

Laboratorio 1

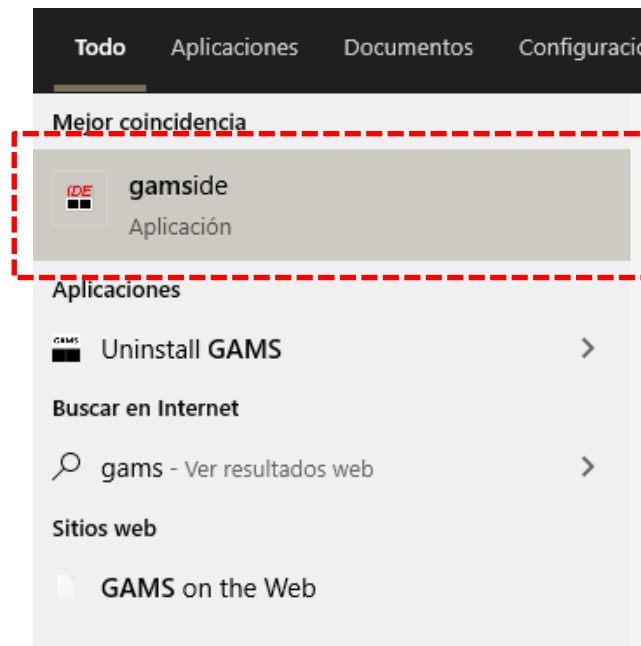
Manejo Básico de GAMS

Modelado, Optimización y Simulación

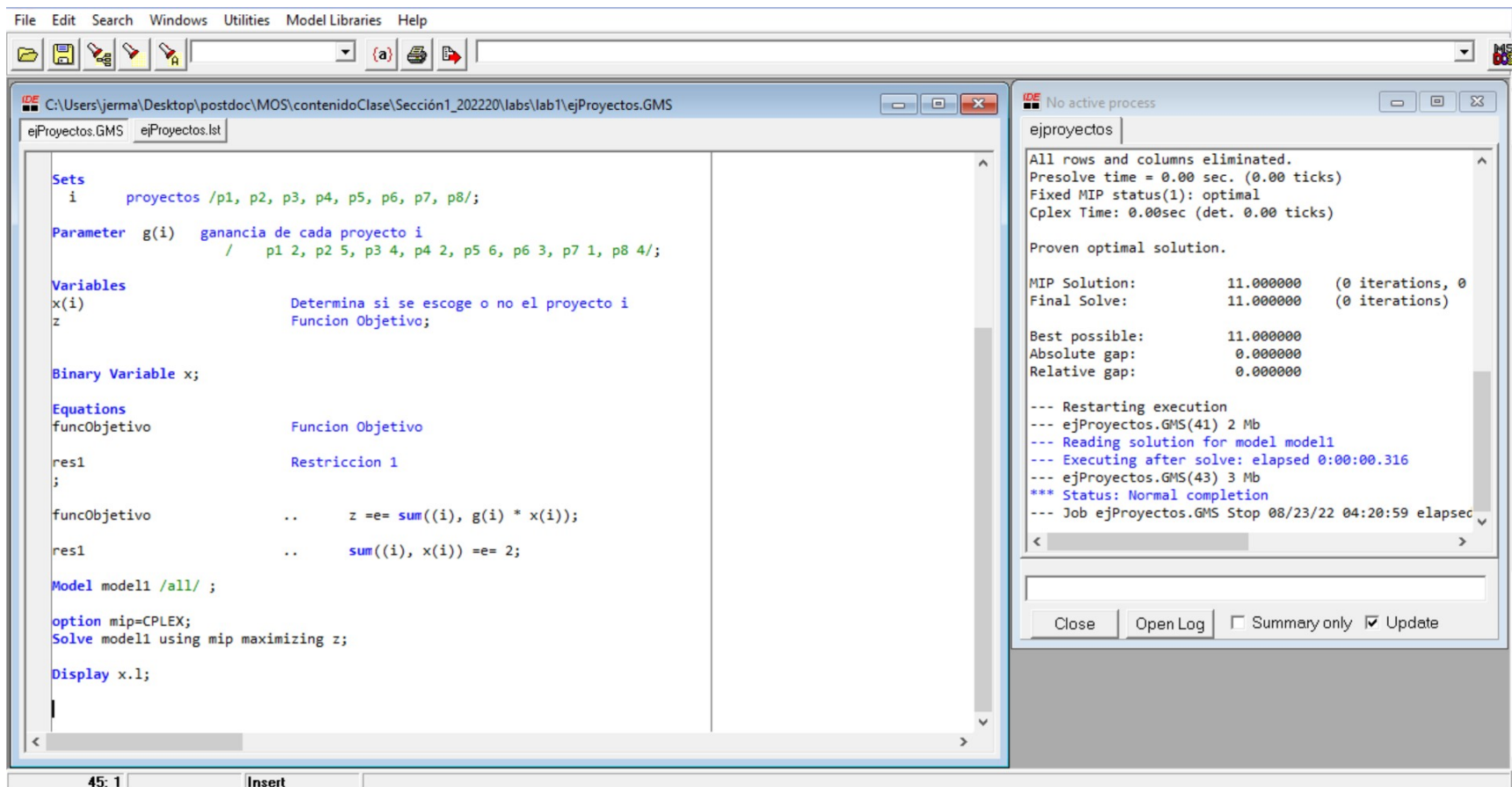
Profesor
Germán Montoya

Generalidades

- Abrir “gamside”.



- Interfaz General:



The screenshot displays the GAMS IDE interface. The main window shows the model file `ejProyectos.GMS` with the following content:

```

Sets
  i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
  x(i)      Determina si se escoge o no el proyecto i
  z          Funcion Objetivo;

Binary Variable x;

Equations
  funcObjetivo      Funcion Objetivo

  res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
  
```

The right-hand pane shows the execution log for `ejproyectos`, indicating a successful solution:

```

All rows and columns eliminated.
Presolve time = 0.00 sec. (0.00 ticks)
Fixed MIP status(1): optimal
Cplex Time: 0.00sec (det. 0.00 ticks)

Proven optimal solution.

MIP Solution:      11.000000   (0 iterations, 0
Final Solve:      11.000000   (0 iterations)

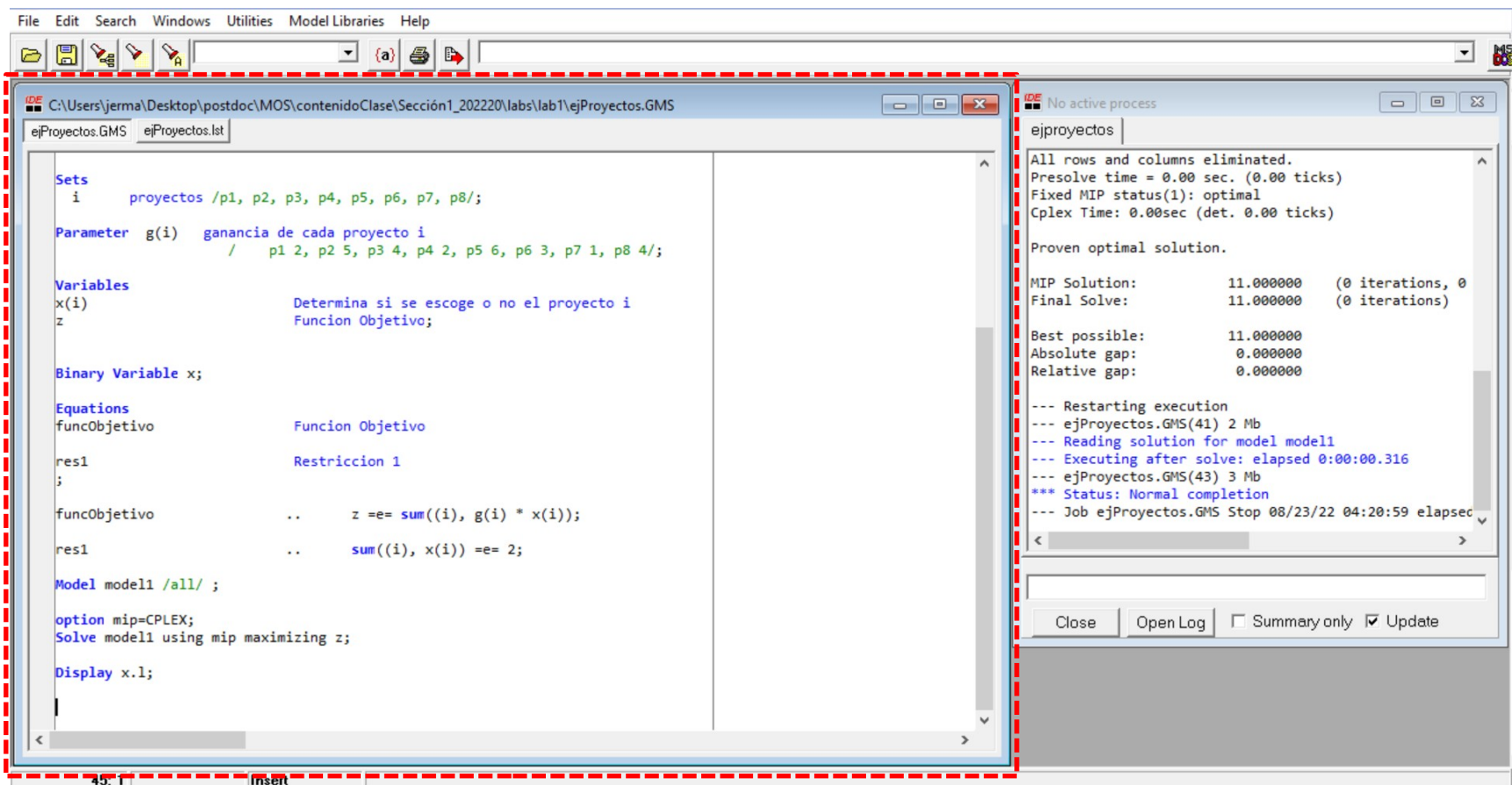
Best possible:     11.000000
Absolute gap:      0.000000
Relative gap:      0.000000

--- Restarting execution
--- ejProyectos.GMS(41) 2 Mb
--- Reading solution for model1
--- Executing after solve: elapsed 0:00:00.316
--- ejProyectos.GMS(43) 3 Mb
*** Status: Normal completion
--- Job ejProyectos.GMS Stop 08/23/22 04:20:59 elapsed
  
```

The status bar at the bottom indicates the current line is 45:1 in Insert mode.

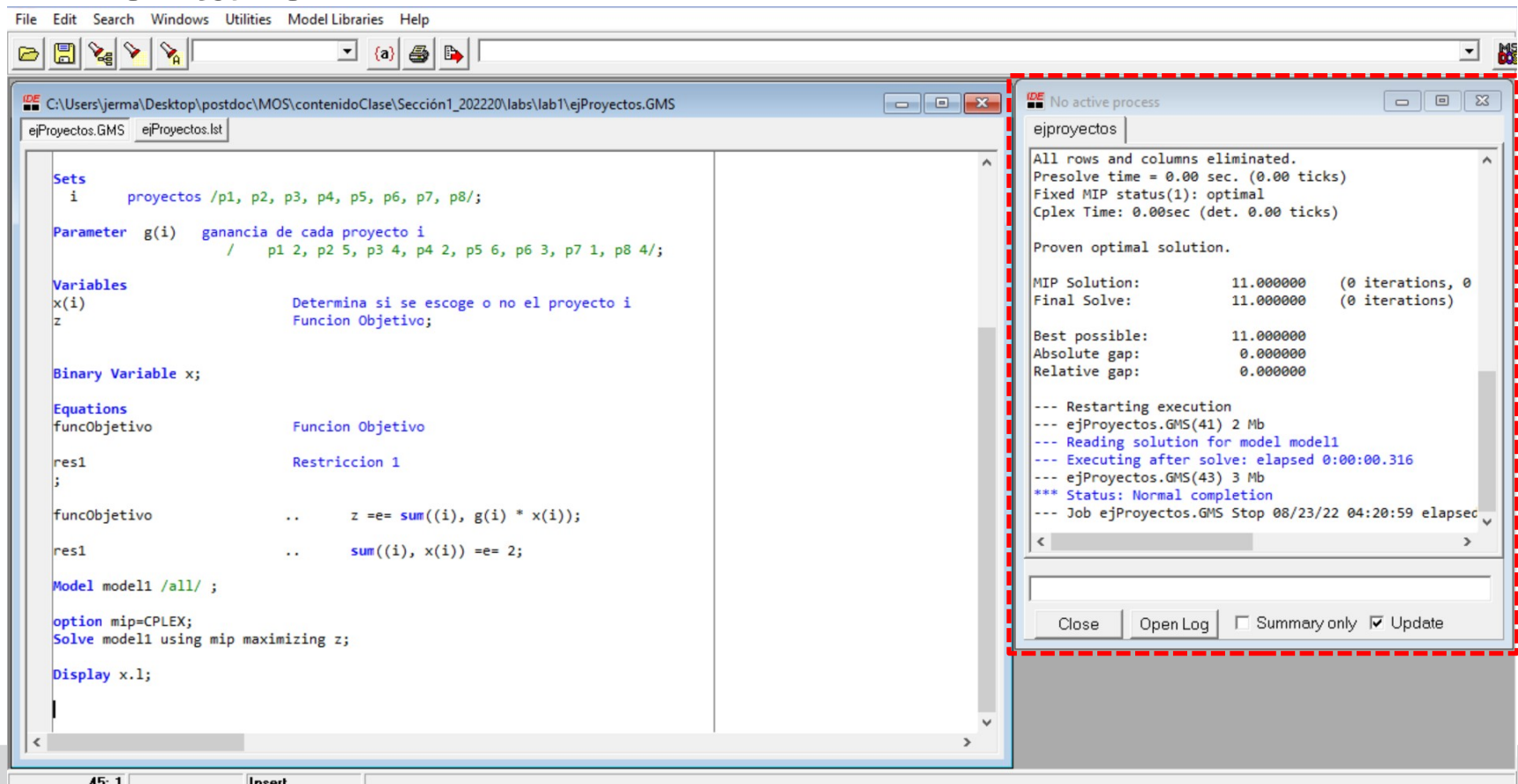
Interfaz

- Editor (*.gms): donde implemento mi modelo matemático.



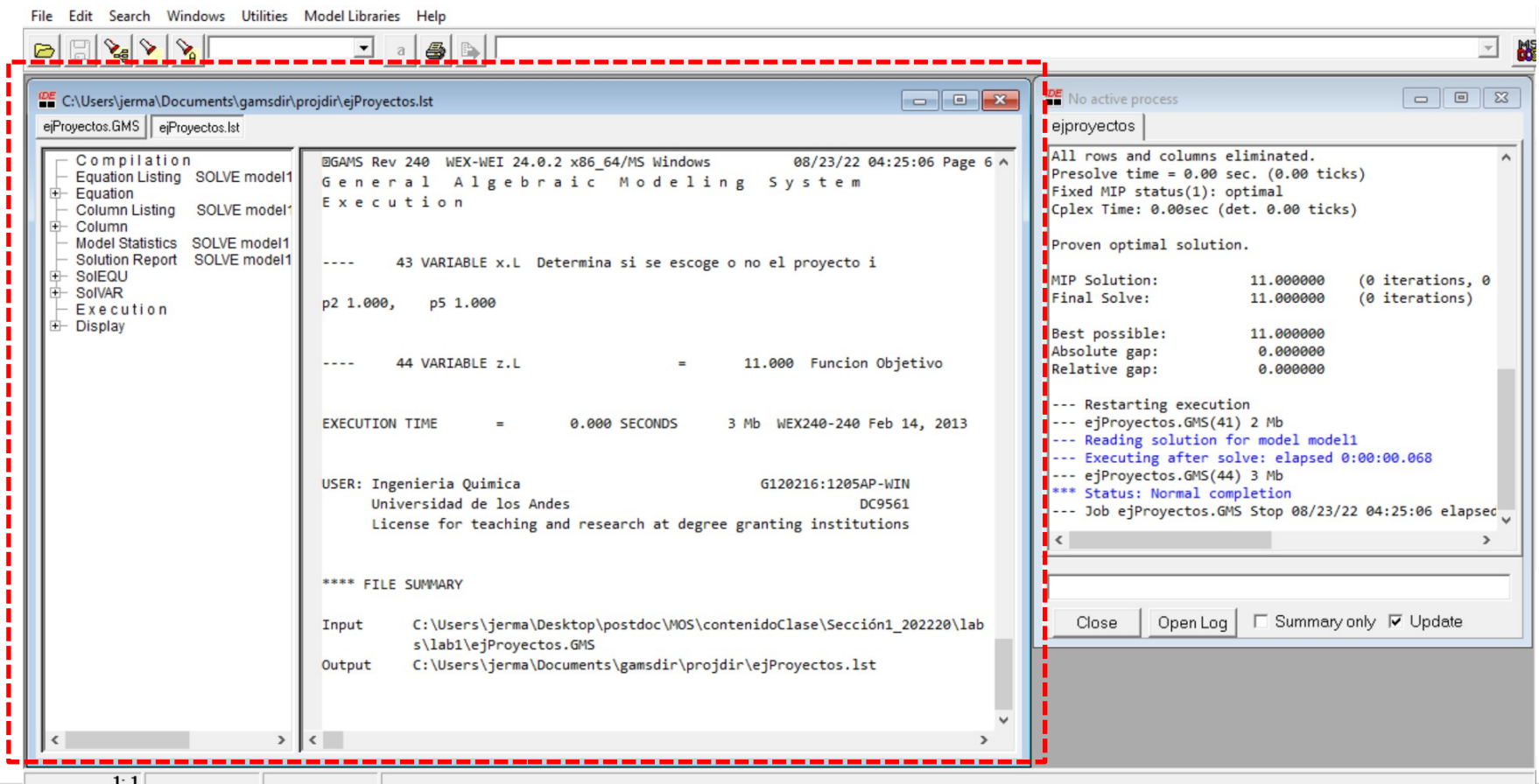
Interfaz

- Consola de resultados (*.log): donde observo si la ejecución del modelo fue correcta o si hay errores de sintaxis.



Interfaz

- Archivo de resultados (*.lst): donde observo el resultado de mi función objetivo y mis variables de decisión.



Modelado en GAMS

- Ejemplo de los proyectos:

```
Sets
i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
x(i)      Determina si se escoge o no el proyecto i
z          Funcion Objetivo;

Binary Variable x;

Equations
funcObjetivo      Funcion Objetivo

res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
Display z.l;
```

Modelado en GAMS

- Conjuntos (Rangos)
- Parámetros
- Variables

```
Sets
  i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
  x(i)      Determina si se escoge o no el proyecto i
  z          Funcion Objetivo;

Binary Variable x;

Equations
  funcObjetivo      Funcion Objetivo

  res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
Display z.l;
```


Modelado en GAMS

- Función Objetivo
- Restricciones

```
Sets
  i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
  x(i)      Determina si se escoge o no el proyecto i
  z          Funcion Objetivo;

Binary Variable x;

Equations
  funcObjetivo      Funcion Objetivo
  res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));
res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
Display z.l;
```

Modelado en GAMS

- Nombre del modelo
- Tipo de problema y solver a usar
- Definición de si vamos a maximizar o minimizar

```
Sets
i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
x(i)      Determina si se escoge o no el proyecto i
z          Funcion Objetivo;

Binary Variable x;

Equations
funcObjetivo      Funcion Objetivo

res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
Display z.l;
```

Modelado en GAMS

- Definiendo las variables que queremos visualizar cuando resolvamos el modelo.

```
Sets
i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
x(i)      Determina si se escoge o no el proyecto i
z          Funcion Objetivo;

Binary Variable x;

Equations
funcObjetivo      Funcion Objetivo

res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

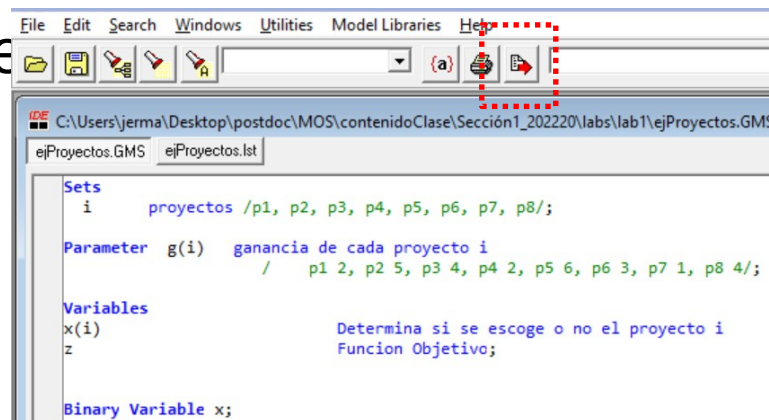
Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

Display x.l;
Display z.l;
```

Implementación del Ejemplo en GAMS

- Ejecutando e



```

File Edit Search Windows Utilities Model Libraries Help
[Icons] (a) [Run] [Exit]
C:\Users\jerma\Desktop\postdoc\MOS\contenidoClase\Sección1_202220\labs\lab1\ejProyectos.GMS
ejProyectos.GMS ejProyectos.lst

Sets
  i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
  x(i)      Determina si se escoge o no el proyecto i
  z          Funcion Objetivo;

Binary Variable x;
  
```

Resultados  Al final del archivo *.lst:

```

----      43 VARIABLE x.L  Determina si se escoge o no el proyecto i
p2 1.000,    p5 1.000

----      44 VARIABLE z.L              =      11.000  Funcion Objetivo
  
```

Características detalladas de GAMS

Sets

- Forma 1:

```
Set i      casas /c1, c2, c3, c4, c5/  
Set j      edificios /e1, e2, e3/
```
- Forma 2:

```
Sets  
i      casas /c1, c2, c3, c4, c5/  
j      edificios /e1, e2, e3/
```
- Forma 3:

```
Sets  
i      casas /c1*c5/  
j      edificios /e1*e3/
```
- Crear un conjunto copia de otro:

```
Set i      nodos /n1*n10/  
alias(j,i)
```

Escalares, Parámetros y Tablas

- Escalares:

```
Scalar BUDGET budget /10/;
```

- Parámetros:

- Parámetro de 1 dimensión (un índice):

```
Parameter value(i) value of each article  
/ a1 12, a2 5, a3 9, a4 6, a5 4 /;
```

- Parámetros de 2 o mas dimensiones:

- Forma 1:

```
Parameter D(i,j);  
D('h1','d1')=1;  
D('h2','d1')=0;  
D('h3','d1')=1;  
D('h1','d2')=1;  
D('h2','d2')=0;
```

- Forma 2:

```
Parameter D(i,j)  
/h1.d1=1  
h2.d1=0  
h3.d1=1  
h1.d2=1  
h2.d2=0/;
```

Escalares, Parámetros y Tablas

- Tablas (parámetro de 2 dimensiones) :

$$\sum_{i \in P} \sum_{j \in B} C_{ij} X_{ij} \longrightarrow ?$$

Table c(i,j) link cost					
	n1	n2	n3	n4	n5
n1	999	1	1	999	999
n2	999	999	999	999	1
n3	999	999	999	1	999
n4	999	999	999	999	1
n5	999	999	999	999	999;

Variables

- Declaración:

```
Variables
x(i)      Indicates if the article is bought or not
z         objective function;
Binary Variable x;
```

- Variable types:

<i>Keyword</i>	<i>Default Lower Bound</i>	<i>Default Upper Bound</i>	<i>Description</i>
free (default)	-inf	+inf	No bounds on variable. Both bounds can be changed from the default values by the user
positive	0	+inf	No negative values are allowed for variable. The user can change the upper bound from the default value.
negative	-inf	0	No positive values are allowed for variables. The user can change the lower bound from the default value.
binary	0	1	Discrete variable that can only take values of 0 or 1
integer	0	100	Discrete variable that can only take integer values between the bounds. The user can change bounds from the default value.

Variables

- Visualización de las variables:

```

Variables
  x(i)      Indicates if the article is bought or not
  z         objective function;

Binary Variable x;
.
.
.

Display x.1;

```

```

-----      29 VARIABLE x.L   if the article is bought

a4 1.000,      a5 1.000

-----      30 VARIABLE z.L           =      10.000  objective function

```

Ecuaciones

- Declaración and definición:

<code>Equations</code>		
<code>funcObjetivo</code>	<code>Funcion Objetivo</code>	} Declaraciones
<code>res1</code>	<code>Restriccion 1</code>	
<code>;</code>		
<code>funcObjetivo</code>	<code>.. z =e= sum((i), g(i) * x(i));</code>	} Definiciones
<code>res1</code>	<code>.. sum((i), x(i)) =e= 2;</code>	

- Operadores relacionales:
 - `=e=` equal
 - `=g=` greater than or equal to
 - `=l=` less than or equal to

Tips adicionales

- Comentario simple:

Equations

funcObjetivo

*res1

;

funcObjetivo

*res1

Funcion Objetivo

Restriccion 1

.. z =e= sum((i), g(i) * x(i));

.. sum((i), x(i)) =e= 2;

Tips adicionales

- Comentario de un fragmento de código:

```
Variables
x(i)                Determina si se escoge o no el proyecto i
z                  Funcion Objetivo;

Binary Variable x;

$ontext

Equations
funcObjetivo        Funcion Objetivo

res1                Restriccion 1
;

funcObjetivo        ..      z =e= sum((i), g(i) * x(i));

res1                ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;

$offtext

Display x.l;
Display z.l;
```

Tips adicionales

- Es possible chequear todos mis conjuntos y parámetros sin resolver el modelo:

```
Sets
i      proyectos /p1, p2, p3, p4, p5, p6, p7, p8/;

Parameter  g(i)  ganancia de cada proyecto i
              /   p1 2, p2 5, p3 4, p4 2, p5 6, p6 3, p7 1, p8 4/;

Variables
x(i)      Determina si se escoge o no el proyecto i
z          Funcion Objetivo;

Binary Variable x;

$ontext

Equations
funcObjetivo      Funcion Objetivo

res1              Restriccion 1
;

funcObjetivo      ..      z =e= sum((i), g(i) * x(i));

res1              ..      sum((i), x(i)) =e= 2;

Model model1 /all/ ;

option mip=CPLEX;
Solve model1 using mip maximizing z;
$offtext

Display i;
Display g;
```

```
----      46 SET i  proyectos

p1,    p2,    p3,    p4,    p5,    p6,    p7,    p8

----      47 PARAMETER g  ganancia de cada proyecto i

p1 2.000,    p2 5.000,    p3 4.000,    p4 2.000,    p5 6.000,    p6 3.000
p7 1.000,    p8 4.000
```

Tips adicionales

- Ejercicios:
 - Implemente la siguiente tabla usando dos métodos distintos: el método “table” y el método “parameter”.

	a1	a2	a3
n1	1	5	2
n2	4	2	6
n3	3	5	2

- Implemente la siguiente tabla usando dos métodos distintos: el método “table” y el método “parameter”.

	Juana	Pedro	Pepita
n1	1	5	2
n2	4	2	6
n3	3	5	2

Tips adicionales

- Ejercicios:
 - Implemente la siguiente tabla usando dos métodos distintos: el método “table” y el método “parameter”.

	n1	n2	n3
n1	1	5	2
n2	4	2	6
n3	3	5	2

Ecuaciones

- ‘Para todo’:

$$\min \sum_{i \in P} \sum_{j \in B} c_{ij} X_{ij}$$

$$\sum_{j \in B} X_{ij} \leq a_i \quad \forall i \in P$$

$$\sum_{i \in P} X_{ij} \geq b_j \quad \forall j \in B$$

Equations

```
cost define objective function
supply(i) observe supply limit at plant i
demand(j) satisfy demand at market j ;
```

```
cost .. z =e= sum((i,j), c(i,j)*x(i,j)) ;
supply(i) .. sum(j, x(i,j)) =l= a(i) ;
demand(j) .. sum(i, x(i,j)) =g= b(j) ;
```

Sentencias para definir el modelo y el solver a usar

```
Model Model1 /all/ ;  
Model Model1 /cost, supply, demand/ ;  
Model Model1 /cost, supply/ ;
```

→

```
cost .. z =e= sum((i,j), c(i,j)*x(i,j)) ;  
supply(i) .. sum(j, x(i,j)) =l= a(i) ;  
demand(j) .. sum(i, x(i,j)) =g= b(j) ;
```

```
option lp=CPLEX  
Solve Model1 using lp minimizing z;
```

```
Display x.l;  
Display z.l;
```

→ Solution for 'x' and 'z'.
'l' : level

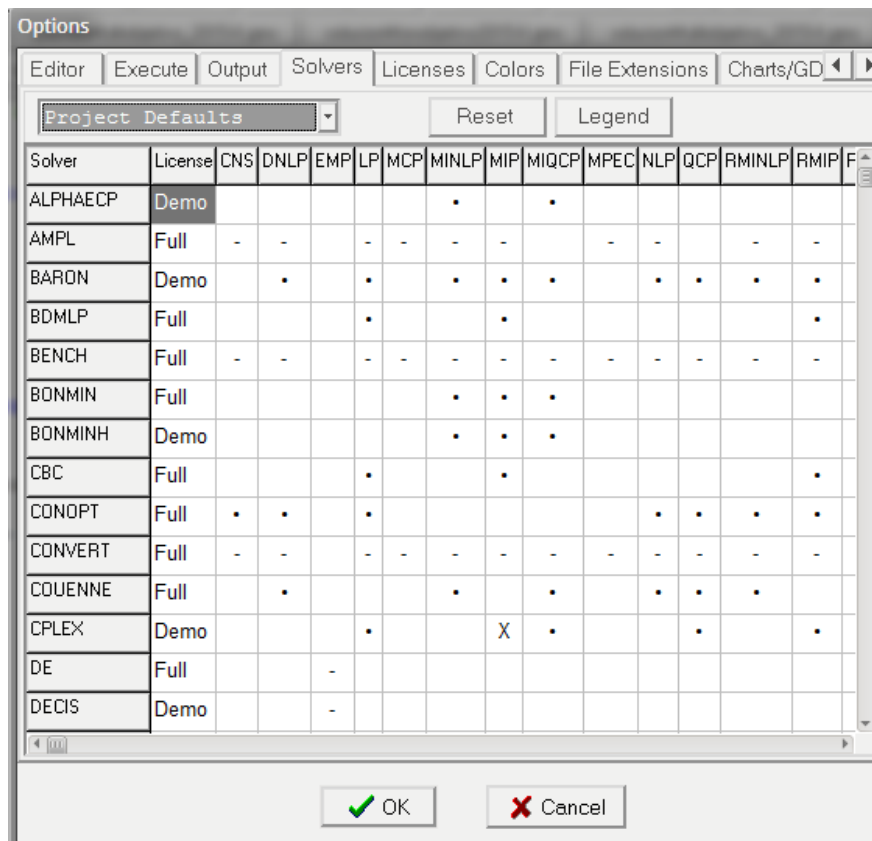
Sentencias para definir el modelo y el solver a usar

- Tipos de problemas:

lp	for linear programming
qcp	for quadratic constraint programming
nlp	for nonlinear programming
dnlp	for nonlinear programming with discontinuous derivatives
mip	for mixed integer programming
rmip	for relaxed mixed integer programming
miqcp	for mixed integer quadratic constraint programming
rmiqcp	for relaxed mixed integer quadratic constraint programming
minlp	for mixed integer nonlinear programming
rminlp	for relaxed mixed integer nonlinear programming
mcp	for mixed complementarity problems
mpec	for mathematical programs with equilibrium constraints
rmpec	for relaxed mathematical program with equilibrium constraints
cns	for constrained nonlinear systems
emp	for extended mathematical programming

Sentencias para definir el modelo y el solver a usar

- Tipos de problemas:
 - File>Options>Solvers



Solver Selection Legend

X Current selection

• Available selection

- Not available (Option statement only)

Condicionales en las restricciones

- Cómo representar un “tal que” en las restricciones?
 - Usando el símbolo “\$”.

```
Eq2(i)$ (cost(i)=5) .. x(i)=1=a(i) ;  
Eq3(i)$ (ord(i)<>1) .. x(i)=1=a(i) ;
```

- Operadores relacionales en los “tal que”:
 - <= (le): less than or equal to
 - < (lt): strictly less than
 - = (eq): equal to
 - <> (ne): not equal to
 - >= (ge): greater than or equal to
 - > (gt): strictly greater than

Errores típicos

- Error 1: No controlar todos los índices en la ecuación genérica.

Ejemplo 1:

- Forma incorrecta:

```
|sourceNode                                .. sum((j), x(i,j)) =e= 1;
```

- Mensaje de error arrojado por GAMS:

```
|--- ejCaminoCostoMinimo.GMS(35) 3 Mb 1 Error  
|*** Error 149 in C:\Users\Germán\Desktop\posi  
|    Uncontrolled set entered as constant
```

- Forma correcta:

```
|sourceNode(i)$(ord(i) = 1)                .. sum((j), x(i,j)) =e= 1;
```

Errores típicos

- Error 1: No controlar todos los índices en la ecuación genérica.

Ejemplo 2:

- Forma incorrecta:

```
sourceNode(i)$ (ord(i) = 1) .. x(i,j) =e= 1;
```

- Mensaje de error arrojado por GAMS:

```
--- ejCaminoCostoMinimo.GMS(35) 3 Mb 1 Error  
*** Error 149 in C:\Users\Germán\Desktop\pos:  
Uncontrolled set entered as constant
```

- Forma correcta:

```
sourceNode(i)$ (ord(i) = 1) .. sum((j), x(i,j)) =e= 1;
```

Errores típicos

- Error 1: No controlar todos los índices en la ecuación genérica.
 - Conclusión:
 - En una expresión genérica todo índice debe estar controlado por algún operador matemático útil para generalizar ecuaciones.
 - Operadores matemáticos para generalizar expresiones:
 - » Sumatoria
 - » Para todo
 - » Productoria

Errores típicos

- Error 2: Que un índice este controlado por mas de un operador matemático.

- Forma incorrecta:

```
sourceNode(i)$ord(i) = 1) .. sum((i,j), x(i,j)) == 1;
```

- Mensaje de error arrojado por GAMS:

```
--- ejCaminoCostoMinimo.GMS(35) 3 Mb 1 Error
*** Error 125 in C:\Users\Germán\Desktop\post
Set is under control already
```

- Forma correcta:

```
sourceNode(i)$ord(i) = 1) .. sum((j), x(i,j)) == 1;
```

Errores típicos

- Error 2: Que un índice este controlado por mas de un operador matemático.
 - Conclusión:
 - Todo índice debe estar controlado por un operador matemático, pero solo puede ser uno, no pueden ser varios operadores para un mismo índice.

Errores típicos

- Error 3: Que ningún índice este controlado por algún operador matemático.

- Forma incorrecta:

```
sourceNode .. x(i,j) =e= 1;
```

- Mensaje de error arrojado por GAMS:

```
--- ejCaminoCostoMinimo.GMS(35) 3 Mb 1 Error  
*** Error 149 in C:\Users\Germán\Desktop\pos:  
Uncontrolled set entered as constant
```

- Forma correcta:

```
sourceNode(i)$(ord(i) = 1) .. sum((j), x(i,j)) =e= 1;
```

Errores típicos

- Error 3: Que ningún índice este controlado por algún operador matemático.

- Forma incorrecta:

```
|sourceNode .. x(i,j) =e= 1;
```

- Mensaje de error arrojado por GAMS:

```
|--- ejCaminoCostoMinimo.GMS(35) 3 Mb 1 Error  
*** Error 149 in C:\Users\Germán\Desktop\pos:  
Uncontrolled set entered as constant
```

- Si queremos especificar un término en particular, debemos realizar lo siguiente:

```
|sourceNode .. x('n1','n2') =e= 1;
```

- O si queremos sumar varios términos específicos, los representamos así:

```
|sourceNode .. x('n1','n2') + x('n1','n3') =e= 1;
```

Errores típicos

- Error 4: incluir la variable de decisión en un “tal que”.

- Forma incorrecta:

```
sourceNode(i)$ (ord(i) = 1) .. sum((j)$ (x(i,j) = 1), x(i,j)) =e= 1;
```

- Mensaje de error arrojado por GAMS:

```
--- ejCaminoCostoMinimo.GMS(43) 3 Mb 2 Errors
*** Error 57 in C:\Users\Germán\Desktop\postdoc\MOS\contenido(
    Endogenous relational operations require model type "dnlp"
```

- Forma correcta: no incluir la variable de decisión en los “tal que”.

```
sourceNode(i)$ (ord(i) = 1) .. sum((j), x(i,j)) =e= 1;
```

Errores típicos

- Error 5: No usar la función “ord” en la evaluación de un índice.

– Forma incorrecta:

- Ej1:

```
sourceNode(i)$(i = 1) .. sum((j), x(i,j)) =e= 1;
```
- Ej2:

```
loop( (i,j),  
    if ((i=j),  
        c(i,j)=999;  
    );
```

– Mensaje de error arrojado por GAMS:

```
--- ejCaminoCostoMinimo.GMS(35) 3 Mb 2 Errors  
*** Error 148 in C:\Users\Germán\Desktop\postdoc\MOS\contenidoClas  
    Dimension different - The symbol is referenced with more/less  
    indices as declared  
*** Error 135 in C:\Users\Germán\Desktop\postdoc\MOS\contenidoClas  
    Incompatible operands for relational operator
```

– Forma correcta:

```
sourceNode(i)$(ord(i) = 1) .. sum((j), x(i,j)) =e= 1;
```

```
loop( (i,j),  
    if ((ord(i)=ord(j)),  
        c(i,j)=999;  
    );
```

Errores típicos

- Error 6: Usar un parámetro para guardar un calculo sin haberlo definido previamente.

– Forma incorrecta:

```
loop( (i,j),  
      D = a(i) - b(j);  
      c(i,j)=D;  
);
```

– Mensaje de error arrojado por GAMS:

```
--- minimoCostoCoordenadas.GMS(31) 3 Mb 1 Error  
*** Error 140 in C:\Users\Germán\Desktop\postdo  
Unknown symbol
```

– Forma correcta:

```
Scalar D / 0 /;  
  
loop( (i,j),  
      D = a(i) - b(j);  
      c(i,j)=D;  
);
```

Sugerencias para desarrollar la guía

- Intentar representar el problema con un grafo.
- Proponer teóricamente el modelo matemático no generico.
 - Definir los conjuntos.
 - Definir los índices a usar.
 - Definir la(s) variable(s) de decision.
 - Definir la función objetivo no generica.
 - Definir las restricciones no genéricas.
- Implementar el modelo no genérico en GAMS.
 - Verificar que el resultado sea coherente (sea lógico) de acuerdo al contexto del problema.
- Implementar el modelo matemático genérico.
 - Convertir la expresiones no genéricas en genéricas.
 - Tener en cuenta los tips de “Errores típicos”.
 - Verificar que el resultado sea coherente (sea lógico) de acuerdo al contexto del problema, es decir, que haya arrojado el mismo resultado arrojado por el modelo matemático no genérico.