МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
"МИРЭА - Российский технологический университет"
РТУ МИРЭА

Институт Информационных Технологий

Кафедра Вычислительной Техники

ПРАКТИЧЕСКАЯ РАБОТА №7

по дисциплине
«Теория принятия решений»
Транспортная задача

Студент группы: ИКБО-04-22 Кликушин В.И.____

(Ф. И.О. студента)

Преподаватель __Железняк Л.М.__

(Ф.И.О. преподавателя)

Москва 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Транспортная задача – это частный случай задачи линейного программирования, где ограничениями являются запасы поставщиков и потребности потребителей, а целевая функция – это стоимость перевозок. Функцию нужно минимизировать.

Транспортную задачу можно решать более простыми методами, чем симплекс-метод. Для этого нужно составить опорный план, используя метод северо-западного угла или метод минимальной стоимости. После построения опорного плана применяется метод потенциалов, который позволяет построить оптимальный план проще, чем симплекс-метод.

Транспортные задачи бывают закрытые и открытые. Закрытой задача является, если сумма количества ресурсов у поставщиков равна сумме потребностей у поставщиков, и открытой в противном случае. В данной работе рассматривается решение закрытой транспортной задачи.

1 ТРАНСПОРТНАЯ ЗАДАЧА

1.1 Постановка задачи

Вариант 13.

Задача. Имеются поставщики и потребители, у которых известны запасы и потребности, соответственно, а также известны стоимости перевозки от каждого поставщика к каждому потребителю. Данные занесены в Таблицу 1.

Таблица 1. Исходные данные задачи.

Потребители / Поставщики	**40**	**30**	**30**	**50**
60	2	3	5	1
70	3	4	9	4
20	2	5	2	5

Определить оптимальный план перевозок.

1.2 Математическая модель транспортной задачи

Математически задачу можно сформулировать следующим образом. Определить переменные x_{ij}, которые минимизируют суммарную стоимость перевозок.

$$f(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

и удовлетворяют системе ограничений

a) $x_{i1} + x_{i2} + \dots + x_{in} = a_i,\ i = 1,2,\dots,m$ – с каждого пункта отправления груз должен быть вывезен полностью;

b) $x_{1j} + x_{2j} + \dots + x_{mj} = b_j,\ j = 1,2,\dots,n$ – потребитель должен получить ровно столько, сколько ему требуется;

c) $x_{ij} \geq 0,\ i = 1,2,\dots,m; j = 1,2,\dots,n.$

Транспортная задача является закрытой, т.к. $\sum_{i=0}^{3} a_i = 60 + 70 + 20 = 150$ и $\sum_{i=0}^{4} b_i = 40 + 30 + 30 + 50 = 150$.

1.3 Метод северо-западного угла

Заполним ячейку a_{11}. Т.к. потребности B_1 меньше запасов A_1 (40 < 60), то $x_{11} = 40$. Запасы первого поставщика не исчерпаны, поэтому происходит переход к ячейке a_{12}. Т.к. потребности B_2 больше запасов A_1 (30 > 20), то $x_{12} = 20$. Запасы первого поставщика исчерпаны, происходит переход к ячейки a_{22}. В составе заявки пункта B_2 остались неудовлетворёнными 10 единиц. Эти 10 единиц покроем за счёт пункта A_2. Далее перейдём к ячейки a_{23}. Запасов второго поставщика достаточно, значит $x_{23} = 30$. Запасы третьего потребителя удовлетворены, значит переходим к ячейки a_{24}. Так как потребность B_4 больше запасов A_2, (50 > 30) $x_{24} = 30$. Из запасов пункта A_3 выделим все доступные 20 единиц, чтобы удовлетворить запрос пункта B_4.

Полученное решение является опорным решением транспортной задачи. Общая стоимость перевозок составляет:

$f_0 = 2 * 40 + 3 * 20 + 4 * 10 + 9 * 30 + 4 * 30 + 5 * 20 = 670$ (единиц).

Все поставки распределены, количество базисных клеток равно 6 = m + n – 1, значит, план невырожденный (Таблица 2).

Таблица 2 – Метод северо-западного угла

Пункты	B1	B2	B3	B4	Запасы
A1	2 40	3 20	5	1	60
A2	3	4 10	9 30	4 30	70
A3	2	5	2	5 20	20
Потребности	40	30	30	50	150

1.4 Метод минимальной стоимости

Выбираем ячейку с минимальной стоимостью $c_{14} = 1$. Значение x_{14} определяется как минимальное из остатков запасов A_1 и потребностей потребителя B_4. Тогда $x_{14} = 50$. Мысленно вычёркиваем из таблицы столбец B_4, так как его запрос удовлетворён. Перейдём к ячейки a_{11} со стоимостью перевозки, равной 2. В рассматриваемую клетку запишем минимальное из остатков запасов A_1 и потребностей потребителя B_1. В данном случае запишем 10 – остаток запасов в пункте A_1, после чего мысленно вычеркнем строку A_1. Далее в ячейку a_{31} запишем минимум из потребностей B_1 (30) и запасов A_3 (20) - 20 и вычеркнем строку A_3. Следующая минимальная стоимость находится в ячейке a_{21}. Сюда запишем 10 и вычеркнем столбец B_1. В ячейку a_{22} запишем 30, так как потребность B_2 (30) меньше, чем запас A_2 (60). В последнюю оставшуюся ячейку a_{23} запишем 30 и зачеркнём соответствующий столбец B_3 и строку A_2.

Общая стоимость перевозок груза составляет:

$f_0 = 2 * 10 + 1 * 50 + 3 * 10 + 4 * 30 + 9 * 30 + 2 * 20 = 530$ (единиц).

Опорный план, составленный способом минимальной стоимости, более близок к оптимальному решению (Таблица 3).

Таблица 3 – Метод минимальной стоимости

Пункты	B1	B2	B3	B4	Запасы
A1	2 10	3	5	1 50	60
A2	3 10	4 30	9 30	4	70
A3	2 20	5	2	5	20
Потребности	40	30	30	50	150

1.5 Метод потенциалов

Для определения исходного плана перевозок воспользуемся методом северо-западного угла. Согласно уже проведённым расчётам, исходный план представлен в Таблице 4. Общее число базисных клеток: m + n – 1 = 3 + 4 – 1 = 6.

Таблица 4 – Метод северо-западного угла

Пункты	B1	B2	B3	B4	Запасы
A1	2 40	3 20	5	1	60
A2	3	4 10	9 30	4 30	70
A3	2	5	2	5 20	20
Потребности	40	30	30	50	150

Стоимость перевозок по этому плану:

$f_0 = 2 * 40 + 3 * 20 + 4 * 10 + 9 * 30 + 4 * 30 + 5 * 20 = 670$ (единиц).

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_2 + v_2 = 4; u_2 + v_3 = 9;$$

$$u_2 + v_4 = 4; u_3 + v_4 = 5;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; u_2 = 1; v_3 = 8; v_4 = 3; u_3 = 2$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{13} = c_{13} - (u_1 + v_3) = 5 - 0 - 8 = -3;$$

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 - 2 - 2 = -2;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 - 2 - 3 = 0;$$

$$\Delta_{33} = c_{33} - (u_3 + v_3) = 2 - 2 - 8 = -8;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{33} = -8$ для клетки (3,3).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 5). Одна из вершин цикла находится в незанятой клетке $(3{,}3)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(30{,}20) = 20$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 6.

Таблица 5 – Таблица перерасчёта

		v1=2	v2=3	v3=8	v4=3	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	2 40	3 20	5	1	60
u2=1	A2	3	4 10	- 9 30	+ 4 30	70
u3=2	A3	2	5	λ 2 +	- 5 20	20
	Потребности	40	30	30	50	150

Таблица 6 – Таблица нового плана

		v1=2	v2=3	v3=8	v4=3	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	2 40	3 20	5	1	60
u2=1	A2	3	4 10	9 10	4 50	70
u3=2	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Стоимость перевозок по этому плану:

$f_1 = f_0 + \Delta_{33}\lambda = 670 - 8 * 20 = 510$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_2 + v_2 = 4; u_2 + v_3 = 9;$$

$$u_2 + v_4 = 4; u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; \mathrm{u}_2 = 1; v_3 = 8; v_4 = 3; \mathrm{u}_3 = -6$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{13} = c_{13} - (u_1 + v_3) = 5 - 0 - 8 = -3;$$

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 6 - 2 = 6;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 6 - 3 = 8;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 6 - 3 = 8;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{13} = -3$ для клетки (1,3).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 7). Одна из вершин цикла находится в незанятой клетке $(1,3)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(10,20) = 10$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 8.

Таблица 7 – Таблица перерасчёта

	Пункты	v1=2 B1	v2=3 B2	v3=8 B3	v4=3 B4	Запасы
u1=0	A1	2 40	- 3 20	λ 5 +	1	60
u2=1	A2	3	+ 4 10	- 9 10	4 50	70
u3=-6	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Таблица 8 – Таблица нового плана

	Пункты	v1=2 B1	v2=3 B2	v3=8 B3	v4=3 B4	Запасы
u1=0	A1	2 40	3 10	5 10	1	60
u2=1	A2	3	4 20	9	4 50	70
u3=-6	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Стоимость перевозок по этому плану:

$f_2 = f_1 + \Delta_{13}\lambda = 510 - 3 * 10 = 480$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_2 = 3; u_1 + v_3 = 5; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4;\ u_3 + v_3 = 2.$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_2 = 3; v_3 = 5; u_2 = 1; v_4 = 3; v_5 = 2; u_3 = -3.$

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{14} = c_{14} - (u_1 + v_4) = 1 - 0 - 3 = -2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 1 - 2 = 0;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 1 - 5 = 3;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 2 = 3;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 3 = 5;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 3 = 5;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{14} = -2$ для клетки (1,4).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 9). Одна из вершин цикла находится в незанятой клетке $(1,4)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(10, 50) = 10$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 10.

Таблица 9 – Таблица перерасчёта

		v1=2	v2=3	v3=5	v4=3	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	2 40	- 3 10	5 10	λ 1 +	60
u2=1	A2	3	+ 4 20	9	- 4 50	70
u3=-3	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Таблица 10 – Таблица нового плана

		v1=2	v2=3	v3=5	v4=3	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	2 40	3	5 10	1 10	60
u2=1	A2	3	4 30	9	4 40	70
u3=-3	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Стоимость перевозок по этому плану:

$f_3 = f_2 + \Delta_{14}\lambda = 480 - 2 * 10 = 460$ (ед.)

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_1 = 2; u_1 + v_3 = 5; u_1 + v_4 = 1; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4;\ u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_1 = 2; v_3 = 5; v_4 = 1; u_2 = 3; v_2 = 1; u_3 = -3$.

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{12} = c_{12} - (u_1 + v_2) = 3 - 0 - 1 = 2;$$

$$\Delta_{21} = c_{21} - (u_2 + v_1) = 3 - 3 - 2 = -2;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 3 - 5 = 1;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 2 = 3;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 1 = 7;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 1 = 7;$$

Условие оптимальности плана перевозок $\Delta_{ij} \geq 0$ не выполняется, поэтому построим замкнутый цикл пересчета и определим величины перераспределения груза.

Минимальной оценкой является $\Delta_{21} = -2$ для клетки (2,1).

Для определения количества груза λ подлежащего распределению, построим замкнутый цикл (указан стрелками) (Таблица 11). Одна из вершин цикла находится в незанятой клетке $(2,1)$, которую отмечаем знаком « $+$ ». Все остальные вершины цикла находятся в базисных клетках, с чередующимися знаками « $-$ » и « $+$ ». Найдем $\lambda = \min(40{,}40) = 40$, равное наименьшему из чисел, стоящих в отрицательных вершинах цикла. Значение λ записываем в незанятую клетку. Двигаясь далее по означенному циклу, вычитаем λ из объемов перевозок, расположенных в клетках, которые обозначены знаком « $-$ », и прибавляем к объемам перевозок, находящихся в клетках, отмеченных знаком « $+$ ». Элементы таблицы, не входящие в цикл, остаются без изменений. Таблица нового плана представлена в Таблице 12.

Таблица 11 – Таблица перерасчёта

		v1=2	v2=1	v3=5	v4=1	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	- 2 40	3	5 10	+ 1 10	60
u2=3	A2	λ 3 +	4 30	9	- 4 40	70
u3=-3	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Таблица 12 – Таблица нового плана

		v1=2	v2=1	v3=5	v4=1	
	Пункты	B1	B2	B3	B4	Запасы
u1=0	A1	2	3	5 10	1 50	60
u2=3	A2	3 40	4 30	9	4 0	70
u3=-3	A3	2	5	2 20	5	20
	Потребности	40	30	30	50	150

Стоимость перевозок по этому плану:

$$f_4 = f_3 + \Delta_{21}\lambda = 460 - 2 * 40 = 380 \text{ (ед.)}$$

Вычислим потенциалы u_i и v_j исходя из базисных переменных. Для их нахождения используем условие $u_i + v_j = c_{ij}$.

$$u_1 + v_3 = 5; u_1 + v_4 = 1; u_2 + v_1 = 3; u_2 + v_2 = 4;$$

$$u_2 + v_4 = 4; u_3 + v_3 = 2;$$

Считая $u_1 = 0$, имеем $u_1 = 0; v_3 = 5; v_4 = 1; u_3 = -3; u_2 = 3; v_2 = 1; v_1 = 0;$

Для каждой свободной клетки вычислим относительные оценки:

$$\Delta_{11} = c_{11} - (u_1 + v_1) = 2 - 0 - 0 = 2;$$

$$\Delta_{12} = c_{12} - (u_1 + v_2) = 3 - 0 - 1 = 2;$$

$$\Delta_{23} = c_{23} - (u_2 + v_3) = 9 - 3 - 5 = 1;$$

$$\Delta_{31} = c_{31} - (u_3 + v_1) = 2 + 3 - 0 = 5;$$

$$\Delta_{32} = c_{32} - (u_3 + v_2) = 5 + 3 - 1 = 7;$$

$$\Delta_{34} = c_{34} - (u_3 + v_4) = 5 + 3 - 1 = 7;$$

Отрицательных оценок нет, значит решение $x_{13} = 10; x_{14} = 50; x_{21} = 40; x_{22} = 30; x_{33} = 20$ является оптимальным. Стоимость перевозок при этом составляет $f_0 = 380$ (ед.)

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы мной была изучена транспортная задача, её методы решения. Была решена конкретная транспортная задача, т.е. найден оптимальный план перевозок. Также была написана программа для решения транспортных задач.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Болотова Л. С. Многокритериальная оптимизация. Болотова Л. С., Сорокин А. Б. [Электронный ресурс] / Метод. указания по вып. курсовой работы — М.: МИРЭА, 2015.
2. Сорокин А. Б. Методы оптимизации: гибридные генетические алгоритмы. Сорокин А. Б. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2016.
3. Сорокин А. Б. Линейное программирование: практикум. Сорокин А. Б., Бражникова Е. В., Платонова О. В. [Электронный ресурс] / Учебно-метод. пособие — М.: МИРЭА, 2017.

ПРИЛОЖЕНИЯ

Приложение А – Код реализации транспортной задачи на языке Python.

Приложение А

Код реализации транспортной задачи на языке Python

Листинг А.1. Реализация транспортной задачи.

```
import csv
import sympy
import copy
import re
import math
import numpy as np
from itertools import product

def transport_task():
    '''Функция, отвечающая за решение закрытой транспортной задачи'''
    # поставщики, потребители, c = input_data()
    поставщики, потребители, c = input_data_from_file()
    assert sum(поставщики) == sum(
        потребители), 'Транспортная задача не является закрытой'
    C = np.vstack(c)  # Стоимости перевозок единицы груза из Ai в Bi
    X = np.zeros_like(C)
    basis = list()  # Базисные переменные (заполненные клетки)
    U = np.zeros_like(поставщики)  # Потенциалы пунктов Ai
    V = np.zeros_like(потребители)  # Потенциалы пунктов Bj
    delta = np.zeros_like(C)  # Относительные оценки клеток
    marks = list(product([f'u{i}' for i in range(
        1, len(поставщики) + 1)], [f'v{i}' for i in range(1, len(потребители)
+ 1)]))
    num_iteration = 0  # Номер итерации в методе потенциалов

    def print_table(potential=True):
        '''Функция для вывода таблицы'''
        print('-' * (14 * (len(потребители) + 2) + len(потребители) + 2))
        if potential:
            print(' ' * 14, end='|')
            for i in range(len(потребители)):
                print(f'v{i + 1} = {V[i]}'.ljust(14), end='|')
            print(''.ljust(14), end='|')
            print('\n' + ' ' * 14 + ('|' + '-' * 14)
                  * len(потребители), end='|')
            print(''.ljust(14), end='|')
            print()
        print('Пункты'.ljust(14), end='|')
        for i in range(len(потребители)):
            print(f'B{i + 1}'.ljust(14), end='|')
        print('Запасы'.ljust(14), end='|')
        for i in range(len(поставщики)):
            print('\n' + '-' * (14 * (len(потребители) + 2) +
len(потребители) + 2))
            if potential:
                print(f'u{i + 1} = {U[i]}'.ljust(14), end='|')
            else:
                print(f'A{i + 1}'.ljust(14), end='|')
            for j in range(len(потребители)):
                print(f'{C[i][j]}'.rjust(14), end='|')
            print('\033[94m' + f'{поставщики[i]
                              }'.ljust(14) + '\033[0m', end='|')
            if potential:
```

Продолжение Листинга А.1.

```
                print('\n' + f'A{i + 1}'.ljust(14), end='|')
            else:
                print('\n' + ' ' * 14, end='|')
            for j in range(len(потребители)):
                if (i, j) not in basis:
                    print(f''.ljust(14), end='|')
                else:
                    print('\033[102m' +
                          f'{X[i][j]}'.ljust(14) + '\033[0m', end='|')
            print(''.ljust(14), end='|')
        print('\n' + '-' * (14 * (len(потребители) + 2) + len(потребители) +
2))
        print('Потребности'.ljust(14), end='|')
        for i in range(len(потребители)):
            print('\033[94m' + f'{потребители[i]
                              }'.ljust(14) + '\033[0m', end='|')
        print('\033[95m' + f'{sum(потребители)
                              }'.ljust(14) + '\033[0m', end='|')
        print('\n' + '-' * (14 * (len(потребители) + 2) +
              len(потребители) + 2))

    def northwest_corner_method():
        '''Метод северо-западного угла нахождения начального опорного
решения'''
        for i in range(len(поставщики)):
            for j in range(len(потребители)):
                X[i][j] = min(поставщики[i] - np.sum(X[i, :]),
                              потребители[j] - np.sum(X[:, j]))
                if X[i][j] != 0:
                    basis.append((i, j))

    def get_min_indexes(suppliers, consumers):
        '''Вспомогательная функция для определения индексов ячейки с
минимальной стоимостью'''
        min_cost = np.inf  # хранит наименьший элемент, который нашли на
текущий момент в матрице C
        min_indexes = (None, None)  # индексы наименьшего элемента
        for i in range(len(C)):
            for j in range(len(C[0])):
                if min_cost > C[i][j] and C[i][j] > 0 and X[i][j] == 0:
                    if suppliers[i] > 0 and consumers[j] > 0:
                        # назначаем (i, j) элемент новым наименьшим
                        min_cost = C[i][j]
                        min_indexes = (i, j)
        return min_indexes

    def min_price_method():
        suppliers = np.copy(поставщики)  # Копия вектора поставщиков
        consumers = np.copy(потребители)  # Копия вектора потребителей
        '''Метод минимальной стоимости нахождения опорного решения'''
        while True:
            i, j = get_min_indexes(suppliers, consumers)
            if i is None and j is None:
                # все потребности удовлетворены и/или все возможности
использованы
                break
            resources = min(suppliers[i], consumers[j])
            suppliers[i] = suppliers[i] - resources
            consumers[j] = consumers[j] - resources
            X[i][j] = resources
            basis.append((i, j))
```

Продолжение Листинга А.1.

```
    def count_function():
        '''Расчёт стоимости перевозок по текущему плану'''
        return np.dot(C.reshape(len(поставщики) * len(потребители)),
X.reshape(len(поставщики) * len(потребители)))

    def count_potentials(calculation_output=False):
        ''' Функция для подсчёта потенциалов Ui, Vi'''
        system = []
        for u, v in marks:
            if (int(u[1]) - 1, int(v[1]) - 1) in basis:
                system.append(f'{u} + {v} = {C[int(u[1]) - 1][int(v[1]) -
1]}')
        if calculation_output:
            print("Вычислим потенциалы ui и vi, исходя из базисных
переменных. Для их нахождения используем условия ui + vj = cij")
            print(*system, sep='\n')
        first_equation = system[0]
        first_variable = first_equation.split()[0]
        variables = set()
        for equation in system:
            for symbol in equation.split():
                if re.fullmatch(r'[uv]\d*', symbol):
                    variables.add(symbol)
        symbols_dict = {symbol: sympy.symbols(symbol) for symbol in
variables}
        system[0] = first_equation.replace(first_variable, '0')
        equations = []
        for equation in system:
            left, right = equation.split('=')
            equations.append(
                sympy.Eq(sympy.sympify(left), sympy.sympify(right)))
        equations.append(sympy.Eq(symbols_dict[first_variable], 0))
        solution = sympy.solve(equations, list(symbols_dict.values()))
        for key, value in solution.items():
            if str(key)[0] == 'u':
                U[int(str(key)[1]) - 1] = value
            else:
                V[int(str(key)[1]) - 1] = value
        if calculation_output:
            print(f'Считая, что {first_variable} = 0, имеем:')
            print(*[f'{key} = {value}' for key,
                   value in solution.items()], sep='; ')

    def count_delta(calculation_output=False):
        nonlocal delta
        if calculation_output:
            print('Для каждой свободной клетки вычислим относительные
оценки:')
        delta = np.zeros_like(C)
        for i in range(len(поставщики)):
            for j in range(len(потребители)):
                if (i, j) not in basis:
                    delta[i][j] = C[i][j] - (U[i] + V[j])
                    if count_delta:
                        print(f'Δ{i + 1}{j + 1} = {delta[i][j]};')

    def recalculate_optimal_plan():
        '''Функкия перерасчёта оптимального плана'''
        min_i, min_j = math.inf, math.inf
        lowest_mark = math.inf
        for i in range(len(поставщики)):
```

```
            for j in range(len(потребители)):
                if delta[i][j] < lowest_mark:
                    lowest_mark = delta[i][j]
                    min_i = i
                    min_j = j
        basis.append((min_i, min_j))
        available_ways = copy.copy(basis)
        cycle = [(min_i, min_j)]
        curr_i = min_i
        curr_j = min_j
        while True:
            dead_end = True
            for i, j in available_ways:
                if i == curr_i and j != curr_j and ((i, j) not in cycle or
(i, j) == (min_i, min_j)):
                    if len(cycle) > 1 and cycle[-2][0] != i:
                        dead_end = False
                        curr_j = j
                        break
                    elif len(cycle) == 1:
                        dead_end = False
                        curr_j = j
                        break
                elif i != curr_i and j == curr_j and ((i, j) not in cycle or
(i, j) == (min_i, min_j)):
                    if len(cycle) > 1 and cycle[-2][1] != j:
                        dead_end = False
                        curr_i = i
                        break
                    elif len(cycle) == 1:
                        dead_end = False
                        curr_i = i
                        break
            if not dead_end:
                cycle.append((curr_i, curr_j))
            elif cycle[0] != cycle[-1]:
                del available_ways[available_ways.index((curr_i, curr_j))]
                curr_i = min_i
                curr_j = min_j
                cycle = [(min_i, min_j)]
            else:
                break
        lam = math.inf
        for index, point in enumerate(cycle[:-1]):
            i, j = point
            if index % 2 and X[i][j] < lam:
                lam = X[i][j]
        deleted_from_basis = 0
        for index, point in enumerate(cycle[:-1]):
            i, j = point
            if index % 2:
                X[i][j] -= lam
            else:
                X[i][j] += lam
            if X[i][j] == 0 and deleted_from_basis == 0:
                del basis[basis.index((i, j))]
                deleted_from_basis += 1

    print('Таблица с исходными данными:')
    print_table(False)
```

Продолжение Листинга А.1.

```
    min_price_method()
    print('Начальный опорный план, полученный методом минимальной
стоимости:')
    print_table()
    print(f'Стоимость перевозок по этому плану: {count_function()} единиц')
    X = np.zeros_like(C)  # Очитка массива X
    basis.clear()  # Очитка базиса
    northwest_corner_method()
    print('\033[101m' + 'Начальный опорный план, полученный методом северо-
западного угла:' + '\033[0m')
    print_table(False)
    print(f'Стоимость перевозок по этому плану: {count_function()} единиц')
    count_potentials(True)
    print_table()
    count_delta(True)
    while np.min(delta) < 0 and num_iteration < 20:
        print('Остались отрицательные оценки, произведём перерасчёт
плана...')
        num_iteration += 1
        print('\033[101m' + f'Итерация № {num_iteration}:' + '\033[0m')
        recalculate_optimal_plan()
        print_table()
        print(f'Стоимость перевозок по этому плану: {count_function()}
единиц')
        count_potentials(True)
        count_delta(True)
    if num_iteration == 20:
        print(f'За {num_iteration} не удалось найти оптимальное решение(')
    else:
        print('\033[101m' + 'Оптимальное решение найдено!' + '\033[0m')

def input_data():
    '''Ручной ввод данных'''
    поставщики = np.array(list(map(int, input(
        "Введите запас груза у каждого поставщика через пробел: ").split())))
    потребители = np.array(list(map(int, input(
        "Введите потребность груза у каждого потребителя через пробел:
").split())))
    c = list()  # Коэффициенты Cij
    for i in range(len(поставщики)):
        c.append(np.array(list(map(int, input(f"Введите стоимости перевозки
от {
                 i + 1}-ого поставщика к каждому потребителю через пробел:
").split()))))
    return поставщики, потребители, c

def input_data_from_file():
    '''Автоматический ввод данных из файла'''
    with open('TPR_PRACT7.csv', encoding='utf-8') as file:
        rows = csv.reader(file)
        поставщики = np.array(list(map(int, next(rows))))
        потребители = np.array(list(map(int, next(rows))))
        c = list()  # Коэффициенты Cij
        for row in rows:
            c.append(np.array(list(map(int, row))))
    return поставщики, потребители, c

transport_task()
```