Vectors and Matrices

```
> restart; with(LinearAlgebra);
```

[&x, Add, Adjoint, BackwardSubstitute, BandMatrix, Basis, BezoutMatrix, BidiagonalForm, BilinearForm, CARE, CharacteristicMatrix, CharacteristicPolynomial, Column, ColumnDimension, ColumnOperation, ColumnSpace, CompanionMatrix, ConditionNumber, ConstantMatrix, ConstantVector, Copy, CreatePermutation, CrossProduct, DARE, DeleteColumn, DeleteRow, Determinant, Diagonal, DiagonalMatrix, Dimension, Dimensions, DotProduct, EigenConditionNumbers, Eigenvalues, Eigenvectors, Equal, ForwardSubstitute, FrobeniusForm, GaussianElimination, GenerateEquations, GenerateMatrix, Generic, GetResultDataType, GetResultShape, GivensRotationMatrix, GramSchmidt, HankelMatrix, HermiteForm, HermitianTranspose, HessenbergForm, HilbertMatrix, HouseholderMatrix, IdentityMatrix, IntersectionBasis, IsDefinite, IsOrthogonal, IsSimilar, IsUnitary, JordanBlockMatrix, JordanForm, KroneckerProduct, LA_Main, LUDecomposition, LeastSquares, LinearSolve, LyapunovSolve, Map, Map2, MatrixAdd, MatrixExponential, MatrixFunction, MatrixInverse, MatrixMatrixMultiply, MatrixNorm, MatrixPower, MatrixScalarMultiply, MatrixVectorMultiply, MinimalPolynomial, Minor, Modular, Multiply, NoUserValue, Norm, Normalize, NullSpace, OuterProductMatrix, Permanent, Pivot, PopovForm, QRDecomposition, RandomMatrix, RandomVector, Rank, RationalCanonicalForm, ReducedRowEchelonForm, Row, RowDimension, RowOperation, RowSpace, ScalarMatrix, ScalarMultiply, ScalarVector, SchurForm, SingularValues, SmithForm, StronglyConnectedBlocks, SubMatrix, SubVector, SumBasis, SylvesterMatrix, SylvesterSolve, ToeplitzMatrix, Trace, Transpose, TridiagonalForm, UnitVector, VandermondeMatrix, VectorAdd, VectorAngle, VectorMatrixMultiply, VectorNorm, VectorScalarMultiply, ZeroMatrix, ZeroVector, Zip] **(1)**

```
> f := x→x^5 + x^2 − 3·x + 2; pp := plot(f(x), x = −2..3); ppp := plot(x^3 + x^2 − 3·x + 2, x = −1..2, color = black);
```

$$f := x \to x^5 + x^2 - 3\,x + 2$$

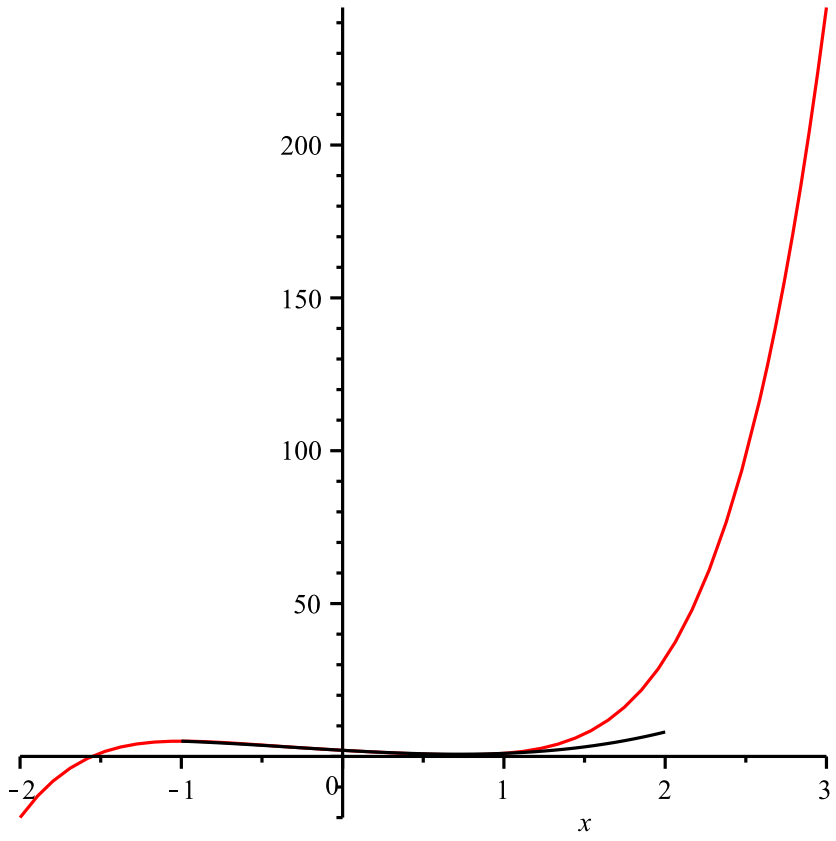
$$pp := PLOT(\ldots)$$

$$ppp := PLOT(\ldots) \quad \textbf{(2)}$$

```
> display(pp);
```

$$display(PLOT(\ldots)) \quad \textbf{(3)}$$

```
> with(plots) :
> display(pp, ppp);
```



Let us inspect (column) vectors.

```
> p := ⟨0, 1⟩; r := ⟨1, 2⟩;
```

$$p := \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$r := \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad (4)$$

```
> p[1];
```

$$0 \quad (5)$$

```
> r[2];
```

$$2 \quad (6)$$

```
> p + r;
```

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix} \quad (7)$$

> $l := p + \lambda \cdot r;$

$$l := \begin{bmatrix} \lambda \\ 1 + 2\,\lambda \end{bmatrix} \quad (8)$$

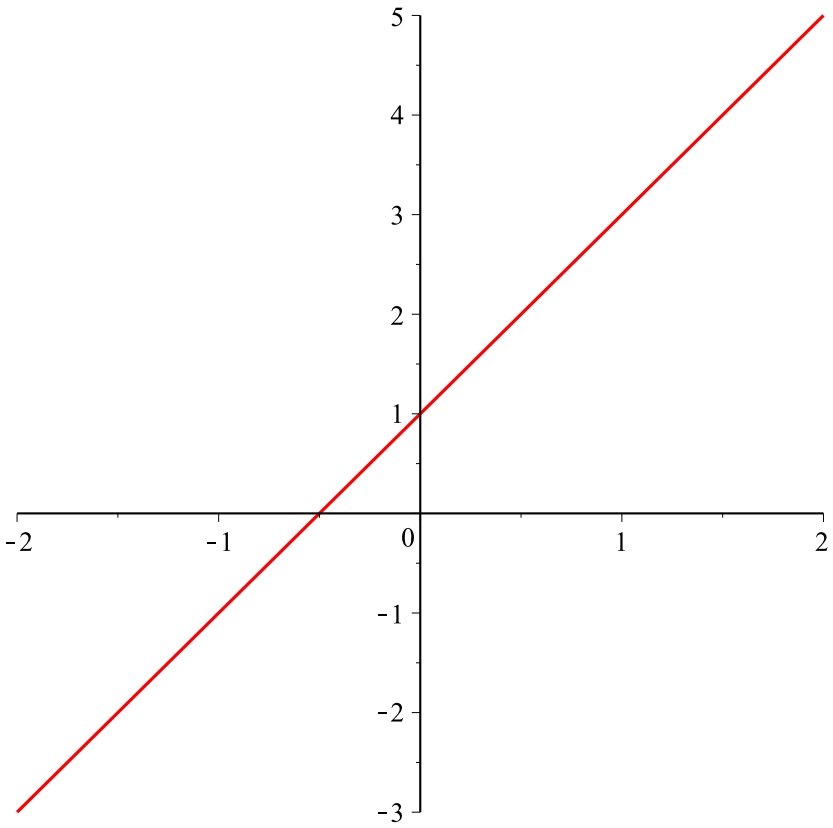
Now, we want to compute the shortest distance from point q := <2,1> to the line.

> $q := \langle 2, 1 \rangle;$

$$q := \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad (9)$$

> $lineplot := plot([l[1], l[2], \lambda = -2..2]); display(lineplot);$

$$lineplot := PLOT(\ldots)$$



> $f := \lambda \rightarrow p + \lambda \cdot r;$

$$f := \lambda \rightarrow p + \lambda\, r \quad (10)$$

> $s := seq([l[1], l[2]], \lambda = -2..2);$

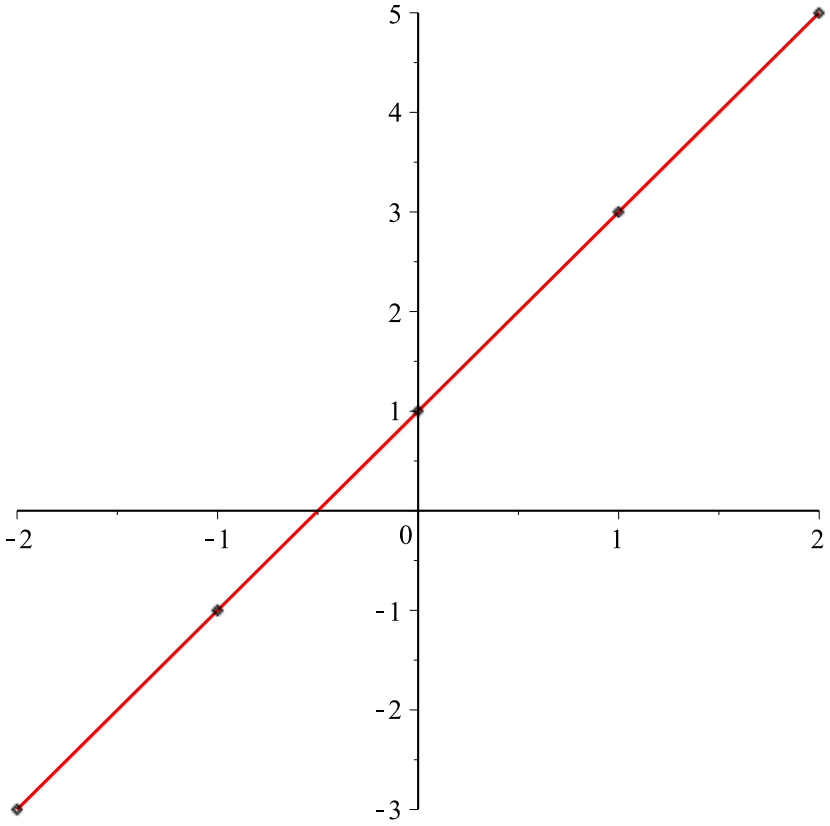
$$s := [-2, -3], [-1, -1], [0, 1], [1, 3], [2, 5] \quad (11)$$

> $t := seq([f(x)[1], f(x)[2]], x = -2..2);$

$$t := [-2, -3], [-1, -1], [0, 1], [1, 3], [2, 5] \quad (12)$$

> $pointline := pointplot([s]); display([lineplot, pointline]);$

$$pointline := PLOT(\ldots)$$



> $Qplot := pointplot(q);$

$$Qplot := PLOT(\ldots) \quad \textbf{(13)}$$

> $a1 := arrow([0, 0], p, width = [0.03, relative = true], head\_length = [0.2, relative = false], color = blue);$

$$a1 := PLOT(\ldots) \quad \textbf{(14)}$$

>

> $a2 := arrow(p, 0.25 \cdot r, width = [0.08, relative = true], head\_length = [0.2, relative = false], color = blue);$

$$a2 := PLOT(\ldots) \quad \textbf{(15)}$$

> $aHitQ := arrow\left(\left\langle \frac{2}{5}, \frac{9}{5} \right\rangle, q - \left\langle \frac{2}{5}, \frac{9}{5} \right\rangle, width = [0.0125, relative = true], head\_length = [0.1, relative = false], color = green\right);$

$$aHitQ := PLOT(\ldots) \quad \textbf{(16)}$$

> $aQ := arrow(\langle 0, 0 \rangle, q, width = [0.0125, relative = true], head\_length = [0.1, relative = false], color = green);$

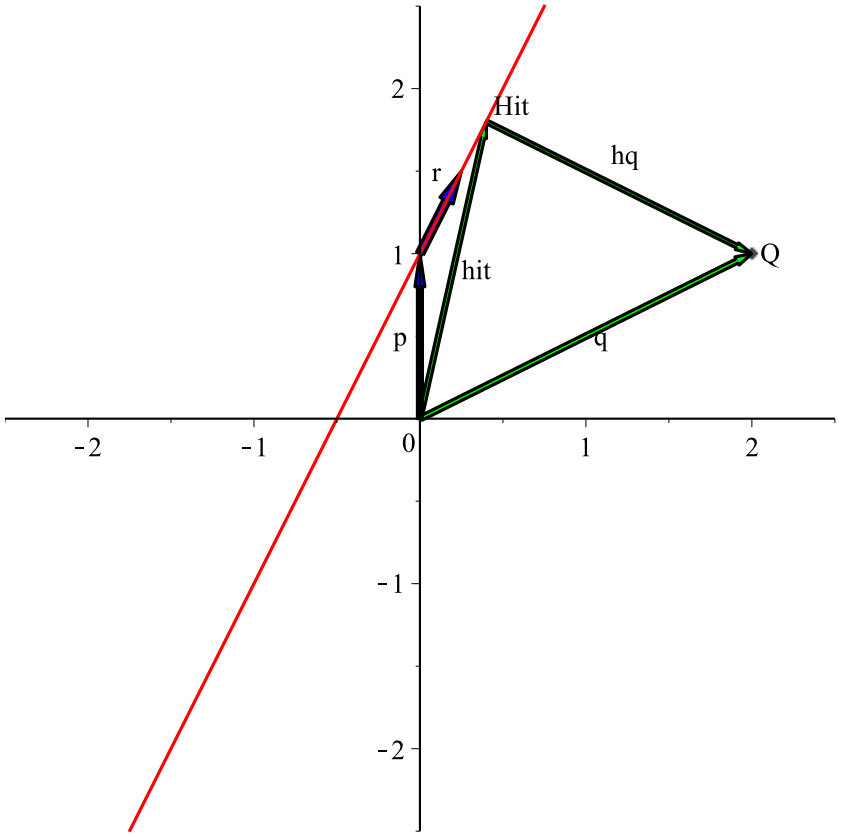
(17)

$$aQ := PLOT(\ldots) \quad \textbf{(17)}$$

> $aHit := arrow\left(\langle 0, 0\rangle, \left\langle \frac{2}{5}, \frac{9}{5}\right\rangle, width = [0.0125, relative = true], head\_length = [0.1, relative = false], color = green\right);$

$$aHit := PLOT(\ldots) \quad \textbf{(18)}$$

> $display\left(a1, a2, aHit, aQ, aHitQ, Qplot, lineplot, view = [-2.5..2.5, -2.5..2.5], textplot([q[1], q[2], "Q"], align = \{right\}), textplot([1.1, 1.5, "hq"], align = \{right, above\}), textplot\left(\left[\frac{1}{5}, \frac{4}{5}, "hit"\right], align = \{right, above\}\right), textplot([1, 0.4, "q"], align = \{right, above\}), textplot\left(\left[\frac{2}{5}, \frac{9}{5}, "Hit"\right], align = \{right, above\}\right), textplot([-.02, 0.5, "p"], align = \{left\}), textplot([0.1, 1.4, "r"], align = \{above\})\right);$



>

Remarks:

q = hit+hq, thus hq=q-hit
moreover there is a lambda such that: hit = p + λr, thus hq = q - (p+λr)
now, we demand that hq ⊥ r, thus hq·r = 0, and therefore (q - (p+r)) · r = 0

Which λ does it? And what is the Point "Hit"?

> $\lambda := solve((q - (p + x \cdot r)).r = 0, x); p + r \cdot \lambda; f\left(\frac{2}{5}\right);$

$$\lambda := \frac{2}{5}$$

$$\begin{bmatrix} \frac{2}{5} \\ \frac{9}{5} \end{bmatrix}$$

$$\begin{bmatrix} \frac{2}{5} \\ \frac{9}{5} \end{bmatrix} \quad \textbf{(19)}$$

Remarks:
resorting (q - (p+λr)) · r = 0 leads to
(q - p - λ·r) . r = 0 and
q.r-p.r = λ·r.r and thus to $\lambda = \frac{(q-p).r}{r.r}$

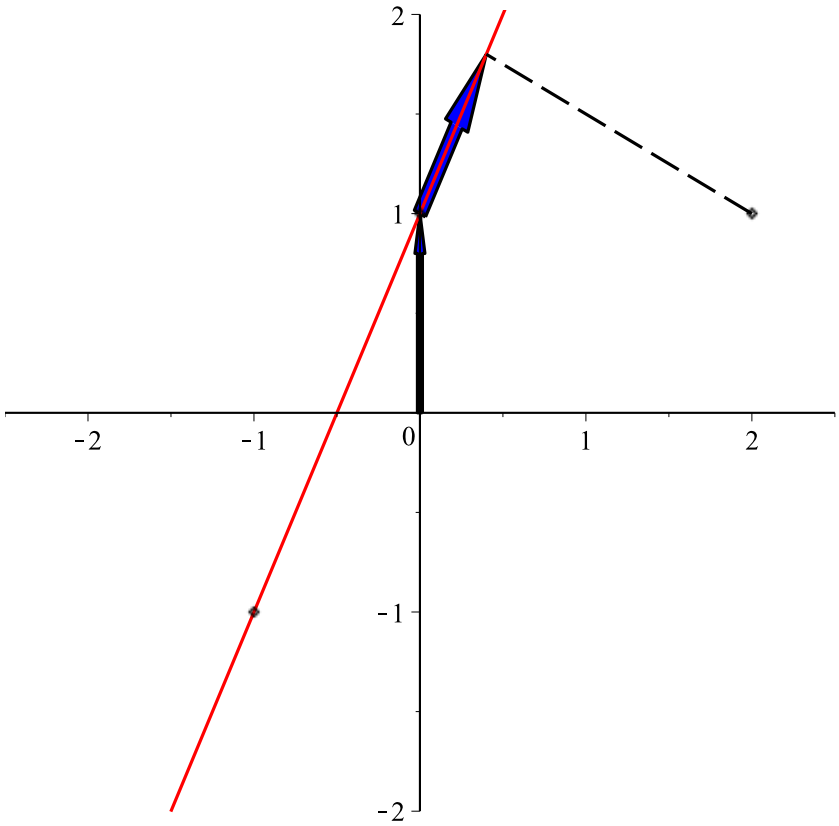
> $a22 := arrow\left(p, \frac{(q-p).r}{r.r} \cdot r, width = [0.075, relative = false], head\_length = [0.4, relative = false], color = blue\right);$

$$a22 := PLOT(\ldots) \quad \textbf{(20)}$$

> $HitPoint := subs\left(\lambda = \frac{DotProduct(q-p, r)}{DotProduct(r, r)}, l\right);$

$$HitPoint := \begin{bmatrix} \frac{2}{5} \\ \frac{9}{5} \end{bmatrix} \quad \textbf{(21)}$$

> $display([a1, a22, Qplot, lineplot, pointline, pointplot([q, HitPoint], connect = true, thickness = 1, linestyle = dash)], view = [-2.5..2.5, -2..2]);$




```
> # at the end, we could plot the big arrow from (0,1), pointing exactly into the HitPoint
>
> DotProduct(q - p, r)/DotProduct(r, r), (q - p).r/r.r;
```

$$\frac{2}{5}, \frac{2}{5} \qquad \textbf{(22)}$$

Aus Vielfalt wird Einheit

Das Internet - ein Hort der Meinungsvielfalt? Theoretisch schon. Doch in der Realität herrscht Einförmigkeit der Ansichten. Das liegt zum einen an uns selbst, zum anderen an den Filtern von Amazon, Google, Facebook & Co. Von Carsten Görig

---------- Carsten Görig -------

full text here:

http://www.stern.de/digital/online/meinungsbildung-im-internet-aus-vielfalt-wird-einheit-1664369.html

---

Zitat, wörtlich:
Neue Techniken führen dazu, dass die Menschen zueinanderfinden. Zugang zu allen Informationen führt dazu, dass wir uns umfassend informieren, Verständnis für andere Meinungen entwickeln und unsere Standpunkte ausgewogener werden - das zumindest propagieren die Verfechter sozialer Netzwerke und Suchmaschinen wie Google. Das Netz ist der große Heilsbringer. Viele Studien zeigen allerdings: Das Gegenteil ist der Fall. Wir denken immer eindimensionaler. Das Netz gibt uns unzählige Möglichkeiten, die eigene Meinung zu stärken und zu verfestigen. Gleichzeitig gibt es uns sehr viele Möglichkeiten, anderen Meinungen auszuweichen. Das ist ein dem System inliegendes Problem: Wenn man es jemandem bequem machen möchte, sucht man ihm Sachen heraus, die er kennt, an die er gewöhnt ist. Und das Ziel der neuen Dienste ist es ja, es dem Menschen im Internet so bequem wie möglich zu machen.

1971 veröffentlicht der amerikanische Wissenschaftler Thomas Schelling eine Studie, die den Titel "Models of Segregation" trägt (Modelle der Trennung), wobei er sich mit dem Begriff der Trennung auf Rassentrennung in Städten bezieht. Auf einem Spielbrett legt er dar, wie selbst eine nur geringe Vorliebe, sich mit Menschen der eigenen Hautfarbe zu umgeben, zu einer vollständigen Trennung von Wohngebieten führen kann. Ein Prozess, der in Städten Jahre und Jahrzehnte dauern, woanders aber deutlich schneller verlaufen kann.

Wir suchen Zustimmung

Dieses Modell lässt sich auf andere Formen des menschlichen Zusammenlebens übertragen, auch auf das Internet, auf unsere Surf-Gewohnheiten. Wir neigen dazu, Dinge zu suchen, die uns näher sind, die uns in unserer Meinung bestärken. Und das tun wir auch im Netz. Mit jedem Klick, mit jeder Seite, die wir uns anschauen, erziehen wir die Suchmaschinen, uns mehr von dem zu zeigen, was wir mögen, weniger von dem, was wir nicht mögen. So kommen wir immer weniger in Kontakt mit Meinungen, die wir nicht so gerne sehen oder lesen. Es ist ein individueller Prozess, bei dem sich die Suchmaschine unseren Bedürfnissen anpasst, oder vielmehr dem, was sie als unsere Bedürfnisse errechnet. Mit dem Ergebnis, dass die Maschine uns irgendwann nur noch die Seiten oben anzeigt, die sie für uns als einzelne Person für wichtig hält.

....

Doch es ist die technische Idee von der Funktion des Gehirns, die in Firmen wie Google oder Facebook vorherrscht. Und in der hat der Zufall keinen Platz. Schon bevor wir einen Begriff eingeben, möchte Google uns die Antwort auf das geben, was wir suchen. Das ist eines der fernen Ziele der Google-Gründer. Doch das funktioniert nur, wenn sie uns genau kennen, wenn sie unsere Gewohnheiten auswerten und wir innerhalb dieser handeln. Und genau deshalb steuern sie uns weiter in eine Ecke, aus der wir nur schwer wieder herauskommen, und verfestigen und bilden damit unsere Meinung.

$$A := \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix}; B := \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix};$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix}$$

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \tag{23}$$

$$B := \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix};$$

$$A := \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \qquad C := A.B$$

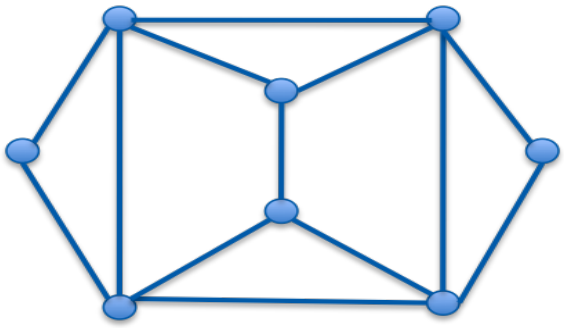
$$\begin{bmatrix} a_{11}\,b_{11} + a_{12}\,b_{21} & a_{11}\,b_{12} + a_{12}\,b_{22} \\ a_{21}\,b_{11} + a_{22}\,b_{21} & a_{21}\,b_{12} + a_{22}\,b_{22} \\ a_{31}\,b_{11} + a_{32}\,b_{21} & a_{31}\,b_{12} + a_{32}\,b_{22} \end{bmatrix} \tag{24}$$

in general: $c_{ik} := a_{i1} \cdot b_{1\,k} + \ldots + a_{\mathbf{in}} \cdot b_{nk}$
cf. G. Fischer, Lineare Algebra, S. 71ff, Verknüpfungen von Matrizen

Graphs

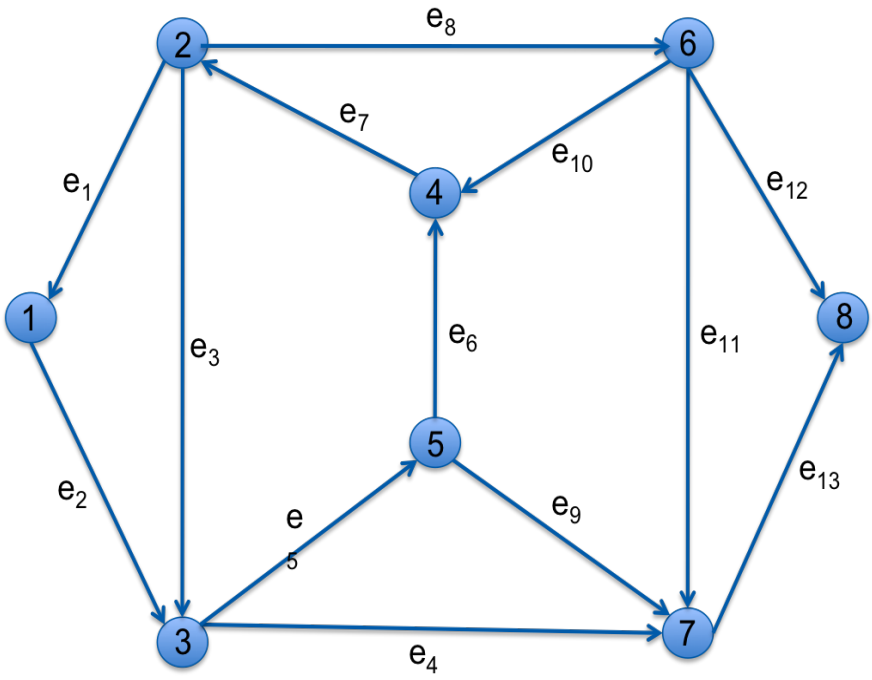
What is a graph?

An undirected graph consists of a pair G=(V,E), where E $\subseteq$ {{u,v} | u,v$\in$ V}.
The elements of E are not ordered..



Elements from V are called nodes (or vertices; Knoten in dt.), elements from E are called edges (Kanten in dt.)

A directed graph (gerichteter Graph) is as well a pair G=(V,E). However, the elements of E are ordered pairs of elements from V. Thus, $E \subseteq \{(u,v) \mid u,v \in V\}$.
Elemente of V are called nodes, elements from E are called edges (im dt.: gerichtete Kanten oder Bögen)
.



Repetition: Flow Control (if, for, while, ...)

if <conditional expression> **then** <statement sequence>
| **elif** <conditional expression> **then** <statement sequence> |
| **else** <statement sequence> |
end if
(Note: Phrases located between | | are optional.)

```
> a := 0;
```

$$a := 0 \quad (25)$$

```
> if (a > 0) then f := x^2 fi;
> if (a = 0) then f := x^2 fi;
```

$$f := x^2 \quad \textbf{(26)}$$

```
> if (a < 9) then
    f := x^2 + 1; # ";" is necessary, because: several statements without structure
    g := x^2      # ";" not necessary
  else
    g := x^2 + 1;
    f := x^2;
  end if;
```

$$f := x^2 + 1$$
$$g := x^2 \quad \textbf{(27)}$$

The **for ...while ... do** loop

```
>
>
```

1) Print even numbers from 6 to 10.

```
> for i from 6 by 2 to 10 do print(i) end do;
```

$$6$$
$$8$$
$$10 \quad \textbf{(28)}$$

2) Find the sum of all two-digit odd numbers from 11 to 99.

```
> mysum := 0;
  for i from 11 by 2 while i < 100 do
    mysum := mysum + i
  end do:
  mysum;
```

$$mysum := 0$$
$$2475 \quad \textbf{(29)}$$

3) Multiply the entries of an expression sequence.

```
> restart;
  total := 1 :
  for z in 1, x, y, q^2, 3 do
    total := total·z
  end do:
  total;
  x := 2 :
  q := 3 :
  total;
```

$$3\,x\,y\,q^2$$
$$54\,y \quad \textbf{(30)}$$

3) Add together the contents of a list.

```
> ?cat
> restart;
```

```
y := 3;
 myconstruction := "";
for z in [1, "+", y, "·", "q^2", "·", 3] do
   myconstruction := cat(myconstruction, z) ;
 end do;
myconstruction;
```

$$y := 3$$
$$myconstruction := ""$$
$$myconstruction := "1"$$
$$myconstruction := "1+"$$
$$myconstruction := "1+3"$$
$$myconstruction := "1+3*"$$
$$myconstruction := "1+3*q^2"$$
$$myconstruction := "1+3*q^2*"$$
$$myconstruction := "1+3*q^2*3"$$
$$"1+3*q^2*3" \quad (31)$$

```
> ?parse
```

```
> q := 4;
```

$$q := 4 \quad (32)$$

```
> qq := parse(myconstruction);
```

$$qq := 1 + 9\,q^2 \quad (33)$$

```
> qq;
```

$$145 \quad (34)$$

Similar-to-Fibonacci-Numbers are
$sff[1] := 0;\ sff[2] := 1; sff[3] := 1; sff[i] := sff[i-1] + sff[i-2] + 1$

```
> restart; sff := [seq(0, i = 1..100)]:
>
> sff[2] := 1; sff[3] := 1;
```

$$sff_2 := 1$$
$$sff_3 := 1 \quad (35)$$

```
> sff[4] := sff[2] + sff[3] + 1;
```

$$sff_4 := 3 \quad (36)$$

```
> for i from 4 to 100 do
    sff[i] := sff[i-1] + sff[i-2] + 1;
    if i = 97 then print(sff[i]) fi;
  end do:
```

$$10336141770971664 6143 \quad (37)$$

```
>  sff1 := sff[1]; sff2 := sff[2]; sff3 := sff[3];
```

$$sff1 := 0$$

$sff2 := 1$

$sff3 := 1$ **(38)**

```
> for i from 4 to 10000 do
    sff4 := sff3 + sff2 + 1;
    sff2 := sff3 : sff3 := sff4;
    if i = 97 then print(sff3) fi;
  end do:
```