proporcionaría el valor correspondiente a la función objetivo.



## 4.4 Lenguajes de Modelización



**@SUM (Producto(j): Beneficio(j)\* UProducidas(j))**



**26 UPRODUCIDAS( P1) +  
35 UPRODUCIDAS( P2) +  
25 UPRODUCIDAS( P3) +  
37 UPRODUCIDAS( P4)**

## 4.4 Lenguajes de Modelización



**@FOR(set : constraint)** Genera restricciones independientes para cada elemento del conjunto **set**.

**@FOR(Maquina(i):**

**[Capacidad]@SUM(Producto(j):HrsProdUtilizadas(i,j)\*UProducidas(j))**  
**<= HrsDisponibles(i););**

CAPACIDAD( ALISADORA) 1.7 UPRODUCIDAS( P1) + 2.1 UPRODUCIDAS( P2)  
+ 1.4 UPRODUCIDAS( P3) + 2.4 UPRODUCIDAS( P4) <= 28

CAPACIDAD( CORTADORA) 1.1 UPRODUCIDAS( P1) + 2.5 UPRODUCIDAS( P2)  
+ 1.7 UPRODUCIDAS( P3) + 2.6 UPRODUCIDAS( P4) <= 34

CAPACIDAD( SOLDADORA) 1.6 UPRODUCIDAS( P1) + 1.3 UPRODUCIDAS( P2)  
+ 1.6 UPRODUCIDAS( P3) + .8 UPRODUCIDAS( P4) <= 21

## 4.4 Lenguajes de Modelización



Utilizando el lenguaje de especificación **LINGO**, el modelo resultante es el siguiente:

**!Ejemplo de Planificación de la producción;**

**SETS:**

**Maquina / alisadora, cortadora, soldadora /: HrsDisponibles;**

**Producto / P1, P2, P3, P4/: Beneficio, UProducidas;**

**MaPr(Maquina, Producto): HrsProdUtilizadas;**

**ENDSETS**

**DATA:**

**HrsDisponibles = 28 34 21;**

**Beneficio =26 35 25 37;**

**HrsProdUtilizadas=        1.7 2.1 1.4 2.4 !alisadora;  
                                 2.5 1.7 2.6 !cortadora;  
                                 1.3 1.6 0.8; !soldadora;**

**ENDDATA**

## 4.4 Lenguajes de Modelización



Utilizando el lenguaje de especificación **LINGO**, el modelo resultante es el siguiente:

**!Maximizar el beneficio total;**

**MAX=@SUM(Producto(j):Beneficio(j)\*UProducidas(j));**

**!Para cada máquina, las horas utilizadas deben ser <= horas disponibles;**

**@FOR(Maquina(i):**

**[Capacidad]@SUM(Producto(j):HrsProdUtilizadas(i,j)\*UProducidas(j))  
                  <= HrsDisponibles(i));**

## 4.4 Lenguajes de Modelización



The screenshot shows the LINGO software window titled "LINGO - LINGO Model - EJEMPLO\_PRODUCCION2". The window has a menu bar (File, Edit, LINGO, Window, Help) and a toolbar with various icons. The main text area contains the following LINGO code:

```
!Ejemplo de Planificación de la producción;
!Definición de Sets primitivos;
SETS:
Maquina /alisadora, cortadora, soldadora/: HrsDisponibles;
Producto /P1, P2, P3, P4/: Beneficio, UProducidas;
!Definición de Sets Derivados;
MaPr(Maquina, Producto): HrsProdUtilizadas;
ENDSETS
DATA:
!Horas disponibles de cada máquina;
HrsDisponibles = 28 34 21;
!Beneficio unitario;
Beneficio =26 35 25 37;
!Horas de máquina necesarias por unidad de producto;
HrsProdUtilizadas= 1.7 2.1 1.4 2.4 !alisadora;
                   1.1  2.5 1.7 2.6 !cortadora;
                   1.6  1.3 1.6 0.8; !soldadora;

ENDDATA
!Maximizar el beneficio total;
MAX=@SUM(Producto(j):Beneficio(j)*UProducidas(j));
!Para cada máquina, las horas deben ser <= horas disponibles;
@FOR(Maquina(i):
[Capacidad]@SUM(Producto(j):HrsProdUtilizadas(i,j)*UProducidas(j))
               <= HrsDisponibles(i);
);
```

## 4.4 Lenguajes de Modelización



**MAX** 26 UPRODUCIDAS( P1) + 35 UPRODUCIDAS( P2) + 25  
UPRODUCIDAS( P3) + 37 UPRODUCIDAS( P4)

### **SUBJECT TO**

CAPACIDAD( ALISADORA)] 1.7 UPRODUCIDAS( P1) + 2.1  
UPRODUCIDAS( P2)

+ 1.4 UPRODUCIDAS( P3) + 2.4 UPRODUCIDAS( P4) <= 28

CAPACIDAD( CORTADORA)] 1.1 UPRODUCIDAS( P1) + 2.5  
UPRODUCIDAS( P2)

+ 1.7 UPRODUCIDAS( P3) + 2.6 UPRODUCIDAS( P4) <= 34

CAPACIDAD( SOLDADORA)] 1.6 UPRODUCIDAS( P1) + 1.3  
UPRODUCIDAS( P2)

+ 1.6 UPRODUCIDAS( P3) + .8 UPRODUCIDAS( P4) <= 21

**END**

## 4.4 Lenguajes de Modelización



**Solution Report - EJEMPLO\_PRODUCCION2**

Global optimal solution found at step: 6  
Objective value: 475.0000

Variable	Value	Reduced Cost
HRSDISPONIBLES( ALISADORA)	28.00000	0.0000000
HRSDISPONIBLES( CORTADORA)	34.00000	0.0000000
HRSDISPONIBLES( SOLDADORA)	21.00000	0.0000000
BENEFICIO( P1)	26.00000	0.0000000
BENEFICIO( P2)	35.00000	0.0000000
BENEFICIO( P3)	25.00000	0.0000000
BENEFICIO( P4)	37.00000	0.0000000
UPRODUCIDAS( P1)	0.0000000	3.577921
UPRODUCIDAS( P2)	10.00000	0.0000000
UPRODUCIDAS( P3)	5.000000	0.0000000
UPRODUCIDAS( P4)	0.0000000	1.441558
HRSPRODUTILIZADAS( ALISADORA,	1.700000	0.0000000
HRSPRODUTILIZADAS( ALISADORA,	2.100000	0.0000000
HRSPRODUTILIZADAS( ALISADORA,	1.400000	0.0000000
HRSPRODUTILIZADAS( ALISADORA,	2.400000	0.0000000
HRSPRODUTILIZADAS( CORTADORA,	1.100000	0.0000000
HRSPRODUTILIZADAS( CORTADORA,	2.500000	0.0000000
HRSPRODUTILIZADAS( CORTADORA,	1.700000	0.0000000
HRSPRODUTILIZADAS( CORTADORA,	2.600000	0.0000000
HRSPRODUTILIZADAS( SOLDADORA,	1.600000	0.0000000
HRSPRODUTILIZADAS( SOLDADORA,	1.300000	0.0000000
HRSPRODUTILIZADAS( SOLDADORA,	1.600000	0.0000000
HRSPRODUTILIZADAS( SOLDADORA,	0.8000000	0.0000000

Row	Slack or Surplus	Dual Price
1	475.0000	1.000000
CAPACIDAD( ALISADORA)	0.0000000	15.25974
CAPACIDAD( CORTADORA)	0.5000000	0.0000000
CAPACIDAD( SOLDADORA)	0.0000000	2.272727

## 4.4 Lenguajes de Modelización



### OPERADORES LÓGICOS Y FUNCIONES MATEMÁTICAS EN LINGO:

**#NOT#** : Es un operador unitario, niega el valor lógico de su argumento.

**#AND#** : Devuelve el valor TRUE sólo si sus dos argumentos son TRUE, en otro caso devuelve el valor FALSE.

**#OR#** : Devuelve el valor FALSE sólo si sus dos argumentos son FALSE, en otro caso devuelve el valor TRUE.

**#EQ#** : Devuelve TRUE si sus dos operandos son iguales y FALSE en caso contrario.

**#NE#** : Devuelve TRUE si sus dos operandos no son iguales y FALSE si lo son.

**#GT#** : Devuelve TRUE si el operando izquierdo es mayor que el derecho, y FALSE en caso contrario.

**#GE#** : Devuelve TRUE si el operando izquierdo es mayor o igual que el derecho, y FALSE en caso contrario.

**#LT#** : Devuelve TRUE si el operando izquierdo es menor que el derecho, y FALSE en caso contrario.

**#LE#** : Devuelve TRUE si el operando izquierdo es menor o igual que el derecho, y FALSE en caso contrario.



## 4.4 Lenguajes de Modelización



Otras funciones en **LINGO**:

**@MAX (set: expresion);**

Devuelve el valor máximo de **expresion** tomado sobre **set**.

**@MIN (set: expresion);**

Devuelve el valor mínimo de **expresion** tomado sobre **set**.

**@ABS(X);**                      Devuelve el valor absoluto de X.

**@SMAX(lista);** Devuelve el máximo de un conjunto de valores.

**@SMIN(lista);**              Devuelve el mínimo de un conjunto de valores.

## 4.4 Lenguajes de Modelización



### IMPORTACIÓN Y EXPORTACIÓN DE DATOS A HOJA DE CALCULO CON LINGO:

- ❑ El ejemplo que acabamos de ver es auto-contenido ya que todos los datos estaban incorporados en la formulación de LINGO. Sin embargo en la práctica es frecuente que los datos se encuentren almacenados en otra fuente (habitualmente Hoja de Cálculo) desde la cual necesitan incorporarse al modelo.
- ❑ Para responder a esta cuestión, LINGO dispone del comando **@OLE()**, para cargar o almacenar datos desde o a una hoja de cálculo.

	A	B	C	D	E	F	G	H	I	J
1										
2										
3	<b>EJEMPLO PLANIFICACION DE LA PRODUCCION</b>									
4										
5										
6	BENEFICIO				P1	P2	P3	P4		
7		Beneficio/unidad:			26	35	25	37		
8		Unidades a Producir:								
9										
10										
11	MAQUINA		HrsDisponibles		Hrs. Utilizadas	Hrs. Producción utilizadas por unidad de producto				
12	ALISADORA		28	>=		1,7	2,1	1,4	2,4	
13	CORTADORA		34	>=		1,2	2,5	1,7	2,6	
14	SOLDADORA		21	>=		1,7	1,3	1,6	0,8	
15										

## 4.4 Lenguajes de Modelización



### IMPORTACIÓN Y EXPORTACIÓN DE DATOS A HOJA DE CALCULO CON LINGO:

Un paso previo para poder utilizar de forma efectiva la función **@OLE()** es la asignación de los **nombres apropiados** a los rangos de datos correspondientes

The screenshot shows a spreadsheet with a table of production hours. The table has columns labeled P1, P2, P3, and P4. The first row contains values 26, 35, 25, and 37. Below this is a green header row, followed by a row with the labels 'Hrs. Utilizadas' and 'Hrs. Producción utilizadas por unidad'. The data rows are:

Hrs. Utilizadas	Hrs. Producción utilizadas por unidad	P1	P2	P3	P4
0	1,7	2,1	1,4		
0	1,2	2,5	1,7		
0	1,7	1,3	1,6		

A context menu is open over the table, with the option 'Asignar nombre a un rango...' highlighted.

The 'Nombre nuevo' dialog box is shown. It has the following fields:

- Nombre: UProducidas
- Ámbito: Libro
- Comentario: (empty)
- Hace referencia a: =Hoja1!\$E\$8:\$H\$8

Buttons: Aceptar, Cancelar

## 4.4 Lenguajes de Modelización



### IMPORTACIÓN Y EXPORTACIÓN DE DATOS A **HOJA DE CALCULO** CON **LINGO**:

Fichero disponible en la dirección **C :\Lingo\EjemploProduccion.xls**:

#### **DATA:**

!Se incorporan los datos al modelo... ;

Maquina=@OLE('C:\Lingo\EjemploProduccion.xls ');

HrsDisponibles=@OLE('C:\Lingo\EjemploProduccion.xls ');

Producto=@OLE('C:\Lingo\EjemploProduccion.xls ');

Beneficio=@OLE('C:\Lingo\EjemploProduccion.xls ');

HrsProdUtilizadas=@OLE('C:\Lingo\EjemploProduccion.xls');

!Se guarda la solución en EXCEL;

@OLE('C:\Lingo\EjemploProduccion.xls')=UProducidas;


**ENDDATA**

## 4.4 Lenguajes de Modelización



### IMPORTACIÓN Y EXPORTACIÓN DE DATOS A **HOJA DE CALCULO** CON **LINGO**:

Por supuesto la solución tras resolver el modelo que toma los datos desde Excel es la misma que se obtuvo inicialmente:



Solution Report - EJEMPLO\_PRODUCCION

Transfer Method:

OLE BASED

Workbook:

C:\MisDocumentos\EjemploProduccion1.xlsx

Ranges Specified:

1

UPRODUCIDAS

Ranges Found:

1

Range Size Mismatches:

0

Values Transferred:

4

	Variable	Value	Reduced Cost
	HRSDISPONIBLES( ALISADORA)	28.00000	0.000000
	HRSDISPONIBLES( CORTADORA)	34.00000	0.000000
	HRSDISPONIBLES( SOLDADORA)	21.00000	0.000000
	BENEFICIO( P1)	26.00000	0.000000
	BENEFICIO( P2)	35.00000	0.000000
	BENEFICIO( P3)	25.00000	0.000000
	BENEFICIO( P4)	37.00000	0.000000
	UPRODUCIDAS( P1)	0.000000	3.805195
	UPRODUCIDAS( P2)	10.00000	0.000000
	UPRODUCIDAS( P3)	5.000000	0.000000
	UPRODUCIDAS( P4)	0.000000	1.441558
	HRSPRODUTILIZADAS( ALISADORA,	1.700000	0.000000
	HRSPRODUTILIZADAS( ALISADORA,	2.100000	0.000000
	HRSPRODUTILIZADAS( ALISADORA,	1.400000	0.000000
	HRSPRODUTILIZADAS( ALISADORA,	2.400000	0.000000
	HRSPRODUTILIZADAS( CORTADORA,	1.200000	0.000000
	HRSPRODUTILIZADAS( CORTADORA,	2.500000	0.000000
	HRSPRODUTILIZADAS( CORTADORA,	1.700000	0.000000
	HRSPRODUTILIZADAS( CORTADORA,	2.600000	0.000000
	HRSPRODUTILIZADAS( SOLDADORA,	1.700000	0.000000
	HRSPRODUTILIZADAS( SOLDADORA,	1.300000	0.000000
	HRSPRODUTILIZADAS( SOLDADORA,	1.600000	0.000000
	HRSPRODUTILIZADAS( SOLDADORA,	0.8000000	0.000000

## 4.4 Lenguajes de Modelización



### IMPORTACIÓN Y EXPORTACIÓN DE DATOS A HOJA DE CALCULO CON LINGO:

Que a su vez transfiere los valores óptimos de las variables a Excel:

	A	B	C	D	E	F	G	H	I
1									
2									
3	<b>EJEMPLO PLANIFICACION DE LA PRODUCCION</b>								
4									
5									
6	BENEFICIO			475	P1	P2	P3	P4	
7		Beneficio/unidad:			26	35	25	37	
8		Unidades a Producir:			0	10	5	0	
9									
10									
11	MAQUINA		HrsDisponibles		Hrs. Utilizadas	Hrs. Producción utilizadas por unidad de producto			
12	ALISADORA		28	>=	28	1,7	2,1	1,4	2,4
13	CORTADORA		34	>=	33,5	1,2	2,5	1,7	2,6
14	SOLDADORA		21	>=	21	1,7	1,3	1,6	0,8
15									