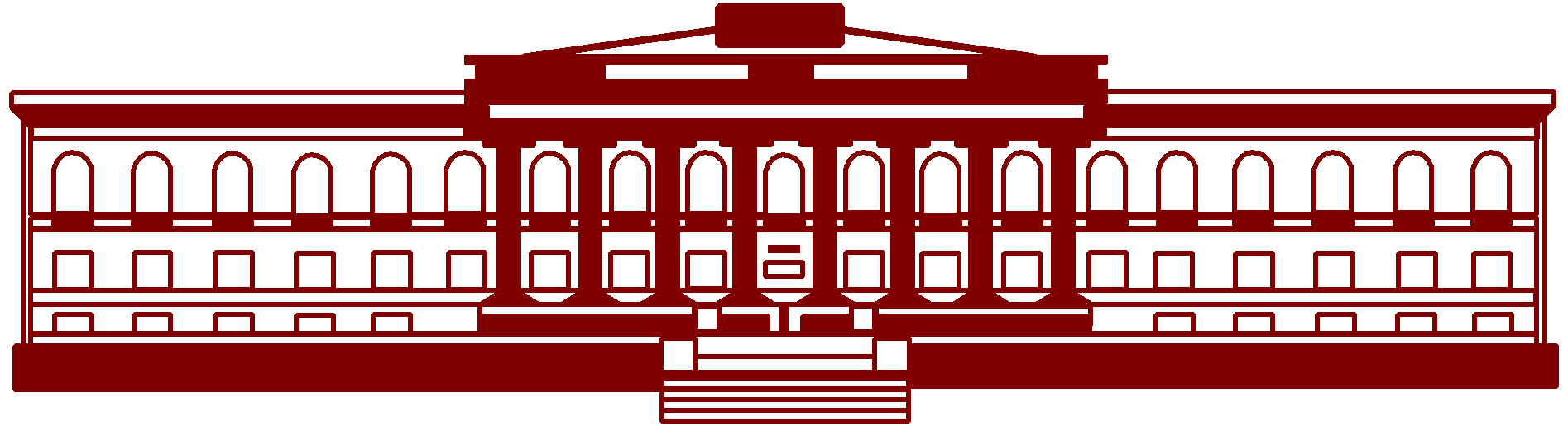
**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до лабораторної роботи №8**

**з курсу**

**«Системний аналіз та теорія прийняття рішень»**

*Студентa 3 курсу*

*групи ПП-31*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

Момотюка Михайла Тарасовича

*Викладач:*

Плескач В.Л.

Білий Р.О.

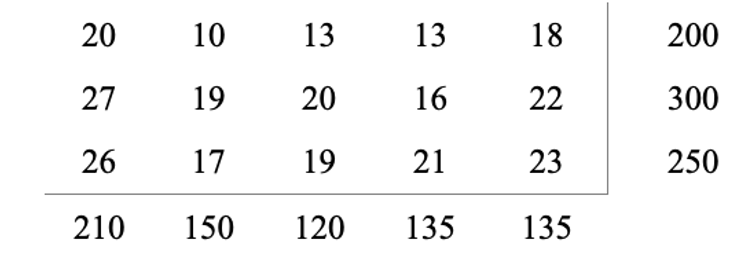
**Київ – 2023**

**Тема:** Транспортна задача. Метод північно-західного кута. Метод потенціалів.

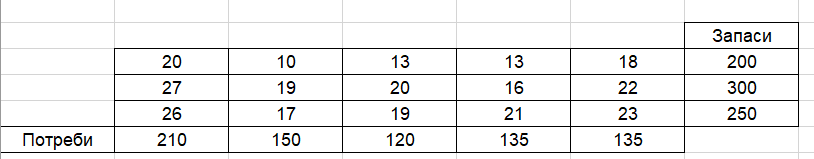
**Мета:** Навчитися розв’язувати транспортну задачу використовуючи метод північно-західного кута та метод потенціалів.

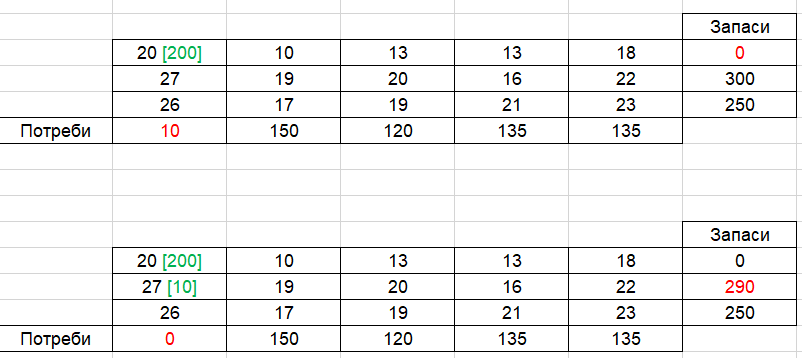
**Завдання**.

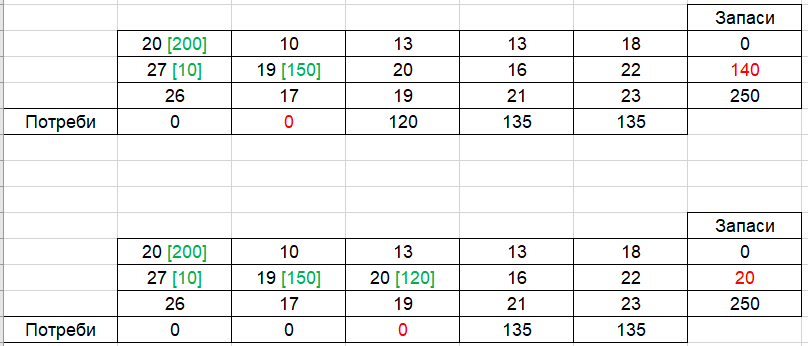
Автоматизувати процес розв’язку задачі методом північно-західного кута та методом потенціалів.

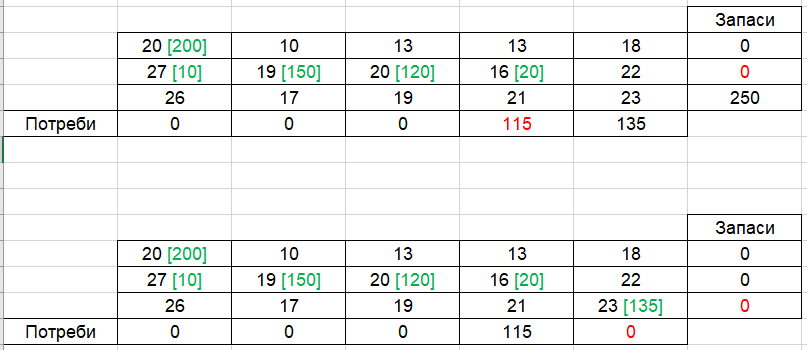


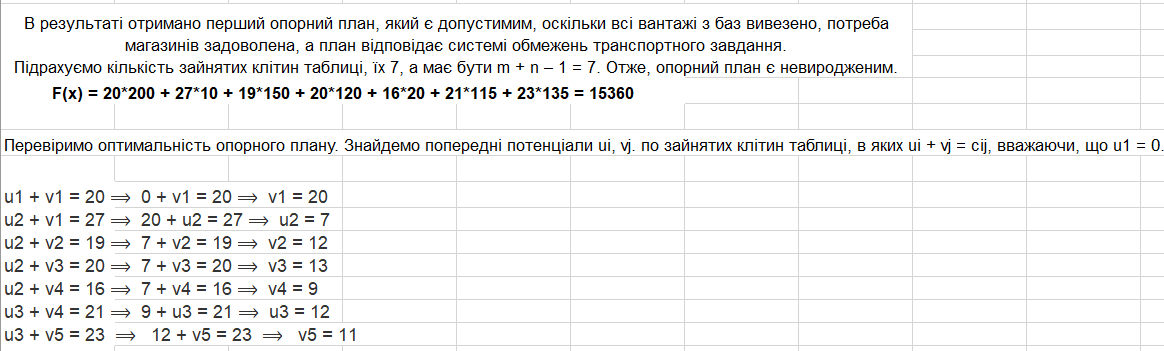
**Хід роботи**

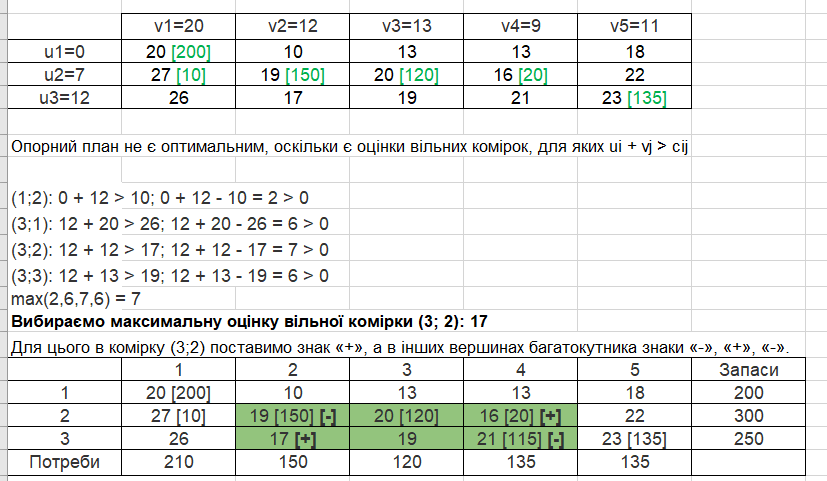
****

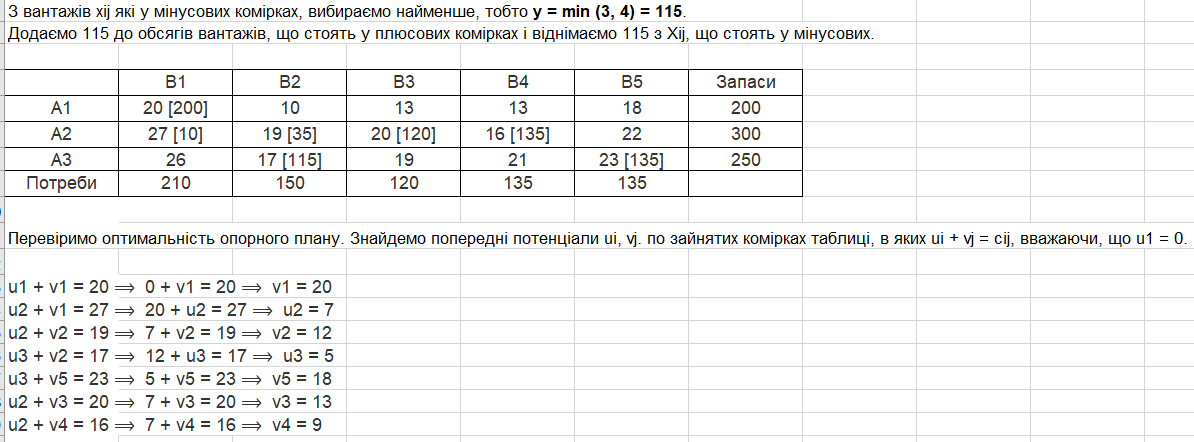
****

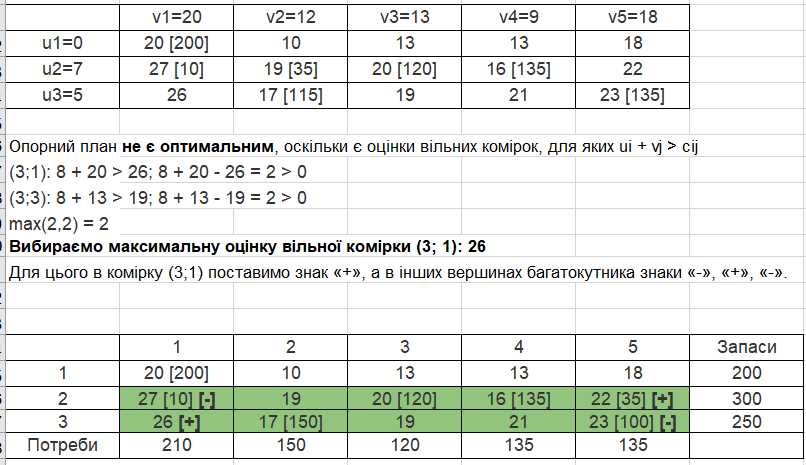
****

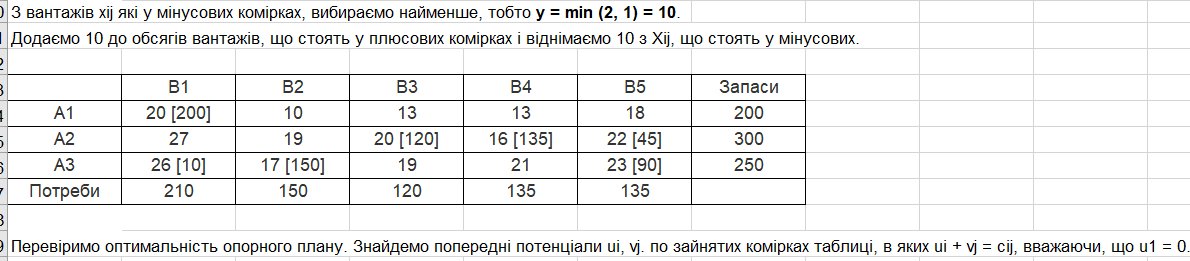
****

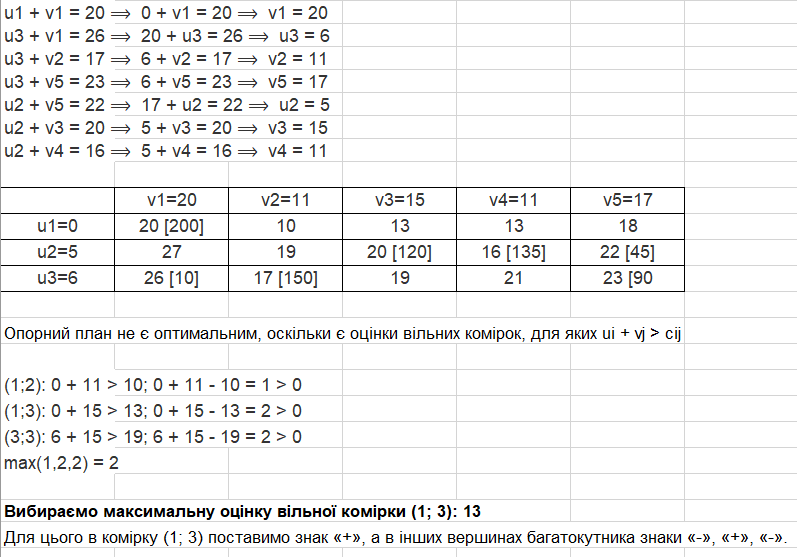
****

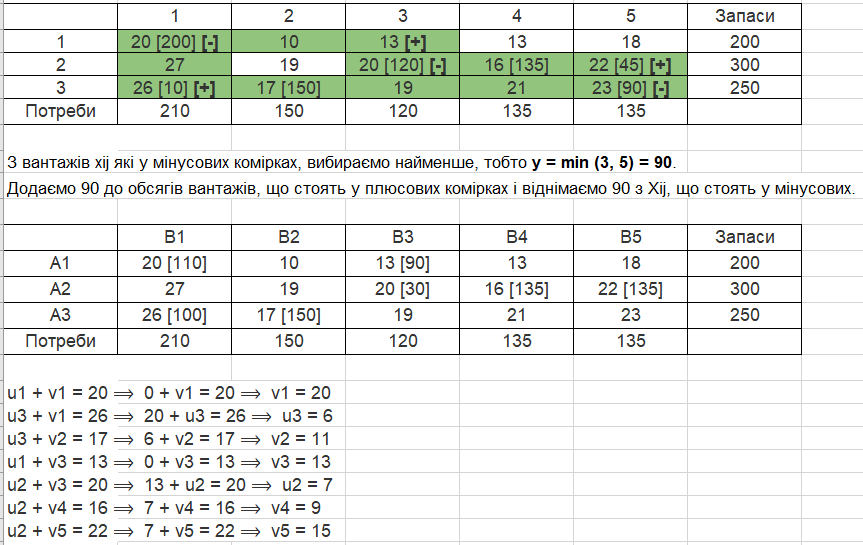
****

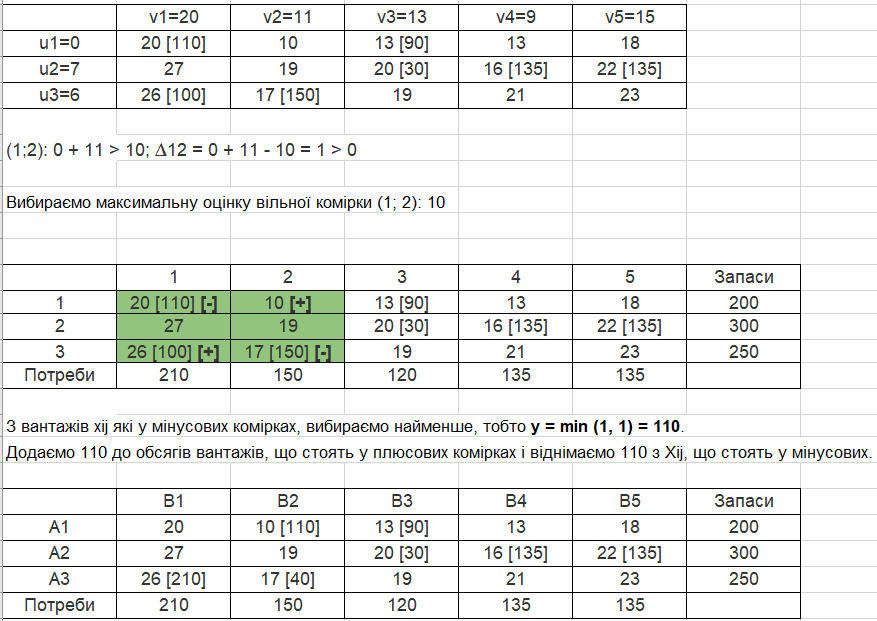
****

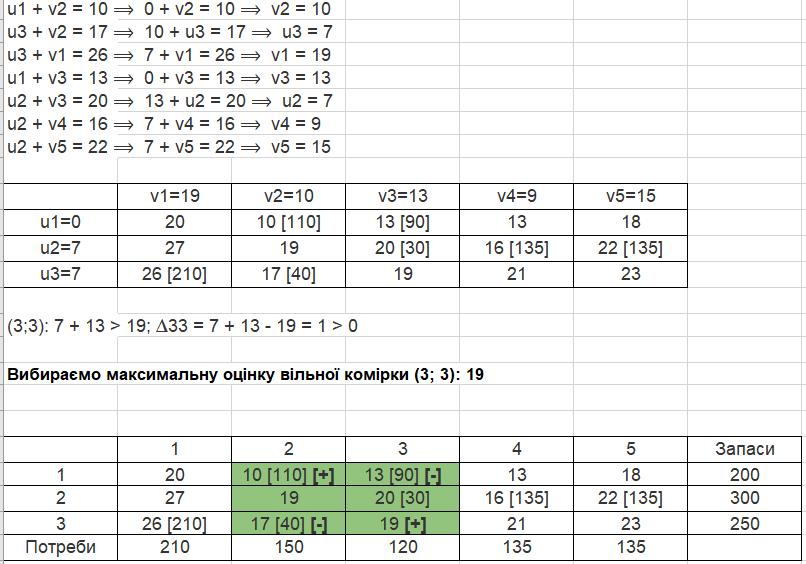
****

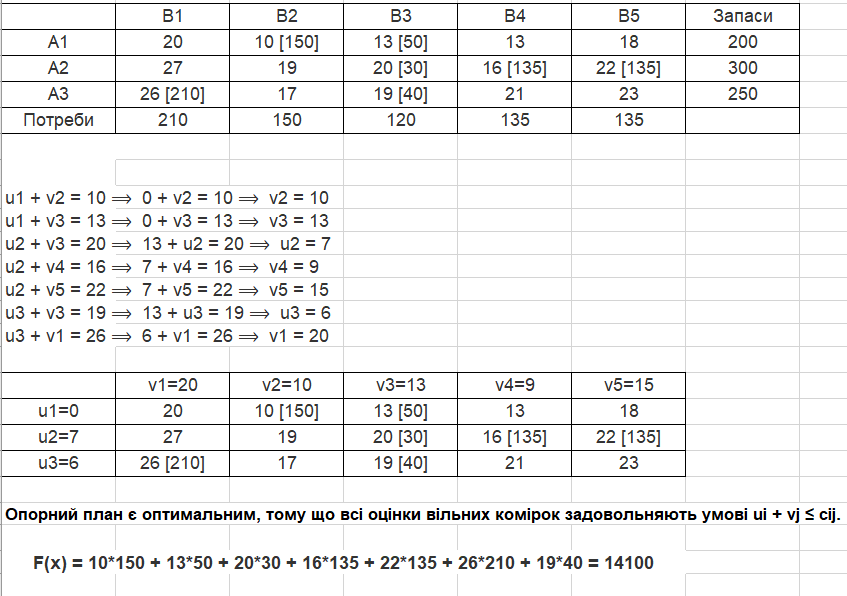
****

****

****

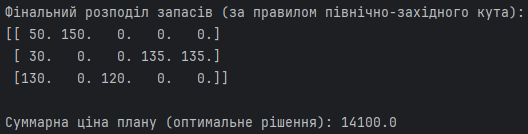
****

****

****

import numpy as np  
from pulp import \*  
  
def northwest\_corner\_rule(supply\_arr, demand\_arr):  
 rows = len(supply\_arr)  
 cols = len(demand\_arr)  
 result = np.zeros((rows, cols))  
  
 i, j = 0, 0  
  
 while i < rows and j < cols:  
 quantity = min(supply\_arr[i], demand\_arr[j])  
 result[i, j] = quantity  
 supply\_arr[i] -= quantity  
 demand\_arr[j] -= quantity  
  
 if supply\_arr[i] == 0:  
 i += 1  
  
 if demand\_arr[j] == 0:  
 j += 1  
  
 return result  
  
def solve\_transportation\_problem(supply, demand, costs):  
 supply\_count, demand\_count = len(supply), len(demand)  
  
 initial\_allocation = northwest\_corner\_rule(supply.copy(), demand.copy())  
  
 costs\_dict = makeDict([range(supply\_count), range(demand\_count)], costs, 0)  
  
 prob = LpProblem("Transportation\_Problem", LpMinimize)  
  
 routes = [(i, j) for i in range(supply\_count) for j in range(demand\_count)]  
 route\_vars = LpVariable.dicts("Route", (range(supply\_count), range(demand\_count)), 0, None, LpInteger)  
  
 prob += lpSum([route\_vars[supplier][consumer] \* costs\_dict[supplier][consumer] for (supplier, consumer) in routes]), "Minimize\_Transportation\_Cost"  
  
 for supplier\_index in range(supply\_count):  
 prob += lpSum([route\_vars[supplier\_index][consumer] for consumer in range(demand\_count)]) <= supply[supplier\_index], f"Supply\_Constraint\_{supplier\_index}"  
  
 for consumer\_index in range(demand\_count):  
 prob += lpSum([route\_vars[supplier][consumer\_index] for supplier in range(supply\_count)]) >= demand[consumer\_index], f"Demand\_Constraint\_{consumer\_index}"  
  
 prob.solve()  
  
 results = np.array([v.varValue for v in prob.variables()]).reshape(supply\_count, demand\_count)  
  
 return results, value(prob.objective)  
  
supply = [200, 300, 250]  
demand = [210, 150, 120, 135, 135]  
costs = [  
 [20, 10, 13, 13, 18],  
 [27, 19, 20, 16, 22],  
 [26, 17, 19, 21, 23]  
]  
  
nw\_result, total\_cost = solve\_transportation\_problem(supply, demand, costs)  
  
print("Фінальний розподіл запасів (за правилом північно-західного кута):")  
print(nw\_result)  
print("\nСуммарна ціна плану (оптимальне рішення):", total\_cost)

Результат співпадає з розрахунками:



**Висновок****:** У ході лабораторної роботи, я провів розрахунки транспортної задачі методом північно-західного кута та методом потенціалів, а також автоматизував процес розв’язку задачі на мові Python.