1. Реализация LU-разложения ¶

In [100]:

```
import numpy as np
from scipy import linalg as LA
import math
```

In [85]:

```
# вспомогательные элементы:

# единичная матрица
def makeI(n):
    I = []
    for i in range(n):
        k = np.zeros(n)
        k[i] = 1
        I.append(k)
    return np.array(I)
```

In [86]:

```
# функция нахождения LU-разложения матрицы
def findLU(A):
    U = np.copy(A)
    n = len(A)
    # единичная матрица
    L = makeI(n)

    for i in range(0, n-1):
        for j in range(i+1, n):
            L[j, i] = U[j, i] / U[i, i]
            U[j, i:n] = U[j, i:n] - L[j, i] * U[i, i:n]

    L = np.array(L)
    U = np.array(U)
    return L, U
```

Проверка:

In [87]:

```
# матрица 2*2
A = [[1, 7], [2, 5]]
```

In [88]:

```
#Наш алгоритм:
l, u = findLU(A)
print("L: ")
print(l)
print("--------------------")
print("U: ")
print(u)
```

```
L:
[[1. 0.]
 [2. 1.]]
--------------------
U:
[[ 1  7]
 [ 0 -9]]
```

In [93]:

```
# должен получиться ноль (нулевая матрица):

N = A - l @ u
print(N)
```

```
[[0. 0.]
 [0. 0.]]
```

Реализация разложения Холецкого

In [263]:

```
def findCholesky(A):
    A = np.array(A)
    n = len(A)
    C = np.array(np.zeros((n, n)))

    for i in range(n):
        for k in range(0, i+1):
            sum_prev = sum(C[i, 0:i] ** 2)
            sum_prev_2 = sum(C[i, 0:k] * C[k, 0:k])

            C[i, i] = math.sqrt(A[i, i] - sum_prev)
            assert(C[i, i] > 0)

            if (i != k):
                elem = (A[i, k] - sum_prev_2)
                C[i, k] = (elem / C[k, k])
    return C
```

Проверка:

In [266]:

```
# матрица 3*3
A = [[6, 3, 4], [3, 6, 5], [4, 5, 10]]
```

In [269]:

```
c = findCholesky(A)
print(c)
```

```
[[2.44948974 0.         0.        ]
 [1.22474487 2.12132034 0.        ]
 [1.63299316 1.41421356 2.30940108]]
```

In [274]:

```
# должен получиться ноль (нулевая матрица):

N = A - c @ np.transpose(c)
Nuls = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]

np.allclose(N, Nuls)
```

Out[274]:

```
True
```