**Федеральное государственное образовательное бюджетное**

**учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**(Финансовый университет)**

**Факультет**

информационных технологий и анализа больших данных

Кафедра «Бизнес-информатика»

**Домашнее задание № 1**

«Решение задачи на поиск кратчайшего пути»

Студенты группы БИ20-4:

Иванова Ксения

Киракосян Виген

Крылов Никита

Мытарева Ангелина

Петрова Арина

Чайковская Анна

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

**ОГЛАВЛЕНИЕ**

[1.ПОСТАНОВКА ЗАДАЧИ (ФИЗИЧЕСКАЯ МОДЕЛЬ) 3](#_Toc97254203)

[2. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 3](#_Toc97254204)

[3.АЛГОРИТМЫ 5](#_Toc97254205)

[4.ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 14](#_Toc97254206)

[5.АРХИТЕКТУРА РЕШЕНИЯ 18](#_Toc97254207)

[6. ТЕСТИРОВАНИЕ 22](#_Toc97254208)

[7.ЗАКЛЮЧЕНИЕ 24](#_Toc97254209)

## ПОСТАНОВКА ЗАДАЧИ (ФИЗИЧЕСКАЯ МОДЕЛЬ)

Компания «Сити такси», занимающаяся пассажирскими перевозками, хочет создать собственное навигационное приложение, составляющее кратчайший маршрут из начальной точки (старт маршрута, заданного клиентом) до конечного адреса. Всего перекрёстков 6. В матрице представлено время, которое занимает водитель при передвижении по участку дороги между перекрёстками.

1. Таблица 1 – «Матрица весовых коэффициентов»

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Wij | A | 1 | 2 | 3 | 4 | 5 | 6 | B |
| A |  |  |  |  |  |  |  |  |
| 1 |  |  | 3 | 5 | 4 |  |  |  |
| 2 |  | 3 |  | 3 |  |  | 1 |  |
| 3 |  | 5 | 3 |  | 1 |  | 2 |  |
| 4 |  | 4 |  | 1 |  | 7 |  |  |
| 5 |  |  |  |  | 7 |  |  |  |
| 6 |  |  | 1 | 2 |  |  |  |  |
| B |  |  |  |  |  |  |  |  |

## 2. МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

В представленном случае нам необходимо решить задачу о поиске кратчайшего пути на графе.

Представим дорожную сеть в виде ориентировочного графа. В качестве вершин возьмём перекрёстки, в качестве дуг – дороги между ними. Вес ребра равен времени проезда по участку дороги. Необходимо построить кратчайший по времени маршрут из пункта отправления A в пункт назначения B.

Для решения данной задачи используем алгоритм Дейкстры:

1. Строим все возможные маршруты из начальной точки A в ближайшие точки и находим длительности пути. Точку A отмечаем, как посещённую;
2. Из первой посещённой вершины строим возможные пути перемещения в соседние вершины и суммируем время движения. Рассмотренные далее вершины отмечаем, как посещённые;
3. Аналогично этому последовательно движемся по вершинам графа;
4. Если при перемещении по вершинам мы приходим в вершину, которую уже посещали, то необходимо записать наименьшее из времён, которое было получено при последовательном прохождении вершин до текущей;
5. Попав в пункт назначения B, завершаем алгоритм, так как закончились те вершины, в которые был осуществлён вход, но не был осуществлён выход.

Для нахождения оптимального маршрута в графе необходимо совершить обратный проход по алгоритму, двигаясь из пункта B каждый раз в сторону вершины с наименьшим значением суммарного времени.

**Математическая постановка задачи.**

**Исходные данные**

Пусть граф G = (V, A), где V – множество вершин графа, A – множество рёбер графа.

Вес рёбер графа Wij задаётся матрицей смежности:

W=

где n – количество вершин графа.

Необходимо найти, по каким рёбрам графа Xij проходит кратчайший маршрут из вершины A в вершину B.

**Переменные**

Переменные представляют собой матрицу искомого маршрута, которая состоит только из 0 и 1:

1 – это рёбра, которые входят в кратчайший маршрут, в остальных случаях – 0.

Целевая функция

Целевая функция представляет собой минимальную суммарную длину пути:

**Ограничения**

Маршрут должен начинаться в пункте A и заканчиваться в пункте B:

Маршрут не должен возвращаться в пункт A и продолжаться дальше пункта B:

Ограничение на связность маршрута – вход в вершину всегда должен соответствовать выходу из неё (сумма по строке i равна сумме по столбцу j), кроме начала A и конца B маршрута:

Переходить можно только по существующим рёбрам (Wij > 0):

Ограничение на бинарность переменных в матрице искомого маршрута:

## 3. АЛГОРИТМЫ

Перед Вами представлено программное решение технического задания. Для этого были разработаны несколько алгоритмов, впоследствии чего каждый из них воплотил математическую модель в виде конечного продукта. Чтобы сразу получить корректный ответ следуйте указаниям, соответствующим выбранному алгоритму. Для этого сначала запустите код в выбранной Вами программе.

**3.1 АЛГОРИТМ РЕШЕНИЯ ПРОИЗВОДСТВЕННОЙ ЗАДАЧИ /ТИП ЗАДАЧИ/ МЕТОДОМ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ**

**3.1.1 ОПИСАНИЕ ВХОДНЫХ ДАННЫХ**

1. Формат входных данных:
   1. число вершин (перекрёстков):

После запуска у вас появится строка «Количество вершин:». Введите показатель в числовом виде, в формате целого неотрицательного числа без пробелов.

1. Матрица весовых коэффициентов:

Три варианта ввода:

* ввод с клавиатуры (для малого количества городов):

После ввода количества вершин появится пользовательский интерфейс (весовая матрица, в которую пользователь заносит значение времени, которое занимает проезд по участку дороги между перекрёстками);

* случайная генерация:

После запуска программа генерирует случайные показатели времени, используя функцию …;

* считывание из CSV файла. Формат CSV: n - строк и столбцов, в ячейках записаны весовые коэффициенты ребер графа:

После запуска .

1. Данные матрицы весовых коэффициентов:
2. После ввода материалов появится поле «Расход материала на товар:». Введите показатель в числовом виде.

2.1 Если является целым неотрицательным – введите без пробелов (Например: 22.0 или 22)

3.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: 2.2 или 0.4)

1. Данные по цене

После вывода матрицы будет предложено указать цену на указанное количество товаров. Введите показатель в числовом виде, предварительно поставив перед ним знак «минус».

 3.1 Если является целым неотрицательным– введите без пробелов (Например: -22.0 или -22)

3.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: -2.2)

1. Вектор ограничений по количеству материалов на складах

При корректном вводе данных о цене, появится поле для ввода данных о количестве материалов на складе.

Введите показатель в числовом виде.

4.1 Если является целым неотрицательны– введите без пробелов (Например: 22.0 или 22)

4.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: 2.2)

**3.1.2 ОПИСАНИЕ АЛГОРИТМА РЕШЕНИЯ**

При запуске кода на входе от пользователя требуется виды товаров и материалов в виде целочисленных неотрицательных показателей. Это станет и размерностью матрицы

Далее составляется пустой список с помощью пустых квадратных скобок. При корректно веденных ранее данных программа запросит значения необходимые для составления матрицы (целое или десятичное неотрицательное число, для этого использовался тип float).  Их (значений) количество будет зависеть от введенного ранее размера1. По окончании программа составит матрицу и выведет её в соответствии количеством строк и столбцов. Для того, чтобы значения в ней были представлены не в строчку через запятую, а в привычном виде используем функцию «np.array()». Для того, чтобы привести к необходимому виду2 транспонируем матрицу с помощью функции «.transpose()». Далее выводим результат через print.

Следующим шагом от пользователя на входе требуется внести цену товаров с минусом (целое или десятичное число, для этого использовался тип float). Это делается для корректности ввода и последующей обработки данных, поскольку для вычисления методом линейного программирования (linprog()) используется библиотека scipy. Optimize, при  определение максимума (по умолчанию определяется минимум коэффициенты целевой функции). В конце будут выведены значения цены в виде списка. Для этого ранее был создан пустой список price с помощью пустых квадратных скобок, в который уже благодаря функции «.append()» были внесены показатели с клавиатуры пользователя, по количеству не превышающему введенному ранее значению, соответствующему «вид товара»/переменной a (for i in range(a))

Переходим к ограничениям. Ими является количество материалов, хранящихся на складе, поскольку значения, как при составлении матрицы по расходам и списка цены, может являться числом дробным и неотрицательным, используется тип данных float. В начале также создаем пустой список под названием «quantity», где количество значений будет равняться указанному числу видов материалов/переменной b. После получим формат характерный для предыдущих данных (по цене).

Следующим шагом приступаем к формированию функции. Для этого создаем её с помощью def и называем как function. После чего инициируем пустой список под названием «f» и вводим переменную. Далее создаем список func1, где с помощью инициируемой функции, поступающие значения будут вычисляться по модулю, поскольку, как было написано ранее, они записываются со знаком «минус», после к ним добавляться значение переменной х, которая представляет собой искомые объёмы производимых изделий каждого вида обеспечивающие максимальную прибыль. Чтобы прослеживалось изменение переменных (х1,x2,x3 и т.д.) используем запись x=x+1. Заполняем пустой список «func» с помощью «.append». Так как у нас содержатся ещё и числовые объекты, которые нельзя совмещать со строками при интерации методом «join()», мы преобразовываем все значения у func1 в строку с помощью функций «map» и «str», где разделителем будет пробел При выводе благодаря совместному использованию функции «"+".join» и «map» со «str» также объединяем и преобразовываем значения, приводя их к единому типу, только разделителем между ними будет являться не проблем, а знак сложения – «+».

Последним действием является нахождение оптимального значения посредством использования линейного программирования. Для этого выбираем необходимое вводимое. Далее с помощью функции linprog() устанавливаем границы с помощью указанных ранее переменных и ищем максимальное значение с помощью метода «interior-point». По итогу на выход получаем оптимальное значение целевой функции, где с помощью функции «o.fun» выводим только нужный по условию найденный показатель по модулю с помощью функции «abs», поскольку присутствует отрицательное число. На следующей строке программа выведет значение переменной при которой выводится оптимальный показатель при вычислении функции linprog() с использованием метода похожего по работе на симплекс - interior-point. Стоит отметить, чтобы все вычисления метода были в одной функции, используется def value

**3.1.3 ОПИСАНИЕ ВЫХОДНЫХ ДАННЫХ**

В конце программа рассчитает оптимальное значение целевой функции, дополнительно указав значение х. Ответ будет представлен в формате:

оптимальное значение целевой функции (): /значение/

При х: /значение/

**3.2 АЛГОРИТМ РЕШЕНИЯ ПРОИЗВОДСТВЕННОЙ ЗАДАЧИ /ТИП ЗАДАЧИ/ СИМПЛЕКС МЕТОДОМ**

**3.2.1 ОПИСАНИЕ ВХОДНЫХ ДАННЫХ**

1. Определите размерность матрицы входных данных
2. После запуска у вас появится строка «Видов товаров:». Введите показатель в числовом виде, в формате целого неотрицательного числа без пробелов (Например: 22)
3. В строке «Видов материалов:». Введите показатель в числовом виде, в формате целого неотрицательного числа без пробелов (Например: 22)
4. Данные матрицы
5. После ввода материалов появится поле «Расход материала на товар:». Введите показатель в числовом виде.

2.1 Если является целым неотрицательным – введите без пробелов (Например: 22.0 или 22)

3.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: 2.2 или 0.4)

1. Данные по цене

После вывода матрицы будет предложено указать цену на указанное количество товаров. Введите показатель в числовом виде, предварительно поставив перед ним знак «минус».

 3.1 Если является целым неотрицательным– введите без пробелов (Например: -22.0 или -22)

3.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: -2.2)

1. Вектор ограничений по количеству материалов на складах

При корректном вводе данных о цене, появится поле для ввода данных о количестве материалов на складе.

Введите показатель в числовом виде.

4.1 Если является целым неотрицательны– введите без пробелов (Например: 22.0 или 22)

4.2 Если является неотрицательным и имеет плавающую запятую – введите с разделителем-точкой без пробелов (Например: 2.2)

**3.2.2 ОПИСАНИЕ АЛГОРИТМА РЕШЕНИЯ**

При запуске кода на входе от пользователя требуется виды товаров и материалов в виде целочисленных неотрицательных показателей. Это станет и размерностью матрицы

Далее составляется пустой список с помощью пустых квадратных скобок. При корректно веденных ранее данных программа запросит значения необходимые для составления матрицы (целое или десятичное неотрицательное число, для этого использовался тип float).  Их (значений) количество будет зависеть от введенного ранее размера. По окончании программа составит матрицу и выведет её в соответствии количеством строк и столбцов. Для того, чтобы значения в ней были представлены не в строчку через запятую, а в привычном виде используем функцию «np.array()». Для того, чтобы привести к необходимому виду транспонируем матрицу с помощью функции «.transpose()». Далее выводим результат через print.

Следующим шагом от пользователя на входе требуется внести цену товаров с минусом (целое или десятичное число, для этого использовался тип float). Это делается для корректности ввода и последующей обработки данных, поскольку для вычисления методом линейного программирования (linprog()) используется библиотека scipy. Optimize, при определение максимума (по умолчанию определяется минимум коэффициенты целевой функции). В конце будут выведены значения цены в виде списка. Для этого ранее был создан пустой список price с помощью пустых квадратных скобок, в который уже благодаря функции «.append()» были внесены показатели с клавиатуры пользователя, по количеству не превышающему введенному ранее значению, соответствующему «вид товара»/переменной a (for i in range(a))

Переходим к ограничениям. Ими является количество материалов, хранящихся на складе, поскольку значения, как при составлении матрицы по расходам и списка цены, может являться числом дробным и неотрицательным, используется тип данных float. В начале также создаем пустой список под названием «quantity», где количество значений будет равняться указанному числу видов материалов/переменной b. После получим формат характерный для предыдущих данных (по цене).

Следующим шагом приступаем к формированию функции. Для этого создаем её с помощью def и называем как function. После чего инициируем пустой список под названием «f» и вводим переменную. Далее создаем список func1, где с помощью инициируемой функции, поступающие значения будут вычисляться по модулю, поскольку, как было написано ранее, они записываются со знаком «минус», после к ним добавляться значение переменной х, которая представляет собой искомые объёмы производимых изделий каждого вида, обеспечивающие максимальную прибыль. Чтобы прослеживалось изменение переменных (х1,x2,x3 и т.д.) используем запись x=x+1. Заполняем пустой список «func» с помощью «.append». Так как у нас содержатся ещё и числовые объекты, которые нельзя совмещать со строками при интерации методом «join()», мы преобразовываем все значения у func1 в строку с помощью функций «map» и «str», где разделителем будет пробел При выводе благодаря совместному использованию функции «"+".join» и «map» со «str» также объединяем и преобразовываем значения, приводя их к единому типу, только разделителем между ними будет являться не проблем, а знак сложения – «+».

Последним действием является нахождение оптимального значения посредством использования симплекс метода. Для этого выбираем необходимое вводимое (в данном случае 1). Далее с помощью функции linprog() устанавливаем границы с помощью указанных ранее переменных и ищем минимальное значение с помощью метода «simplex». По итогу на выход получаем оптимальное значение целевой функции, где с помощью функции «o.fun» выводим только нужный по условию найденный показатель по модулю с помощью функции «abs», поскольку присутствует отрицательное число. На следующей строке программа выведет значение переменной при которых выводится оптимальный показатель при вычислении функции linprog() с использованием симплекс метода. Стоит отметить, чтобы все вычисления метода были в одной функции, используется def value

**3.2.3 ОПИСАНИЕ ВЫХОДНЫХ ДАННЫХ**

В конце программа рассчитает оптимальное значение целевой функции, дополнительно указав значение х. Ответ будет представлен в формате:

оптимальное значение целевой функции (симплекс): /значение/

При х: /значение/

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

Последовательность действий пользователя:

1. Пользователь запускает программу
2. Пользователю запускается диалоговое окно, куда необходимо заполнить количество видов товаров;



Рисунок 1 – Диалоговое окно №1

1. Пользователь вводит необходимое число;
2. Пользователю выводится количество, которое он ввел;



Рисунок 2 – Вывод результата из диалогового окна №1

1. Пользователю запускается диалоговое окно, куда необходимо заполнить количество видов материалов;



Рисунок 3 - Диалоговое окно №2

1. Пользователь вводит необходимое число;
2. Пользователю выводится количество, которое он ввел;



Рисунок 3 – Вывод результата из диалогового окна №2

1. Пользователю выводится диалоговое окно, куда необходимо внести расход каждого материала на товар;



Рисунок 4 – Диалоговое окно №3

1. Пользователь вводит необходимые данные в последовательности: 1) сколько материала уходит на один товар, потом на следующий; 2) так каждый материал.
2. Пользователю выводятся написанные значения, и они же в виде матрицы, где каждая строка — это товар, а столбец – материал.

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 5 – Вывод результата из диалогового окна №3

1. Пользователю выводится диалоговое окно, куда необходимо внести цену на товар при этом добавив перед ним знак минус, что необходимо для работы функции;



Рисунок 6 – Диалоговое окно №4

1. Пользователь вводит цены;
2. Пользователю выводятся написанные значения;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 – Вывод результата из диалогового окна №4

1. Пользователю выводится диалоговое окно, куда необходимо внести количество материалов на складе;



Рисунок 8 – Диалоговое окно №5

1. Пользователь вводит количество материала на складе;
2. Пользователю выводятся написанные значения;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 – Вывод результата из диалогового окна №5

1. Пользователю выводится целевая функция из полученных данных;



Рисунок 8 – Вывод целевой функции

1. Пользователю выводится оптимальное значение функции и при каких значениях x оно достигается;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 – Вывод оптимального значения функции и x

1. Пользователю выводится диалоговое окно, где пользователь выбирает решить ли эту же задачу симплекс-методом;



Рисунок 10 – Диалоговое окно №6

1. Пользователь вводит «1»-если да или «2»-если нет
2. Если пользователь вводит «1», то выводится оптимальное значение функции и при каких значениях x оно достигается, решенное симплекс-методом;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 11 – Вывод оптимального значения функции и x, решённое симплекс-методом

1. Если пользователь вводит «2», то выводится «Конец».



Рисунок 12 – Вывод сообщения при выборе «Нет»

## АРХИТЕКТУРА РЕШЕНИЯ

Для решения задачи использовались методы(функции), которые можно разделить на 3 принципиальных кода.

**5.1. ФУНКЦИИ СЧИТЫВАНИЯ ИНФОРМАЦИИ**

Метод ввода данных (не осуществлен внутри функции)

Что делает: Позволяет заполнить весовой массив, вектор цены и вектор ограничений.

Входные параметры:

* нет входных параметров

Выходные параметры:

* переменная a - количество видов товара (тип данных int);
* переменная b - количество видов продукции (тип данных int);
* переменная data – весовая матрица (тип данных list);
* переменная price – вектор цены (тип данных list);
* переменная quantity – вектор ограничений (тип данных list).

Затрагиваемые в ходе работы переменные:

* переменная a - количество видов товара (тип данных int);
* переменная b - количество видов продукции (тип данных int);
* переменная data – весовая матрица (тип данных list);
* переменная price – вектор цены (тип данных list);
* переменная quantity – вектор ограничений (тип данных list).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13 – Часть кода, отвечающая за считывание информации

**5.2. ФУНКЦИИ ОБРАБОТКИ ИНФОРМАЦИИ**

*Функция function:*

Что делает: формирует целевую функцию.

Входные параметры:

* переменная f – вектор цены, применяем в целевой функции (тип данных list)

Выходные параметры:

* переменная func –часть целевой функции (тип данных list).

Затрагиваемые в ходе работы переменные:

* переменная price – вектор цены (тип данных list);
* переменная func1 – формирует значения для func (тип данных list);
* переменная x –i в значении xi (тип данных int), по умолчанию равна 1 и увеличивается с каждым новым элементом списка.

*Функция optimal\_value:*

Что делает: Решает задачу методом “interior-point”

Входные параметры:

* переменная p – используемая в linprog вектор цены (тип данных list);
* переменная d –используемая в linprog весовая матрица (тип данных list);
* переменная q – используемая в linprog вектор ограничений  (тип данных list).

Выходные параметры:

* ov-решение методом “interior-point”.

Затрагиваемые в ходе работы переменные:

* переменная data – весовая матрица (тип данных list);
* переменная price – вектор цены (тип данных list);
* переменная quantity – вектор ограничений (тип данных list).

*Функция simplex\_method:*

Что делает: Решает задачу методом “simplex”

Входные параметры:

* переменная p – используемая в linprog вектор цены (тип данных list);
* переменная d –используемая в linprog весовая матрица (тип данных list);
* переменная q – используемая в linprog вектор ограничений  (тип данных list).

Выходные параметры:

* sm-решение методом “simplex”.

Затрагиваемые в ходе работы переменные:

* переменная data – весовая матрица (тип данных list);
* переменная price – вектор цены (тип данных list);
* переменная quantity – вектор ограничений (тип данных list).

**Изображение выглядит как текст

Автоматически созданное описание**

Рисунок 14 – Часть кода, отвечающая за обработку информации

**5.3. ФУНКЦИИ ВЫВОДА ИНФОРМАЦИИ**

Метод вывода информации (он заключен внутри каждой функции(метода)

Что делает: осуществляет вывод необходимой информации

Вывод информации осуществляется с помощью функции print()  
Затрагиваемые переменные:

* переменная data2 – матрица значений (тип данных list);
* переменная price – вектор цены (тип данных list);
* переменная quantity – вектор ограничений (тип данных list);
* ov.fun – оптимальное значение полученное решение методом “interior-point” ( тип данных float)
* ov.x – значения x полученные при решение методом “interior-point” ( тип данных list)
* sm.fun – оптимальное значение полученное решение методом “simplex” ( тип данных float)
* sm.x – значения x полученные при решение методом “simplex” ( тип данных list)
* func – часть целевой функции (тип данных list).

Также имеется вывод значения симплекс в зависимости от выбора пользователя, где входная переменная xy- значение, показывающее выводить ли пользователю решение методом симплекс или нет  (тип данных int)

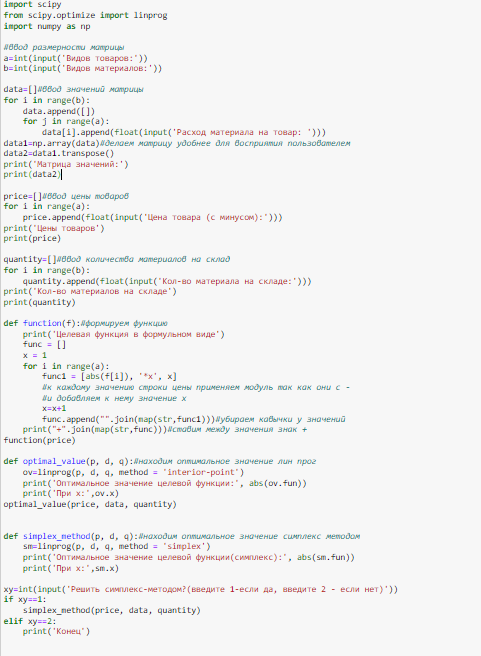


Рисунок 15 – Код полностью

## 6. ТЕСТИРОВАНИЕ

После проверки кода было выполнено тестирование. Также данная задача была решена через MS Excel и онлайн калькулятор (<https://matworld.ru/calculator/simplex-method-online.php>).

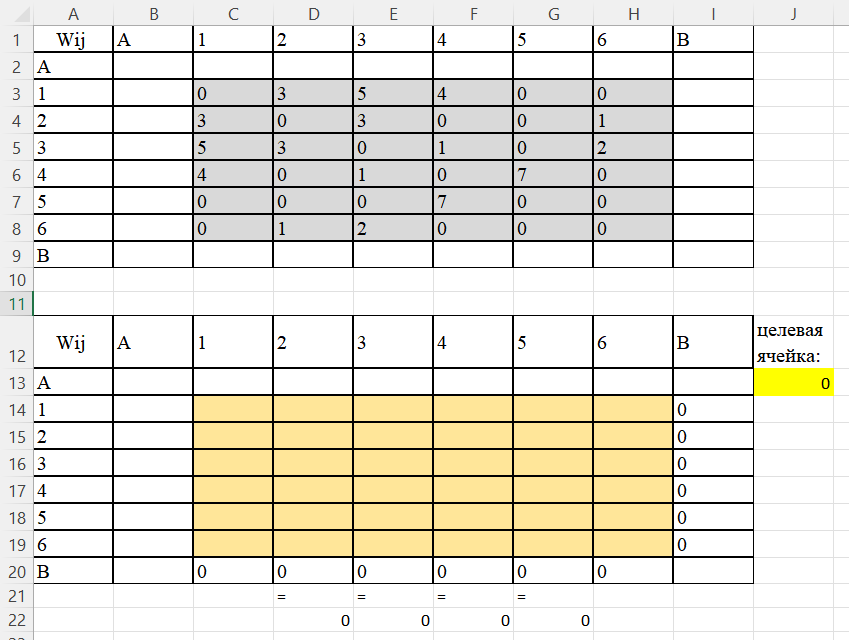
**6.1 МЕТОД ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ**

Результат оптимального значения – 7999.999995514576, результат x’ов: x1 = 9.90286644e-02, x2 = 1.98019427e+01, x3 = 4.57054455e-10 (рис. 9).

**6.2 СИМПЛЕКС МЕТОД**

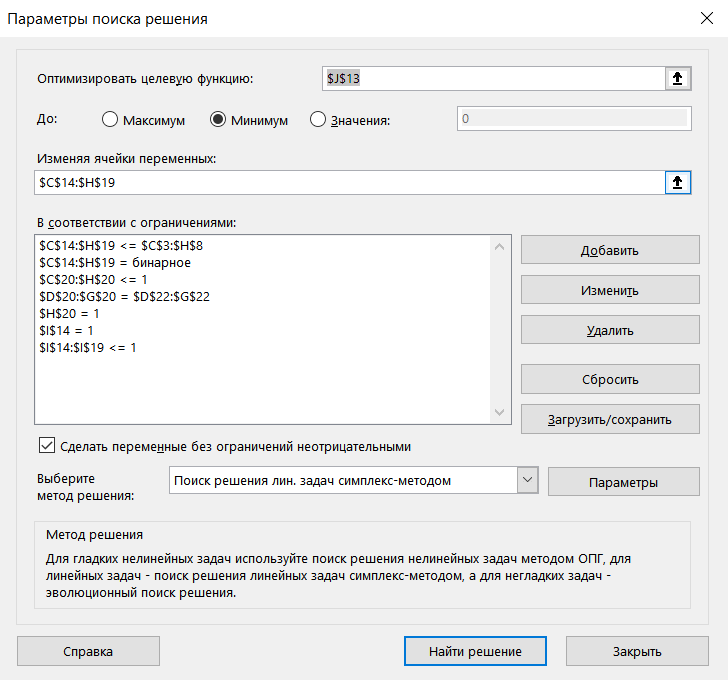
Результатом оптимального значения является 8000, x1 = 0, x2 = 20, x3 = 0 (рис. 11).

**6.3 MS EXCEL**

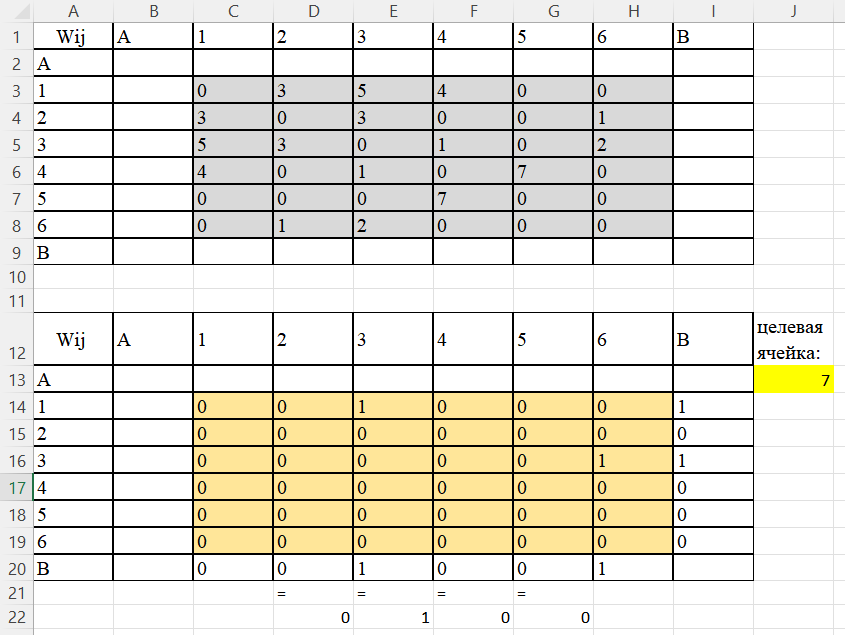
Запишем исходную матрицу в MS Excel и продублируем соразмерную матрицу без значений. В ячейках С20:Н20 напишем формулу суммы значений соответствующих столбцов, а в ячейках I14:I19 - формулу суммы соответствующих строк. В ячейке с целевой формулой укажем формулу =СУММПРОИЗВ и в качестве аргументов напишем обе матрицы.

Обозначим ограничения для использования надстройки Поиск решения:

1. Область изменяемых ячеек должна быть бинарной (значения могут быть только равные 0 или 1)
2. Значения изменяемых ячеек должны быть больше или равны значений соответствующих исходных ячеек (то есть, если значение исходной ячейки 0, то значение соответствующей ячейки в изменяющихся ячейках не может быть отличным от нуля)
3. Ячейка выхода из пункта 1 должна быть равна единице и ячейка входа в пункт 6 должна быть равна единице
4. Так как мы не можем выйти из одной вершины дважды одновременно, как и войти в вершину из двух вершин одновременно, обозначим суммы по столбцам не превышающими единицу
5. Для того, чтобы обозначить послежовательность вхождения из ожной вершины в другую, поместим снизу изменяющихся ячеек строчку, в которую вставим формулы: для D22 - ‘=15’ и так далее соответственно. Таким образом, ячейки D20:G20 = D22:G22 (если это равенство выполняется, значит мы вошли в вершину и вышли из нее, что логично и иначе быть не может)



Таким образом, применив данные настройки, мы получили ответ 7.



**6.4 СТОРОННЕЕ РЕШЕНИЕ**

В качестве стороннего решения был использован онлайн калькулятор Симплекс метод онлайн. Заполнение количества ограничений и количества переменных:

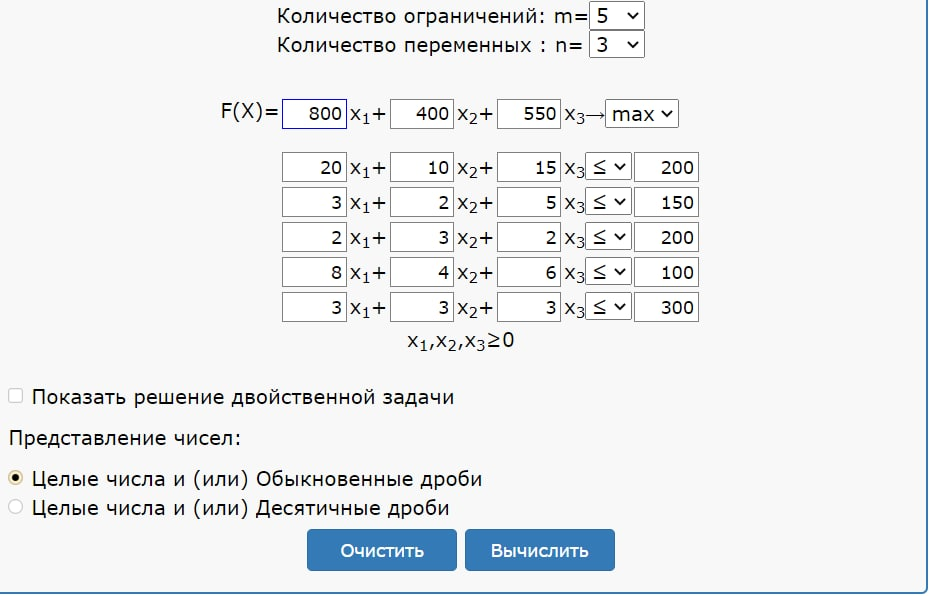


Рисунок 17 — Ввод данных для решения ЗЛП через онлайн калькулятора

Результат оптимального значения – 8000, x’ы: x1 = 10, x2 = 0, x3 = 0.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18 — Результат решения ЗЛП

## ЗАКЛЮЧЕНИЕ

Мы смогли решить задачу при помощи методов «interior-point» и «simplex». Их сравнение можно наблюдать в Таблице 4.

Таблица 5 - Сравнение алгоритмов.

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | interior-point | simplex | граф метод |
| Пример:  Количество возможных параметров | Не ограничено | Не ограничено | Не более 2 |
| Простота восприятия | не всем  понятно | легки в понимании | сложно определить точное значение,но позволяет увидеть как оно получилось |
| Время | быстрее достичь оптимальности | быстрее для небольших задач | затрачивает достаточно много времени |
| Тестирование | Число приближенное к тестированию | Число такое же как при тестировании в Excel и калькуляторе | Сложно сравнить |
| Точность | До 1000 | До 1000 | - |

Наиболее оптимальным алгоритмом мы считаем “simplex” метод, т.к. он предоставляет выходные данные, которые совпадают с результатами сторонних методов проверки

Предоставленный нами алгоритм позволил нам решить задачу в кратчайшие сроки, и предоставить заказчику наиболее точный ответ на поставленную задачу.

Варианты улучшения:

1. Предоставить пользователю возможность увидеть по какому материалу и продукции он набирает число в весовой матрице;
2. Возможность редакции данных;
3. Возможность выгрузки данных из excel формата;
4. Ввод цены без минуса, так как пользователь может не обратить на это внимания.