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

[Оглавление 2](#_Toc187608759)

[Задание 3](#_Toc187608760)

[Результат выполнения программы 4](#_Toc187608761)

[Исходный код 5](#_Toc187608762)

[Вывод 9](#_Toc187608763)

# Задание

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

4

Используя метод потенциалов, проверьте оптимальность указанного опорного решения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | 11 | 5 | 4 | 2  **80** | 80 |
|  | 1  **70** | 4  **60** | 5  **40** | 9 | 170 |
|  | 9 | 8 | 7  **140** | 10  **10** | 150 |
|  | 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

# с\_ij - затраты (истинные тарифы)

# с\_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.