МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ НЕПРЕРЫВНОГО И ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

доцент		Н. А. Волкова
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

Транспортная задача линейного программирования

по дисциплине: Прикладные модели оптимизации

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №	Z1431		М.Д.Быстров
	номер группы	подпись, дата	инициалы, фамилия

Студенческий билет № 2021/3572

Санкт-Петербург 2025

Оглавление

Задание

Цель работы: изучение методики решения транспортной задачи линейного программирования для определения оптимального плана перевозок.

4

Используя метод потенциалов, проверьте оптимальность указанного опорного решения

	B_1	B_2	B_3	B_4	a_i
A_1	11	5	4	2 **80**	80
A_2	1 **70**	4 **60**	5 **40**	9	170
A_3	9	8	7 **140**	10 **10**	150
b_j	70	60	180	90	400

Результат выполнения программы

Работа выполнена в виде программы на языке программирования Python. Ниже представлены шаги работы программы при решении заданного варианта.

Нумерация поставщиков и потребителей начинается с 0.

Запасы: [80, 170, 150]

Потребности: [70, 60, 180, 90]

Маршруты:

[[11. 5. 4. 2.]
 [1. 4. 5. 9.]
 [9. 8. 7. 10.]]

План отгрузок:

[[0.   0.   0. 80.]
 [70. 60. 40.   0.]
 [0.   0. 140. 10.]]

Начальный базис:

[[0, 3], [1, 0], [1, 1], [1, 2], [2, 2], [2, 3]]

Потенциалы поставщиков

{0: 2.0, 1: 8.0, 2: 10.0}

Потенциалы потребителей

{0: -7.0, 1: -4.0, 2: -3.0, 3: 0.0}

Оценки свободных ячеек:

[[16. 7. 5. 0.]
 [0. 0. 0. 1.]
 [6. 2. 0. 0.]]

Отрицательные оценки не найдены, решение оптимально.

Исходный код

Файл “lab2.py”

```
# 4
# Используя метод потенциалов, проверьте оптимальность указанного опорного решения

# Постановка задачи:
#       B_1 B_2 B_3 B_4 a_i
# A_1   11  5   4   2   80
# A_2   1   4   5   9   170
# A_3   9   8   7   10  150
# b_j   70  60  180 90  400

# Данное оптимальное решение: (!=0: базисные ячейки)
#       B_1 B_2 B_3 B_4 a_i
# A_1   0   0   0   80  80
# A_2   70  60  40  0   170
# A_3   0   0   140 10  150
# b_j   70  60  180 90  400

# критерий оптимальности dij:  d_ij=c_ij^*-c_ij
# c_ij - затраты (истинные тарифы)
# c_ij^* - расчётные затраты (косвенные тарифы)

import numpy as np;

# получить переменные базиса из плана отгрузок
def get_basis(shipments):

    basis = [];

    for i in range(shipments.shape[0]):
        for j in range(shipments.shape[1]):
            if shipments[i][j] != 0:
                basis.append([i, j]);

    return basis;

# получить потенциалы
def get_potentials(basis, paths):

    u = dict(); # потенциалы поставщика
    v = dict(); # потенциалы потребителя

    # кол-во u и v
    u_num = paths.shape[0];
    v_num = paths.shape[1];

    [prod, cons] = basis[0];
```

```
    path = paths[prod][cons];

    # для первого уравнения
    # берем коэффициент потребителя = 0
    v[cons] = float(0);
    u[prod] = float(path);

    # расчет потенциалов перебором
    while (u.__len__() != u_num
           or v.__len__() != v_num):
        for [prod, cons] in basis:
            path = paths[prod][cons];
            if prod in u and not(cons in v):
                v[cons] = float(path - u[prod]);
            elif cons in v and not(prod in u):
                u[prod] = float(path - v[cons]);

    u = dict(sorted(u.items()));
    v = dict(sorted(v.items()));

    return [u,v];


# получить ключи свободных ячеек
def get_free_keys(shipments):

    free_keys = list();

    for i in range(shipments.shape[0]):
        for j in range(shipments.shape[1]):
            if shipments[i][j] == 0:
                free_keys.append([i, j]);

    return free_keys;


# найти оценки свободных ячеек
def get_scores(keys, u, v, paths):

    scores = np.zeros(paths.shape)

    for [prod, cons] in keys:
        path = paths[prod, cons];
        scores[prod, cons] = path - (u[prod] + v[cons]);

    return scores;

producers_num = 3; # кол-во поставщиков
consumers_num = 4; # кол-во потребителей
```

```
stocks = [80, 170, 150]; # запасы
needs = [70, 60, 180, 90]; # потребности

# длины путей от поставщика к потребителю
paths = np.array(
    #B1  B2  B3  B4
    [
    [11,  5,  4,  2], #A1
    [ 1,  4,  5,  9], #A2
    [ 9,  8,  7, 10], #A3
    ],
    dtype = float);

# опорный план отрузок от поставщиков к потребителям
shipments = np.array(
    # B1   B2   B3   B4
    [
    [  0,   0,   0,  80], #A1
    [ 70,  60,  40,   0], #A2
    [  0,   0, 140,  10], #A3
    ],
    dtype = float);

basis = get_basis(shipments);

print("Запасы: " + str(stocks));
print("Потребности: " + str(needs));

print("Маршруты:");
print(paths);

print("План отгрузок:");
print(shipments);

print("Начальный базис: ");
print(basis);

# проверка на ограничение кол-ва базисных клеток
if (len(basis) != producers_num + consumers_num - 1):
    raise ValueError("Кол-во маршрутов должно быть равным"
     + str(producers_num + consumers_num - 1));

# рассчитаем потенциалы поставщиков и потребителей
[u, v] = get_potentials(basis, paths);

print("Потенциалы поставщиков");
print(u);

print("Потенциалы потребителей");
print(v);
```

```
# получим свободные ячейки
free_keys = get_free_keys(shipments);

# найдем оценки свободных ячеек
scores = get_scores(free_keys, u, v, paths);

print("Оценки свободных ячеек:");
print(scores);

# подсчет негативных оценок
negative_scores = [];

for [prod, cons] in free_keys:
    if scores[prod][cons] < 0:
        negative_scores.append(scores[prod][cons]);

if (len(negative_scores) > 0):
    print("Найдены негативные оценки, решение не оптимально");
else:
    print("Отрицательные оценки не найдены, решение оптимально");
```

Вывод

В ходе выполнения второй лабораторной работы создана программа на языке Python для проверки опорного решения транспортной задачи с помощью метода потенциалов.

Изучена методика решения транспортной задачи линейного программирования для определения оптимального плана перевозок.