Ejemplos instrucciones display y printf

Las funciones de **display** y **printf** se pueden usar para arrojar resultados específicos de un modelo de programación lineal en GLPK y Gusek.

- Justo debajo de todas las restricciones debe colocarse la instrucción **solve;**
- Debajo de **solve;** se pueden colocar las instrucciones de **display** o **printf**.
- Debajo de los **display** o **printf** debe aparecer la instrucción de los datos **data;**
- Las salidas de los **display** o **statement** se muestran en la partición del lado derecho del Gusek una vez se ejecuta el código. Si está trabajando en GLPK online se muestran en la pestaña que dice Output.
- Se pueden hacer **display** o **printf** de la función objetivo, de variables de decisión, de los valores del lado izquierdo de las restricciones, o de cálculos adicionales de las variables resultantes o cálculos de los lados izquierdos de las restricciones.
- En los **printf** siempre debe indicarse el formato de lo que se va a mostrar, si es un entero debe colocarse %d, si es un número real con decimales debe colocarse %f y siempre se debe colocar \n antes de cerrar las comillas para que aparezca un salto de línea (enter) entre las salidas de texto.

Ejemplo 1: Una grande superficie, está considerando la posibilidad de expandir una de sus sucursales adquiriendo espacios desocupados en un centro comercial. Para hacer la expansión se requieren dos años de trabajo. La siguiente tabla presenta la cantidad de millones de dólares que cada expansión costaría en cada uno de los dos años y el espacio que aumentaría la sucursal (m2) si hace la expansión. La grande superficie cuenta con 10 millones de dólares para invertir en cada año. Se desea maximizar el espacio total ampliado.

	Expansión							
	1	2	3	4	5	6	7	8
Año 1 (millones US$)	1.7	5.0	7.3	1.9	2.5	2.3	1.6	3.6
Año 2 (millones US$)	3.5	1.8	5.4	4.2	3.8	7.1	6.2	4.4
Espacio (m^2)	2200	9100	5300	8600	4200	7500	3800	8100

Formule y resuelva un modelo COMPACTO de programación lineal para este problema.

```
set I:=1..8; #expansiones
set J:=1..2; #años

param c{I,J}; #costo de la expansión i en el año j
param e{I}; #incremento espacio

var x{I} binary; #1 si se hace la expansión i, 0 dlc

maximize z: sum{i in I} e[i]*x[i];

s.t. presupuesto{j in J}:sum{i in I}c[i,j]*x[i]<=10;

solve;
```

```
display z;
display "la función objetivo es " & z;
display x;
display {i in I:x[i]=1} x[i];
display presupuesto;
display {j in J} "del presupuesto del año " & j & " sobran " & 10-presupuesto[j] & " millones de
dólares";
printf z;
printf: "la función objetivo es %f \n",z;
printf: "la función objetivo es %d \n",z;
printf {i in I:x[i]=1} "se debe hacer la expasión %d \n",i;
printf {j in J} "lo gastado del presupuesto del año %d es %f \n",j,presupuesto[j];
printf {j in J} "lo que sobra del presupuesto del año %d es %f \n",j,10-presupuesto[j];
printf {j in J} "la proporción del presupuesto del año %d es %f \n",j,presupuesto[j]/10;
printf "la cantidad de espacios que se adquirieron fue %d \n",sum{i in I}x[i];

data;

param c:1       2:=
1       1.7     3.5
2       5.0     1.8
3       7.3     5.4
4       1.9     4.2
5       2.5     3.8
6       2.3     7.1
7       1.6     6.2
8       3.6     4.4;

param e:=
1       2200
2       9100
3       5300
4       8600
5       4200
6       7500
7       3800
8       8100;
```

Ejemplo 2: A continuación, se presenta el código de Gusek completo del problema del video de inventarios del segundo corte, titulado PLANEACIÓN DE PRODUCCIÓN E INVENTARIO COLUMNAS DE ACERO. Posteriormente, con diversos ejemplos de las instrucciones **display** y **printf**. En esos ejemplos verán listados en dos líneas seguidas de código, como arrojar el mismo resultado con la función de **display** o con la de **printf**.

```
#Conjuntos:
set J:=1..2; #:Tipos de trabajadores {1:Tiempo Completo, 2:Teimpo Parcial}
set I:=1..3; #Tipos de Columnas
set S:=1..4; #Semanas

#Parámetros:
param D{I,S}; #Demanda del tipo de columna i in I para cada semana  s in S.
param N{I}; #Inventario mínimo de la columna tipo i in I
```

```
param F{I}; #Costo de Almacenamiento por semana de la columna tipo i in I
param T{J}; #Tiempo que trabaja el trabajador tipo j in J
param SS{J}; #salario semanal en $/trabajador del trabajador tipo j in J
param P{I}; #Tiempo necesario para producir la columna tipo I
param V; #Capacidad de la Bodega

#Variables de decisión
var x{I,S,J}>=0 integer; #cantidad de trabajadores tipo j in J,contratados en la semana s in S,para hacer columnas tipo i in I
var y{I,S}>=0 integer; #cantidad de columnas tipo i in I almacenadas al final de la semana s in S
var w{I,S}>=0 integer; #cantidad de columnas tipo i in I fabricadas en la semana s in S


#Función objetivo:
minimize z: sum{i in I, s in S, j in J}x[i,s,j]*SS[j]+sum{i in I, s in S}y[i,s]*F[i];

#s.a
s.t. capacidadProduccion{s in S, i in I}:w[i,s]*P[i]/60<=sum{j in J}x[i,s,j]*T[j];
s.t. inventarioMinimo{s in S, i in I}:y[i,s]>=N[i];
s.t. inventarioSemana1{i in I}:y[i,1]=w[i,1]-D[i,1];
s.t. inventarioOtrasSemanas{s in S, i in I:s>1}:y[i,s]=y[i,s-1]+w[i,s]-D[i,s];
s.t. CapacidadBodega{s in S}:sum{i in I}y[i,s]<=V;
s.t. trabajadoresTC{s in S}:sum{i in I}x[i,s,1]>=0.2*sum{i in I, j in J, h in S} x[i,h,j];

solve;

#con las siguientes instrucciones aparecerá el valor de la función objetivo solamente
display z;
printf z;

#con las siguientes instrucciones aparecerá el texto “la función objetivo es 210800”
display: "la función objetivo es " & z;
printf: "la función objetivo es %d \n", z;

#aquí se mostrará la cantidad de trabajadores de cada tipo contratados en cada semana para hacer cada tipo de columnas
display{i in I, s in S, j in J:x[i,s,j]>0}:'la cantidad de trabajadores tipo ' & j & 'contratados en la semana ' & s & 'para hacer columnas tipo ' & i & 'es ' & x[i,s,j];
printf{i in I, s in S, j in J:x[i,s,j]>0}"la cantidad de trabajadores tipo %d contratados en la semana %d para hacer columnas tipo %d es %d \n",j,s,i,x[i,s,j];

#aquí se mostrará la cantidad de columnas de cada tipo que se almacenan en inventario al final de cada semana
display {i in I, s in S:y[i,s]>0} :"la cantidad de columnas tipo " & i & " almacenadas al final de la semana " & s & " es " & y[i,s];
printf{i in I, s in S:y[i,s]>0}"la cantidad de columnas tipo %d almacenadas al final de la semana %d es %d \n",i,s,y[i,s];

#aquí se mostrará la cantidad de columnas de cada tipo que se fabrican en cada semana
display{i in I, s in S:w[i,s]>0}"la cantidad de columnas tipo " & i & " fabricadas en la semana " & s & " es " & w[i,s];
printf{i in I, s in S:w[i,s]>0}"la cantidad de columnas tipo %d fabricadas en la semana %d es %d \n",i,s,w[i,s];
```

```
#esto mostrará el porcentaje ocupado de la bodega en cada semana
display{s in S}: "el porcentaje que se llenó de la bodega en la semana " & s & " fue " & sum{i in I}y[i,s]*100/V & " porciento";
printf{s in S}: "el porcentaje que se llenó de la bodega en la semana %d fue %f porciento \n",s,sum{i in I}y[i,s]*100/V;

data;

param D:        1       2       3       4:=
1                       20      18      21      17
2                       20      22      18      25
3                       25      23      28      24;

param N:=
1       5
2       10
3       3;

param F:=
1       100
2       200
3       50;

param T:=
1       40
2       20;

param SS:=
1       300000
2       200000;

param P:=
1       120
2       60
3       180;

param V:=50;

end data;
```