MA4702. Programación Lineal Mixta : Teoría y Laboratorio. 2018.
Profesor: José Soto.
Profesor Auxiliar: Arturo Merino
Profesor Ayudante: Obed Ulloa

Laboratorio 1: Modelamiento de kendokus

- **Reglas:** Este laboratorio tiene una parte presencial **obligatoria** y una parte no-presencial **opcional**.
  - Si su grupo hace solo la parte obligatoria, su nota será 30 % (TI) + 70 % (TP).
  - Si hacen la parte opcional su nota será 30 % (TI) + 40 % (TP) + 30 % (TNP) pero tendrán el siguiente beneficio: podrá volver a entregar si lo desea (una versión corregida de) su tarea y su parte presencial.
- **Entregables:**
  - Cada archivo que entregue o genere mediante un script **debe** comenzar con una línea de identificación:
    `## Archivo (nombrearchivo). Entregado por Grupo (xx): (nombreintegrantes) ##`
    donde `(nombrearchivo)` es el nombre del archivo incluyendo extensión, (xx) es el número de su grupo, y `(nombreintegrantes)` son los nombres y apellidos de cada integrante del grupo.
  - Presencial. Se entregan en ucursos antes de la hora de término del laboratorio los 9 archivos siguientes:
    ◦ `grupoxx-facil.mod` ◦ `grupoxx-facil.run` ◦ `grupoxx-facil-reporte.txt`.
    ◦ `grupoxx-normal.mod` ◦ `grupoxx-normal.run` ◦ `grupoxx-normal-reporte.txt`.
    ◦ `grupoxx-dificil.mod` ◦ `grupoxx-dificil.run` ◦ `grupoxx-dificil-reporte.txt`.
  - No presencial. Se entrega en ucursos antes de las **23:59 del día lunes 2 de abril**, los siguientes archivos:
    ◦ `grupoxx-noop.mod` ◦ `grupoxx-noop.run` ◦ `grupoxx-noop-reporte.txt`.

1. Parte presencial: Kendoku

Un **tablero** M es una matriz en $[N]^{N\times N}$. Los subconjuntos no vacíos de índices $C \subseteq [N]\times[N]$ se llaman **jaulas**. Una jaula C se dice **bloque de M** si todos los elementos $\{M_{ij} : ij \in C\}$ son diferentes. Un par ordenado $(k; \text{op})$, con $k \in \mathbb{N}$ es una **declaración válida** para una jaula C si alguno de los siguientes ocurre:

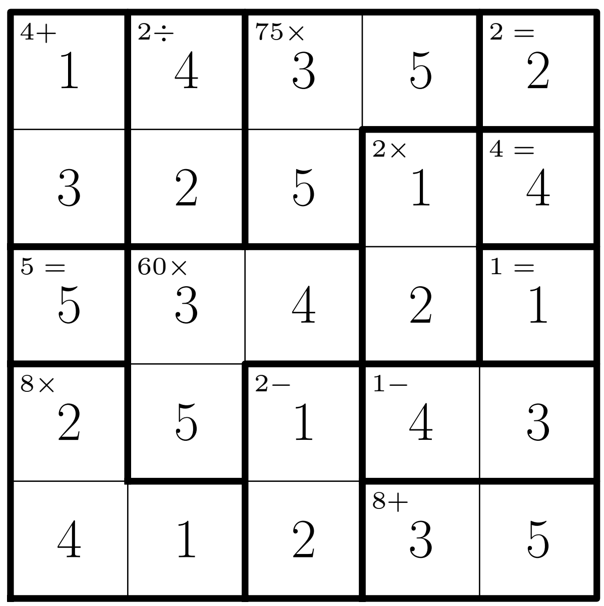
Declaración	**Codificación**	**Condición**
$(k; =)$	$(k, 0)$	$\lvert C\rvert = 1$ y el único elemento $(i, j) \in C$ satisface $M_{ij} = k$.
$(k; +)$	$(k, 1)$	$\sum_{(i,j)\in C} M_{ij} = k$.
$(k; -)$	$(k, 2)$	$\lvert C\rvert = 2$, digamos $C = \{(i, j), (i', j')\}$ y además $\lvert M_{ij} - M_{i'j'}\rvert = k$.
$(k; \times)$	$(k, 3)$	$\prod_{(i,j)\in C} M_{ij} = k$.
$(k; \div)$	$(k, 4)$	$\lvert C\rvert = 2$, digamos $C = \{(i, j), (i', j')\}$ y además $k \in \left\{\dfrac{M_{ij}}{M_{i'j'}}, \dfrac{M_{i'j'}}{M_{ij}}\right\}$.

En palabras: la declaración es válida si es posible combinar los valores presentes en la jaula usando el operador op, de modo de obtener el resultado k. Solo se permite jaulas de tamaño 2 para los operadores $-$ y $\div$, ya que estos no son asociativos. La columna codificación contiene una codificación de las declaraciones que solo usa números naturales. Por ejemplo $(k, 3)$ es una codificación de la declaración $(k; \times)$.

Un **kendoku**, es un tablero M cuyas filas y columnas son bloques, junto una partición de sus casilleros en jaulas $\{C_1, \ldots, C_q\}$ con declaraciones válidas para cada uno. En el **juego del kendoku** recibimos como entrada un tablero vacío particionado en jaulas, junto con una declaración para cada jaula. El objetivo es llenar el tablero con valores en $[N]$ de modo tal que el tablero resultante sea un kendoku.

Las instancias tienen kendokus vacíos con el siguiente formato. La primera linea contiene dos números: el tamaño N y el número de jaulas q. Luego recibirá una matriz de N por N llena con símbolos en $[q]$ que representa la partición del tablero en las q jaulas. Finalmente recibirá q líneas con dos números naturales k, op de modo que la i-ésima línea es la codificación de una declaración para la i-ésima jaula. Por ejemplo, la descripción del kendoku vacío siguiente (guardado en `kendoku-21.txt`) y su kendoku solución se encuentran a continuación:

```
5 13
1  2  3  3  4
1  2  3  5  6
7  8  8  5  9
10 8  11 12 12
10 10 11 13 13
4  1
2  4
75 3
2  0
2  3
4  0
5  0
60 3
1  0
8  3
2  2
1  2
8  1
```



Las siguientes partes son incrementales: La parte 3 contiene a la parte 2 y esta a su vez contiene a la parte 1. Aún así, **no intente hacer la parte 3 sin hacer la parte 2, ni la parte 2 sin hacer la parte 1** ya que así se asegura de cometer menos errores.

Parte 1: Escriba un archivo de modelo `grupoxx-facil.mod` y un script `grupoxx-facil.run`, donde xx es el número de su grupo que realicen lo siguiente.

- Deben permitir resolver kendokus que usen solo declaraciones del tipo $(k, =)$ y $(k, +)$.
- El archivo de modelo debe describir un conjunto lineal binario que represente las condiciones que satisface un kendoku. debe incluir los parámetros N y q asociados a la descripción del kendoku vacío. Use variables binarias $\{x[i, j, k] \colon i, j, k \in N\}$, tal que $x[i, j, k] = 1$ si y solo si $M[i, j] = k$, donde M representa el kendoku resuelto. Si lo desea puede usar variables, parámetros y conjuntos auxiliares.

- El archivo de script debe al menos:

1. Tener la linea de identificación indicada en las instrucciones.
2. Cargar su modelo.
3. Ejecutar los comandos siguientes:

```
option solver cplex; option display_1col 0; option display_width 300;
```

4. Leer y resolver los kendokus de cada archivo `kendoku-nn.txt` con nn entre 01 y 10.
5. Escribir un archivo `grupoxx-facil-reporte.txt` que siga el siguiente formato:

```
## Archivo (nombrearchivo). Entregado por (nombreintegrantes) ##

Instancia: kendoku-01.txt
tiempo de resolucion: ampl (_ampl_time ), solver (_solve_elapsed_time)
solucion:
1 2
2 1

Instancia: kendoku-2.txt
(...)
```

Los archivos a entregar son `grupoxx-facil.mod`, `grupoxx-facil.run` y `grupoxx-facil-reporte.txt`.

Parte 2: Repita las mismas instrucciones anteriores para kendokus que usen declaraciones del tipo $(k,=)$, $(k,+)$ y $(k,-)$. Los archivos de instancias están en `kendoku-nn.txt`, con nn entre 01 y 20 (note que contiene las instancias de la parte anterior)

Los archivos a entregar son `grupoxx-normal.mod`, `grupoxx-normal.run` y `grupoxx-normal-reporte.txt`.

Indicación: si J denota un conjunto con 2 elementos (digamos $J = \{(i1, j1); (i2, j2)\}$), y necesitamos agregar una restricción del tipo **`y[i1,j1]-y[i2,j2]=LADO DERECHO`** en AMPL, puede intentar lo siguiente:

```
(...)
subject to mirestriccion {(i1,j1,i2,j2): (i1,j1) in J and (i2,j2) in J
        and ((i1=i2 and j1<j2) or (i1<i2))} y[i1,j1]-y[i2,j2] = LADO DERECHO;
(...)
```

Parte 3: Repita las mismas instrucciones de la parte anterior para kendokus con todo tipo de declaraciones. Los archivos de instancias están en `kendoku-nn.txt`, con nn entre 01 y 30 (note que contiene las instancias de la parte anterior)

Los archivos a entregar son `grupoxx-dificil.mod`, `grupoxx-dificil.run` y `grupoxx-dificil-reporte.txt`.

Indicación: En AMPL, **`log(t)`** es el comando para obtener el logaritmo natural de **`t`**.

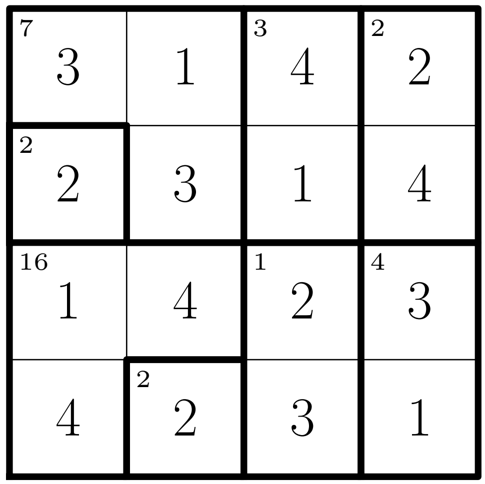
2. Parte no presencial: Kendoku No-op

Kendoku No-op es una variante del juego de kendoku donde se reciben declaraciones incompletas, es decir declaraciones que **no tienen operadores fijos** Es decir, recibimos como entrada un tablero vacío particionado en jaulas, junto con un valor objetivo para cada jaula. El objetivo es llenar el tablero con valores en $[N]$ de modo tal que existan símbolos op para cada jaula que hagan que el tablero resultante sea un kendoku.

Los archivos de instancias tienen formatos similares a los del kendoku tradicional, a excepción que las q líneas que representaban las declaraciones completas de cada jaula son reemplazadas por q líneas que solo tienen los valores objetivos.

Por ejemplo, la descripción del kendoku no-op vacío siguiente (guardado en `noop-01.txt`) y su solución se encuentran a continuación:

```
4 8
1 1 2 3
4 1 2 3
5 5 6 7
5 8 6 7
7
3
2
2
16
2
1
4
```



Parte única: Repita las mismas instrucciones de las partes anteriores para kendokus no-op. Los archivos de instancias están en `noop-nn.txt`, con nn entre 01 y 08.

Los archivos a entregar son `grupoxx-noop.mod`, `grupoxx-noop.run` y `grupoxx-noop-reporte.txt`.

Indicación: Le puede ser útil generalizar la siguiente idea: si A_1 y A_2 son variables acotadas (digamos $0 \leq A_1, A_2 \leq M$), entonces la condición $A_1 = a_1$ ó $A_2 = a_2$ con a_1 y a_2 constantes se puede escribir como

$$\begin{cases} A_1 = z \cdot a_1 + b, \\ A_2 = w \cdot a_2 + c, \\ z + w = 1; \quad z, w \in \{0,1\} \\ 0 \leq b \leq Mw \\ 0 \leq c \leq Mz \end{cases}$$

de ese modo:

$$\begin{array}{lll} z = 0, w = 1 & \text{equivale a} & 0 \leq A_1 \leq M, A_2 = a_2 \\ z = 1, w = 0 & \text{equivale a} & A_1 = a_1, 0 \leq A_2 \leq M. \end{array}$$

Notas: Las instancias fueron extraidas de diversas fuentes. En particular se usaron instancias de `www.calcudoku.org`, `kenkenpuzzle.com`, `www.webkendoku.com`.