```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
#
#Created on Mon Nov 20 12:22:52 2017
#
#@author: christianehelzel
#

import numpy as np
```

Zeilentausch

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])


# A[[i,j]]= A[[j,i]] vertauscht (i+1)-te mit (j+1)-ter Zeile

A[[1,2]] = A[[2,1]]

# Beachte den Unterschied zu A[1,2]=A[2,1]
```

multipliziere (2)-te Zeile mit k

```
k=2
A[1]*=k
```

x =*y ist eine Kurzschreibweise für x=x* y. Dies gibt es auch für +, -, /

Addiere das k-Fache der 2. Zeile zur 3. Zeile

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])

k=2
A[2]+=k*A[1]
```

0.1 Gaußelimnination Version 1

```
def Gaussian_elimination1(A):
    m, n = np.shape(A)

    for j in range(n - 1):
        for i in range(j + 1, m):
            A[i,j:] = A[i,j:] - (A[i,j]/A[j,j])*A[j,j:]
    return A
```

Beispiel 1:

```
A = np.array([[1,2],[2,1]])
Gaussian_elimination1(A)
```

```
array([[ 1,  2],
       [ 0, -3]])
```

Beispiel 2:

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])
Gaussian_elimination1(A)
```

```
/home/troll/miniconda3/bin/pweave:6: RuntimeWarning: divide by zero
encountered in long_scalars
  sys.exit(pweave.scripts.weave())
/home/troll/miniconda3/bin/pweave:6: RuntimeWarning: invalid value
encountered in multiply
  sys.exit(pweave.scripts.weave())
```

```
array([[                  10,                   -7,
0],
       [                   0,                    0,
6],
       [                   0, -9223372036854775808,
-9223372036854775808]])
```

0.2 Gaußelimnination Version 2

```
def Gaussian_elimination2(A):
    m, n = np.shape(A)

    for j in range(n - 1):
        for i in range(j + 1, m):
            if A[j,j]!=0.:
                A[i,j:] = A[i,j:] - (A[i,j]/A[j,j])*A[j,j:]
            else:
                print('Division durch Null')
                break
    return A
```

Beispiel:

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])
Gaussian_elimination2(A)
```

```
Division durch Null
```

```
array([[10, -7,  0],
       [ 0,  0,  6],
       [ 0,  2,  5]])
```

0.3 Gaußelimnination Version 3

```
def Gaussian_elimination3(A):
    m, n = np.shape(A)
```

```
    for j in range(n - 1):
        for i in range(j + 1, m):
            try:
                A[i,j:] = A[i,j:] - (A[i,j]/A[j,j])*A[j,j:]

            except ZeroDivisionError:
                print('Division durch Null')
                A = np.zeros([m,n])
    return A
```

Beispiel:

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])
Gaussian_elimination3(A)
```

```
/home/troll/miniconda3/bin/pweave:7: RuntimeWarning: divide by zero
encountered in long_scalars
/home/troll/miniconda3/bin/pweave:7: RuntimeWarning: invalid value
encountered in multiply
```

```
array([[                  10,                   -7,
0],
       [                   0,                    0,
6],
       [                   0, -9223372036854775808,
-9223372036854775808]])
```

Achtung: Es gibt keinen ZeroDivisionError sondern nur eine Warnung, da A[i,j]/A[j,j] eine Rechenoperation mit NumPy-Typen ist. NumPy hat eine eigene Fehlerverwaltung.

0.4 Gaußelimnination Version 4

```
def Gaussian_elimination4(A):
    m, n = np.shape(A)
    i = 0
    for j in range(n):
        p = np.argmax(abs(A[i:m,j]))
        if p > 0: # Zeilentausch
            A[[i,p+i]] = A[[p+i, i]]
        if A[i,j] != 0:
            for r in range(i+1, m):
                A[r,j:] =  A[r,j:] - (A[r,j]/A[i,j])*A[i,j:]
            i += 1
        if i >= m:
            break
    return A
```

Beispiel:

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])
Gaussian_elimination4(A)
```

```
array([[10, -7,  0],
       [ 0,  2,  5],
       [ 0,  0,  6]])
```

0.5 Nun berechnen wir auch den Rang der Matrix

```
def Gaussian_elimination5(A):
    rank = 0
    m, n = np.shape(A)
    i = 0
    for j in range(n):
        p = np.argmax(abs(A[i:m,j]))
        if p > 0: # Zeilentausch
            A[[i,p+i]] = A[[p+i, i]]
        if A[i,j] != 0:
            rank += 1
            for r in range(i+1, m):
                A[r,j:] =  A[r,j:] - (A[r,j]/A[i,j])*A[i,j:]
            i += 1
        if i >= m:
            break
    return A, rank
```

Beispiel:

```
A = np.array([[10,-7,0],[-3,2,6],[5,-1,5]])
Gaussian_elimination5(A)
```

```
(array([[10, -7,  0],
       [ 0,  2,  5],
       [ 0,  0,  6]]), 3)
```