Cuaderno de Prácticas

Asignatura

MATEMÁTICAS DISCRETAS

CURSO 2024/25

Datos personales

Nombre y apellidos : Francisco Javier Martín - Lunas Escobar
DNI : 26268082
Grupo de teoría : 2
Grupo de prácticas :

Archivos

In[ ]:=

```
indice={
        {"Introducción","Apartado 1","Apartado 2","Apartado 3"},
        {"Capítulo 9: Retículos y Alg. de Boole finitas","Orden, max, min, inf, etc","Reticulo
        {"Capítulo 10: Funcones booleanas.","Minterminos, maxterminos"
            ,"Tablas de funciones boole","Formas canonicas"},
        {"Capítulo 11: Numeros enteros 1: Divisibilidad",
            "Euclides","Bezout" ,"Diofanticas, comprobar coprimos"},
        {"Capítulo 12: Numeros enteros 2: Congruencias y sist. de numeración",
            "Chino","Inverso","Bases decimal, ocatal, etc"}
};
TextGrid[indice,Frame→All,Spacings→{2,1}]
```

Out[ ]=

Introducción	Apartado 1	Apartado 2	Apartado 3
Capítulo 9: Retículos y Alg. de Boole finitas	Orden, max, min, inf, etc	Reticulos	
Capítulo 10: Funcones booleanas.	Minterminos, maxterminos	Tablas de funciones boole	Formas canonicas
Capítulo 11: Numeros enteros 1: Divisibilidad	Euclides	Bezout	Diofanticas, comprobar coprimos
Capítulo 12: Numeros enteros 2: Congruencias y sist. de numeración	Chino	Inverso	Bases decimal, ocatal, etc

Práctica 8

```
(* Para saber si es de orden *)
ORDEN[A_,R_]:=Module[{falloReflexiva,falloAntisimetrica,falloTransitiva},falloReflexiva={};
 Do[If[MemberQ[R,{A[[n]],A[[n]]}],Null,AppendTo[falloReflexiva,A[[n]]]],{n,Length[A]}];
 falloAntisimetrica={};
 Do[If[MemberQ[R,{R[[r,2]],R[[r,1]]}]&&!(ToString[R[[r,1]]]==ToString[R[[r,2]]]),
AppendTo[falloAntisimetrica,{R[[r,1]],R[[r,2]]}];];,{r,Length[R]}];
 falloTransitiva={};
 Do[Do[If[R[[p,1]]==R[[q,2]],If[MemberQ[R,{R[[q,1]],R[[p,2]]}],Null,
AppendTo[falloTransitiva,{{R[[q,1]],R[[q,2]]},{R[[p,1]],R[[p,2]]}}]]];,
{q,Length[R]}];,{p,Length[R]}];
 If[falloReflexiva=={},Print["R es reflexiva"],
Print["R no es reflexiva, falla en los elementos: ",falloReflexiva]];
 If[falloAntisimetrica=={},Print["R es antisimétrica"],
Print["R no es antisimétrica, falla en los pares: ",falloAntisimetrica]];
 If[falloTransitiva=={},Print["R es transitiva"],
Print["R no es transitiva, falla en los pares: ",falloTransitiva]];
 If[Union[falloReflexiva,falloAntisimetrica,falloTransitiva]=={},
Print["R es una relación de orden"],Print["R no es relación de orden"]];];
```

```
(* Para hacer reticulo *)
RETICULO[A_,R_]:=Module[{reticulo,cotassuperiores,cotasinferiores,csuper,cinfer,supremo,
    infimo,mini,maxi,n,m,x1,x2},reticulo=True;
Do[Do[cotassuperiores={};cotasinferiores={};
 Do[csuper=True;cinfer=True;
 If[Intersection[{{A⟦x1⟧,A⟦n⟧}},R]=={},csuper=False];
 If[Intersection[{{A⟦x2⟧,A⟦n⟧}},R]=={},csuper=False];
 If[Intersection[{{A⟦n⟧,A⟦x1⟧}},R]=={},cinfer=False];
 If[Intersection[{{A⟦n⟧,A⟦x2⟧}},R]=={},cinfer=False];
 If[csuper,AppendTo[cotassuperiores,A⟦n⟧]];If[cinfer,AppendTo[cotasinferiores,A⟦n⟧]];,
    {n,1,Length[A]}];
 supremo={};infimo={};
 Do[mini=True;
 Do[If[Intersection[{{cotassuperiores⟦n⟧,cotassuperiores⟦m⟧}},R]=={},mini=False],
    {m,1,Length[cotassuperiores]}];
 If[mini,AppendTo[supremo,cotassuperiores⟦n⟧]];,{n,1,Length[cotassuperiores]}];
 If[supremo=={},reticulo=False];
 Do[maxi=True;
 Do[If[Intersection[{{cotasinferiores⟦m⟧,cotasinferiores⟦n⟧}},R]=={},maxi=False],
    {m,1,Length[cotasinferiores]}];
If[maxi,AppendTo[infimo,cotasinferiores⟦n⟧]];
         ,{n,1,Length[cotasinferiores]}];
If[infimo=={},reticulo=False];
      ,{x1,1,Length[A]}];
   ,{x2,1,Length[A]}];
reticulo
]
(* Maximales *)
MAXIMALES[A_,R_]:=Module[{maximales,maximal,n,m},
maximales={};
Do[maximal=True;
Do[If[MemberQ[R,{A⟦n⟧,A⟦m⟧}]&&n≠m,maximal=False;Break[];],{m,1,Length[A]}];
If[maximal,AppendTo[maximales,A⟦n⟧]];,{n,1,Length[A]}];
maximales
]

(* Minimales *)
MINIMALES[A_,R_]:=Module[{minimales,minimal,n,m},
minimales={};
Do[minimal=True;
Do[If[MemberQ[R,{A⟦m⟧,A⟦n⟧}]&&n≠m,minimal=False;Break[];],{m,1,Length[A]}];
If[minimal,AppendTo[minimales,A⟦n⟧]];,{n,1,Length[A]}];
minimales
]

(* Supremo *)
SUPREMO[A_,B_,R_]:=Module[{cotassuperiores,csuper,supremo,mini,n,m},
cotassuperiores={};
Do[csuper=True;
Do[If[Intersection[{{B⟦m⟧,A⟦n⟧}},R]=={},csuper=False;Break[];],{m,1,Length[B]}];
If[csuper,AppendTo[cotassuperiores,A⟦n⟧]];,{n,1,Length[A]}];
supremo={};
Do[mini=True;
Do[If[MemberQ[R,{cotassuperiores⟦n⟧,cotassuperiores⟦m⟧}], ,mini=False;Break[];],
```

```
    {m,1,Length[cotassuperiores]}];
 If[mini,AppendTo[supremo,cotassuperiores⟦n⟧];
 Break[];];,{n,1,Length[cotassuperiores]}];
supremo
]

(* Complemento *)
COMPLEMENTO[A_,R_,n_]:=Module[{complemento,k},complemento={};
Do[If[INFIMO[A,{A⟦k⟧,n},R]==MINIMALES[A,R]&&SUPREMO[A,{A⟦k⟧,n},R]==MAXIMALES[A,R],
    AppendTo[complemento,A⟦k⟧]],{k,1,Length[A]}];
complemento]

(* Infimo *)
INFIMO[A_,B_,R_]:=Module[{cotasinferiores,cinfer,infimo,maxi,n,m},
cotasinferiores={};
Do[cinfer=True;
Do[If[Intersection[{{A⟦n⟧,B⟦m⟧}},R]=={},cinfer=False;Break[];],{m,1,Length[B]}];
If[cinfer,AppendTo[cotasinferiores,A⟦n⟧]];,{n,1,Length[A]}];
infimo={};
Do[maxi=True;
Do[If[MemberQ[R,{cotasinferiores⟦m⟧,cotasinferiores⟦n⟧}],,maxi=False;Break[];],
    {m,1,Length[cotasinferiores]}];
If[maxi,AppendTo[infimo,cotasinferiores⟦n⟧];Break[];];,{n,1,Length[cotasinferiores]}];
infimo
]

(* Algebra de Boole *)
BOOLE[L_,R_]:=Module[{minimales,cero,preimagen,supremo,i,j,k,l,n,conjunto,imagen,f,nmin},
ABoole=True;minimales=MINIMALES[L,R];
If[Length[minimales]≠1, ABoole=False,
cero=minimales⟦1⟧;minimales=MINIMALES[Complement[L,{cero}],R];
n=Length[minimales];f[cero]:=Table[0,{i,n}];
Do[f[minimales⟦i⟧]=Table[If[j≠i,0,1],{j,n}];,{i,n}];
ABoole=(Length[L]==(2^n) && Length[R]==Sum[(n!/(k!(n-k)!))*2^(n-k),{k,0,n}]);
If[ABoole,
imagen=Union[{f[cero]},Table[f[minimales⟦i⟧],{i,Length[minimales]}]];
preimagen=Union[{cero},minimales];
l=2;
While[l≤n && ABoole,
conjunto={};j=1;nmin=Length[minimales];
While[j≤nmin && ABoole,
i=j+1;
While[i≤nmin && ABoole,
If[Length[Position[f[minimales⟦i⟧]+f[minimales⟦j⟧],0]]==n-l,
supremo=SUPREMO[L,{minimales⟦i⟧,minimales⟦j⟧},R]⟦1⟧;
If[Intersection[preimagen,{supremo}]=={},
conjunto=Union[conjunto,{supremo}];
f[supremo]=Table[If[(f[minimales⟦i⟧]+f[minimales⟦j⟧])⟦k⟧≠0,1,0],{k,n}];
imagen=Union[imagen,{f[supremo]}];
preimagen=Union[preimagen,{supremo}];
,If[Intersection[{supremo},conjunto]=={},ABoole=False;];
];];i++;];j++;];
minimales=conjunto;
ABoole=(Length[conjunto]==(n!/(l!*(n-l)!)));
l++;];
```

```
ABoole=(Length[imagen]==2^n && Length[preimagen]==2^n);
];];
If[ABoole,
Print["Es álgebra de Boole, un isomorfismo con ", Superscript["𝔹2",n]," viene dado por:"];
Do[Print[L[[i]]," → ",f[L[[i]]]];,{i,2^n}];
,
Print["No es álgebra de Boole"];];
];
```

Ejercicio 9.4 .

Sea D el conjunto formado por los divisores positivos del producto de los tres últimos dígitos no nulos de tu DNI con la relación de orden, a≤b⟺b|a Comprobar:

In[∘]:=
```
ult_dig=8*6*2;
d=Subsets[{8*6*2}];
R={};
For[i=1, i≤Length[d],i++,
    For[j=1, j≤Length[d],j++,
        If[SubsetQ[d[[j]],d[[i]]],
            AppendTo[R,{d[[i]],d[[j]]}]]
    ]
]
R
d1=Divisors[8*6*2];
R1={};
For[i=1, i≤Length[d1],i++,
    For[j=1, j≤Length[d1],j++,
        If[Mod[d1[[j]],d1[[i]]]==0,
            AppendTo[R1,{d1[[i]],d1[[j]]}]]
    ]
]
R1
```

Out[∘]=
```
{{{}, {}}, {{}, {96}}, {{96}, {96}}}
```

Out[∘]=
```
{{1, 1}, {1, 2}, {1, 3}, {1, 4}, {1, 6}, {1, 8}, {1, 12}, {1, 16}, {1, 24}, {1, 32},
 {1, 48}, {1, 96}, {2, 2}, {2, 4}, {2, 6}, {2, 8}, {2, 12}, {2, 16}, {2, 24}, {2, 32},
 {2, 48}, {2, 96}, {3, 3}, {3, 6}, {3, 12}, {3, 24}, {3, 48}, {3, 96}, {4, 4},
 {4, 8}, {4, 12}, {4, 16}, {4, 24}, {4, 32}, {4, 48}, {4, 96}, {6, 6}, {6, 12},
 {6, 24}, {6, 48}, {6, 96}, {8, 8}, {8, 16}, {8, 24}, {8, 32}, {8, 48}, {8, 96},
 {12, 12}, {12, 24}, {12, 48}, {12, 96}, {16, 16}, {16, 32}, {16, 48}, {16, 96},
 {24, 24}, {24, 48}, {24, 96}, {32, 32}, {32, 96}, {48, 48}, {48, 96}, {96, 96}}
```

a) Si es un conjunto ordenado

In[∘]:=
```
ORDEN[d1,R1]
```

```
R es reflexiva
```

```
R es antisimétrica
```

```
R es transitiva
```

```
R es una relación de orden
```

b) Si es un retículo

In[ ]:=
```
RETICULO[d1,R1]
```

Out[ ]=
```
True
```

c) Si es un álgebra de Boole

In[ ]:=
```
BOOLE[d1,R1]
```

```
No es álgebra de Boole
```

Ejercicio 9.5 .

Sea D el conjunto formado por los tres últimos dígitos distintos de tu DNI con relación de orden,

$$A \leq B \Leftrightarrow A \subseteq B$$

Comprobar:

In[ ]:=
```
ult_dig={2,6,8};
d=Subsets[{2,6,8}];
R={};
For[i=1, i≤Length[d],i++,
    For[j=1, j≤Length[d],j++,
        If[SubsetQ[d⟦j⟧,d⟦i⟧],
            AppendTo[R,{d⟦i⟧,d⟦j⟧}]]
    ]
]
R
```

Out[ ]=
```
{{{}, {}}, {{}, {2}}, {{}, {6}}, {{}, {8}}, {{}, {2, 6}}, {{}, {2, 8}}, {{}, {6, 8}},
 {{}, {2, 6, 8}}, {{2}, {2}}, {{2}, {2, 6}}, {{2}, {2, 8}}, {{2}, {2, 6, 8}},
 {{6}, {6}}, {{6}, {2, 6}}, {{6}, {6, 8}}, {{6}, {2, 6, 8}}, {{8}, {8}}, {{8}, {2, 8}},
 {{8}, {6, 8}}, {{8}, {2, 6, 8}}, {{2, 6}, {2, 6}}, {{2, 6}, {2, 6, 8}}, {{2, 8}, {2, 8}},
 {{2, 8}, {2, 6, 8}}, {{6, 8}, {6, 8}}, {{6, 8}, {2, 6, 8}}, {{2, 6, 8}, {2, 6, 8}}}
```

a) Si es un conjunto ordenado

In[ ]:=
```
ORDEN[d,R]
```

```
R es reflexiva
```

```
R es antisimétrica
```

```
R es transitiva
```

```
R es una relación de orden
```

b) Si es un retículo

In[ ]:=
```
RETICULO[d,R]
```

Out[ ]=
```
True
```

c) Si es un álgebra de Boole

In[ ]:=
```
BOOLE[d,R]
```

Es álgebra de Boole, un isomorfismo con $\mathbb{B}_2{}^3$ viene dado por:

```
{} → {0, 0, 0}
```

```
{2} → {1, 0, 0}
```

```
{6} → {0, 1, 0}
```

```
{8} → {0, 0, 1}
```

```
{2, 6} → {1, 1, 0}
```

```
{2, 8} → {1, 0, 1}
```

```
{6, 8} → {0, 1, 1}
```

```
{2, 6, 8} → {1, 1, 1}
```

d) Calcular el complementario del conjunto formado por los dos primeros dígitos

In[ ]:=
```
y={2,6};
COMPLEMENTO[d,R,y]
```

Out[ ]=
```
{{8}}
```

e) Calcular los átomos y sus complementos

```
Ra={{{2},{2}},{{2},{2,6}},
{{2},{2,8}},{{2},{2,6,8}},{{6},{6}},{{6},{2,6}},{{6},{6,8}},{{6},{2,6,8}},{{8},{8}},
{{8},{2,8}},{{8},{6,8}},{{8},{2,6,8}},{{2,6},{2,6}},{{2,6},{2,6,8}},{{2,8},{2,8}},
{{2,8},{2,6,8}},{{6,8},{6,8}},{{6,8},{2,6,8}},{{2,6,8},{2,6,8}}}
```

Out[∘]=

```
{{{2}, {2}}, {{2}, {2, 6}}, {{2}, {2, 8}}, {{2}, {2, 6, 8}}, {{6}, {6}}, {{6}, {2, 6}},
 {{6}, {6, 8}}, {{6}, {2, 6, 8}}, {{8}, {8}}, {{8}, {2, 8}}, {{8}, {6, 8}},
 {{8}, {2, 6, 8}}, {{2, 6}, {2, 6}}, {{2, 6}, {2, 6, 8}}, {{2, 8}, {2, 8}},
 {{2, 8}, {2, 6, 8}}, {{6, 8}, {6, 8}}, {{6, 8}, {2, 6, 8}}, {{2, 6, 8}, {2, 6, 8}}}
```

In[∘]:=

```
da = {{2},{6},{8},{2,6},{2,8},{6,8},{2,6,8}};
MINIMALES[da,Ra]
```

Out[∘]=

```
{{2}, {6}, {8}}
```

In[∘]:=

```
COMPLEMENTO[d,R,{2}]
COMPLEMENTO[d,R,{6}]
COMPLEMENTO[d,R,{8}]
```

Out[∘]=

```
{{6, 8}}
```

Out[∘]=

```
{{2, 8}}
```

Out[∘]=

```
{{2, 6}}
```

Práctica 10

In[∘]:=

```
fAnd[x_,y_]:=x*y;
fOr[x_,y_]:=If[x+y==0,0,1];

(* Complementario *)
Compl[a_]:=Mod[a+1,2];

(* Tabla de verdad *)
TABLADEVERDAD[funcion_,nombres_]:=Module[{n,x,j,f,tabla,resto},
 n=Length[nombres];
 x=Table[0,{t,n}];
 tabla=Table[0,{r,(2^n)},{s,n+1}];
 Do[j=i;
  For[f=n,f>0,f--,
  resto=Mod[j,2];j=Floor[j/2];
  If[resto==0,x[[f]]=0;tabla[[i+1,f]]=0,x[[f]]=1;tabla[[i+1,f]]=1];
  ];
 tabla[[i+1,n+1]]=funcion[x];
```

```
,{i,0,2^n-1}];
TableForm[tabla, TableHeadings→{None,Join[nombres,{ToString[funcion]}]},
    TableAlignments→Center]
];

(* Maxterminos *)
MAXTERM[d_,x_]:=Boole[BooleanMaxterms[{BitXor[d,1]},x]];

(* Minterminos *)
MINTERM[d_,x_]:=Boole[BooleanMinterms[{d},x]];

(* Formas canonicas *)
FCANONICAS[funcion_,nombres_]:=Module[{mint, maxt,x,n,disyminterm,conjmaxterm,i},
n=Length[nombres];mint={};maxt={};
For[i=0,i<2^n,i++,
x=IntegerDigits[i,2,n];
If[funcion[x]==1,AppendTo[mint,i];,AppendTo[maxt,i];];
];
If[mint=={},Print["Es una contradicción"];
,

Do[mint[[i]]=Subscript["m",IntegerDigits[mint[[i]],2,n]],{i,Length[mint]}];
Do[mint=Insert[mint," + ",j];,{j,Length[mint],2,-1}];
Print["Supremo de mintérminos: ",funcion," = ",Row[mint]];
];
If[maxt=={},Print["Es una tautología"];
,
Do[maxt[[i]]=Subscript["M",IntegerDigits[maxt[[i]],2,n]],{i,Length[maxt]}];
Do[maxt=Insert[maxt," · ",j];,{j,Length[maxt],2,-1}];
Print["Ínfimo de maxtérminos: ",funcion," = ",Row[maxt]];
];];

(* Formas canonicas 2 *)
FCANONICAS2[funcion_,nombres_]:=Module[{mint,maxt,n,cad,cad2,x,j,i,f,cadfto,var,resto,
    varneg,fcminterminos,fcmaxterminos,Nfunc},
n=Length[nombres];x=Table[0,{t,n}];
mint={};maxt={};
For[i=0,i<2^n,i++,cad={};cad2={};j=i;
For[f=1,f≤n,f++,resto=Mod[j,2];j=Floor[j/2];
var=nombres[[f]];
varneg=OverBar[nombres[[f]]];
If[resto==0,x[[f]]=1;AppendTo[cad,var];
AppendTo[cad2,varneg];,x[[f]]=0;AppendTo[cad,varneg];
AppendTo[cad2,var];];];
If[funcion[x]==1,AppendTo[mint,cad];,AppendTo[maxt,cad2];];];
cadfto=mint;
Do[Do[cadfto[[i]]=Insert[cadfto[[i]],"∧",j];,{j,n,2,-1}];
cadfto[[i]]=Row[Join[{"("},cadfto[[i]],{")"}]];
   ,{i,1,Length[mint]}];
Do[cadfto=Insert[cadfto," ∨ ",i];,{i,Length[mint],2,-1}];
fcminterminos=Row[cadfto];
cadfto=maxt;
Do[Do[cadfto[[i]]=Insert[cadfto[[i]],"∨",j];,{j,n,2,-1}];
cadfto[[i]]=Row[Join[{"("},cadfto[[i]],{")"}]];,{i,1,Length[maxt]}];
```

```
Do[cadfto=Insert[cadfto," ∧ ",i];,{i,Length[maxt],2,-1}];
fcmaxterminos=Row[cadfto];
Nfunc=StringJoin[ToString[funcion],"("];
Do[Nfunc=StringJoin[Nfunc,ToString[nombres⟦i⟧],","],{i,Length[nombres]-1}];
Nfunc=StringJoin[Nfunc,nombres⟦Length[nombres]⟧,") = "];
Print["Forma canónica en mintérminos es: ",Nfunc,fcminterminos];
Print["Forma canónica en maxtérminos es: ",Nfunc,fcmaxterminos];];
```

Ejercicio 10.1

Determinar las expresiones booleanas que representen las funciones booleanas elementales de dos variables de la tablas 10.3

$x \oplus y$	f6[x_y_]:=	BitXor[...]

Tabla 10.4. Funciones booleanas elementales más habituales de ...

Ejemplo 10.3. Implementar de distintas maneras la función booleana de tres variables dada por $f(x, y, z) = (x \wedge \bar{y}) \vee z$. Para comprobar que realmente funcionan evaluarlas para alguna de las combinaciones (0, 0,1), (1, 0, 1) y (1, 1, 1).

Comencemos definiendo de distintas formas la función f dándole en cada caso un nombre distinto:

```
In[]:=  f[{x_,y_,z_}]:=BitOr[BitAnd[x,BitNot[y]],z]

In[]:=  Compl[x_]:=Mod[x+1,2]
        f8[x_,y_]:=x*y
        f14[x_,y_]:=Compl[f8[Compl[x],Compl[y]]]
        ff[{x_,y_,z_}]:=f14[f8[x,Compl[y]],z]

In[]:=  Complb[x_]:=If[x==0,1,0]
        f8b[x_,y_]:=If[x==y==1,1,0]
        f14b[x_,y_]:=If[x==y==0,0,1]
        fff[{x_,y_,z_}]:=f14b[f8b[x,Complb[y]],z]
```

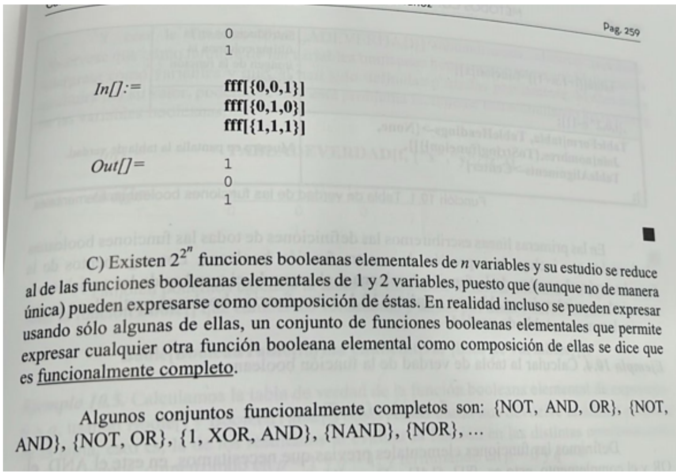
Ahora evaluamos para las combinaciones solicitadas:

```
In[]:=  f[{0,0,1}]
        f[{0,1,0}]
        f[{1,1,1}]

Out[]=  1
        0
        1

In[]:=  ff[{0,0,1}]
        ff[{0,1,0}]
        ff[{1,1,1}]

Out[]=  1
```


Pag. 259

```
        0
        1
In[]:=  fff[{0,0,1}]
        fff[{0,1,0}]
        fff[{1,1,1}]
Out[]=  1
        0
        1
```

■

C) Existen 2^{2^n} funciones booleanas elementales de *n* variables y su estudio se reduce al de las funciones booleanas elementales de 1 y 2 variables, puesto que (aunque no de manera única) pueden expresarse como composición de éstas. En realidad incluso se pueden expresar usando sólo algunas de ellas, un conjunto de funciones booleanas elementales que permite expresar cualquier otra función booleana elemental como composición de ellas se dice que es funcionalmente completo.

Algunos conjuntos funcionalmente completos son: {NOT, AND, OR}, {NOT, AND}, {NOT, OR}, {1, XOR, AND}, {NAND}, {NOR}, ...

In[ ]:=
```
f0[x_,y_]:=0;(*Constante igual a 0*)
f10[x_,y_]:=y;(*Igual a la segunda variable*)
f12[x_,y_]:=x;(*Igual a la primera variable*)
f15[x_,y_]:=1;(*Constante igual a 1*)
```

In[ ]:=
```
f8[x_,y_]:=x*y;(*Conjunción o producto*)
f7[x_,y_]:=Compl[f8[x,y]];(*NAND*)
f3[x_,y_]:=Compl[x];(*Complemento o negación de x*)
f5[x_,y_]:=Compl[y];(*Complemento o negación de y*)
```

In[ ]:=
```
f14[x_,y_]:=Compl[f8[Compl[x],Compl[y]]];(*Disyunción o suma*)
```

In[ ]:=
```
f1[x_,y_]:=Compl[f14[x,y]];(*NOR*)
f2[x_,y_]:=f8[Compl[x],y];(*--*)
f4[x_,y_]:=f8[x,Compl[y]];(*--*)
f6[x_,y_]:=Mod[x+y,2];(*XOR*)
f9[x_,y_]:=Compl[f6[x,y]];(*bicondicional*)
f11[x_,y_]:=f7[x,Compl[y]];(*Implicacion*)
f13[x_,y_]:=f11[y,x];(*Contraimplicacion*)
```

Ejercicio 10.3

Calcular la tabla de verdad, la forma canónica en mintérminos y maxtérminos de las funciones booleanas

a) f (x, y, z) = (x ∨ (y ∧ Compl[z]) ∧ z

In[∘]:=
```
fa[{x_,y_,z_}]:=BitAnd[BitOr[x,BitAnd[y,Compl[z]]],z];
TABLADEVERDAD[fa,{"x","y","z"}]
```

Out[∘]//TableForm=

x	y	z	fa
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

In[∘]:=
```
FCANONICAS[fa,{"x","y","z"}]
```

Supremo de mintérminos: fa $= m_{\{1,0,1\}} + m_{\{1,1,1\}}$

Ínfimo de maxtérminos: fa $= M_{\{0,0,0\}} \cdot M_{\{0,0,1\}} \cdot M_{\{0,1,0\}} \cdot M_{\{0,1,1\}} \cdot M_{\{1,0,0\}} \cdot M_{\{1,1,0\}}$

b) g (x, y, z) = (x ∧ (Compl[y] ∨ z)) ∨ (((x ∧ y) ∨ Compl[z]) ∧ x)

In[∘]:=
```
fb[{x_,y_,z_}]:=BitOr[BitAnd[x,BitOr[Compl[y],z]] , BitAnd[BitOr[BitAnd[x,y],Compl[z]],x]];
TABLADEVERDAD[fb,{"x","y","z"}]
```

Out[∘]//TableForm=

x	y	z	fb
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

In[∘]:=
```
FCANONICAS[fb,{"x","y","z"}]
```

Supremo de mintérminos: fb $= m_{\{1,0,0\}} + m_{\{1,0,1\}} + m_{\{1,1,0\}} + m_{\{1,1,1\}}$

Ínfimo de maxtérminos: fb $= M_{\{0,0,0\}} \cdot M_{\{0,0,1\}} \cdot M_{\{0,1,0\}} \cdot M_{\{0,1,1\}}$

c) h (x, y, z,t) = (x + t) ⊻ (z*y)

In[ ]:=
```
fc[{x_,y_,z_,t_}]:=f6[f14[x,t],f8[z,y]];
TABLADEVERDAD[fc,{"x","y","z","t"}]
```

Out[ ]//TableForm=

x	y	z	t	fc
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

In[ ]:=
```
FCANONICAS[fc,{"x","y","z","t"}]
```

Supremo de mintérminos: fc $= m_{\{0,0,0,1\}} + m_{\{0,0,1,1\}} + m_{\{0,1,0,1\}} + m_{\{0,1,1,0\}} + m_{\{1,0,0,0\}} + m_{\{1,0,0,1\}} + m_{\{1,0,1,0\}} + m_{\{1,0,1,1\}} + m_{\{1,1,0,0\}} + m_{\{1,1,0,1\}}$

Ínfimo de maxtérminos: fc $= M_{\{0,0,0,0\}} \cdot M_{\{0,0,1,0\}} \cdot M_{\{0,1,0,0\}} \cdot M_{\{0,1,1,1\}} \cdot M_{\{1,1,1,0\}} \cdot M_{\{1,1,1,1\}}$

Ejercicio 10.13 .

Consideramos las funciones booleanas:
f(x,y,z)=(x⊻y) ⇒(x∧z)
g(x,y,z)=Compl[x]+y*z
Calcular:

In[ ]:=
```
ff[{x_,y_,z_}]:=BitOr[BitXor[BitXor[x,y],1],BitAnd[x,z]]
fg[{x_,y_,z_}]:=f14[Compl[x],f8[y,z]]
```

a) Su expresión en mintérminos y en maxtérminos

In[ ]:=
```
FCANONICAS[ff,{"x","y","z"}]
FCANONICAS2[ff,{"x","y","z"}]
```

Supremo de mintérminos: ff = $m_{\{0,0,0\}}$ + $m_{\{0,0,1\}}$ + $m_{\{1,0,1\}}$ + $m_{\{1,1,0\}}$ + $m_{\{1,1,1\}}$

Ínfimo de maxtérminos: ff = $M_{\{0,1,0\}}$ · $M_{\{0,1,1\}}$ · $M_{\{1,0,0\}}$

Forma canónica en mintérminos es: ff(x,y,z) = $(x\wedge y\wedge z) \vee (x\wedge\overline{y}\wedge z) \vee (\overline{x}\wedge\overline{y}\wedge z) \vee (x\wedge y\wedge\overline{z}) \vee (\overline{x}\wedge\overline{y}\wedge\overline{z})$

Forma canónica en maxtérminos es: ff(x,y,z) = $(x\vee\overline{y}\vee\overline{z}) \wedge (x\vee\overline{y}\vee z) \wedge (\overline{x}\vee y\vee z)$

In[∘]:=
```
FCANONICAS[fg,{"x","y","z"}]
FCANONICAS2[fg,{"x","y","z"}]
```

Supremo de mintérminos: fg = $m_{\{0,0,0\}}$ + $m_{\{0,0,1\}}$ + $m_{\{0,1,0\}}$ + $m_{\{0,1,1\}}$ + $m_{\{1,1,1\}}$

Ínfimo de maxtérminos: fg = $M_{\{1,0,0\}}$ · $M_{\{1,0,1\}}$ · $M_{\{1,1,0\}}$

Forma canónica en mintérminos es: fg(x,y,z) = $(x\wedge y\wedge z) \vee (\overline{x}\wedge y\wedge z) \vee (\overline{x}\wedge\overline{y}\wedge z) \vee (\overline{x}\wedge y\wedge\overline{z}) \vee (\overline{x}\wedge\overline{y}\wedge\overline{z})$

Forma canónica en maxtérminos es: fg(x,y,z) = $(\overline{x}\vee y\vee\overline{z}) \wedge (\overline{x}\vee\overline{y}\vee z) \wedge (\overline{x}\vee y\vee z)$

b) Encontrar expresiones equivalentes en las que sólo se usen los siguientes conjuntos

b .1) {∧ , ¬}

In[∘]:=
```
BooleanConvert[Equivalent[Xor["x","y"],And["x","z"]],"AND"]
```

Out[∘]=

! (x && ! y && ! z) && ! (! x && y) && ! (y && z)

b .4) { ⊼ }

```
Clear[x,y,z,ExpNor]
ExpNor=BooleanConvert[Equivalent[Xor["x","y"],And["x","z"]],"NOR"]
A=ExpNor/. Not[x_]:→(x⊽x);
A
```

Out[∘]=

(! x ⊽ y ⊽ z) ⊽ (x ⊽ ! y) ⊽ (! y ⊽ ! z)

Out[∘]=

((x ⊽ x) ⊽ y ⊽ z) ⊽ (x ⊽ (y ⊽ y)) ⊽ ((y ⊽ y) ⊽ (z ⊽ z))

Caso a parte {¬,∧,∨}

In[∘]:=
```
BooleanConvert[Equivalent[Xor["x","y"], And["x","z"]]]
```

Out[∘]=

(x && y && ! z) || (! x && ! y) || (! y && z)

Práctica 11

In[1]:=

```
(* Euclides *)
Euclides[a_,b_]:=If[Mod[a,b]==0,Abs[b],Euclides[b,Mod[a,b]]]

(* Diofanticas *)
DIOFANTICA[a_,b_,c_]:=Module[{particular,k},If[Mod[Abs[c],GCD[Abs[a],Abs[b]]]≠0,
    Print["La ecuación no tiene solución"];,particular=FindInstance[a*x+b*y==c,{x,y},
        Integers];
 Print["La solución particular es:" ];
 Print["  x0 =  ",particular[[1]][[1]][[2]]," e  y0 = ",particular[[1]][[2]][[2]],"."];
 Print["La solución general es:"];
 Print["  x = ",particular[[1]][[1]][[2]]," + ",b/GCD[a,b],"t"];
 Print["  y = ",particular[[1]][[2]][[2]]," - ",a/GCD[a,b],"t."];];];

(* Bezout *)
Bezout2[x_,y_]:=Module[{a,b,temp,temp2,n1,n2,cocientes,aux1,aux2,euclides,coef1,coef2,coef3,
    coef4,listam,temporal},
n1=y;n2=x;b=Abs[n1];a=Abs[n2];
If[Mod[a,b]==0,euclides={{a,b,Quotient[a,b],0}};,r=1;euclides={};
While[r>0,q=Quotient[a,b];r=Mod[a,b];
 AppendTo[euclides,{a,b,q,r}];
 a=b;b=r;];
s=Length[euclides];];
Print["Algoritmo de Euclides"];
Do[Print["(",i,") ",euclides[[i,1]]," = ",euclides[[i,2]],"·",euclides[[i,3]]," + ",
    euclides[[i,4]]],{i,Length[euclides]}];
Print["m.c.d.{",n1,",",n2,"}=",a];
Print["m.c.m.{",n1,",",n2,"}=",(Abs[n1*n2])/a];
listam=Table[0,{i,s}];
Print["Cálculo de la Identidad de Bezout"];
If[Mod[Abs[n2],Abs[n1]]==0||Mod[Abs[n1],Abs[n2]]==0,
If[Mod[Abs[n2],Abs[n1]]==0,Print["Identidad de Bezout: ",n2,"·(0)+",n1,"·(",Sign[n1],
    ")=",Abs[n1],"."];];
If[Mod[Abs[n1],Abs[n2]]==0,Print["Identidad de Bezout: ",n1,"· 0)+",n2,"·(",Sign[n2],
    ")=",Abs[n2],"."];];
,Print["(",s-1,") ",euclides[[s-1,4]]," = ",euclides[[s-1,1]]," - ",euclides[[s-1,2]],"·",
    euclides[[s-1,3]]];
coef4=1;coef3=(-1)*euclides[[s-1,3]];
Do[Clear[aux1,aux2];
 listam[[i]]=aux1;listam[[i+1]]=aux2;
 For[f=i+2,f<s+1,f++,listam[[f]]=listam[[f-2]]-(listam[[f-1]]*euclides[[f-2,3]]);];
 temporal:=Simplify[listam[[s-1]]-(listam[[s]]*euclides[[s-1,3]])];
 Print["(",i,") ",euclides[[i,4]]," = ",euclides[[i,1]]," - ",euclides[[i,2]]," · ",
    euclides[[i,3]]];
 aux1=0;aux2=1;coef1=temporal;
 aux1=1;aux2=0;coef2=temporal;
 Print["    ",euclides[[s-1,4]]," = ",euclides[[i+1,1]]," · (",coef4,") + ","(",euclides[[i,1]],"
     - ",euclides[[i,2]]," · ",euclides[[i,3]],") · (",coef3,") = ",euclides[[i+1,1]]," ·
    (",coef1,") + ",euclides[[i,1]]," · (",coef2,")"];
 coef3=coef1;coef4=coef2;,{i,Length[euclides]-2,1,-1}];
 Print["Identidad de Bezout: ",n1,"· (",coef1*Sign[n1],") + ",n2," · (",coef2*Sign[n2],")
     = ",a,"."];];];
```

Ejercicio 11.3

Hallar el máximo común divisor d y el mínimo común múltiplo m usando el algoritmo de Euclides, calculando posteriormente la identidad de Bezout para los siguientes pares de números enteros:
a) 1400 y 6237

In[ ]:=
```
n1a=1400;
n2a=6237;
a=Abs[n1a];b=Abs[n2a];
m=1;
While[m>0,m=Mod[a,b];a=b;b=m;];
Print["m.c.d.(",n1a,",",n2a,")=",a]
Print["m.c.m.(",n1a,",",n2a,")=",Abs[(n1a*n2a)/a]]
```

```
m.c.d.(1400,6237)=7
m.c.m.(1400,6237)=1 247 400
```

In[ ]:=
```
Bezout2[1400,6237]
```

```
Algoritmo de Euclides
(1) 1400 = 6237·0 + 1400
(2) 6237 = 1400·4 + 637
(3) 1400 = 637·2 + 126
(4) 637 = 126·5 + 7
(5) 126 = 7·18 + 0
m.c.d.{6237,1400}=7
m.c.m.{6237,1400}=1 247 400
Cálculo de la Identidad de Bezout
(4) 7 = 637 - 126·5
(3) 126 = 1400 - 637 · 2
    7 = 637 · (1) + (1400 - 637 · 2) · (-5) = 637 · (11) + 1400 · (-5)
(2) 637 = 6237 - 1400 · 4
    7 = 1400 · (-5) + (6237 - 1400 · 4) · (11) = 1400 · (-49) + 6237 · (11)
(1) 1400 = 1400 - 6237 · 0
    7 = 6237 · (11) + (1400 - 6237 · 0) · (-49) = 6237 · (11) + 1400 · (-49)
Identidad de Bezout: 6237· (11) + 1400 · (-49) = 7.
```

b) 123840 y 4720

In[ ]:=
```
n1b=123840;
n2b=4720;
a=Abs[n1b];b=Abs[n2b];
m=1;
While[m>0,m=Mod[a,b];a=b;b=m;];
Print["m.c.d.(",n1b,",",n2b,")=",a]
Print["m.c.m.(",n1b,",",n2b,")=",Abs[(n1b*n2b)/a]]
```

m.c.d.(123 840,4720)=80

m.c.m.(123 840,4720)=7 306 560

In[ ]:=
```
Bezout2[123840,4720]
```

Algoritmo de Euclides

(1) 123 840 = 4720·26 + 1120

(2) 4720 = 1120·4 + 240

(3) 1120 = 240·4 + 160

(4) 240 = 160·1 + 80

(5) 160 = 80·2 + 0

m.c.d.{4720,123 840}=80

m.c.m.{4720,123 840}=7 306 560

Cálculo de la Identidad de Bezout

(4) 80 = 240 - 160·1

(3) 160 = 1120 - 240 · 4

80 = 240 · (1) + (1120 - 240 · 4) · (-1) = 240 · (5) + 1120 · (-1)

(2) 240 = 4720 - 1120 · 4

80 = 1120 · (-1) + (4720 - 1120 · 4) · (5) = 1120 · (-21) + 4720 · (5)

(1) 1120 = 123 840 - 4720 · 26

80 = 4720 · (5) + (123 840 - 4720 · 26) · (-21) = 4720 · (551) + 123 840 · (-21)

Identidad de Bezout: 4720· (551) + 123 840 · (-21) = 80.

c) 4394 y 1040

In[ ]:=
```
n1c=4394;
n2c=1040;
a=Abs[n1c];b=Abs[n2b];
m=1;
While[m>0,m=Mod[a,b];a=b;b=m;];
Print["m.c.d.(",n1c,",",n2c,")=",a]
Print["m.c.m.(",n1c,",",n2c,")=",Abs[(n1c*n2c)/a]]
```

```
m.c.d.(4394,1040)=2
m.c.m.(4394,1040)=2 284 880
```

In[ ]:=
```
Bezout2[4394,1040]
```

```
Algoritmo de Euclides
(1) 4394 = 1040·4 + 234
(2) 1040 = 234·4 + 104
(3) 234 = 104·2 + 26
(4) 104 = 26·4 + 0
m.c.d.{1040,4394}=26
m.c.m.{1040,4394}=175 760
Cálculo de la Identidad de Bezout
(3) 26 = 234 - 104·2
(2) 104 = 1040 - 234 · 4
    26 = 234 · (1) + (1040 - 234 · 4) · (-2) = 234 · (9) + 1040 · (-2)
(1) 234 = 4394 - 1040 · 4
    26 = 1040 · (-2) + (4394 - 1040 · 4) · (9) = 1040 · (-38) + 4394 · (9)
Identidad de Bezout: 1040· (-38) + 4394 · (9) = 26.
```

Ejercicio 11.4 .

Resolver la ecuación diofántica ax+by=c donde a es el año de tu nacimiento, b es el día del mes en que naciste y c = ab.

In[•]:=
```
a=2004;
b=3;
c=a*b
```

Out[•]=
```
6012
```

In[•]:=
```
Bezout2[a,b]
```

```
Algoritmo de Euclides
(1) 2004 = 3·668 + 0
m.c.d.{3,2004}=2004
m.c.m.{3,2004}=3
Cálculo de la Identidad de Bezout
Identidad de Bezout: 2004·(0)+3·(1)=3.
```

(2004,3)=3, como 3 es divisor de 6012 tiene solución

In[•]:=
```
d=3;
y0=1
x0=0
```

Out[•]=
```
1
```

Out[•]=
```
0
```

In[•]:=
```
y=y0-(b/d)*t
x=x0+(a/d)*t
```

Out[•]=
```
1 - t
```

Out[•]=
```
668 t
```

```
DIOFANTICA[a,b,c]
```

La solución particular es:

$x_0 = 0$ e $y_0 = 2004$.

La solución general es:

$x = 0 + 1t$

$y = 2004 - 668t$.

Ejercicio 11. 9.

Sea c el resto de dividir tu DNI entre 100. Encontrar, si existe, los números x∈Z tales que verifiquen simultáneamente:

I. (c+2)|x

II. El resto de dividir x entre (c+3) es 1.

In[4]:=
```
c=Mod[26268082,100];
x=(c+2)*k
x=(c+3)*q+1
```

Out[5]= 84 k

Out[6]= 1 + 85 q

84k - 85q = 1

In[7]:=
```
Bezout2[84,-85]
```

```
Algoritmo de Euclides
(1) 84 = 85·0 + 84
(2) 85 = 84·1 + 1
(3) 84 = 1·84 + 0
m.c.d.{-85,84}=1
m.c.m.{-85,84}=7140
Cálculo de la Identidad de Bezout
(2) 1 = 85 - 84·1
(1) 84 = 84 - 85 · 0
    1 = 85 · (1) + (84
     - 85 · 0) · (-1) = 85 ·
    (1) + 84 · (-1)
Identidad de Bezout: -85· (-1) + 84 · (-1)
     = 1.
```

El MCD es 1, es divisor de 85, existe solución

In[8]:=
```
a=84;
b=-85;
c=1;
d=1;
y0=-1
x0=-1
```

Out[12]= -1

Out[13]= -1

In[14]:=
```
y=y0-(b/d)*t
x=x0+(a/d)*t
```

Out[14]=

$-1+85\,t$

Out[15]=

$-1+84\,t$

Ejercicio 11.10 .

Si x, y, z son los ángulos de un triángulo expresados en grados. Suponiendo que x,y,z son números enteros, calcular todos los triángulos cuyos ángulos verifiquen la siguiente ecuación:
2x+5y=3z

z=180-x-y
2x+5y=3(180-x-y)
5x+8y=540

In[ ]:=
```
Bezout2[8,5]
```

```
Algoritmo de Euclides
(1) 8 = 5·1 + 3
(2) 5 = 3·1 + 2
(3) 3 = 2·1 + 1
(4) 2 = 1·2 + 0
m.c.d.{5,8}=1
m.c.m.{5,8}=40
Cálculo de la Identidad de Bezout
(3) 1 = 3 - 2·1
(2) 2 = 5 - 3 · 1
    1 = 3 · (1) + (5 - 3 · 1) · (-1) = 3 · (2) + 5 · (-1)
(1) 3 = 8 - 5 · 1
    1 = 5 · (-1) + (8 - 5 · 1) · (2) = 5 · (-3) + 8 · (2)
Identidad de Bezout: 5· (-3) + 8 · (2) = 1.
```

El MCD es 1, es divisor de 540, existe solución

In[ ]:=
```
a=8;
b=5;
c=540;
d=1;
y0=2*(c/d)
x0=-3*(c/d)
```

Out[ ]=
1080

Out[ ]=
-1620

In[ ]:=
```
y=y0-(b/d)*t
x=x0+(a/d)*t
```

Out[ ]=
1080 - 5 t

Out[ ]=
-1620 + 8 t

In[ ]:=
```
z=180-x-y
```

Out[ ]=
720 - 3 t

```
Clear[t]
Reduce[1080-5t≥1 && 1080-5t≤178 && -1620+8t≥1 && -1620+8t≤178 && 720-3t≥1 &&
    720-3t≤178,t,Integers]
```

Out[ ]=
$t \in \mathbb{Z}$ && $203 \leq t \leq 215$

In[ ]:=
```
For[t=203,t≤215,t++,
    Print["PARA t=",t];
    Print["x=",x];
    Print["y=",y];
    Print["z=",y];
]
```

PARA t=203

x=4

y=65

z=65

PARA t=204

x=12

y=60

z=60

```
PARA t=205
x=20
y=55
z=55
PARA t=206
x=28
y=50
z=50
PARA t=207
x=36
y=45
z=45
PARA t=208
x=44
y=40
z=40
PARA t=209
x=52
y=35
z=35
PARA t=210
x=60
y=30
z=30
PARA t=211
x=68
y=25
z=25
PARA t=212
x=76
y=20
z=20
PARA t=213
x=84
y=15
z=15
PARA t=214
x=92
y=10
```

z=10

PARA t=215

x=100

y=5

z=5

In[∘]:=
```
DIOFANTICA[5,8,540]
```

La solución particular es:

x_0 = 4 e y_0 = 65.

La solución general es:

x = 4 + 8t

y = 65 - 5t.

Práctica 12

In[19]:=
```
Inverso[x_,n_]:=
    Module[{inverso},
        If[GCD[x,n]≠ 1,Print[“No existe”];,
            Do[If[Mod[j*x,n]==1,inverso=j;
                Break[]],{j,1,n-1}];
             Return[inverso]
        ];
    ];

(* Bezout *)
Bezout[x_,y_]:=
    Module[{Cocientes,Euclides,Resto,b,a1,a2},
        Euclides[n1_,n2_]:=
            If[Mod[n1,n2]==0,n2,
                AppendTo[Cocientes,Quotient[n1,n2]];
                Euclides[n2,Mod[n1,n2]]];
     Resto[1,x1_,x2_]:=x1-x2*Cocientes[[1]];
     Resto[2,x1_,x2_]:=
        x2-Resto[1,x1,x2]*Cocientes[[2]];
     Resto[k_,x1_,x2_]:=
        Expand[Resto[k-2,x1,x2]-
            Resto[k-1,x1,x2]*Cocientes[[k]]];
     a1=Abs[x];a2=Abs[y];
     Cocientes={};b=Euclides[a1,a2];
     Print["Identidad de Bezout: (",
        Resto[Length[Cocientes],1,0]*Sign[x],")·(",x,
            ")+(",Resto[Length[Cocientes],0,1]*Sign[y],
```

```
      ")·(",  y, ")=", b]
];
(* Algoritmo Chino *)
AlgChino2[a_,m_]:=Module[{M,b,Teorema,n,u,w,x0},
    Teorema="sí";
    n=Length[a];
    Do[
        Do[
            If[GCD[m⟦i⟧,m⟦j⟧]≠1,Teorema="no";Break[];],
        {j,i+1,Length[m]}],
    {i,Length[m]-1}];
    If[Teorema=="sí",
        Inverso[x_,n_]:=
            Module[{inverso},Do[If[Mod[j*x,n]==1,inverso=j;
                Break[]],{j,1,n-1}];
             Return[inverso]];
    M[1]=1;M[i_]:=M[i-1]*m⟦i-1⟧;
    u[i_]:=Inverso[M[i],m⟦i⟧];
    b[1]=Mod[a⟦1⟧,m⟦1⟧];
    w[i_]:=Mod[(a⟦i⟧-b[i-1])*u[i],m⟦i⟧];
    b[i_]:=b[i-1]+w[i]×M[i];
    x0=b[Length[a]];
    Print["Solución: x = ",x0," + t*",M[n+1]];
    Print[TableForm[Join[{{a⟦1⟧,m⟦1⟧,1,1,"-",b[1]}},
        Table[{a⟦i⟧,m⟦i⟧,M[i],u[i],w[i],b[i]},
            {i,2,n}]],TableHeadings→{Table[i,{i,n}],{"a","m","M","u","w","b"}}]];,
    Print["No podemos aplicar el Tª Chino del Resto"];];
];

(* Euclides para comprobar coprimos *)
Euclides[a_,b_]:=If[Mod[a,b]==0,Abs[b],Euclides[b,Mod[a,b]]]
```

Ejercicio 11. 9.

Sea c el resto de dividir tu DNI entre 100. Encontrar, si existe, los números $x \in Z$ tales que verifiquen simultáneamente :

I . $(c+2) \mid x$

II . El resto de dividir x entre $(c+3)$ es 1.

In[ ]:=
```
c=Mod[26268082,100];
x=(c+2)*k
x=(c+3)*q+1
```

Out[ ]=

84 k

Out[ ]=

1 + 85 q

$x \equiv 0 \bmod 84$

$x \equiv 1 \bmod 85$

In[∘]:=
```
Euclides[85,84]
```

Out[∘]=

1

Como el resultado es 1 son coprimos, por ende tiene solución

In[∘]:=
```
AlgChino2[{0,1},{(c+2),(c+3)}]
```

Solución: x = 7056 + t*7140

	a	m	M	u	w	b
1	0	84	1	1	–	0
2	1	85	84	84	84	7056

Ejercicio 12.1.

Calcular el inverso de tu DNI, si es posible, Z5, Z27 y Z1001.

In[∘]:=
```
Bezout[26268082,5]
```

Identidad de Bezout: (−2) · (26 268 082) + (10 507 233) · (5) =1

In[∘]:=
```
PowerMod[26268082,-1,5]
```

Out[∘]=

3

El inverso de mi DNI es 3.

In[∘]:=
```
Bezout[26268082,27]
```

Identidad de Bezout: (13) · (26 268 082) + (−12 647 595) · (27) =1

El inverso de mi DNI es 13

In[∘]:=
```
Bezout[26268082,1001]
```

Identidad de Bezout: (−244) · (26 268 082) + (6 403 009) · (1001) =1

In[∘]:=
```
PowerMod[26268082,-1,1001]
```

Out[∘]=

757

El inverso de mi DNI es 757

Ejercicio 12.5

Resolver los siguientes sistema de congruencias.

a) x ≡ 2 mod 3
x ≡ 5 mod 7
x ≡ 6 mod 11
x ≡ DNI mod 17

In[∘]:=
```
Euclides[3,7]
Euclides[3,11]
Euclides[3,17]
Euclides[7,11]
Euclides[7,17]
Euclides[11,17]
```

Out[∘]=
1

Out[∘]=
1

Out[∘]=
1

Out[∘]=
1

Out[∘]=
1

Out[∘]=
1

Como da 1 todos son comprimos. (3,7)=(3,11)=(3,17)=(7,11)(7,17)=(11,17)=1

In[∘]:=
```
AlgChino2[{2,5,6,26268082},{3,7,11,17}]
```

Solución: x = 3218 + t*3927

	a	m	M	u	w	b
1	2	3	1	1	–	2
2	5	7	3	5	1	5
3	6	11	21	10	10	215
4	26 268 082	17	231	12	13	3218

b) x ≡ 0 mod 3
x ≡ 5 mod 7
x ≡ DNI mod 100

In[•]:=
```
Euclides[3,7]
Euclides[3,100]
Euclides[7,100]
```

Out[•]=
1

Out[•]=
1

Out[•]=
1

Como da 1 todos son comprimos . (3, 7) = (3, 100) = (7, 100) = 1

In[•]:=
```
AlgChino2[{0,5,26268082},{3,7,100}]
```

Solución: x = 1482 + t*2100

	a	m	M	u	w	b
1	0	3	1	1	–	0
2	5	7	3	5	4	12
3	26 268 082	100	21	81	70	1482

Práctica 13

```
Bezout2[x_,y_]:=Module[{a,b,temp,temp2,n1,n2,cocientes,aux1,aux2,euclides,coef1,coef2,coef3,
                                                            coef4,listam,temporal},
n1=y;n2=x;b=Abs[n1];a=Abs[n2];
If[Mod[a,b]==0,euclides={{a,b,Quotient[a,b],0}};,r=1;euclides={};
While[r>0,q=Quotient[a,b];r=Mod[a,b];
AppendTo[euclides,{a,b,q,r}];
a=b;b=r;];
s=Length[euclides];];
Print["Algoritmo de Euclides"];
Do[Print["(",i,") ",euclides⟦i,1⟧," = ",euclides⟦i,2⟧,"·",euclides⟦i,3⟧," + ",euclides⟦i,4⟧]
{i,Length[euclides]}];
Print["m.c.d.{",n1,",",n2,"}=",a];
Print["m.c.m.{",n1,",",n2,"}=",(Abs[n1*n2])/a];
listam=Table[0,{i,s}];
Print["Cálculo de la Identidad de Bezout"];
If[Mod[Abs[n2],Abs[n1]]==0||Mod[Abs[n1],Abs[n2]]==0,
If[Mod[Abs[n2],Abs[n1]]==0,Print["Identidad de Bezout: ",n2,"·(0)+",n1,"·(",Sign[n1],")=",
Abs[n1],"."];];
If[Mod[Abs[n1],Abs[n2]]==0,Print["Identidad de Bezout: ",n1,"· 0)+",n2,"·(",Sign[n2],")=",
Abs[n2],"."];];
,Print["(",s-1,") ",euclides⟦s-1,4⟧," = ",euclides⟦s-1,1⟧," - ",euclides⟦s-1,2⟧,"·",
euclides⟦s-1,3⟧];
coef4=1;coef3=(-1)*euclides⟦s-1,3⟧;
Do[Clear[aux1,aux2];
listam⟦i⟧=aux1;listam⟦i+1⟧=aux2;
For[f=i+2,f<s+1,f++,listam⟦f⟧=listam⟦f-2⟧-(listam⟦f-1⟧*euclides⟦f-2,3⟧);];
temporal:=Simplify[listam⟦s-1⟧-(listam⟦s⟧*euclides⟦s-1,3⟧)];
Print["(",i,") ",euclides⟦i,4⟧," = ",euclides⟦i,1⟧," - ",euclides⟦i,2⟧," · ",euclides⟦i,3⟧];
aux1=0;aux2=1;coef1=temporal;
aux1=1;aux2=0;coef2=temporal;
Print["    ",euclides⟦s-1,4⟧," = ",euclides⟦i+1,1⟧," · (",coef4,") + ","(",euclides⟦i,1⟧," -
euclides⟦i,2⟧," · ",euclides⟦i,3⟧,") · (",coef3,") = ",euclides⟦i+1,1⟧," · (",coef1,") + ",
euclides⟦i,1⟧," · (",coef2,")"];
coef3=coef1;coef4=coef2;,{i,Length[euclides]-2,1,-1}];
Print["Identidad de Bezout: ",n1,"· (",coef1*Sign[n1],") + ",n2," · (",coef2*Sign[n2],") = "
a,"."];];];

FromDecimal[n_,b_]:=Module[{n2,d},d={};n2=n;
While[Quotient[n2,b]>0,PrependTo[d,Mod[n2,b]];
n2=Quotient[n2,b]];
PrependTo[d,Mod[n2,b]];
Do[If[d⟦i⟧≥10,d⟦i⟧+=7],{i,Length[d]}];
FromCharacterCode[d+48]];

ToDecimal[n_,b_]:=Module[{M,n2},n2=ToCharacterCode[ToUpperCase[n]]-48;
Do[If[n2⟦i⟧≥17,n2⟦i⟧-=7],{i,Length[n2]}];
M=Sum[n2⟦i⟧*b^(Length[n2]-i),{i,Length[n2]}]];

CambioBase[n_,b1_,b2_]:=FromDecimal[ToDecimal[n,b1],b2]
```

Ejercicio 11. 17.

Plantear una ecuación diofántica[30] que permita calcular, si existen, números enteros x e y tales que:

$$(5\,F)_x - (50)_y = 7$$

5x+F

-5y

5x + f -5y = +7 - > 5x-5y = -8

In[ ]:=
```
Bezout2[5,-5]
```

```
Algoritmo de Euclides
(1)  5 = 5·1 + 0
m.c.d.{-5,5}=5
m.c.m.{-5,5}=5
Cálculo de la Identidad de Bezout
Identidad de Bezout: 5·(0)+-5·(-1)=5.
Identidad de Bezout: -5· 0)+5·(1)=5.
```

(5,-5)=5, como 5 no es divisor de -8 no existe solución

Ejercicio 12.13.

Consideramos el número de tu DNI (incluida la letra) en base 40, expresarlo en decimal y en bases 2,5,8,13,16 y 23.

In[ ]:=
```
dni="26268082N";
dniBase40 = CambioBase[dni,40,10]
dniBase40 = CambioBase[dni,40,2]
dniBase40 = CambioBase[dni,40,5]
dniBase40 = CambioBase[dni,40,8]
dniBase40 = CambioBase[dni,40,13]
dniBase40 = CambioBase[dni,40,16]
dniBase40 = CambioBase[dni,40,23]
```

Out[ ]=
```
14099066892903
```

Out[ ]=
```
11001101001010110001110111111011001001100111
```

Out[ ]=
```
3321444342111033103
```

Out[ ]=
```
315126167731147
```

Out[ ]=
```
7B36C83479C5
```

Out[ ]=
```
CD2B1DFB267
```

Out[ ]=
```
7J0KK080F2
```