LAB8: Zagadnienie przydziału – algorytm (metoda) węgierska

Zadanie - realizacja części algorytmu węgierskiego - redukacja macierzy i szkielet metody

In [ ]:
```
import numpy as np
from copy import deepcopy


SIZE = 0

def reduce_rows(matrix):
    fi = 0
    SIZE = len(matrix)
    for row in range(SIZE):
        min_value = min(matrix[row])
        fi += min_value
        for col in range(SIZE):
            matrix[row][col] -= min_value
    return fi

def total_reduction(matrix):
    fi = 0
    #redukcja wierszy i dodanie do ograniczenia dolnego
    fi += reduce_rows(matrix)
    #redukcja kolumn i dodanie do ogranicznenia dolnego, w tym celu transpozycja ma
    matrix = matrix.T
    fi += reduce_rows(matrix)
    #ponowny powrót
    matrix = matrix.T
    return fi

def indep_zeros_search(start, matrix):
    pass

def line_out(indep_zeros, dep_zeros, matrix):
    pass

def additional_zeros(rows, cols, matrix):
    pass

def hunarian_algorithm(matrix):
    REDUCTION = 0
    SIZE = len(matrix)
    #1
    REDUCTION += total_reduction(matrix)
    print(matrix)
    #2
    indep_zeros, dep_zeros = indep_zeros_search(0, matrix)
    while len(indep_zeros) != SIZE:
        #3
```

```
        min_rows, min_cols = line_out(indep_zeros, dep_zeros, matrix)
        #4
        REDUCTION += additional_zeros(min_rows, min_cols, matrix)
        indep_zeros, dep_zeros = indep_zeros_search(len(indep_zeros), matrix)
        break
    #return Xij and value of func
    Xij = np.zeros((SIZE, SIZE))
    for pair in indep_zeros:
        Xij[pair[0]][pair[1]] = 1
    return Xij, REDUCTION









if __name__ == '__main__':
    matrix = np.array([
        [6, 1, 5, 9, 2, 8],
        [7, 4, 9, 6, 9, 8],
        [9, 4, 9, 7, 1, 8],
        [0, 6, 5, 9, 7, 3],
        [4, 4, 4, 0, 1, 5],
        [7, 9, 3, 5, 8, 8]
    ])
    cp_matrix = deepcopy(matrix)


    print(hunarian_algorithm(matrix))
```