МIНIСТЕРСТВО ОСВIТИ I НАУКИ УКРАЇНИ

НАЦIОНАЛЬНИЙ ТЕХНIЧНИЙ УНIВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛIТЕХНIЧНИЙ IНСТИТУТ»

Кафедра прикладної математики

Звіт

із лабораторної роботи №6

з дисципліни «Алгоритми і системи комп’ютерної математики.

Математичні алгоритми»

на тему:

«Транспортна задача»

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-63 | *Старший викладач Бай Ю.П.* |
| *Артеменко Я.К.* |  |

Київ — 2019

# **ЗМІСТ**

[**ВСТУП** 3](#_Toc27604480)

[**2 ОСНОВНА ЧАСТИНА** 4](#_Toc27604481)

[**2.1 Завдання** 4](#_Toc27604482)

[**2.2 Описання методу** 4](#_Toc27604483)

[**2.3 Використані тест кейси** 6](#_Toc27604484)

[**3 ВИСНОВКИ** 7](#_Toc27604485)

[**ПЕРЕЛІК ПОСИЛАНЬ** 8](#_Toc27604486)

[**ДОДАТКИ** 9](#_Toc27604487)

[**Додаток А (код програми для алгоритму)** 9](#_Toc27604488)

[**Додаток Б (код програми для тестів)** 10](#_Toc27604489)

[**Додаток В (результати виконання програми)** 12](#_Toc27604490)

[**Додаток Г (підтвердження правильності роботи алгоритму)** 15](#_Toc27604491)

# **ВСТУП**

Мета лабораторної роботи:

Розробити програмне забезпечення для розв’язання транспортної задачі та провести аналіз задачі, що розв’язується, та методу її розв’язання на предмет виключних ситуацій, які можуть виникати під час застосування заданого методу до розв’язання поставленої задачі.

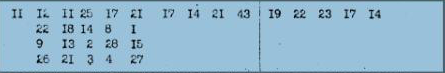
Методи лінійного програмування, одним з яких є транспортна задача, мають широке застосування в економіці. З їх допомогою можна оптимізувати завантаження устаткування, розподіл ресурсів або фахівців за завданнями, підвищувати ефективність використання фінансових ресурсів.

У розподільній логістиці транспортна задача застосовується для організації ефективного розподілу потоків товарів від декількох виробників до декільком споживачам.

# **2 ОСНОВНА ЧАСТИНА**

# **2.1 Завдання**

Розв’язати транспортну задачу для поставленого варіанту №11:



**Математична постановка даної задачі має наступний вигляд:**

**Балансування:**

**Обмеження виробництва:**

**Рівності для потреб замовників:**

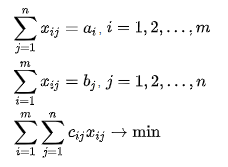
**Цільова функція:**

# **2.2 Описання методу**

Нехай є m пунктів виробництва деякого однорідного продукту та n пунктів його споживання. Для кожного пункту виробництва i = 1,2, ..., m і для кожного пункту споживання j = 1,2, ..., n задані наступні величини: обсяг виробництва ai в пункті виробництва i, обсяг споживання bj в пункті споживання j, витрати на перевезення одиниці продукту сij від пункту виробництва i до пункту споживання j. Передбачається, що сумарне виробництво дорівнює сумарному споживанню:



Потрібно скласти план перевезень, що дозволяє повністю вивезти продукти всіх виробників, повністю забезпечує потреби всіх споживачів і дає мінімум сумарних витрат на перевезення. Позначимо хij як обсяги перевезень від постачальника i до споживача j.



Метод найменшого елемента:

Одним із способів вирішення завдання є метод мінімального (найменшого) елемента. Його суть полягає в зведенні до мінімуму побічних перерозподілів товарів між споживачами.

Алгоритм:

1. З таблиці вартостей вибирають найменшу вартість і в клітинку, яка їй відповідає, вписують більше з чисел.
2. Перевіряються рядки постачальників на наявність рядки з витраченими запасами і стовпці споживачів на наявність стовпчика, потреби якого повністю задоволені. Такі стовпці і рядки далі не розглядаються.
3. Якщо не всі споживачі задоволені і не всі постачальники витратили товари, повернення до п. 1, в іншому випадку задача вирішена.

# **2.3 Використані тест кейси**

Для заданого алгоритму було розроблено ряд тест кейсів. Задача вирішується для трьох варіантів вхідних даних, серед яких знаходиться варіант, який потрібно вирішити. Також на розв’язання поступають масиви з числами, які згенеровано за допомогою генератора випадкових чисел та які також перевіряються на наявність літер, тобто неправильного вводу.

Додаткові варіанти для перевірки:

a1 = [12, 17, 18, 13]  
b1 = [10, 8, 12, 14, 16]  
C1 = [[6, 11, 20, 17, 8],  
 [1, 25, 3, 18, 17],  
 [9, 39, 16, 30, 31],  
 [23, 15, 4, 3, 28]]

a2 = [14, 14, 14, 18]  
b2 = [12, 12, 12, 12, 12]  
C2 = [[29, 4, 7, 6, 16],  
 [21, 13, 25, 21, 7],  
 [20, 10, 12, 6, 2],  
 [17, 7, 4, 6, 19]]

При виконанні тест кейсів, якщо дані введено правильно, то програма виводить результат, де варіанти вхідних даних підписано (чи це дані для прикладу, чи дані для поставленої задачі) та виводиться результат підрахунку.

Для даних, що використовуються для прикладу, виконується функція для порівняння даних:

self.assertEqual

Якщо під час виконання тесту була виявлена помилка, то біля вхідних даних виводиться повідомлення про помилку.

# **3 ВИСНОВКИ**

В даній лабораторній роботі було розроблено програмне забезпечення для розв’язання транспортної задачі. Також було розроблено ряд тестів, які перевіряють наявність помилок і виконують певні дії з помилками, які можуть виникати при вводі даних, а також, якщо дані введено правильно, то виводиться результат обчислення.

# **ПЕРЕЛІК ПОСИЛАНЬ**

1. <https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B0>
2. <https://www.coin-or.org/PuLP/CaseStudies/a_transportation_problem.html>
3. <https://towardsdatascience.com/linear-programming-and-discrete-optimization-with-python-using-pulp-449f3c5f6e99>
4. <http://matecos.ru/mat/matematika/kak-reshit-transportnuyu-zadachu-2.html>
5. <https://math.semestr.ru/transp/index.php>

# **ДОДАТКИ**

# **Додаток А (код програми для алгоритму)**

from pulp import \*  
  
def main(a, b, C):  
  
 x1 = pulp.LpVariable("x1", lowBound=0)  
 x2 = pulp.LpVariable("x2", lowBound=0)  
 x3 = pulp.LpVariable("x3", lowBound=0)  
 x4 = pulp.LpVariable("x4", lowBound=0)  
 x5 = pulp.LpVariable("x5", lowBound=0)  
 x6 = pulp.LpVariable("x6", lowBound=0)  
 x7 = pulp.LpVariable("x7", lowBound=0)  
 x8 = pulp.LpVariable("x8", lowBound=0)  
 x9 = pulp.LpVariable("x9", lowBound=0)  
 x10 = pulp.LpVariable("x10", lowBound=0)  
 x11 = pulp.LpVariable("x11", lowBound=0)  
 x12 = pulp.LpVariable("x12", lowBound=0)  
 x13 = pulp.LpVariable("x13", lowBound=0)  
 x14 = pulp.LpVariable("x14", lowBound=0)  
 x15 = pulp.LpVariable("x15", lowBound=0)  
 x16 = pulp.LpVariable("x16", lowBound=0)  
 x17 = pulp.LpVariable("x17", lowBound=0)  
 x18 = pulp.LpVariable("x18", lowBound=0)  
 x19 = pulp.LpVariable("x19", lowBound=0)  
 x20 = pulp.LpVariable("x20", lowBound=0)  
 problem = pulp.LpProblem('0', pulp.LpMaximize)  
 problem += -C[0][0] \* x1 - C[0][1] \* x2 - C[0][2] \* x3 - C[0][3] \* x4 - C[0][4] \* x5 \  
 -C[1][0] \* x6 - C[1][1] \* x7 - C[1][2] \* x8 - C[1][3] \* x9 - C[1][4] \* x10 \  
 -C[2][0] \* x11 - C[2][1] \* x12 - C[2][2] \* x13 - C[2][3] \* x14 - C[2][4] \* x15 \  
 -C[3][0] \* x16 - C[3][1] \* x17 - C[3][2] \* x18 - C[3][3] \* x19 - C[3][4] \* x20 , "Функция цели"  
 problem += x1 + x2 + x3 + x4 + x5<= a[0], "1"  
 problem += x6 + x7 + x8 + x9 + x10<= a[1], "2"  
 problem += x11 + x12 + x13 + x14 + x15 <= a[2], "3"  
 problem += x16 + x17 + x18 + x19 + x20 <= a[3], "4"  
 problem += x1 + x6 + x11 + x16 == b[0], "5"  
 problem += x2 + x7 + x12 + x17 == b[1], "6"  
 problem += x3 + x8 + x13 + x18 == b[2], "7"  
 problem += x4 + x9 + x14 + x19 == b[3], "8"  
 problem += x5 + x10 + x15 + x20 == b[4], "9"  
 problem.solve()  
 result = {}  
 result\_unsort = {}  
 for variable in problem.variables():  
 result\_unsort[variable.name] = variable.varValue  
 for i in sorted(result\_unsort.keys(), key= lambda x: int(x[1:])):  
 result[i] = result\_unsort[i]  
  
 for i in result.keys():  
 print(result[i], end=" ")  
  
 cost = abs(value(problem.objective))  
 print("Стоимость: ", cost)  
 return result, cost

# **Додаток Б (код програми для тестів)**

import unittest  
from transport import main  
import random  
  
  
  
a = [random.randint(-10, 10) for i in range(20)]  
a.append("A")  
A\_rand1 = [[random.choice(a) for i in range(5)] for j in range(4)]  
A\_rand2 = [[random.choice(a) for i in range(5)] for j in range(4)]  
A\_rand3 = [[random.choice(a) for i in range(5)] for j in range(4)]  
A\_rand4 = [[random.choice(a) for i in range(5)] for j in range(4)]  
  
  
a = [17, 14, 21, 43]  
b = [19, 22, 23, 17, 14]  
C = [[12, 11, 25, 17, 21],  
 [22, 18, 14, 8, 1],  
 [9, 13, 2, 28, 15],  
 [26, 21, 3, 4, 27]]  
res = 598.0  
  
  
a1 = [12, 17, 18, 13]  
b1 = [10, 8, 12, 14, 16]  
C1 = [[6, 11, 20, 17, 8],  
 [1, 25, 3, 18, 17],  
 [9, 39, 16, 30, 31],  
 [23, 15, 4, 3, 28]]  
res1 = ({'x1': 0.0, 'x2': 8.0, 'x3': 0.0, 'x4': 0.0, 'x5': 4.0,  
 'x6': 0.0, 'x7': 0.0, 'x8': 5.0, 'x9': 0.0, 'x10': 12.0,  
 'x11': 10.0, 'x12': 0.0, 'x13': 7.0, 'x14': 1.0, 'x15': 0.0,  
 'x16': 0.0, 'x17': 0.0, 'x18': 0.0, 'x19': 13.0, 'x20': 0.0},  
 610.0)  
  
  
a2 = [14, 14, 14, 18]  
b2 = [12, 12, 12, 12, 12]  
C2 = [[29, 4, 7, 6, 16],  
 [21, 13, 25, 21, 7],  
 [20, 10, 12, 6, 2],  
 [17, 7, 4, 6, 19]]  
  
res2 = ({'x1': 0.0, 'x2': 12.0, 'x3': 0.0, 'x4': 2.0, 'x5': 0.0,  
 'x6': 12.0, 'x7': 0.0, 'x8': 0.0, 'x9': 0.0, 'x10': 2.0,  
 'x11': 0.0, 'x12': 0.0, 'x13': 0.0, 'x14': 4.0, 'x15': 10.0,  
 'x16': 0.0, 'x17': 0.0, 'x18': 12.0, 'x19': 6.0,'x20': 0.0}, 454.0)  
  
  
class main\_test(unittest.TestCase):  
  
 def test\_main(self):  
 try:  
 print("УСЛОВИЕ ЗАДАНИЯ:\n[{},\n{},\n{}]\n".format(a, b, C))  
 x = main(a, b, C)  
 #self.assertEqual(main(a, b, C), res)  
 except:  
 pass  
  
 try:  
 print("\nУСЛОВИЕ ДЛЯ ПРИМЕРА №1:\n[{},\n{},\n{}]\n".format(a1, b1, C1))  
 x = main(a1, b1, C1)  
 self.assertEqual(main(a1, b1, C1), res1)  
 except:  
 pass  
  
 try:  
 print("\nУСЛОВИЕ ДЛЯ ПРИМЕРА №2:\n[{},\n{},\n{}]\n".format(a2, b2, C2))  
 x = main(a2, b2, C2)  
 self.assertEqual(main(a2, b2, C2), res2)  
 except:  
 pass  
  
  
 try:  
 print("\nГенератор рандомных чисел(ПРИМЕР 1):\n[{},\n{},\n{}]\n".format(a, b, A\_rand1))  
 x = main(a, b, A\_rand1)  
 except TypeError:  
 print("НЕ ПОДХОДЯЩИЕ ДАННЫЕ")  
  
 try:  
 print("\nГенератор рандомных чисел(ПРИМЕР 2):\n[{},\n{},\n{}]\n".format(a, b, A\_rand2))  
 x = main(a, b, A\_rand2)  
 except TypeError:  
 print("НЕ ПОДХОДЯЩИЕ ДАННЫЕ")  
  
 try:  
 print("\nГенератор рандомных чисел(ПРИМЕР 3):\n[{},\n{},\n{}]\n".format(a, b, A\_rand3))  
 x = main(a, b, A\_rand3)  
 except TypeError:  
 print("НЕ ПОДХОДЯЩИЕ ДАННЫЕ")  
  
 try:  
 print("\nГенератор рандомных чисел(ПРИМЕР 4):\n[{},\n{},\n{}]\n".format(a, b, A\_rand4))  
 x = main(a, b, A\_rand4)  
 except TypeError:  
 print("НЕ ПОДХОДЯЩИЕ ДАННЫЕ")  
  
  
if \_\_name\_\_=="\_\_main\_\_":  
 unittest.main()

# **Додаток В (результати виконання програми)**

Testing started at 23:12 ...

"E:\учеба\4 курс\Автоматизация\6лаб\lab6\env\Scripts\python.exe" "C:\Program Files\JetBrains\PyCharm 2019.1.3\helpers\pycharm\\_jb\_unittest\_runner.py" --target transport\_test.main\_test.test\_main

Launching unittests with arguments python -m unittest transport\_test.main\_test.test\_main in E:\учеба\4 курс\Автоматизация\6лаб\lab6

Ran 1 test in 0.194s

OK

Process finished with exit code 0

УСЛОВИЕ ЗАДАНИЯ:

[[17, 14, 21, 43],

[19, 22, 23, 17, 14],

[[12, 11, 25, 17, 21], [22, 18, 14, 8, 1], [9, 13, 2, 28, 15], [26, 21, 3, 4, 27]]]

0.0 17.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 14.0 19.0 2.0 0.0 0.0 0.0 0.0 3.0 23.0 17.0 0.0 Стоимость: 598.0

УСЛОВИЕ ДЛЯ ПРИМЕРА №1:

[[12, 17, 18, 13],

[10, 8, 12, 14, 16],

[[6, 11, 20, 17, 8], [1, 25, 3, 18, 17], [9, 39, 16, 30, 31], [23, 15, 4, 3, 28]]]

0.0 8.0 0.0 0.0 4.0 0.0 0.0 5.0 0.0 12.0 10.0 0.0 7.0 1.0 0.0 0.0 0.0 0.0 13.0 0.0 Стоимость: 610.0

0.0 8.0 0.0 0.0 4.0 0.0 0.0 5.0 0.0 12.0 10.0 0.0 7.0 1.0 0.0 0.0 0.0 0.0 13.0 0.0 Стоимость: 610.0

УСЛОВИЕ ДЛЯ ПРИМЕРА №2:

[[14, 14, 14, 18],

[12, 12, 12, 12, 12],

[[29, 4, 7, 6, 16], [21, 13, 25, 21, 7], [20, 10, 12, 6, 2], [17, 7, 4, 6, 19]]]

0.0 12.0 0.0 2.0 0.0 12.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0 4.0 10.0 0.0 0.0 12.0 6.0 0.0 Стоимость: 454.0

0.0 12.0 0.0 2.0 0.0 12.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0 4.0 10.0 0.0 0.0 12.0 6.0 0.0 Стоимость: 454.0

Генератор рандомных чисел(ПРИМЕР 1):

[[17, 14, 21, 43],

[19, 22, 23, 17, 14],

[[7, -1, 'A', 7, 8], [9, -7, -7, 1, 8], [-7, -7, 9, -7, -7], [-1, 7, -7, 7, -7]]]

НЕ ПОДХОДЯЩИЕ ДАННЫЕ

Генератор рандомных чисел(ПРИМЕР 2):

[[17, 14, 21, 43],

[19, 22, 23, 17, 14],

[[1, 7, -7, -4, 1], [-7, 1, 6, -4, -10], [-7, 1, -7, -9, 7], [-10, -1, 8, -4, 7]]]

0.0 0.0 17.0 0.0 0.0 0.0 0.0 0.0 0.0 14.0 0.0 0.0 6.0 15.0 0.0 19.0 22.0 0.0 2.0 0.0 Стоимость: 656.0

Генератор рандомных чисел(ПРИМЕР 3):

[[17, 14, 21, 43],

[19, 22, 23, 17, 14],

[[-7, -7, 1, 9, -4], [-10, -10, 'A', -9, 8], [-7, -1, 1, 9, 'A'], [-7, 9, 7, -7, -4]]]

НЕ ПОДХОДЯЩИЕ ДАННЫЕ

Генератор рандомных чисел(ПРИМЕР 4):

[[17, 14, 21, 43],

[19, 22, 23, 17, 14],

[[-7, 7, -7, 'A', -7], [8, 8, -7, -1, -7], [7, -4, 'A', -7, -1], [8, 1, -7, 7, 1]]]

НЕ ПОДХОДЯЩИЕ ДАННЫЕ

# **Додаток Г (підтвердження правильності роботи алгоритму)**

1. Для прикладу №1:

Результат виконання програми:

УСЛОВИЕ ДЛЯ ПРИМЕРА №1:

[[12, 17, 18, 13],

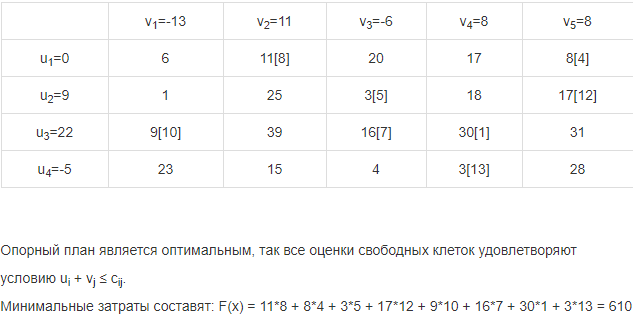
[10, 8, 12, 14, 16],

[[6, 11, 20, 17, 8], [1, 25, 3, 18, 17], [9, 39, 16, 30, 31], [23, 15, 4, 3, 28]]]

0.0 8.0 0.0 0.0 4.0 0.0 0.0 5.0 0.0 12.0 10.0 0.0 7.0 1.0 0.0 0.0 0.0 0.0 13.0 0.0

Стоимость: 610.0

Результат з онлайн джерела:



1. Для прикладу №2:

Результат виконання програми:

УСЛОВИЕ ДЛЯ ПРИМЕРА №2:

[[14, 14, 14, 18],

[12, 12, 12, 12, 12],

[[29, 4, 7, 6, 16], [21, 13, 25, 21, 7], [20, 10, 12, 6, 2], [17, 7, 4, 6, 19]]]

0.0 12.0 0.0 2.0 0.0 12.0 0.0 0.0 0.0 2.0 0.0 0.0 0.0 4.0 10.0 0.0 0.0 12.0 6.0 0.0

Стоимость: 454.0

Результат з онлайн джерела:

