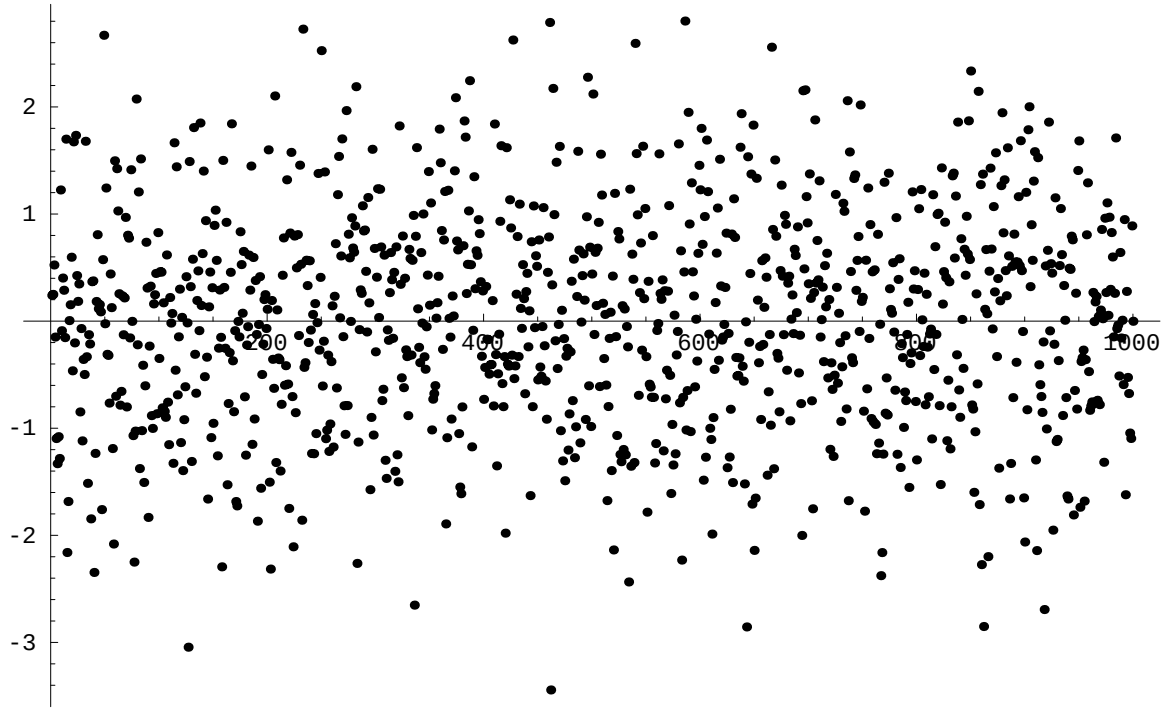
Medie, mediana e moda

In[1]:=
```
Media[X_List] := Apply[Plus,X]/Length[X]
Media[x__] := Media[{x}]
```
In[3]:=
```
Media[x[1],x[2],x[3],x[4]]
```
Out[3]=
$$\frac{x[1] + x[2] + x[3] + x[4]}{4}$$
In[4]:=
```
Sum[(x - x[i])^2,{i,1,4}]
```
Out[4]=
$$(x - x[1])^2 + (x - x[2])^2 + (x - x[3])^2 + (x - x[4])^2$$
In[5]:=
```
D[%,x]
```
Out[5]=
$$2\,(x - x[1]) + 2\,(x - x[2]) + 2\,(x - x[3]) + 2\,(x - x[4])$$
In[6]:=
```
Solve[% == 0,x]
```
Out[6]=
$$\{\{x \to \frac{x[1] + x[2] + x[3] + x[4]}{4}\}\}$$

In[7]:=
```
D[%%,x] /. %[[1]]
```
Out[7]=
$$8$$
In[8]:=
```
MediaGen[e_][X_List] := Media[X^e]^(1/e)
MediaGen[e_][x__] := MediaGen[e][{x}]
```
In[10]:=
```
MediaArm := MediaGen[-1]
MediaQuad := MediaGen[2]
```
In[12]:=
```
MediaArm[x[1],x[2],x[3],x[4]]
```
Out[12]=
$$\frac{4}{\frac{1}{x[1]}+\frac{1}{x[2]}+\frac{1}{x[3]}+\frac{1}{x[4]}}$$
In[13]:=
```
MediaQuad[x[1],x[2],x[3],x[4]]
```
Out[13]=
$$\frac{\text{Sqrt}[x[1]^2+x[2]^2+x[3]^2+x[4]^2]}{2}$$
In[14]:=
```
MediaGeom[X_List] :=
  Apply[Times,X]^(1/Length[X])
MediaGeom[x__] := MediaGeom[{x}]
```
In[16]:=
```
MediaGeom[x[1],x[2],x[3],x[4]]
```
Out[16]=
$$(x[1]\ x[2]\ x[3]\ x[4])^{1/4}$$

In[17]:=
```
dati = Table[Random[Real,10],{1000}];
```
In[18]:=
```
{Media[dati],MediaQuad[dati],
 MediaGeom[dati]}
```
Out[18]=
```
{5.06127, 5.82622, 3.725726770147483}
```
In[19]:=
```
MediaPesata[X:{{_,_}..}] :=
  Sum[X[[i,1]] X[[i,2]],{i,Length[X]}] /
  Sum[X[[i,2]],{i,Length[X]}]
MediaPesata[xp:{_,_}..] :=
  MediaPesata[{xp}]
```
In[21]:=
```
MediaPesata[{x[1],p[1]},
            {x[2],p[2]},
            {x[3],p[3]}]
```
Out[21]=
$$\frac{p[1]\ x[1] + p[2]\ x[2] + p[3]\ x[3]}{p[1] + p[2] + p[3]}$$
In[22]:=
```
RandomGauss[med_,var_] :=
  med + Cos[2 Pi Random[]] *
        Sqrt[-2 Log[Random[]]] var
```
In[23]:=
```
dati = Table[RandomGauss[0,1],{1000}];
```

In[24]:=
```
ListPlot[dati]
```



Out[24]=
```
-Graphics-
```

In[25]:=
```
Mediana[X_List] :=
  Sort[X][[IntegerPart[Length[X]/2]]]
```

In[26]:=
```
Mediana[dati]
```

Out[26]=
```
0.0278943
```

In[27]:=
```
Moda[X_List] :=
  First[Last[Sort[Split[Sort[X]]]]]
```

```
General::spell:
   Possible spelling error: new symbol
     name "Moda" is similar to existing
     symbols {Mod, Mode}.
```

In[28]:=

```
Moda[dati]
```

Out[28]=

```
2.80203
```

In[29]:=

```
Moda[Round[dati]]
```

Out[29]=

```
0
```

Diverse definizioni del fattoriale

In[30]:=

```
FattDo[n_Integer] :=
  Module[{p = 1},Do[p = p i,{i,n}];p]
```

In[31]:=

```
FattWhile[n_Integer /; n >= 0] :=
  Module[{i = 1,p = 1},
    While[i <= n,p = p i;i = i + 1];p]
```

In[32]:=

```
FattMat[n_Integer] := Product[i,{i,1,n}]
```

In[33]:=

```
FattRic[n_Integer /; n >= 0] :=
  If[n == 0,1,n FattRic[n - 1]]
```

In[34]:=

```
Fatt[0] := 1
Fatt[n_Integer /; n > 0] := n Fatt[n - 1]
```

In[36]:=

```
{FattDo[5],FattWhile[5],
 FattMat[5],FattRic[5],Fatt[5]}
```

Out[36]=

```
{120, 120, 120, 120, 120}
```

In[37]:=

```
{Timing[FattDo[500];],
 Timing[FattWhile[500];],
 Timing[FattMat[500];],
 Timing[FattRic[500];],
 Timing[Fatt[500];]} // TableForm
```

```
$RecursionLimit::reclim:
   Recursion depth of 256 exceeded.
$RecursionLimit::reclim:
   Recursion depth of 256 exceeded.
```

Out[37]//TableForm=

```
0.0166667 Second   Null

0.0166667 Second   Null

0.0166667 Second   Null

0.0333333 Second   Null

0.0333333 Second   Null
```

In[38]:=

```
? $RecursionLimit
$RecursionLimit gives the current limit
   on the number of levels of recursion
   that Mathematica can use.
```

In[39]:=

```
$RecursionLimit = 2000;
```

In[40]:=

```
{Timing[FattDo[1000];],
 Timing[FattWhile[1000];],
 Timing[FattMat[1000];],
 Timing[FattRic[1000];],
 Timing[Fatt[1000];]} // TableForm
```

Out[40]//TableForm=

0.0333333 Second	Null
0.05 Second	Null
0. Second	Null
0.0833333 Second	Null
0.05 Second	Null

Diverse definizioni di MCD

In[41]:=
```
MCD[n_Integer,m_Integer] :=
  If[n == 0,Abs[m],MCD[Abs[m - n],n]]
```
In[42]:=
```
MCD[24,36]
```
Out[42]=
```
12
```
In[43]:=
```
MCD[24,36,40]
```
Out[43]=
```
MCD[24, 36, 40]
```
In[44]:=
```
MCD[x_Integer] := Abs[x]
MCD[x_,y_,z__] := MCD[x,MCD[y,z]]
```
In[46]:=
```
{MCD[24],MCD[24,36],MCD[24,36,40]}
```
Out[46]=
```
{24, 12, 4}
```
In[47]:=
```
MCD[3,10000]
$IterationLimit::itlim:
   Iteration limit of 4096 exceeded.
```
Out[47]=
```
Hold[MCD[Abs[3 - 5905], 5905]]
```
In[48]:=
```
? $IterationLimit
$IterationLimit gives the maximum length
   of evaluation chain used in trying to
   evaluate any expression.
```

In[49]:=

```
$IterationLimit = 100000;
```

In[50]:=

```
MCD[3,10000]
```

Out[50]=

```
1
```

In[51]:=

```
MCDMod[n_Integer] := Abs[n]

SetAttributes[MCDMod,{Flat,OneIdentity}]

MCDMod[n_Integer,m_Integer] :=
  If[n == 0,Abs[m],MCDMod[Mod[m,n],n]]
```

In[54]:=

```
{MCDMod[24],MCDMod[24,36],
 MCDMod[24,36,40]}
```

Out[54]=

```
{24, 12, 4}
```

In[55]:=

```
Divisors[24]
```

Out[55]=

```
{1, 2, 3, 4, 6, 8, 12, 24}
```

In[56]:=

```
MCDDiv[n_Integer,m_Integer] :=
  Max[Intersection[Divisors[n],
                   Divisors[m]]]
```

In[57]:=

```
MCDDiv[24,36]
```

Out[57]=

```
12
```

In[58]:=

```
MCDDiv[n__Integer] :=
  Max[Apply[Intersection,
            Map[Divisors,{n}]]]
```

In[59]:=

```
{MCDDiv[24],MCDDiv[24,36],
 MCDDiv[24,36,40]}
```

Out[59]=

```
{24, 12, 4}
```

In[60]:=

```
{Timing[GCD[123456,12]],
 Timing[MCD[123456,12]],
 Timing[MCDMod[123456,12]],
 Timing[MCDMod[123456,12]]} // TableForm
```

Out[60]//TableForm=

```
0.          Second   12
0.866667    Second   12
0.          Second   12
0.          Second   12
```

Principi di equivalenza

In[61]:=
```
IPrincEq[m1_ == m2_,esp_] :=
  m1 + esp == m2 + esp
```
In[62]:=
```
IIPrincEq[m1_ == m2_,esp_] :=
  m1 esp == m2 esp
```
```
General::spell1:
   Possible spelling error: new symbol
     name "IIPrincEq"
     is similar to existing symbol
    "IPrincEq".
```
In[63]:=
```
a x + b == 0
```
Out[63]=
```
b + a x == 0
```
In[64]:=
```
IPrincEq[%,-b]
```
Out[64]=
```
a x == -b
```
In[65]:=
```
IIPrincEq[%,1/a]
```
Out[65]=
$$x == -\left(\frac{b}{a}\right)$$

In[66]:=
```
SplitEq[e1_ e2_ == 0] :=
  e1 == 0 || e2 == 0
```

In[67]:=
```
a x^2 + b x + c == 0
```
Out[67]=
$$c + b x + a x^2 == 0$$
In[68]:=
```
IIPrincEq[%,4 a]
```
Out[68]=
$$4 a (c + b x + a x^2) == 0$$
In[69]:=
```
MapAt[Expand,%,1]
```
Out[69]=
$$4 a c + 4 a b x + 4 a^2 x^2 == 0$$
In[70]:=
```
IPrincEq[%,b^2 - 4 a c]
```
Out[70]=
$$b^2 + 4 a b x + 4 a^2 x^2 == b^2 - 4 a c$$
In[71]:=
```
MapAt[Factor,%,1]
```
Out[71]=
$$(b + 2 a x)^2 == b^2 - 4 a c$$
In[72]:=
```
% /. {b + 2 a x -> y,
      b^2 - 4 a c -> d^2}
```
Out[72]=
$$y^2 == d^2$$

In[73]:=
```
IPrincEq[%,-d^2]
```
Out[73]=
$$-d^2 + y^2 == 0$$
In[74]:=
```
MapAt[Factor,%,1]
```
Out[74]=
$$-((d - y)\,(d + y)) == 0$$
In[75]:=
```
IIPrincEq[%,-1]
```
Out[75]=
$$(d - y)\,(d + y) == 0$$
In[76]:=
```
SplitEq[%]
```
Out[76]=
$$d - y == 0 \;||\; d + y == 0$$
In[77]:=
```
Map[IPrincEq[#,-d]&,%]
```
Out[77]=
$$-y == -d \;||\; y == -d$$
In[78]:=
```
MapAt[IIPrincEq[#,-1]&,%,1]
```
Out[78]=
$$y == d \;||\; y == -d$$

In[79]:=
```
% /. {y -> b + 2 a x,
      d -> Sqrt[b^2 - 4 a c]}
```
Out[79]=
$$b + 2 a x == \sqrt{b^2 - 4 a c} \;||\; b + 2 a x == -\sqrt{b^2 - 4 a c}$$
In[80]:=
```
Map[IPrincEq[#,-b]&,%]
```
Out[80]=
$$2 a x == -b + \sqrt{b^2 - 4 a c} \;||\; 2 a x == -b - \sqrt{b^2 - 4 a c}$$
In[81]:=
```
Map[IIPrincEq[#,1/(2 a)]&,%]
```
Out[81]=
$$x == \frac{-b + \sqrt{b^2 - 4 a c}}{2 a} \;||\; x == \frac{-b - \sqrt{b^2 - 4 a c}}{2 a}$$
In[82]:=
```
SolveEq[x_^n_Integer == c_] :=
  If[OddQ[n],
     x == c^(1/n),
     x == c^(1/n) || x == -c^(1/n)]
```

In[83]:=
```
%71
```
Out[83]=
$$(b + 2 a x)^2 == b^2 - 4 a c$$
In[84]:=
```
SolveEq[%]
```
Out[84]=
$$b + 2 a x == \text{Sqrt}[b^2 - 4 a c] \;||\; b + 2 a x == -\text{Sqrt}[b^2 - 4 a c]$$
In[85]:=
```
Map[IPrincEq[#,-b]&,%]
```
Out[85]=
$$2 a x == -b + \text{Sqrt}[b^2 - 4 a c] \;||\; 2 a x == -b - \text{Sqrt}[b^2 - 4 a c]$$
In[86]:=
```
Map[IIPrincEq[#,1/(2 a)]&,%]
```
Out[86]=
$$x == \frac{-b + \text{Sqrt}[b^2 - 4 a c]}{2 a} \;||\; x == \frac{-b - \text{Sqrt}[b^2 - 4 a c]}{2 a}$$

Insiemi e relazioni

In[87]:=
```
Insieme[x___] :=
  Module[{sx = Union[{x}]},
         Apply[Insieme,sx] /; sx =!= {x}]
```
In[88]:=
```
X = Insieme[0,1,2,3]
```
Out[88]=
```
Insieme[0, 1, 2, 3]
```
In[89]:=
```
Y = Insieme[0,-1,-2,-3]
```
Out[89]=
```
Insieme[-3, -2, -1, 0]
```
In[90]:=
```
ElementoQ[x_,
  Insieme[a___,x_,z___]] := True
ElementoQ[_,Insieme[___]] := False
```
In[92]:=
```
ElementoQ[1,X]
```
Out[92]=
```
True
```
In[93]:=
```
ElementoQ[x,X]
```
Out[93]=
```
False
```
In[94]:=
```
ContenutoQ[A_Insieme,B_Insieme] :=
  Apply[And,Map[ElementoQ[#,B]&,A]]
```

In[95]:=
```
ContenutoQ[Insieme[],X]
```
Out[95]=
```
True
```
In[96]:=
```
ContenutoQ[X,X]
```
Out[96]=
```
True
```
In[97]:=
```
ContenutoQ[X,Y]
```
Out[97]=
```
False
```
In[98]:=
```
SetAttributes[{Unione,Intersezione},
  {Flat,OneIdentity,Orderless}]
```
```
General::spell1:
   Possible spelling error: new symbol
     name "Unione"
     is similar to existing symbol
    "Union".
```
In[99]:=
```
Unione[Insieme[x___],
       Insieme[y___]] := Insieme[x,y]
```
In[100]:=
```
Intersezione[A_Insieme,B_Insieme] :=
  Select[A,ElementoQ[#,B]&]
```
In[101]:=
```
Unione[X,X]
```
Out[101]=
```
Insieme[0, 1, 2, 3]
```

```
In[102]:=
 Unione[X,Y]
Out[102]=
 Insieme[-3, -2, -1, 0, 1, 2, 3]
In[103]:=
 Intersezione[X,X]
Out[103]=
 Insieme[0, 1, 2, 3]
In[104]:=
 Intersezione[X,Y]
Out[104]=
 Insieme[0]
In[105]:=
 Prodotto[A_Insieme,B_Insieme] :=
   Flatten[Outer[{#1,#2}&,A,B]]
In[106]:=
 Prodotto[Insieme[],X]
Out[106]=
 Insieme[]
In[107]:=
 Prodotto[X,X]
Out[107]=
 Insieme[{0, 0}, {0, 1}, {0, 2}, {0, 3},
   {1, 0}, {1, 1}, {1, 2}, {1, 3},
   {2, 0}, {2, 1}, {2, 2}, {2, 3},
   {3, 0}, {3, 1}, {3, 2}, {3, 3}]
In[108]:=
 RelQ[R_] := MatchQ[R,Insieme[{_,_}...]]
```

In[109]:=
```
Rel1 = Select[Prodotto[X,X],
          (#[[2]] == Mod[#[[1]],2])&]
```
Out[109]=
```
Insieme[{0, 0}, {1, 1}, {2, 0}, {3, 1}]
```
In[110]:=
```
Rel2 = Select[Prodotto[X,X],
          (#[[2]] - #[[1]] == 1)&]
```
Out[110]=
```
Insieme[{0, 1}, {1, 2}, {2, 3}]
```
In[111]:=
```
Rel3 = Select[Prodotto[X,X],
          (Mod[#[[2]] - #[[1]],2] == 0)&]
```
Out[111]=
```
Insieme[{0, 0}, {0, 2}, {1, 1}, {1, 3},
  {2, 0}, {2, 2}, {3, 1}, {3, 3}]
```
In[112]:=
```
Map[RelQ,{Rel1,Rel2,Rel3}]
```
Out[112]=
```
{True, True, True}
```
In[113]:=
```
Dominio[R_?RelQ] := Map[First,R]
Codominio[R_?RelQ] := Map[Last,R]
```
In[115]:=
```
Map[Dominio,{Rel1,Rel2,Rel3}]
```
Out[115]=
```
{Insieme[0, 1, 2, 3], Insieme[0, 1, 2],
  Insieme[0, 1, 2, 3]}
```

In[116]:=
```
Map[Codominio,{Rel1,Rel2,Rel3}]
```
Out[116]=
```
{Insieme[0, 1], Insieme[1, 2, 3],
  Insieme[0, 1, 2, 3]}
```
In[117]:=
```
UnivocaQ[R_?RelQ] :=
  Length[Dominio[R]] == Length[R]
```
In[118]:=
```
IniettivaQ[R_?RelQ] :=
  Length[Codominio[R]] == Length[R]
```
In[119]:=
```
BiunivocaQ[R_?RelQ] :=
  UnivocaQ[R] && IniettivaQ[R]
```
In[120]:=
```
Map[UnivocaQ,{Rel1,Rel2,Rel3}]
```
Out[120]=
```
{True, True, False}
```
In[121]:=
```
Map[IniettivaQ,{Rel1,Rel2,Rel3}]
```
Out[121]=
```
{False, True, False}
```
In[122]:=
```
Map[BiunivocaQ,{Rel1,Rel2,Rel3}]
```
Out[122]=
```
{False, True, False}
```
In[123]:=
```
R_Insieme[x_,y_] /; RelQ[R] :=
  ElementoQ[{x,y},R]
```

In[124]:=
```
{Rel1[1,2],Rel2[1,2],Rel3[1,2]}
```
Out[124]=
```
{False, True, False}
```
In[125]:=
```
Composizione[R1_?RelQ,R2_?RelQ] :=
  Apply[Insieme,
    Cases[Prodotto[R1,R2],
          {{x_,y_},{y_,z_}} :> {x,z}]]
```
In[126]:=
```
Inversa[R_?RelQ] := Map[Reverse,R]
```
```
General::spell1:
   Possible spelling error: new symbol
     name "Inversa"
     is similar to existing symbol
    "Inverse".
```
In[127]:=
```
Identità[A_Insieme] :=
  Inner[List,A,A,Insieme]
```
In[128]:=
```
Composizione[Rel1,Rel2]
```
Out[128]=
```
Insieme[{0, 1}, {1, 2}, {2, 1}, {3, 2}]
```
In[129]:=
```
Inversa[%]
```
Out[129]=
```
Insieme[{1, 0}, {1, 2}, {2, 1}, {2, 3}]
```
In[130]:=
```
Identità[X]
```
Out[130]=
```
Insieme[{0, 0}, {1, 1}, {2, 2}, {3, 3}]
```

In[131]:=

```
SimmQ[R_?RelQ] := R === Inversa[R]
```

In[132]:=

```
RiflQ[R_?RelQ] :=
  ContenutoQ[Identità[Dominio[R]],R]
```

In[133]:=

```
TranQ[R_?RelQ] :=
  ContenutoQ[Composizione[R,R],R]
```

In[134]:=

```
EquivQ[R_?RelQ] :=
  RiflQ[R] && SimmQ[R] && TranQ[R]
```

In[135]:=

```
{EquivQ[Rel1],EquivQ[Rel2],EquivQ[Rel3]}
```

Out[135]=

```
{False, False, True}
```

In[136]:=

```
MatriceRel[R_?RelQ] :=
  Outer[R,Apply[List,Dominio[R]],
          Apply[List,Codominio[R]]]
```

In[137]:=

```
MatriceRel[Rel3] // MatrixForm
```

Out[137]//MatrixForm=

True	False	True	False
False	True	False	True
True	False	True	False
False	True	False	True

In[138]:=

```
SetAttributes[TavolaRel,HoldAll];

TavolaRel[R_?RelQ] :=
  Module[{c = Apply[List,Dominio[R]],
          d = Apply[List,Codominio[R]],
          m = MatriceRel[R]},
    Join[{Join[{HoldForm[R]},c]},
         Table[Join[{d[[i]]},m[[i]]],
               {i,Length[d]}]] //
    TableForm
```

In[140]:=

```
TavolaRel[Rel3]
```

Out[140]//TableForm=

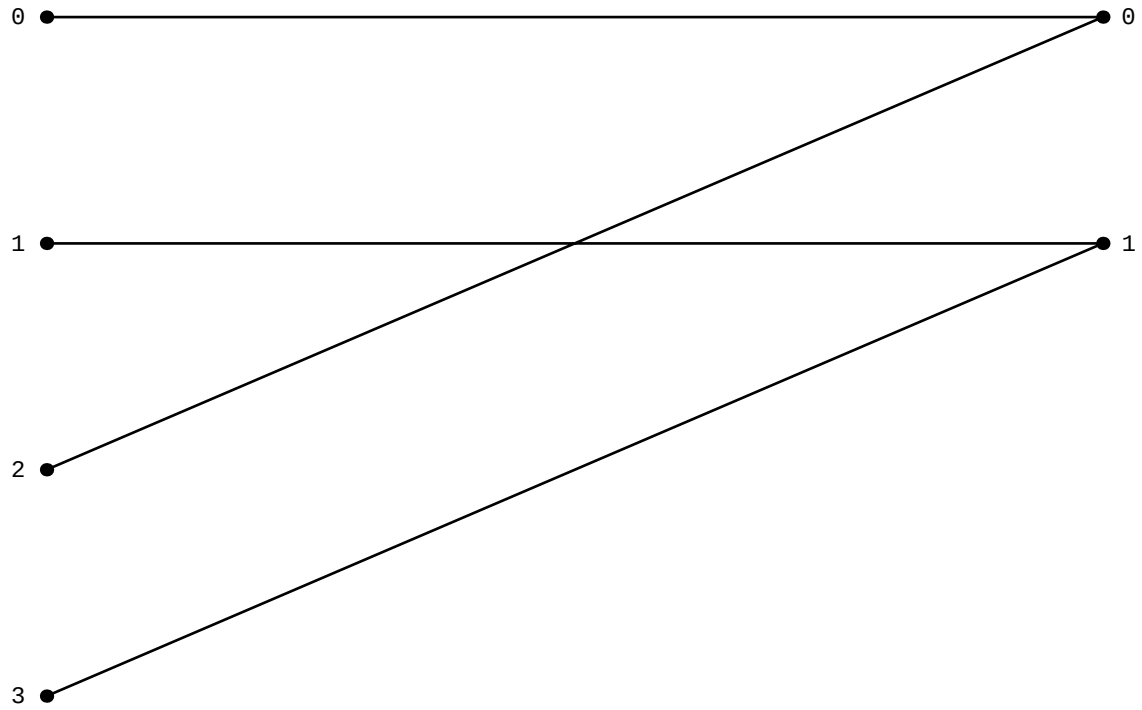
Rel3	0	1	2	3
0	True	False	True	False
1	False	True	False	True
2	True	False	True	False
3	False	True	False	True

In[141]:=

```
GrafoRel[R_?RelQ] :=
  Module[{d = Dominio[R],
          c = Codominio[R],
          m = MatriceRel[R]},
  Show[Graphics[{AbsolutePointSize[5],
                 AbsoluteThickness[1],
    Table[{Text[d[[i]],{-.02,-i},{1,0}],
          Point[{0,-i}]},{i,Length[d]}],
    Table[{Text[c[[i]],{1.02,-i},{-1,0}],
          Point[{1,-i}]},{i,Length[c]}],
    Table[If[m[[i,j]],
          Line[{{0,-i},{1,-j}}],{}],
          {i,Length[d]},{j,Length[c]}]},
    PlotRange->All]]]
```

In[142]:=

```
GrafoRel[Rel1]
```



Out[142]=

```
-Graphics-
```

Pattern matching (modelli)

```
(* da completare ... *)
```

Strutture e operatori

```
(* da completare ... *)
```