# НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики Кафедра прикладної математики

#### Звіт

# із лабораторної роботи із дисципліни «АЛГОРИТМИ І СИСТЕМИ КОМП'ЮТЕРНОЇ МАТЕМАТИКИ 1.МАТЕМАТИЧНІ АЛГОРИТМИ»

на тему:

Транспортна задача

Виконав: Перевірила: студент групи КМ-13 асистент кафедри ПМА Онищенко В.С. Ковальчук-Химюк Л.О.

# **3MICT**

1	BC	ľ¥II		4
2	Пор	эядок в	иконання роботи	5
3	Осн	овна ч	астина	6
	3.1	Вихід	ні дані по роботі.	. 6
	3.2	Типов	вий перелік вимог до ПЗ	. 7
	3.3	Опис	методів	. 9
		3.3.1	Оптимізація розподілу ресурсів	. 9
		3.3.2	Обмеження логістичної мережі	. 10
	3.4	Побуд	цова математичної моделі	. 12
		3.4.1	Транспортна задача без обмежень	. 12
		3.4.2	Транспортна задача із обмеженнями	. 13
	3.5	Блок-	схеми алгоритмів методів	. 16
		3.5.1	Транспортна задача	. 16
		3.5.2	Транспортна задача з обмеженнями	. 17
	3.6	Аналі	з правильності функціонування програми	. 18
		3.6.1	Python	. 18
		3.6.2	Octave	. 18
	3.7	Прикл	пад роботи програми	. 19
		3.7.1	Python	. 19
		3.7.2	Octave	. 21
4	вис	СНОВЬ	ки	23
5	СП	исок	ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	24

## 1. ВСТУП

Мета роботи: дослідження методів розв'язання транспортної задачі та практичне застосування цих методів для розв'язку задачі лінійного програмування на електронно-обчислювальній машині (ЕОМ) з використанням систем комп'ютерної математики (СКМ). Для розробки використати мови Python [1] та Octave[2]

## 2. Порядок виконання роботи

- 1. Ознайомитися з теоретичним матеріалом і методичними вказівками до роботи.
- 2. Одержати варіант індивідуального завдання /вихідні дані для задачі, метод розв'язання задачі/. Формалізувати задачу лінійного програмування: скласти математичну модель у загальному вигляді; використовуючи конкретні числові дані перетворити вихідну математичну модель до вигляду, що допускає застосування підпрограм СКМ.
- 3. Розробити блок-схему алгоритму розв'язання задачі.
- 4. Написати, налагодити і виконати програму. У програмі передбачити опис усіх використовуваних масивів і введення значень їх елементів, вивід вихідних даних, результатів розрахунків і значення ознаки результату.
- 5. Розв'язати задачу лінійного програмування.
- 6. Оформити звіт по роботі.

## 3. Основна частина

## 3.1. Вихідні дані по роботі.

Умови для варіанту 9 зазначені в таблицях 3.1.1 та 3.1.2

Таблиця 3.1.1 – умови варіанту 9, транспортна здача

Варіант	Тран	спор	гні ви	трати	[	Об'єм	и вир	обн	ицтва	Об'е	см п	отре	еб	
	(Мат	риця	C)			(векто	op a)			(век	гор 1	b)		
9	30	24	11	12	25	21	19	15	25	15	15	15	15	20
	26	4	29	20	24									
	27	14	14	10	18									
	6	14	28	8	2									

Таблиця 3.1.2 – Умови варіанту 9, транспортна задача з обмеженнями на пропускну здатність

Bapi-	Тран	нспо	ртні	витр	ати	Обм	еже	ня п	на		Об'єм ви-	Об'єм
ант						пере	евезе	кнн			робництва	потреб
						(мат	риця	4 D)			(вектор а)	(вектор b)
9	5 19 9 7	4 20 6 15	2 8	5 4 19 6	14 5 1 11	_	8	14 15 15 2	4 5	13 12 10 20	60 50 40 20	10 50 40 50 20

### 3.2. Типовий перелік вимог до ПЗ

Програмне забезпечення, розроблюване в рамках виконання кожної лабораторної роботи, повинно задовольняти низку вимог, які можна розділити на *обов'язкові* (які ПЗ повинно задовольняти незалежно від лабораторної роботи) та *варіативні* (які для кожної лабораторної роботи унікальні). До обов'язкових вимог належать:

- У програмі повинно бути передбачено перевірки на некоректне введення для всіх полів введення, зокрема:
  - 1. порожнє введення;
  - 2. синтаксично некоректне введення (наприклад, літери в полі для числових коефіцієнтів);
  - 3. уведення спеціальних символів;
  - 4. уведення чисел, які перевищують максимальний розмір для чисел відповідного типу даних (для перевірки на переповнення розрядної сітки).

У випадку некоректного введення повинно з'являтися діалогове вікно з відповідним повідомленням.

- Для всіх полів введення повинно бути визначено гранично допустиму кількість символів (для числових полів гранично допустимі значення).
- У програмі повинно відслідковуватися переповнення розрядної сітки під час виконання обчислень. У випадку переповнення повинно з'являтися діалогове вікно з відповідним повідомленням.

• У графічному інтерфейсі користувача повинно бути передбачено можливість гнучкого налаштування розмірності розв'язуваної задачі (наприклад, можливість зміни розмірності матриці чи кількості складів у транспортній задачі).

До варіативних вимог належать вимоги щодо перевірки на коректне опрацювання виключних ситуацій, які можуть виникати під час застосування заданого методу до розв'язання поставленої задачі (наприклад, коли сума заявок не збігається з сумою ресурсів у транспортній задачі, нижня межа інтегрування перевищує верхню тощо).

#### 3.3. Опис методів

### 3.3.1. Оптимізація розподілу ресурсів

#### Постановка задачі

Задача оптимізації розподілу ресурсів  $\epsilon$  ключовою складовою в управлінні ланцюгами поставок. Метою  $\epsilon$  забезпечення ефективного використання доступних ресурсів для доставки продукції від кількох постачальників до споживачів при мінімізації витрат [3].

#### Математична постановка задачі

Розглянемо систему з p джерел постачання  $S_k$  ( $k=1,\ldots,p$ ) і q точок споживання  $T_l$  ( $l=1,\ldots,q$ ), де кожне джерело має обсяг ресурсів  $r_k$ , а кожна точка споживання має потребу в обсязі  $d_l$ . Вартість переміщення одиниці ресурсу між  $S_k$  і  $T_l$  позначається через  $c_{kl}$ .

Задача формулюється наступним чином:

Мінімізувати: 
$$C = \sum_{k=1}^p \sum_{l=1}^q c_{kl} z_{kl}$$

при виконанні умов:

$$\sum_{l=1}^{q} z_{kl} \le r_k, \quad \forall k,$$

$$\sum_{k=1}^{p} z_{kl} \ge d_l, \quad \forall l,$$

$$z_{kl} \ge 0, \quad \forall k, l.$$

де  $z_{kl}$  — кількість ресурсу, переміщеного між  $S_k$  і  $T_l$ .

#### Методи вирішення задачі | Метод Північно-Західного кута

Метод Північно-Західного кута — це простий алгоритм побудови початкового розподілу для задач оптимізації. Він полягає у поступовому заповненні матриці перевезень, починаючи з верхнього лівого кута, з урахуванням обмежень попиту та пропозиції.

### Етапи методу:

- 1. Почати заповнення таблиці з верхнього лівого кута, призначаючи максимально можливий обсяг перевезення, обмежений  $r_k$  і  $d_l$ .
- 2. Відповідно до використаних обсягів зменшити доступний ресурс джерела  $r_k$  або попит точки  $d_l$ .
- 3. Перейти до наступного невикористаного елемента таблиці праворуч або вниз.
- 4. Повторювати, поки всі ресурси і попити не будуть розподілені.

Метод Північно-Західного кута підходить для створення початкового плану, який далі можна покращувати за допомогою методів оптимізації.

#### 3.3.2. Обмеження логістичної мережі

Додаткові обмеження, такі як пропускна здатність маршрутів  $b_{kl}$ , можуть бути включені в модель [3]:

Мінімізувати 
$$C = \sum_{k=1}^p \sum_{l=1}^q c_{kl} z_{kl}$$
 з обмеженнями:  $\sum_{l=1}^q z_{kl} \leq r_k, \quad k=1,\ldots,p,$   $\sum_{k=1}^p z_{kl} \geq d_l, \quad l=1,\ldots,q,$   $0 \leq z_{kl} \leq b_{kl}, \quad \forall k,l.$ 

Ця модель дозволяє враховувати реальні обмеження, забезпечуючи більш точне планування. Алгоритм включає побудову початкового плану, перевірку його оптимальності та ітеративне вдосконалення до досягнення найкращого результату.

## 3.4. Побудова математичної моделі

### 3.4.1. Транспортна задача без обмежень

### Формулювання задачі

#### Цільова функція, яке треба мінімізувати

$$\begin{aligned} 30x_{1,1} + 24x_{1,2} + 11x_{1,3} + 12x_{1,4} + 25x_{1,5} + \\ \min Z &= \frac{26x_{2,1} + 4x_{2,2} + 29x_{2,3} + 20x_{2,4} + 24x_{2,5} + }{27x_{3,1} + 14x_{3,2} + 14x_{3,3} + 10x_{3,4} + 18x_{3,5} + } \\ 6x_{4,1} + 14x_{4,2} + 28x_{4,3} + 8x_{4,4} + 2x_{4,5}. \end{aligned}$$

#### Обмеження за умовою

#### 1. Обмеження постачання:

$$\sum_{j=1}^{5} x_{1,j} = 21,$$

$$\sum_{j=1}^{5} x_{2,j} = 19,$$

$$\sum_{j=1}^{5} x_{3,j} = 15,$$

$$\sum_{j=1}^{5} x_{4,j} = 25.$$

#### 2. Обмеження попиту:

$$\sum_{i=1}^{4} x_{i,1} = 15,$$

$$\sum_{i=1}^{4} x_{i,2} = 15,$$

$$\sum_{i=1}^{4} x_{i,3} = 15,$$

$$\sum_{i=1}^{4} x_{i,4} = 15,$$

$$\sum_{i=1}^{4} x_{i,5} = 20.$$

#### 3. Обмеження невід'ємності:

$$x_{i,j} \geq 0, \quad \forall i, j.$$

### 3.4.2. Транспортна задача із обмеженнями

## Цільова функція, яку треба мінімізувати

### Обмеження за умовою

#### 1. Обмеження постачання:

$$\sum_{j=1}^{5} x_{1,j} = 60,$$

$$\sum_{j=1}^{5} x_{2,j} = 50,$$

$$\sum_{j=1}^{5} x_{3,j} = 40,$$

$$\sum_{j=1}^{5} x_{4,j} = 20.$$

#### 2. Обмеження попиту:

$$\sum_{i=1}^{4} x_{i,1} = 10,$$

$$\sum_{i=1}^{4} x_{i,2} = 50,$$

$$\sum_{i=1}^{4} x_{i,3} = 40,$$

$$\sum_{i=1}^{4} x_{i,4} = 50,$$

$$\sum_{i=1}^{4} x_{i,5} = 20.$$

### 3. Обмеження пропускної здатності:

$$0 \le x_{ij} \le d_{ij}, \quad \forall i, j,$$

## 4. Обмеження невід'ємності:

$$x_{i,j} \ge 0, \quad \forall i, j.$$

# 3.5. Блок-схеми алгоритмів методів

#### 3.5.1. Транспортна задача

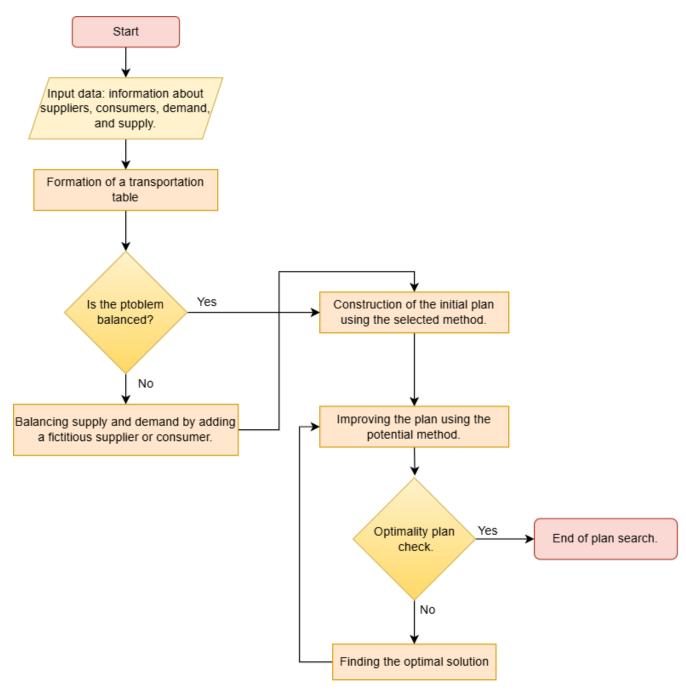


Рисунок 3.5.1.1 – Діаграма алгоримту для вирішення транспортної задачі

#### 3.5.2. Транспортна задача з обмеженнями

Діаграма виконання алгоритму для розв'язання транспортної задачі з обмеженнями представлена на рисунку 3.5.2.1

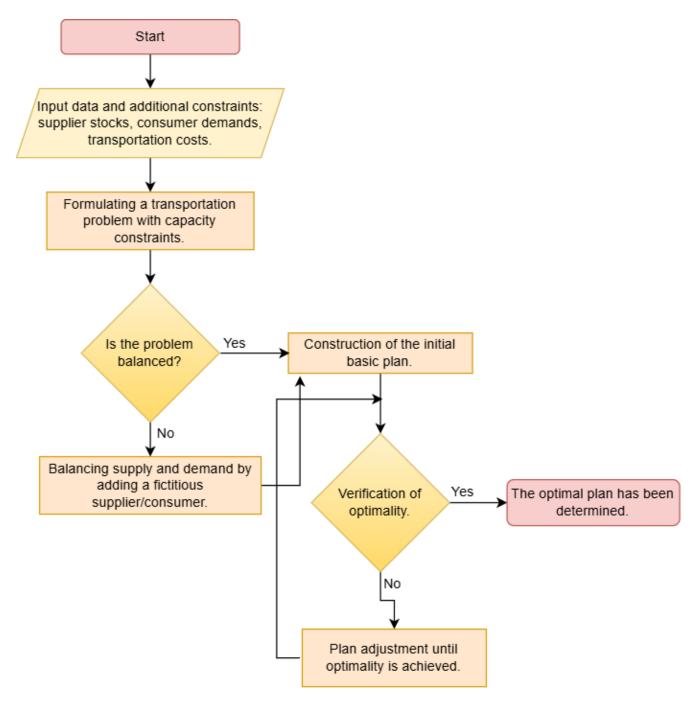


Рисунок 3.5.2.1 – Діаграма алгоримту для вирішення транспортної задачі з урахуванням обмежень.

## 3.6. Аналіз правильності функціонування програми

### **3.6.1. Python**

- Кожна таблиця повинна включати числові значення та мати фіксований розмір.
- Потрібно виконати перевірку розмірів векторів а і b, а також матриці С.
- Необхідно перевірити, чи співпадає сума елементів вектора а з сумою елементів вектора b.

#### 3.6.2. Octave

Програма, створена для GNU Octave, не здійснює перевірку коректності введених даних.

## 3.7. Приклад роботи програми

## **3.7.1. Python**

### Транспортна задача

За програмної реалізації розв'язку заданої задачі отримуємо наступні результати (рис. 3.7.1.1):

				ортної задачі ======
1атриця роз	в'язку:			
0.00	0.00	15.00	6.00	0.00
4.00	15.00	0.00	0.00	0.00
0.00	0.00	0.00	9.00	6.00
11.00	0.00	0.00	0.00	14.00
	ртість пере : 693.00 од			
Вартість	: 693.00 од		споживач):	
Вартість цеталі пост	ачань (пост	иниць		
Вартість Петалі пост Постачалостачалостачалоста	: 693.00 од ачань (пост пьник Р1 ->	ачальник ->	: 15.00 од	иниць
Вартість Цеталі пост Постачал	ачань (пост пьник Р1 -> пьник Р1 ->	ачальник -> Споживач РЗ	: 15.00 оди : 6.00 оди	иниць ниць
Вартість Цеталі пост Постачал Постачал	ачань (пост пьник Р1 -> пьник Р2 ->	ачальник -> Споживач Р3 Споживач Р4	: 15.00 оди : 6.00 один : 4.00 один	иниць ниць ниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал	ачань (пост пьник Р1 -> пьник Р2 -> пьник Р2 -> пьник Р2 -> пьник Р3 ->	ачальник -> Споживач Р3 Споживач Р4 Споживач Р1 Споживач Р2 Споживач Р4	: 15.00 оди : 6.00 оди : 4.00 оди : 15.00 оди : 9.00 оди	иниць ниць ниць иниць ниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	ачань (пост пьник Р1 -> пьник Р2 -> пьник Р2 -> пьник Р3 -> пьник Р3 ->	ачальник -> Споживач РЗ Споживач Р4 Споживач Р1 Споживач Р2 Споживач Р4 Споживач Р5	: 15.00 оди : 6.00 один : 4.00 один : 15.00 один : 9.00 один : 6.00 один	иниць ниць ниць иниць ниць ниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	ачань (пост пьник Р1 -> пьник Р2 -> пьник Р2 -> пьник Р3 -> пьник Р3 -> пьник Р4 ->	ачальник -> Споживач Р3 Споживач Р4 Споживач Р1 Споживач Р2 Споживач Р4	: 15.00 оди : 6.00 оди : 4.00 оди : 15.00 оди : 9.00 оди : 6.00 оди : 11.00 оди	иниць ниць иниць ниць ниць иниць иниць

Рисунок 3.7.1.1 – Вихідні дані розв'язання транспортної задачі (Python)

## Транспортна задача із обмеженнями

За програмної реалізації розв'язку заданої задачі отримуємо наступні результати (рис. 3.7.1.2):

				портної задачі
Матриця обм	еження:			
777		14.00		
		15.00		
6.00	8.00	15.00	5.00	10.00
14.00	7.00	2.00	23.00	20.00
Матриця роз	в'язку:			
5.00	20.00	10.00	25.00	0.00
0.00	21.00	15.00	4.00	10.00
5.00	8.00	15.00	2.00	10.00
0.00	1.00	0.00	19.00	0.00
_				
Вартість	: 1,336.00	одиниць	CDOWNESTI	
Вартість  Деталі пост	: 1,336.00	одиниць тачальник ->		
Вартість Деталі постачал	: 1,336.00 ачань (пос тьник Р1 ->	одиниць тачальник -> Споживач Р1	: 5.00 оди	иниць
Вартість Деталі пост	: 1,336.00 ачань (пос тьник Р1 ->	одиниць тачальник -> Споживач Р2 Споживач Р2	: 5.00 оди : 20.00 од	иниць циниць
Вартість Деталі постачал Постачал Постачал Постачал Постачал Постачал	: 1,336.00 ачань (пос тьник Р1 -> тьник Р1 ->	одиниць тачальник -> Споживач Р2 Споживач Р2	: 5.00 OD : 20.00 OD : 10.00 OD	иниць диниць
Вартість Деталі поста  Постачал  Постачал  Постачал  Постачал  Постачал	: 1,336.00 ачань (пос тьник Р1 -> тьник Р1 -> тьник Р1 ->	одиниць  тачальник ->  Споживач РЗ  Споживач РЗ  Споживач РЗ	: 5.00 оди : 20.00 од : 10.00 од : 25.00 од	иниць диниць диниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	: 1,336.00 ачань (пос тьник Р1 -> тьник Р1 -> тьник Р1 -> тьник Р1 ->	одиниць Тачальник -> Споживач РЗ Споживач РЗ Споживач РЗ Споживач РЗ	: 5.00 ODV : 20.00 OD : 10.00 OD : 25.00 OD : 21.00 OD	иниць диниць диниць диниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	ачань (посльник Р1 -> пьник Р2 -> пьник Р2 ->	ОДИНИЦЬ  Тачальник ->  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ	: 5.00 CDV : 20.00 OD : 10.00 OD : 25.00 OD : 21.00 OD : 15.00 OD	иниць циниць циниць циниць
Вартість  Деталі постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	зчань (постыник Р1 -> пьник Р1 -> пьник Р1 -> пьник Р1 -> пьник Р2 -> пьник Р2 -> пьник Р2 ->	одиниць  тачальник ->  Споживач РЗ	: 5.00 cg/ : 20.00 cg/ : 10.00 cg/ : 25.00 cg/ : 15.00 cg/ : 4.00 cg/	иниць диниць диниць диниць диниць диниць
Вартість  Деталі пост  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал  Постачал	ачань (посльник Р1 - 20 пьник Р1 - 20 пьник Р2 - 20 пьник	ОДИНИЦЬ  Тачальник ->  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ  Споживач РЗ	: 5.00 ODV : 20.00 OD : 10.00 OD : 25.00 OD : 21.00 OD : 15.00 OD : 4.00 ODV	иниць диниць диниць диниць диниць
Вартість  Деталі пост  Постачал	ачань (посльник Р1 - 21 гоник Р1 - 21 гоник Р2 - 21 гоник Р3 - 2	ОДИНИЦЬ  Тачальник ->  Споживач РЗ	: 5.00 CDV : 20.00 OV : 10.00 OV : 25.00 OV : 21.00 OV : 15.00 OV : 10.00 OV : 5.00 ODV	иниць диниць диниць диниць иниць иниць иниць иниць
Вартість  Деталі постачал	ачань (постыник Р1 - 2) тыник Р1 - 20 тыник Р2 - 20 тыник Р3 - 20	ОДИНИЦЬ  Тачальник -> Споживач РЗ	: 5.00 CDV : 20.00 OI : 10.00 OI : 25.00 OI : 15.00 OI : 4.00 CDV : 5.00 CDV : 8.00 CDV	иниць диниць диниць диниць иниць диниць диниць иниць иниць
Вартість  Деталі постачал	ачань (послыник Р1 - 2) пьник Р1 - 2) пьник Р2 - 2) пьник Р3 - 2) пьник Р3 - 2) пьник Р3 - 2	ОДИНИЦЬ  Тачальник -> Споживач РЗ	: 5.00 CDV : 20.00 OV : 10.00 OV : 25.00 OV : 15.00 OV : 4.00 OV : 5.00 OV : 8.00 OV : 15.00 OV	иниць диниць диниць диниць диниць иниць иниць иниць
Вартість  Деталі пост  Постачал	ачань (посльник Р1 - 2 тыник Р1 - 2 тыник Р2 - 2 тыник Р2 - 2 тыник Р3 - 2	ОДИНИЦЬ  Тачальник ->  Споживач РЗ	: 5.00 CDV : 20.00 OD : 10.00 OD : 25.00 OD : 15.00 OD : 4.00 OD : 5.00 OD : 8.00 OD : 15.00 OD : 2.00 OD	иниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць пиниць
Постачал По	ачань (посльник Р1 - 2) тыник Р1 - 2) тыник Р2 - 2) тыник Р2 - 2) тыник Р2 - 2) тыник Р3 - 2	ОДИНИЦЬ  Тачальник ->  Споживач Р2  Споживач Р3  Споживач Р3  Споживач Р3  Споживач Р3  Споживач Р3	: 5.00 CDV : 20.00 OV : 10.00 OV : 25.00 OV : 15.00 OV : 10.00 OV : 5.00 OV : 15.00 OV : 2.00 OV : 10.00 OV	иниць иниць иниць иниць иниць иниць иниць иниць иниць иниць

Рисунок 3.7.1.2 – Вихідні дані розв'язання транспортної задачі з обмеженнями (Python)

#### **3.7.2.** Octave

## Транспортна задача

За програмної реалізації розв'язку заданої задачі отримуємо наступні результати (рис. 3.7.2.1):

0.00	0.	00   1	5.00	1	6.00		0.00	
4.00	15.	00	0.00	1	0.00		0.00	1
0.00	0.	00	0.00	1	9.00		6.00	1
11.00	0.	00	0.00	1	0.00		14.00	1
Вартість: 69	3.00 0	линиць						
Деталі поста	чань (	постачаль						
Деталі поста	чань (	постачаль						
Деталі поста Постачальник	чань (: Pl ->	постачаль.	53:	15.00	одиниц	ць		
Вартість: 69  Деталі поста Постачальник Постачальник Постачальник	чань (: Pl -> Pl ->	постачаль Споживач Споживач	S3: S4:	15.00 6.00	одиниц одиниц	ĮЪ		
Деталі поста Постачальник Постачальник Постачальник	чань (; P1 -> P1 -> P2 ->	постачаль Споживач Споживач Споживач	S3: S4: S1:	15.00 6.00 4.00	одиниц одиницы одиницы	ĮЬ o		
Деталі поста Постачальник Постачальник Постачальник Постачальник	чань (я P1 -> P1 -> P2 -> P2 ->	постачаль: Споживач Споживач Споживач Споживач	S3: S4: S1: S2:	15.00 6.00 4.00 15.00	одиниц одиница одиница одиница	ЦБ ) ЦБ		
Деталі поста Постачальник Постачальник Постачальник Постачальник Постачальник	чань (; P1 -> P1 -> P2 -> P2 ->	постачаль Споживач Споживач Споживач Споживач Споживач	S3: S4: S1: S2: S4:	15.00 6.00 4.00 15.00 9.00	одиниц одиница одиница одиница одиница	lp o		
Деталі поста Постачальник Постачальник	чань (; P1 -> P1 -> P2 -> P2 -> P3 ->	постачаль Споживач Споживач Споживач Споживач Споживач Споживач	S3: S4: S1: S2: S4: S5:	15.00 6.00 4.00 15.00 9.00 6.00	одиниц одиница одиница одиница одиница одиница	lp c		

Рисунок 3.7.2.1 – Вихідні дані розв'язання транспортної задачі (Octave)

## Транспортна задача із обмеженнями

За програмної реалізації розв'язку заданої задачі отримуємо наступні результати (рис. 3.7.2.2):

19.00									
	-		1 14						
6 00							-		-
			15						
14.00	l 	7.00	3	2.00	1 :	23.00	I	20.00	<u> </u>
атриця роз	B' #3	ку:							
5.00	:	20.00	10	0.00		25.00	1	0.00	ı
0.00	1 2	21.00	15	5.00	1	4.00	1	10.00	1
5.00	I	8.00	15	5.00	1	2.00	1	10.00	1
0.00	I	1.00	1 (	0.00	1	19.00	1	0.00	1
еталі пос	гачан	ь (по	стачалы	ник -	-> спо	кивач			
			01414111						
остачальн	nk Pl			51:	5.00				
остачальні остачальні		-> C	поживач			одиниц	ць		
остачальні остачальні	nk Pl	-> C: -> C: -> C:	поживач поживач поживач	52: 53:	20.00	одиниі одини одини	иць иць		
остачальні остачальні остачальні	nk Pl nk Pl nk Pl	-> C: -> C: -> C: -> C:	равижоп Равижоп Равижоп Равижоп	S2: S3: S4:	20.00 10.00 25.00	одиниі одині одині одині одині	иць иць иць		
остачальні остачальні остачальні остачальні	uk Pl uk Pl uk Pl uk P2	-> C: -> C: -> C: -> C: -> C:	PARUWON PARUWON PARUWON PARUWON PARUWON	S2: S3: S4: S2:	20.00 10.00 25.00 21.00	одиниі одині одині одині одині одині	ипр ипр ипр ипр		
остачальні остачальні остачальні остачальні остачальні	MK P1 MK P1 MK P1 MK P2 MK P2	-> C:	nomubay nomubay nomubay nomubay nomubay	S2: S3: S4: S2: S3:	20.00 10.00 25.00 21.00 15.00	одини одини одини одини одини одини	итр итр итр итр		
остачальні остачальні остачальні остачальні остачальні остачальні	MK P1 MK P1 MK P1 MK P2 MK P2 MK P2	-> C:	nomubay nomubay nomubay nomubay nomubay nomubay	S2: S3: S4: S2: S3: S4:	20.00 10.00 25.00 21.00 15.00 4.00	одини одини одини одини одини одини	тр итр итр итр итр		
остачальні остачальні остачальні остачальні остачальні остачальні	MK P1 MK P1 MK P1 MK P2 MK P2 MK P2 MK P2 MK P2	-> C:	nownbay nownbay nownbay nownbay nownbay nownbay nownbay nownbay	S2: S3: S4: S2: S3: S4: S5:	20.00 10.00 25.00 21.00 15.00 4.00	одиниі одини одини одини одини одини одиниі	итр итр итр итр итр		
остачальні остачальні остачальні остачальні остачальні остачальні остачальні	MR P1 MR P1 MR P2 MR P2 MR P2 MR P2 MR P2 MR P3	-> C:	nowneau  commeau  com	S2: S3: S4: S2: S3: S4: S5: S5:	20.00 10.00 25.00 21.00 15.00 4.00 5.00	одиниі одині одині одині одині одині одиниі одиниі	тр птр птр птр птр птр		
остачальні остачальні остачальні остачальні остачальні остачальні остачальні	MR P1 MR P1 MR P2 MR P2 MR P2 MR P2 MR P2 MR P3 MR P3	-> C:	nowneau  commeau  com	S2: S3: S4: S2: S3: S4: S5: S1: S2:	20.00 10.00 25.00 21.00 15.00 4.00 5.00 8.00	одинио сдину одину одину сдину одинио одинио одинио одинио одинио	тр птр птр птр птр птр		
остачальні остачальні остачальні остачальні остачальні остачальні остачальні остачальні	MR P1 MR P1 MR P2 MR P2 MR P2 MR P2 MR P3 MR P3 MR P3 MR P3	-> C:	OWNBAY	S2: S3: S4: S2: S3: S4: S5: S5: S1: S2: S3:	20.00 10.00 25.00 21.00 15.00 4.00 5.00 8.00 15.00	одиниі одини одини одини одини одиниі одиниі одиниі одиниі	MP  TP  MP  MP  MP  MP  MP  MP  MP  MP		
остачальні остачальні остачальні остачальні остачальні остачальні остачальні	MR P1 MR P1 MR P2 MR P2 MR P2 MR P2 MR P3 MR P3 MR P3 MR P3 MR P3	-> C:	OXUBAY	S2: S3: S4: S2: S3: S4: S5: S1: S2: S3: S4:	20.00 10.00 25.00 21.00 15.00 4.00 5.00 8.00 15.00 2.00	одиниі одині одині одині одині одиниі одиниі одиниі одиниі	тр тр тр тр тр тр тр тр тр		

Рисунок 3.7.2.2 – Вихідні дані розв'язання транспортної задачі з обмеженнями (Octave)

## 4. ВИСНОВКИ

У ході цієї лабораторної роботи здобуто навички застосування методів розв'язання транспортних задач та їх реалізації для лінійного програмування з використанням програмного забезпечення.

Розроблено інструменти для розв'язання двох типів оптимізаційних задач: стандартної "Транспортної задачі" та "Транспортної задачі з обмеженнями". Також було впроваджено механізм перевірки введених даних для підвищення точності результатів.

За результатами роботи можемо зробити висновки щодо застосування методів розв'язання транспортних задач. Для першої задачі без обмежень було сформовано оптимальну матрицю розв'язку, в якій загальна вартість перевезення склала 693 одиниці. Система поставок показує, як ефективно можна здійснити транспортування між постачальниками і споживачами, при цьому мінімізуючи витрати. Для другого випадку, коли задачею додаються певні обмеження, отримано нову матрицю розв'язку, де загальна вартість транспортування складає 1336 одиниць, що значно більше через впровадження додаткових обмежень. Ці зміни підвищують загальні витрати на транспортування, але водночас вони можуть відображати умови реальних логістичних задач, де часто є обмеження на наявність ресурсів, транспорту, чи інших чинників. Нові обмеження значно змінили розподіл ресурсів, зокрема призвели до істотного зростання вартості, що вказує на важливість ретельного врахування таких обмежень під час моделювання подібних задач.

# 5. СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

- 1. Python Software Foundation. *Python Documentation* (2023). https://docs.python.org/3.12/.
- 2. Eaton, J. W., Bateman, D., Hauberg, S. & Wehbring, R. *GNU Octave 4.0 Reference Manual: Free Your Numbers* (Free Software Foundation, 2015). ISBN: 978-9888381050. https://docs.octave.org/octave-4.0.0.pdf.
- 3. Johnson, R. *Supply Chain Management: Strategies and Practices* 2nd (Pearson, New York, 2024).

# 6. ДОДАТОК А (код роботи програм)

## **Python**

```
import numpy as np
  from scipy.optimize import linprog
3
  costs 1 = np.array([
      [30, 24, 11, 12, 25],
      [26, 4, 29, 20, 24],
      [27, 14, 14, 10, 18],
7
      [6, 14, 28, 8, 2]
8
  ])
9
10
  supply 1 = np.array([21, 19, 15, 25])
11
  demand 1 = np.array([15, 15, 15, 15, 20])
13
  costs 2 = np.array([
      [5, 4, 20, 5, 14],
15
      [19, 20, 2, 4, 5],
16
      [9, 6, 8, 19, 1],
      [7, 15, 20, 6, 11]
  ])
19
20
  capacities = np.array([
      [5, 20, 14, 25, 13],
22
      [19, 22, 15, 4, 12],
23
      [6, 8, 15, 5, 10],
24
      [14, 7, 2, 23, 20]
25
  ])
26
27
supply 2 = np.array([60, 50, 40, 20])
```

```
demand 2 = np.array([10, 50, 40, 50, 20])
30
  import numpy as np
  from scipy.optimize import linprog
  def check numeric data(C, supply, demand, capacity=None):
      if not (np.issubdtype(C.dtype, np.number) and np.all(np.isfinite(C)))
35
          raise ValueError("Cost matrix 'C' must contain only numeric and fi
36
      if not (np.issubdtype(supply.dtype, np.number) and np.all(np.isfinite)
37
          raise ValueError("Supply vector must contain only numeric and fin:
38
      if not (np.issubdtype(demand.dtype, np.number) and np.all(np.isfinite
39
          raise ValueError("Demand vector must contain only numeric and final
40
      if capacity is not None:
41
          if not (np.issubdtype(capacity.dtype, np.number) and np.all(np.is
42
              raise ValueError("Capacity matrix must contain only numeric and
43
  def validate dimensions(C, supply, demand, capacity=None):
      check numeric data(C, supply, demand, capacity)
      if len(supply) != C.shape[0]:
47
          raise ValueError("Supply vector length must match the number of ro
48
      if len(demand) != C.shape[1]:
49
          raise ValueError("Demand vector length must match the number of co
50
      if capacity is not None and capacity.shape != C.shape:
51
          raise ValueError("Capacity matrix dimensions must match those of o
52
53
  def ensure balanced supply demand(supply, demand):
54
      if not np.isclose(np.sum(supply), np.sum(demand)):
          raise ValueError("Total supply and demand must be equal.")
  def adjust_imbalance(supply, demand, cost_matrix, capacity=None, high_valu
      supply_total = np.sum(supply)
59
      demand total = np.sum(demand)
```

```
if supply total > demand total:
          demand = np.append(demand, supply_total - demand_total)
          cost_matrix = np.hstack((cost_matrix, np.zeros((cost_matrix.shape)))
          if capacity is not None:
               capacity = np.hstack((capacity, np.full((capacity.shape[0], 1)
66
      elif demand total > supply total:
67
          supply = np.append(supply, demand total - supply total)
68
          cost_matrix = np.vstack((cost_matrix, np.zeros((1, cost_matrix.sha
69
          if capacity is not None:
70
               capacity = np.vstack((capacity, np.full((1, capacity.shape[1])
71
72
      return supply, demand, cost matrix, capacity
73
  def transport simple(C, supply, demand):
      rows, cols = C.shape
76
      cost vector = C.flatten()
      equality_constraints = np.zeros((rows + cols, rows * cols))
79
      supply_demand_vector = np.concatenate((supply, demand))
80
81
      for i in range(rows):
82
          equality constraints[i, i * cols:(i + 1) * cols] = 1
83
84
      for j in range(cols):
85
          equality_constraints[rows + j, j::cols] = 1
86
87
      bounds = [(0, None)] * (rows * cols)
88
      solution = linprog(cost_vector, A_eq=equality_constraints, b_eq=supply
      if solution.success:
```

```
return solution.x.reshape(rows, cols), solution.fun
                                  raise ValueError("No feasible solution found for the transportation programme to the transport
             def transport_with_capacity(C, capacity, supply, demand):
                                  rows, cols = C.shape
  97
                                  cost vector = C.flatten()
  98
  99
                                 equality_constraints = np.zeros((rows + cols, rows * cols))
100
                                 supply_demand_vector = np.concatenate((supply, demand))
101
102
                                  upper bounds = np.eye(rows * cols)
103
                                  capacity vector = capacity.flatten()
104
105
                                 for i in range(rows):
106
                                                      equality constraints[i, i * cols:(i + 1) * cols] = 1
                                 for j in range(cols):
                                                      equality constraints[rows + j, j::cols] = 1
111
                                  bounds = [(0, None)] * (rows * cols)
112
113
                                  solution = linprog(cost vector, A eq=equality constraints, b eq=supply
114
115
                                 if solution.success:
                                                      return solution.x.reshape(rows, cols), solution.fun
117
                                  raise ValueError("No feasible solution found for the transportation programme to the transport
118
119
             def solve transport problem(C, supply, demand, use capacity=False, capacit
120
                                 try:
                                                     validate_dimensions(C, supply, demand, capacity)
                                                     ensure_balanced_supply_demand(supply, demand)
```

```
if use capacity:
               if capacity is None:
                   raise ValueError("Capacity matrix must be provided when us
               supply, demand, C, capacity = adjust_imbalance(supply, demand)
               return transport with capacity(C, capacity, supply, demand)
130
           supply, demand, C, _ = adjust_imbalance(supply, demand, C)
131
           return transport_simple(C, supply, demand)
132
      except ValueError as error:
133
           return str(error)
134
135
  solution, cost = solve transport problem(costs 1, supply 1, demand 1, use
137
  print("=" * 60)
  print(" " * 15 + "Результати розвязання' транспортної задачі")
  print("=" * 60)
  print("\Матрицяп розвязку':")
  print("-" * 60)
  for row in solution:
143
       print(" | ".join(f"{cell:8.2f}" for cell in row))
144
  print("-" * 60)
145
146
  print("\Загальнап вартість перевезення:")
147
  print(f@@" Вартість: {cost:,.2f} одиниць")
148
  print("-" * 60)
149
150
  print("\Detanin постачань постачальник( -> споживач):")
  for i, provider in enumerate([f'P\{i\}'] for i in range(1, len(supply 1)+1)]
152
      for j, customer in enumerate([f'P{i}' for i in range(1, len(demand 1)-
           if solution[i][j] > 0: # Show only active connections
               print(f"@@ Постачальник {provider} -> Споживач {customer}: {se
```

```
print("=" * 60)
  solution, cost = solve transport problem(costs 2, supply 2, demand 2, use
  print("="*60)
  print(""*15 + "Результати розв'язання транспортної задачі")
163 print("=" * 60)
164 print("\Матрицяп обмеження:")
print("-" * 60)
  for row in capacities:
166
       print(" | ".join(f"{cell:8.2f}" for cell in row))
167
  print("-" * 60)
  print("\Maтрицяп розв'язку:")
  print("-" * 60)
  for row in solution:
      print(" | ".join(f"{cell:8.2f}" for cell in row))
172
  print("-" * 60)
174
print("\Загальнап вартість перевезення:")
  print(f@@" Вартість: {cost:,.2f} одиниць")
176
  print("-" * 60)
177
178
  print("\Деталіп постачань постачальник( -> споживач):")
179
  for i, provider in enumerate([f'P{i}' for i in range(1, len(supply_2)+1)])
180
      for j, customer in enumerate([f'P{i}' for i in range(1, len(demand 2)-
181
           if solution[i][j] > 0:
182
               print(f"@@ Постачальник {provider} -> Споживач {customer}: {se
183
184
185 print("=" * 60)
```

### **Octave**

```
costs_1 = [
      30, 24, 11, 12, 25;
2
      26, 4, 29, 20, 24;
3
      27, 14, 14, 10, 18;
       6, 14, 28, 8, 2
5
  ];
  supply_1 = [21; 19; 15; 25];
  demand_1 = [15; 15; 15; 15; 20];
10
  costs 2 = [
11
       5, 4, 20, 5, 14;
12
      19, 20, 2, 4,
                       5;
13
       9, 6, 8, 19,
                       1;
14
       7, 15, 20, 6, 11
15
  ];
16
17
  capacities = [
       5, 20, 14, 25, 13;
      19, 22, 15, 4, 12;
       6, 8, 15, 5, 10;
      14, 7, 2, 23, 20
  ];
  supply_2 = [60; 50; 40; 20];
  demand_2 = [10; 50; 40; 50; 20];
27
28
  disp('Solving transportation problem without capacity constraints:');
  try
30
      [x no capacity, cost no capacity] = transportation problem no capacity
31
      fprintf("%s\n", repmat("=", 1, 60));
```

```
fprintf("%s\n", repmat(" ", 1, 15), "Результати розвязання' транспорти
      fprintf("%s\n", repmat("=", 1, 60));
      disp(x_no_capacity);
      fprintf("\Maтрицяп розвязку':\n");
      fprintf("%s\n", repmat("-", 1, 60));
37
      for i = 1:size(x no capacity, 1)
38
          fprintf(" %8.2f |", x no capacity(i, :));
39
          fprintf("\n");
40
      end
41
      fprintf("%s\n", repmat("-", 1, 60));
42
43
      fprintf("\Загальнап вартість перевезення:\n");
44
      fprintf("Bapтiсть: %.2f одиниць\n", cost no capacity);
45
      fprintf("%s\n", repmat("-", 1, 60));
46
      fprintf("\Detanin постачань постачальник( -> споживач):\n");
48
      for i = 1:size(x no capacity, 1)
           for j = 1:size(x no capacity, 2)
               if x_{no} capacity(i, j) > 0
51
                   fprintf("Постачальник P%d -> Споживач S%d: %.2f одиниць\n'
52
               end
53
           end
54
      end
55
56
      fprintf("%s\n", repmat("=", 1, 60));
57
58
  catch err
59
      disp('Error in solving transportation problem without capacity constra
60
      disp(err.message);
  end
```

```
disp('Solving transportation problem with capacity constraints:');
  try
      [x_with_capacity, cost_with_capacity] = transportation_problem_with_capacity.
      fprintf("%s\n", repmat("=", 1, 60));
      fprintf("%s\n", repmat(" ", 1, 15), "Результати розвязання' транспорти
69
      fprintf("%s\n", repmat("=", 1, 60));
70
71
      fprintf("\Maтрицяn обмеження:\n");
72
      fprintf("%s\n", repmat("-", 1, 60));
73
      for i = 1:size(capacities, 1)
74
          fprintf(" %8.2f |", capacities(i, :));
75
          fprintf("\n");
76
      end
77
      fprintf("%s\n", repmat("-", 1, 60));
78
      fprintf("\Maтрицяп розвязку':\n");
80
      fprintf("%s\n", repmat("-", 1, 60));
      for i = 1:size(x with capacity, 1)
          fprintf(" %8.2f |", x_with_capacity(i, :));
83
          fprintf("\n");
84
      end
85
      fprintf("%s\n", repmat("-", 1, 60));
86
87
      fprintf("\Загальнап вартість перевезення:\n");
88
      fprintf("Вартість: %.2f одиниць\n", cost with capacity);
89
      fprintf("%s\n", repmat("-", 1, 60));
90
91
      fprintf("\Detanin постачань постачальник( -> споживач):\n");
92
      for i = 1:size(x_with_capacity, 1)
          for j = 1:size(x with capacity, 2)
               if x with capacity(i, j) > 0
                   fprintf("Постачальник P%d -> Споживач S%d: %.2f одиниць\n
```

```
end
end
fprintf("%s\n", repmat("=", 1, 60));

catch err
disp('Error in solving transportation problem with capacity constraint disp(err.message);
end
end
end
fprintf("%s\n", repmat("=", 1, 60));

catch err
disp('Error in solving transportation problem with capacity constraint disp(err.message);
end
```