CP.2.2.1.a, Sauer3

Use code fragments for Gaussian elimination in the previous section to write a python script to take a matrix A as input and output L and U. No row exchanges are allowed—the program should be designed to shut down if it encounters a zero pivot. Check your program by factoring the matrices in EX.2.2.2.a.

(a) $\begin{pmatrix} 3 & 1 & 2 \\ 6 & 3 & 4 \\ 3 & 1 & 5 \end{pmatrix}$

CP.2.2.1.a, Sauer3, solution, Langou

See what was already done for **EX.2.2.2.a, Sauer3, solution, Langou**

Colab link: https://colab.research.google.com/drive/1Av8Vn_gwXXKtLulKxGETXHEuK_ER_ldL

```
import numpy as np
import copy
```

```
A = np.array([  [  3.,  1.,  2. ],
                [  6.,  3.,  4. ],
                [  3.,  1.,  5. ] ] )
```

```
# using our bucket algorithm `lu_no_pivoting`

L, U = lu_no_pivoting( A )

print("L=\n",L)
print("U=\n",U)

# check that A = LU
print("\nL @ U=\n", L @ U)
print("A=\n",A)
print("\n|| A - LU ||_oo / || A ||_oo = ",\
 np.linalg.norm(A - L@U,np.infty)/np.linalg.norm(A,np.infty) )
```

```
L=
 [[1. 0. 0.]
 [2. 1. 0.]
 [1. 0. 1.]]
U=
 [[3. 1. 2.]
 [0. 1. 0.]
 [0. 0. 3.]]
```

```
L @ U=
 [[3. 1. 2.]
 [6. 3. 4.]
 [3. 1. 5.]]
A=
 [[3. 1. 2.]
 [6. 3. 4.]
 [3. 1. 5.]]

|| A - LU ||_oo / || A ||_oo =  0.0
```