In [1]:
```
j=2
```

In [2]:
```
j**2
```

Out[2]: 4

In [3]:
```
1j**2
```

Out[3]: (-1+0j)

In [4]:
```
-2+3*j
```

Out[4]: 4

In [5]:
```
-2+3j
```

Out[5]: (-2+3j)

In [6]:
```
from numpy import *
```

In [7]:
```
sqrt(8)
```

Out[7]: 2.8284271247461903

In [8]:
```
cos(pi/5)
```

Out[8]: 0.8090169943749475

In [9]:
```
from sympy import *
```

In [10]:
```
sqrt(8)
```

Out[10]: $2\sqrt{2}$

In [11]:
```
cos(pi/5)
```

Out[11]: $\frac{1}{4} + \frac{\sqrt{5}}{4}$

In [12]:
```
cos(pi/7)
```

Out[12]: $\cos\left(\frac{\pi}{7}\right)$

In [13]:
```
I**2
```

Out[13]: -1

In [14]:
```
-2+3*I
```

Out[14]: $-2 + 3i$

In [15]:
```
-2+3I
```

```
  File "<ipython-input-15-263665c59c98>", line 1
    -2+3I
        ^
SyntaxError: invalid syntax
```

In [16]:
```
v = Matrix([1,2,3])
v
```
Out[16]:
$$\begin{bmatrix}1\\2\\3\end{bmatrix}$$

In [17]:
```
transpose(v)
```
Out[17]: $\begin{bmatrix}1 & 2 & 3\end{bmatrix}$

In [18]:
```
v.transpose()
```
Out[18]: $\begin{bmatrix}1 & 2 & 3\end{bmatrix}$

In [19]:
```
v.transpose
```
Out[19]:
```
<bound method MatrixOperations.transpose of Matrix([
[1],
[2],
[3]])>
```

In [20]:
```
w = Matrix([0,1,0])
```

In [21]:
```
v.dot(w)
# Skalarprodukt
```
Out[21]: 2

In [22]:
```
v.cross(w)
# Kryssprodukt
```
Out[22]:
$$\begin{bmatrix}-3\\0\\1\end{bmatrix}$$

In [23]:
```
A = Matrix([[1,2,3],[4,5,6],[7,8,9]])
A
```
Out[23]:
$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$$

In [24]:
```
A[1,:]
```
Out[24]: $\begin{bmatrix}4 & 5 & 6\end{bmatrix}$

In [25]:
```
A[1,2]
```
Out[25]: 6

In [26]:
```
A*v
```
Out[26]:
$$\begin{bmatrix}14\\32\\50\end{bmatrix}$$

In [27]:
```
v*A
```

```
---------------------------------------------------------------------------
ShapeError                                Traceback (most recent call last)
<ipython-input-27-cb1e8bf78ac1> in <module>
----> 1 v*A

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/core/decorators.py
in binary_op_wrapper(self, other)
    127                     if f is not None:
    128                         return f(self)
--> 129             return func(self, other)
    130         return binary_op_wrapper
    131     return priority_decorator

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/matrices/common.py
in __mul__(self, other)
   2103             if self.shape[1] != other.shape[0]:
   2104                 raise ShapeError("Matrix size mismatch: %s * %s." % (
-> 2105                     self.shape, other.shape))
   2106
   2107         # honest sympy matrices defer to their class's routine

ShapeError: Matrix size mismatch: (3, 1) * (3, 3).
```

In [28]:
```
A*A
```

Out[28]:

$$\begin{bmatrix} 30 & 36 & 42 \\ 66 & 81 & 96 \\ 102 & 126 & 150 \end{bmatrix}$$

In [29]:
```
eye(3)
```

Out[29]:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In [30]:
```
B = A + eye(3)
```

In [31]:
```
B
```

Out[31]:

$$\begin{bmatrix} 2 & 2 & 3 \\ 4 & 6 & 6 \\ 7 & 8 & 10 \end{bmatrix}$$

In [32]:
```
B**-1
```

Out[32]:

$$\begin{bmatrix} -6 & -2 & 3 \\ -1 & \frac{1}{2} & 0 \\ 5 & 1 & -2 \end{bmatrix}$$

In [33]:
```
det(B)
```

Out[33]: -2

In [34]:
```
B.det()
```

Out[34]: -2

In [35]:
```
A.rank()
```

Out[35]: 2

In [36]:
```
A**-1
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-36-1bff7ad18ede> in <module>
----> 1 A**-1

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/core/decorators.py
in binary_op_wrapper(self, other)
    127                     if f is not None:
    128                         return f(self)
--> 129             return func(self, other)
    130         return binary_op_wrapper
    131     return priority_decorator

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/matrices/common.py
in __pow__(self, num)
   2138             if num < 0:
   2139                 num = -num
-> 2140                 a = a.inv()
   2141             # When certain conditions are met,
   2142             # Jordan block algorithm is faster than

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/matrices/matrices.
py in inv(self, method, **kwargs)
   3192         if method is not None:
   3193             kwargs['method'] = method
-> 3194         return self._eval_inverse(**kwargs)
   3195 
   3196     def is_nilpotent(self):

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/matrices/dense.py
in _eval_inverse(self, **kwargs)
    267         M = self.as_mutable()
    268         if method == "GE":
--> 269             rv = M.inverse_GE(iszerofunc=iszerofunc)
    270         elif method == "LU":
    271             rv = M.inverse_LU(iszerofunc=iszerofunc)

/srv/conda/envs/notebook/lib/python3.6/site-packages/sympy/matrices/matrices.
py in inverse_GE(self, iszerofunc)
   3107         red = big.rref(iszerofunc=iszerofunc, simplify=True)[0]
   3108         if any(iszerofunc(red[j, j]) for j in range(red.rows)):
-> 3109             raise ValueError("Matrix det == 0; not invertible.")
   3110 
   3111         return self._new(red[:, big.rows:])

ValueError: Matrix det == 0; not invertible.
```

In [37]:
```
A.eigenvals()
```

Out[37]: {15/2 - 3*sqrt(33)/2: 1, 15/2 + 3*sqrt(33)/2: 1, 0: 1}

In [38]:
```
A.eigenvects()
```

Out[38]:
```
[(0, 1, [Matrix([
   [ 1],
   [-2],
   [ 1]])]), (15/2 - 3*sqrt(33)/2, 1, [Matrix([
   [-(-18*sqrt(33) + 3*(-13/2 + 3*sqrt(33)/2)*(-5/2 + 3*sqrt(33)/2) + 78)/((-
8 + (-13/2 + 3*sqrt(33)/2)*(-5/2 + 3*sqrt(33)/2))*(-13/2 + 3*sqrt(33)/2))],
   [
-(-51 + 9*sqrt(33))/(-8 + (-13/2 + 3*sqrt(33)/2)*(-5/2 + 3*sqrt(33)/2))],
   [
1]])]), (15/2 + 3*sqrt(33)/2,
  1,
  [Matrix([
   [-(78 + 18*sqrt(33) + 3*(-3*sqrt(33)/2 - 13/2)*(-3*sqrt(33)/2 - 5/2))/((-8
+ (-3*sqrt(33)/2 - 13/2)*(-3*sqrt(33)/2 - 5/2))*(-3*sqrt(33)/2 - 13/2))],
   [
-(-9*sqrt(33) - 51)/(-8 + (-3*sqrt(33)/2 - 13/2)*(-3*sqrt(33)/2 - 5/2))],
   [
1]])])]
```

In [39]:
```
V = A.eigenvects()
```

In [40]:
```
V[0]
```

Out[40]:
```
(0, 1, [Matrix([
  [ 1],
  [-2],
  [ 1]])])
```

In [41]:
```
V[0][2]
```

Out[41]:
```
[Matrix([
 [ 1],
 [-2],
 [ 1]])]
```

In [42]:
```
V[0][2][0]
```

Out[42]:

$$\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

In [43]:
```
A.nullspace()
```

Out[43]:
```
[Matrix([
 [ 1],
 [-2],
 [ 1]])]
```

In [45]:
```
A.columnspace()
```

Out[45]:
```
[Matrix([
 [1],
 [4],
 [7]]), Matrix([
 [2],
 [5],
 [8]])]
```

In [46]:
```
p = A.charpoly()
p
```

Out[46]:

$$\operatorname{PurePoly}\left(\lambda^{3} - 15\lambda^{2} - 18\lambda, \lambda, domain = \mathbb{Z}\right)$$

In [47]:
```
roots(p)
```

Out[47]: {15/2 - 3*sqrt(33)/2: 1, 15/2 + 3*sqrt(33)/2: 1, 0: 1}

In [48]:
```
var = Symbol('t')
var
```

Out[48]: t

In [50]:
```
a,b,c,x,y,z=symbols("a,b,c,x,y,z")
```

In [51]:
```
roots(x**2+2)
```

Out[51]: {-sqrt(2)*I: 1, sqrt(2)*I: 1}

In [52]:
```
factor(x**6-1)
```

Out[52]: $(x-1)(x+1)\left(x^2-x+1\right)\left(x^2+x+1\right)$

In [53]:
```
det(A-x*eye(3))
```

Out[53]: $77x+(1-x)(5-x)(9-x)-45$

In [54]:
```
expand(det(A-x*eye(3)))
```

Out[54]: $-x^3+15x^2+18x$

In [55]:
```
# Vi vil loese likningssystemet
# 2x + y - 3z = 4
# -x + 2y     = 1
# x  + 3y - 3z = 5

A = Matrix([[2,1,-3,4],[-1,2,0,1],[1,3,-3,5]])
A
```

Out[55]:

$$\begin{bmatrix} 2 & 1 & -3 & 4 \\ -1 & 2 & 0 & 1 \\ 1 & 3 & -3 & 5 \end{bmatrix}$$

In [56]:
```
A.rref()
```

Out[56]:
```
(Matrix([
 [1, 0, -6/5, 7/5],
 [0, 1, -3/5, 6/5],
 [0, 0,    0,   0]]), (0, 1))
```

In [57]:
```
# Likningssystemet ovenfor er ekvivalent med
# x - 6/5 z = 7/5
# y - 3/5 z = 6/5
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
In [ ]:
```