**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»**

**Расчетно-аналитическая работа по дисциплине**

«Математические методы принятия решений»

Студент группы БИ20-5:

Андронюк А О.

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

**ОГЛАВЛЕНИЕ**

[РЕШЕНИЕ ТРАНСПОРТНОЙ ЗАДАЧИ МЕТОДОМ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ №1 6](#_Toc106574676)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 6](#_Toc106574677)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ РЕШЕНИЯ ЗАДАЧИ 7](#_Toc106574678)

[АЛГОРИТМЫ РЕШЕНИЯ 11](#_Toc106574679)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 13](#_Toc106574680)

[АРХИТЕКТУРА РЕШЕНИЯ 14](#_Toc106574681)

[Архитектура решения Python (линейное программирование) 14](#_Toc106574682)

[Архитектура решения Excel (симплекс-метод) 17](#_Toc106574683)

[ТЕСТИРОВАНИЕ 18](#_Toc106574684)

[ЗАКЛЮЧЕНИЕ 1](#_Toc106574685)

[РЕШЕНИЕ ЗАДАЧИ КОММИВОЯЖЕРА И ЗАДАЧИ О КРАТЧАЙШЕМ ПУТИ №2 2](#_Toc106574686)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 2](#_Toc106574687)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 4](#_Toc106574688)

[АЛГОРИТМ РЕШЕНИЯ 16](#_Toc106574689)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 17](#_Toc106574690)

[АРХИТЕКТУРА РЕШЕНИЯ 18](#_Toc106574691)

[Архитектура решения Python 18](#_Toc106574692)

[Архитектура решения Excel 26](#_Toc106574693)

[ТЕСТИРОВАНИЕ 28](#_Toc106574694)

[ЗАКЛЮЧЕНИЕ 31](#_Toc106574695)

[РЕШЕНИЕ ЗАДАЧ ПО ТЕМЕ «ТЕОРИЯ ИГР» №3-4 32](#_Toc106574696)

[ПОСТАНОВКА ЗАДАЧИ (ФИЗИЧЕСКАЯ МОДЕЛЬ) 32](#_Toc106574697)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 40](#_Toc106574698)

[АЛГОРИТМЫ 59](#_Toc106574699)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 63](#_Toc106574700)

[АРХИТЕКТУРА РЕШЕНИЯ 78](#_Toc106574701)

[ТЕСТИРОВАНИЕ 84](#_Toc106574702)

[ЗАКЛЮЧЕНИЕ 86](#_Toc106574703)

[РЕШЕНИЕ ЗАДАЧ ПО ТЕМЕ «МЕТОД НАИМЕНЬШИХ КВАДРАТОВ» №5 87](#_Toc106574704)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 87](#_Toc106574705)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 88](#_Toc106574706)

[Входные данные 92](#_Toc106574707)

[Алгоритм решения 92](#_Toc106574708)

[Выходные данные 92](#_Toc106574709)

[ВХОДНЫЕ ДАННЫЕ 93](#_Toc106574710)

[АЛГОРИТМ РЕШЕНИЯ 93](#_Toc106574711)

[ТЕСТИРОВАНИЕ 102](#_Toc106574712)

[ЗАКЛЮЧЕНИЕ 121](#_Toc106574713)

[ЭКСПЕРТНЫЕ ОЦЕНКИ №6 122](#_Toc106574714)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 122](#_Toc106574715)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 123](#_Toc106574716)

[АЛГОРИТМЫ РЕШЕНИЯ 125](#_Toc106574717)

[Параметры входных данных 125](#_Toc106574718)

[Алгоритм решения 125](#_Toc106574719)

[Параметры выходных данных 126](#_Toc106574720)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 127](#_Toc106574721)

[АРХИТЕКТУРА РЕШЕНИЯ 130](#_Toc106574722)

[ТЕСТИРОВАНИЕ 143](#_Toc106574723)

[ЗАКЛЮЧЕНИЕ 144](#_Toc106574724)

[ЭКСПЕРТНЫЕ ОЦЕНКИ №7 145](#_Toc106574725)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 145](#_Toc106574726)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 146](#_Toc106574727)

[АЛГОРИТМЫ РЕШЕНИЯ 156](#_Toc106574728)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ 159](#_Toc106574729)

[АРХИТЕКТУРА РЕШЕНИЯ 164](#_Toc106574730)

[ТЕСТИРОВАНИЕ 171](#_Toc106574731)

[ЗАКЛЮЧЕНИЕ 176](#_Toc106574732)

[СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ №8 177](#_Toc106574733)

[ФИЗИЧЕСКАЯ МОДЕЛЬ 177](#_Toc106574734)

[МАТЕМАТИЧЕСКАЯ МОДЕЛЬ 178](#_Toc106574735)

[АЛГОРИТМЫ РЕШЕНИЯ 184](#_Toc106574736)

[Параметры входных данных 184](#_Toc106574737)

[Алгоритм решения 184](#_Toc106574738)

[Параметры выходных данных 186](#_Toc106574739)

[ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ 187](#_Toc106574740)

[АРХИТЕКТУРА РЕШЕНИЯ 191](#_Toc106574741)

[ТЕСТИРОВАНИЕ 195](#_Toc106574742)

[ЗАКЛЮЧЕНИЕ 196](#_Toc106574743)

# РЕШЕНИЕ ТРАНСПОРТНОЙ ЗАДАЧИ МЕТОДОМ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ №1

## ФИЗИЧЕСКАЯ МОДЕЛЬ

Данная работа посвящена решению транспортной задачи методом линейного программирования для логистической минимизации стоимости грузов. Для начала следует обозначить условие задачи:

Формат входных д анных:

а) Размер матрицы входных данных;

б) Матрица входных данных;

в) Вектор ограничений

Формат выходных данных (решение задачи):

а) Целевая функция в формульном виде.

б) Оптимальное значение целевой функции.

Визуализация данных:

а) График визуализации:

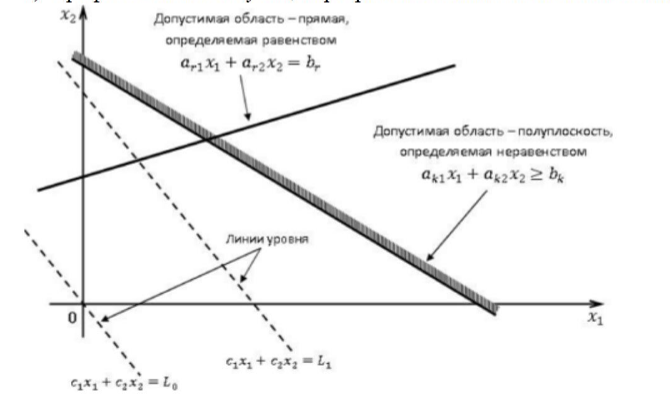


Рисунок 1

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ РЕШЕНИЯ ЗАДАЧИ

Рассмотрим транспортную задачу, где в качестве критерия оптимальности взята минимальная стоимость перевозок всего груза. Обозначим через Сij тарифы перевозки единицы груза из пункта отправления i в пункт назначения j. Обозначим через Ai запасы груза i-м пункте отправления, а через Bj потребности груза j-м пункте назначения, а через Xj количество единиц груза переводимого из пункта отправления i в пункт назначения j.

Тогда математическая модель транспортной задачи состоит в определении минимального значения функции:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2

при условиях, что:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3

Поскольку удовлетворяется условия (1.2)−(1.4), то обеспечивается доставка необходимого количества груза в каждый из пунктов назначения, вывоз груза из всех пунктов отправления, а также исключаются обратные перевозки.

F = ∑∑cijxij, (1)  
При условиях:  
∑xij = ai, i = 1,2,…, m, (2)  
∑xij = bj, j = 1,2,…, n, (3)  
xij ≥ 0

Запишем экономико-математическую модель для нашей задачи.  
Переменные:  
x11 – количество груза из 1-го склада в 1-й магазин.  
x12 – количество груза из 1-го склада в 2-й магазин.  
x13 – количество груза из 1-го склада в 3-й магазин.  
x21 – количество груза из 2-го склада в 1-й магазин.  
x22 – количество груза из 2-го склада в 2-й магазин.  
x23 – количество груза из 2-го склада в 3-й магазин.  
x31 – количество груза из 3-го склада в 1-й магазин.  
x32 – количество груза из 3-го склада в 2-й магазин.  
x33 – количество груза из 3-го склада в 3-й магазин.

Ограничения по запасам:

x11  + x12  + x13  ≤ 74 (для 1 базы)  
x21  + x22  + x23  ≤ 40 (для 2 базы)  
x31  + x32  + x33  ≤ 36 (для 3 базы)

Ограничения по потребностям:

x11  + x21  + x31  = 20 (для 1-го магазина)  
x12  + x22  + x32  = 45 (для 2-го магазина)  
x13  + x23  + x33  = 30 (для 3-го магазина)

Целевая функция:

7x11 + 3x12 + 6x13 + 4x21 + 8x22 + 2x23 + 1x31 + 5x32 + 9x33 → min

Стоимость доставки единицы груза из каждого пункта отправления в соответствующие пункты назначения задана матрицей тарифов:

Таблица 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | B1 | B2 | B3 | Запасы |
| A1 | 7 | 3 | 6 | 74 |
| A2 | 4 | 8 | 2 | 40 |
| A3 | 1 | 5 | 9 | 36 |
| Потребности | 20 | 45 | 30 |  |

Проверим необходимое и достаточное условие разрешимости задачи.  
∑a = 74 + 40 + 36 = 150  
∑b = 20 + 45 + 30 = 95

Как видно, суммарная потребность груза в пунктах назначения меньше запасов груза на базах. Следовательно, модель исходной транспортной задачи является открытой. Чтобы получить закрытую модель, введем дополнительную (фиктивную) потребность, равной 55 (150—95). Тарифы перевозки единицы груза к этому магазину полагаем равны нулю.

Занесем исходные данные в распределительную таблицу.

Таблица 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | B1 | B2 | B3 | B4 | Запасы |
| A1 | 7 | 3 | 6 | 0 | 74 |
| A2 | 4 | 8 | 2 | 0 | 40 |
| A3 | 1 | 5 | 9 | 0 | 36 |
| Потребности | 20 | 45 | 30 | 55 |  |

Граф решения:



Рисунок 4

## АЛГОРИТМЫ РЕШЕНИЯ

Для решения данной задачи следует вычислить (метод линейного программирования).

**Параметры входных данных:**

а) Размер матрицы входных данных;

б) Матрица входных данных;

в) Вектор ограничений

**Алгоритм решения:**

1. Минимальное значение функции, которая вытекает из входных данных.
2. Обеспечить доставку необходимого количества груза в каждый из пунктов назначения, вывоз груза из всех пунктов отправления, а также исключить обратные перевозки.
3. Стоимость доставки единицы груза из каждого пункта отправления в соответствующие пункты назначения (матрица).
4. Вычислить суммарную потребность груза.
5. Получить данные в виде таблицы.

**Параметры выходных данных:**

а) Целевая функция в формульном виде.

б) Оптимальное значение целевой функции.

Алгоритмы для решения производственной задачи (транспортная или задача о назначениях) Симплекс-методом. **(опционально)**

**Параметры входных данных:**

а) Размер матрицы входных данных;

б) Матрица входных данных;

в) Вектор ограничений

**Алгоритм решения:**

1. Определим минимальное значение целевой функции F(X).
2. Систему неравенств приведем к системе уравнений путем введения дополнительных переменных (переход к канонической форме).
3. Решим систему уравнений относительно базисных переменных.
4. Проверим критерий оптимальности.
5. Составим симплекс-таблицу.
6. Запишем оптимальный план.

**Параметры выходных данных:**

а) Целевая функция в формульном виде.

б) Оптимальное значение целевой функции.

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

Частью линейного программирования являются транспортные задачи, которые играют особую роль в уменьшении транспортных издержек предприятия. Это является актуальным вопросом в условиях рыночной экономики, когда любые затраты должны быть минимизированы, ведь тогда издержки покрываются меньшей частью прибыли, а также позволяют снизить себестоимость продукции на рынке, что делает предприятие более конкурентоспособным.

Транспортная задача – задача об оптимальном плане перевозок продукта из пункта наличия в пункт потребления. Их целью является доставка продукции в определенное время и место при минимальных совокупных затратах трудовых, материальных и финансовых ресурсов.

Она считается достигнутой, если нужный товар требуемого качества и в необходимом количестве доставляется в нужное время и в нужное место с минимальными затратами.

Для решения транспортных задач разработан специальный метод, имеющий следующие этапы:

1) Нахождение исходного опорного решения

2) Проверка этого решения на оптимальность

3) Переход от одного опорного решения к другому

Данная задача решается с помощью Excel (при вводе нужных матричных данных), ручным решением и с помощью кодирования в Python.

## АРХИТЕКТУРА РЕШЕНИЯ

### Архитектура решения Python (линейное программирование)

На основе условия задачи с форматом входных данных и требуемых выходных данных, основная программа (код) решения данной задачи была технически разделена три составные части. Во-первых, вызов необходимых библиотек – linprog, используемая для расчета оптимизации линейного программирования.

Пример из кода:

from scipy.optimize import linprog

Во-вторых, для выполнения выходных данных пункта а используется функция для создания матрицы в зависимости от входных данных пользователя – заданной оси, количеству вариационных рядов. То есть для визуализации входных данных пользователя для обращения в формат целевой функции в формульном виде.



Рисунок 5

В-третьих, далее программа, ориентируясь на созданную матрицу на основе входных данных пользователя, «переходит» к третьей части задачи – оптимизации. Для этого используется функция для оптимизации linprog. Перед ней пользователю нужно ввести входные данные, состоящие из размерности матрицы, получившейся ранее, чисел по осям X и Y и нужное количество чисел для оптимизации.



Рисунок 6

В конце программы (кода) используется возвращение функций для вызова результата (построенной на основе введенных пользователем входных данных матрица, целевая функция в формульном виде и оптимизация целевой функции).

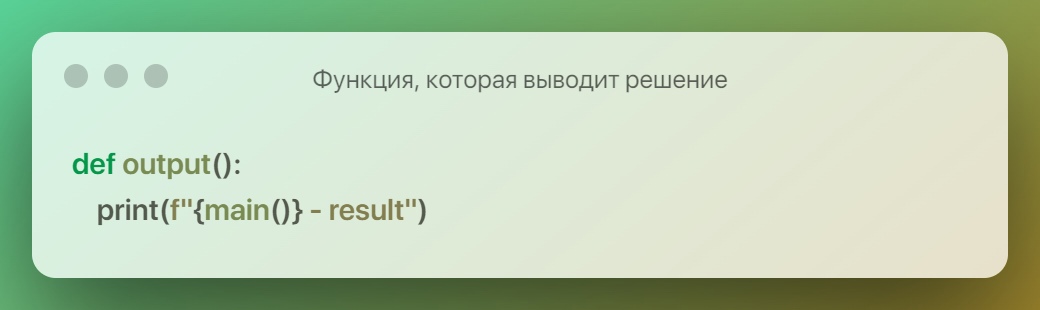
****

Рисунок 7

## Архитектура решения Excel (симплекс-метод)

В программе Excel решение задачи производится сервисом «Поиск решения».Изображение выглядит как стол

Автоматически созданное описание

Рисунок 8

Для решения следует выделить целевую функцию, обратится к сервису Поиск решения, отметить линейную модель, добавить неотрицательные значения, выделить изменить пустые ячейки для знаний, добавить нужные ограничения, свести решение к минимуму и выполнить команду.

## ТЕСТИРОВАНИЕ

Проверим решение задачи ручным способом через Excel (с данными из математической модели, с учетом приведения к целым неотрицательным числам в матрице).

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 9

Во-первых, введем искомую матрицу и объемы запасов и спроса (зеленые ячейки). Далее введем в матрице формулы сумм по столбцам и строкам (изначально матрица, не выделенная цветом с пустыми значениями).

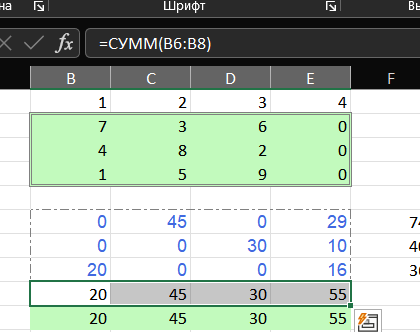


Рисунок 10

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 11

Далее введем формулу целевой функции. Она равна:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 12

Далее с помощью надстроек Excel и функции Поиск решения находим искомую матрицу и целевую функцию. Целевая функция по итогам решения 215.

Проверка выполнена.

## ЗАКЛЮЧЕНИЕ

Опираясь на данные из условия математической модели, ответ на задачу будет следующим: опорный план является оптимальным, так все оценки свободных клеток удовлетворяют условию ui + vj ≤ cij. Минимальные затраты составят: F(x) = 3\*45 + 0\*29 + 2\*30 + 0\*10 + 1\*20 + 0\*16 = 215

Итог:

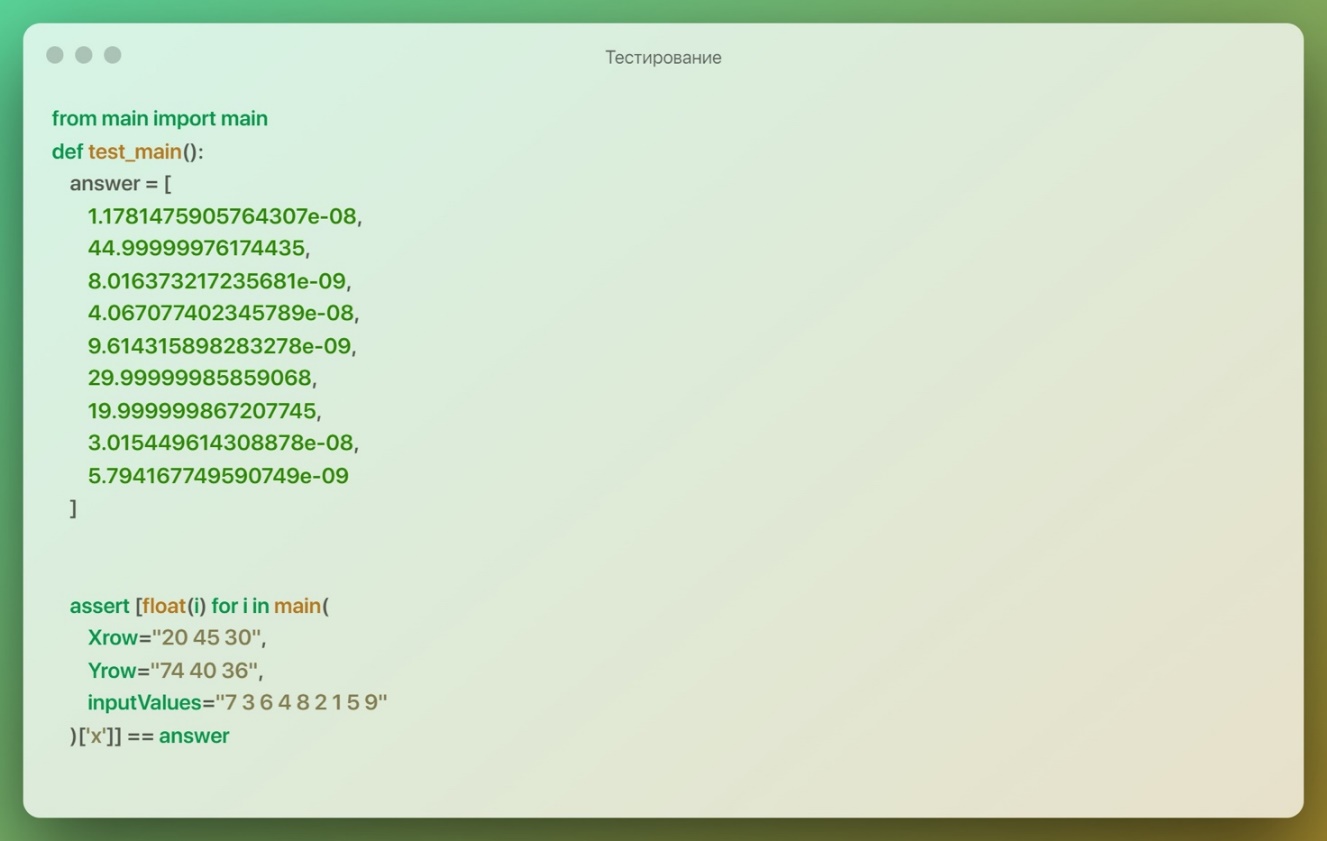


Рисунок 13

Тестирование при помощи pytest:

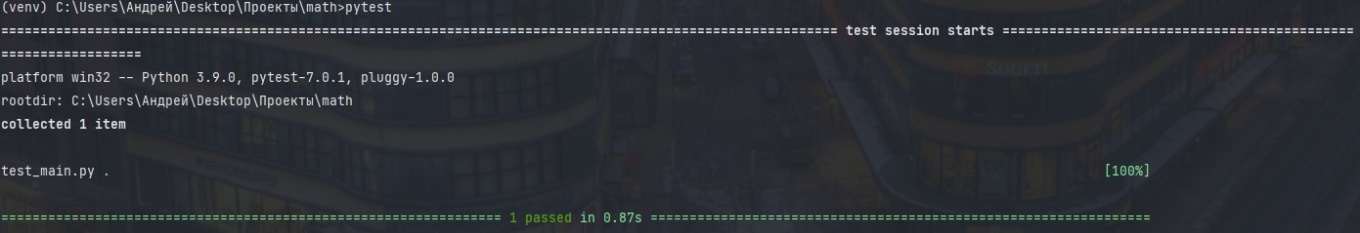


Рисунок 14

Данная задача имеет перспективы развития, например, визуализированное решение с помощью диаграмм, точечных построений и графиков для более быстрого нахождения решения задачи и оперативного принятия решений. Решение выполнено линейным программированием и симплекс-методом.

# РЕШЕНИЕ ЗАДАЧИ КОММИВОЯЖЕРА И ЗАДАЧИ О КРАТЧАЙШЕМ ПУТИ №2

## ФИЗИЧЕСКАЯ МОДЕЛЬ

**Задание 1.** Написать (добавить в собственный класс/библиотеку) следующие функции:

1. Решение задачи о нахождении кротчайшего пути при помощи **Алгоритма Дейкстры**;

Формат входных данных:

а) число вершин (городов);

б) матрица весовых коэффициентов;

\* необходимо организовать 3 варианта получения матрицы весов:

А) ввод с клавиатуры (для малого количества городов)

Б) случайная генерация.

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

Формат выходных данных:

а) строка с номерами вершин в формате:

где номера соответствуют последовательности посещения вершин при оптимальном маршруте.

1. Решение задачи коммивояжера при помощи **алгоритма ветвей и границ** (branch and bound)

Формат входных данных:

а) число вершин (городов);

б) матрица весовых коэффициентов;

\* необходимо организовать 3 варианта получения матрицы весов:

А) ввод с клавиатуры (для малого количества городов)

Б) случайная генерация.

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

Формат выходных данных:

а) строка с номерами вершин в формате:

где номера соответствуют последовательности посещения вершин при оптимальном маршруте.

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Задача коммивояжера.

Возьмем в качестве произвольного маршрута:

X0 = (1,2);(2,3);(3,4);(4,5);(5,1), тогда F(X0) = 2 + 7 + 6 + 5 + 5 = 25

Для определения нижней границы множества воспользуемся операцией редукции или приведения матрицы по строкам, для чего необходимо в каждой строке матрицы D найти минимальный элемент.

di = min(j) dij

i j 1 2 3 4 5 di

1 M 2 3 4 5 2

2 2 M 7 3 4 2

3 3 4 M 6 2 2

4 4 2 3 M 5 2

5 5 6 4 3 M 3

Затем вычитаем di из элементов рассматриваемой строки. В связи с этим во вновь полученной матрице в каждой строке будет как минимум один ноль.

i j 1 2 3 4 5

1 M 0 1 2 3

2 0 M 5 1 2

3 1 2 M 4 0

4 2 0 1 M 3

5 2 3 1 0 M

Такую же операцию редукции проводим по столбцам, для чего в каждом столбце находим минимальный элемент:

dj = min(i) dij

i j 1 2 3 4 5

1 M 0 1 2 3

2 0 M 5 1 2

3 1 2 M 4 0

4 2 0 1 M 3

5 2 3 1 0 M

dj 0 0 1 0 0

После вычитания минимальных элементов получаем полностью редуцированную матрицу, где величины di и dj называются константами приведения.

i j 1 2 3 4 5

1 M 0 0 2 3

2 0 M 4 1 2

3 1 2 M 4 0

4 2 0 0 M 3

5 2 3 0 0 M

Сумма констант приведения определяет нижнюю границу H:

H = ∑di + ∑dj

H = 2+2+2+2+3+0+0+1+0+0 = 12

Элементы матрицы dij соответствуют расстоянию от пункта i до пункта j. Поскольку в матрице n городов, то D является матрицей nxn с неотрицательными элементами dij ≥ 0

Каждый допустимый маршрут представляет собой цикл, по которому коммивояжер посещает город только один раз и возвращается в исходный город.

Длина маршрута определяется выражением:

F(Mk) = ∑dij

Причем каждая строка и столбец входят в маршрут только один раз с элементом dij.

Шаг №1.

Определяем ребро ветвления и разобьем все множество маршрутов относительно этого ребра на два подмножества (i,j) и (i\*,j\*).

С этой целью для всех клеток матрицы с нулевыми элементами заменяем поочередно нули на М(бесконечность) и определяем для них сумму образовавшихся констант приведения, они приведены в скобках.

i j 1 2 3 4 5 di

1 M 0(0) 0(0) 2 3 0

2 0(2) M 4 1 2 1

3 1 2 M 4 0(3) 1

4 2 0(0) 0(0) M 3 0

5 2 3 0(0) 0(1) M 0

dj 1 0 0 1 2 0

d(1,2) = 0 + 0 = 0; d(1,3) = 0 + 0 = 0; d(2,1) = 1 + 1 = 2; d(3,5) = 1 + 2 = 3; d(4,2) = 0 + 0 = 0; d(4,3) = 0 + 0 = 0; d(5,3) = 0 + 0 = 0; d(5,4) = 0 + 1 = 1;

Наибольшая сумма констант приведения равна (1 + 2) = 3 для ребра (3,5), следовательно, множество разбивается на два подмножества (3,5) и (3\*,5\*).

Исключение ребра (3,5) проводим путем замены элемента d35 = 0 на M, после чего осуществляем очередное приведение матрицы расстояний для образовавшегося подмножества (3\*,5\*), в результате получим редуцированную матрицу.

i j 1 2 3 4 5 di

1 M 0 0 2 3 0

2 0 M 4 1 2 0

3 1 2 M 4 M 1

4 2 0 0 M 3 0

5 2 3 0 0 M 0

dj 0 0 0 0 2 3

Нижняя граница гамильтоновых циклов этого подмножества:

H(3\*,5\*) = 12 + 3 = 15

Включение ребра (3,5) проводится путем исключения всех элементов 3-ой строки и 5-го столбца, в которой элемент d53 заменяем на М, для исключения образования негамильтонова цикла.

В результате получим другую сокращенную матрицу (4 x 4), которая подлежит операции приведения.

После операции приведения сокращенная матрица будет иметь вид:

i j 1 2 3 4 di

1 M 0 0 2 0

2 0 M 4 1 0

4 2 0 0 M 0

5 2 3 M 0 0

dj 0 0 0 0 0

Сумма констант приведения сокращенной матрицы:

∑di + ∑dj = 0

Нижняя граница подмножества (3,5) равна:

H(3,5) = 12 + 0 = 12 ≤ 15

Поскольку нижняя граница этого подмножества (3,5) меньше, чем подмножества (3\*,5\*), то ребро (3,5) включаем в маршрут с новой границей H = 12

Шаг №2.

Определяем ребро ветвления и разобьем все множество маршрутов относительно этого ребра на два подмножества (i,j) и (i\*,j\*).

С этой целью для всех клеток матрицы с нулевыми элементами заменяем поочередно нули на М(бесконечность) и определяем для них сумму образовавшихся констант приведения, они приведены в скобках.

i j 1 2 3 4 di

1 M 0(0) 0(0) 2 0

2 0(3) M 4 1 1

4 2 0(0) 0(0) M 0

5 2 3 M 0(3) 2

dj 2 0 0 1 0

d(1,2) = 0 + 0 = 0; d(1,3) = 0 + 0 = 0; d(2,1) = 1 + 2 = 3; d(4,2) = 0 + 0 = 0; d(4,3) = 0 + 0 = 0; d(5,4) = 2 + 1 = 3;

Наибольшая сумма констант приведения равна (1 + 2) = 3 для ребра (2,1), следовательно, множество разбивается на два подмножества (2,1) и (2\*,1\*).

Исключение ребра (2,1) проводим путем замены элемента d21 = 0 на M, после чего осуществляем очередное приведение матрицы расстояний для образовавшегося подмножества (2\*,1\*), в результате получим редуцированную матрицу.

i j 1 2 3 4 di

1 M 0 0 2 0

2 M M 4 1 1

4 2 0 0 M 0

5 2 3 M 0 0

dj 2 0 0 0 3

Нижняя граница гамильтоновых циклов этого подмножества:

H(2\*,1\*) = 12 + 3 = 15

Включение ребра (2,1) проводится путем исключения всех элементов 2-ой строки и 1-го столбца, в которой элемент d12 заменяем на М, для исключения образования негамильтонова цикла.

В результате получим другую сокращенную матрицу (3 x 3), которая подлежит операции приведения.

После операции приведения сокращенная матрица будет иметь вид:

i j 2 3 4 di

1 M 0 2 0

4 0 0 M 0

5 3 M 0 0

dj 0 0 0 0

Сумма констант приведения сокращенной матрицы:

∑di + ∑dj = 0

Нижняя граница подмножества (2,1) равна:

H(2,1) = 12 + 0 = 12 ≤ 15

Поскольку нижняя граница этого подмножества (2,1) меньше, чем подмножества (2\*,1\*), то ребро (2,1) включаем в маршрут с новой границей H = 12

Шаг №3.

Определяем ребро ветвления и разобьем все множество маршрутов относительно этого ребра на два подмножества (i,j) и (i\*,j\*).

С этой целью для всех клеток матрицы с нулевыми элементами заменяем поочередно нули на М(бесконечность) и определяем для них сумму образовавшихся констант приведения, они приведены в скобках.

i j 2 3 4 di

1 M 0(2) 2 2

4 0(3) 0(0) M 0

5 3 M 0(5) 3

dj 3 0 2 0

d(1,3) = 2 + 0 = 2; d(4,2) = 0 + 3 = 3; d(4,3) = 0 + 0 = 0; d(5,4) = 3 + 2 = 5;

Наибольшая сумма констант приведения равна (3 + 2) = 5 для ребра (5,4), следовательно, множество разбивается на два подмножества (5,4) и (5\*,4\*).

Исключение ребра (5,4) проводим путем замены элемента d54 = 0 на M, после чего осуществляем очередное приведение матрицы расстояний для образовавшегося подмножества (5\*,4\*), в результате получим редуцированную матрицу.

i j 2 3 4 di

1 M 0 2 0

4 0 0 M 0

5 3 M M 3

dj 0 0 2 5

Нижняя граница гамильтоновых циклов этого подмножества:

H(5\*,4\*) = 12 + 5 = 17

Включение ребра (5,4) проводится путем исключения всех элементов 5-ой строки и 4-го столбца, в которой элемент d45 заменяем на М, для исключения образования негамильтонова цикла.

В результате получим другую сокращенную матрицу (2 x 2), которая подлежит операции приведения.

После операции приведения сокращенная матрица будет иметь вид:

i j 2 3 di

1 M 0 0

4 0 0 0

dj 0 0 0

Сумма констант приведения сокращенной матрицы:

∑di + ∑dj = 0

Нижняя граница подмножества (5,4) равна:

H(5,4) = 12 + 0 = 12 ≤ 17

Поскольку нижняя граница этого подмножества (5,4) меньше, чем подмножества (5\*,4\*), то ребро (5,4) включаем в маршрут с новой границей H = 12

В соответствии с этой матрицей включаем в гамильтонов маршрут ребра (1,3) и (4,2).

В результате по дереву ветвлений гамильтонов цикл образуют ребра:

(3,5), (5,4), (4,2), (2,1), (1,3),

Длина маршрута равна F(Mk) = 12

Алгоритм Дейкстры:

Граф:

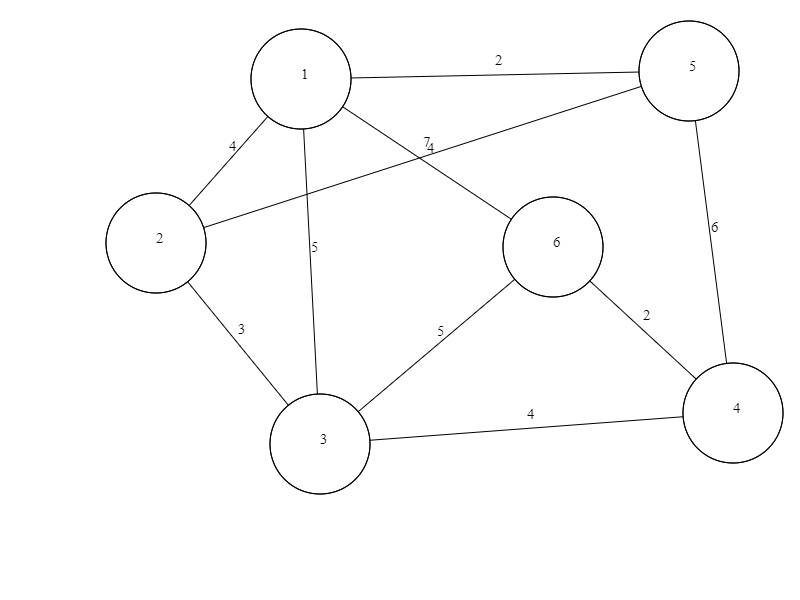


Рисунок 1

Задача о кратчайшем пути между парой вершин. Алгоритм Дейкстры

1. Устанавливаем расстояние D[i] от начальной вершины s до всех остальных в ∞.

Таблица 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| D | 0 | ∞ | ∞ | ∞ | ∞ |

2. Полагаем D[1] = 0.

Помещаем все вершины в очередь с приоритетом Q.

4. Запускаем цикл из n итераций (по числу вершин).

Итерация №0

Извлекаем из очереди Q вершину v=1 с минимальным приоритетом – ближайшую к s=1 вершину.

Отмечаем вершину v как посещенную (помещаем v во множество H).

Возможны пути из s=1 через вершину v=1 стали короче, выполняем проверку: для каждой вершины u смежной с вершиной v и не включенной в H проверяем и корректируем расстояние D[u].

Таблица 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| D | 0 | 0+3=3 | 0+7=7 | ∞ | 0+20=20 |

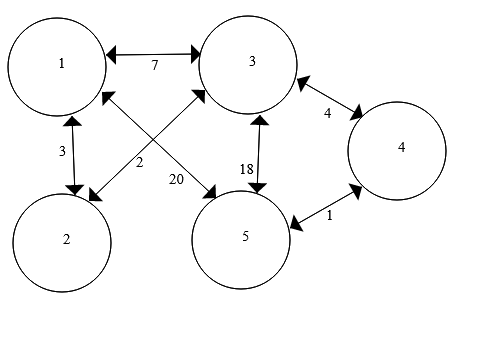


Рисунок 2

Итерация №1

Извлекаем из очереди Q вершину v=2 с минимальным приоритетом – ближайшую к s=1 вершину.

Отмечаем вершину v как посещенную (помещаем v во множество H).

Корректируем расстояние D[u].

Таблица 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| D | 0 | 3 | 3+2=5 | ∞ | 20 |

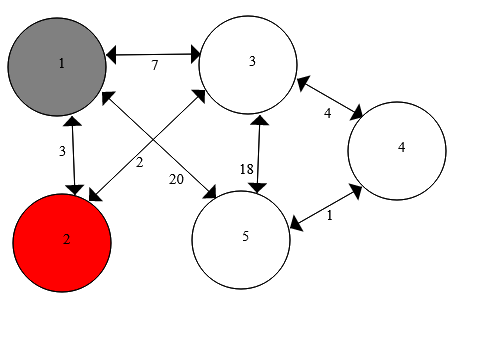


Рисунок 3

Итерация №2

Извлекаем из очереди Q вершину v=3 с минимальным приоритетом – ближайшую к s=1 вершину.

Отмечаем вершину v как посещенную (помещаем v во множество H).

Корректируем расстояние D[u].

Таблица 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| D | 0 | 3 | 5 | 5+4=9 | 20 |

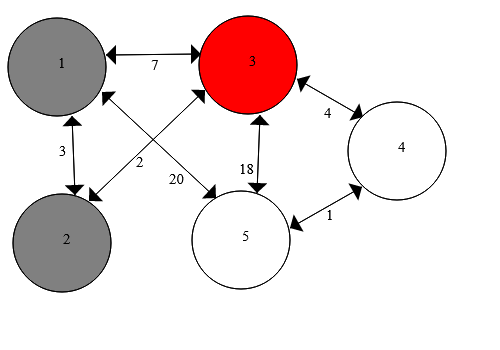


Рисунок 4

Итерация №3

Извлекаем из очереди Q вершину v=4 с минимальным приоритетом – ближайшую к s=1 вершину.

Отмечаем вершину v как посещенную (помещаем v во множество H).

Корректируем расстояние D[u].

Таблица 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n | 1 | 2 | 3 | 4 | 5 |
| D | 0 | 3 | 5 | 9 | 9+1=10 |

Изображение выглядит как iPod, векторная графика

Автоматически созданное описание

Рисунок 5

Путь: 1→2→3→4→5. Длина: 10

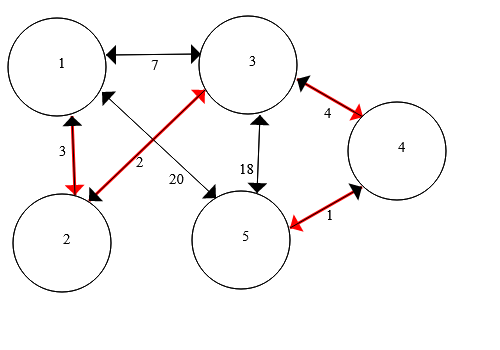


Рисунок 6

## АЛГОРИТМ РЕШЕНИЯ

**Параметры входных данных:**

Число вершин и матрица весовых коэффициентов.

**Алгоритм решения:**

1. Совершим операцию редукции для матрицы.
2. Определим нижнюю границу множества.
3. Вычислим длину маршрута.
4. Определяем ребро ветвления и разобьем все множество маршрутов относительно этого ребра.
5. Повторяем операции приведения матрицы.
6. Вычисляем длину маршрута по сокращенной матрице.

**Параметры выходных данных:**

Вектор кратчайшего пути и длина кратчайшего пути.

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

Применение задачи коммивояжера довольно обширно. Вот лишь некоторые возможные варианты применения задачи коммивояжера на практике в экономике:

* определение оптимального маршрута грузоперевозок;
* расчет наилучших маршрутов для курьеров и почтальонов.

Кроме логистики, задачи коммивояжера находит применение и в других областях экономики и человеческой деятельности: финансы (оптимизация денежных потоков, например, поиск путей перевода денежных средств с минимальными транзакционными издержками), туризм (расчет маршрутов экскурсий и туров), шоу-бизнес (организация турне музыкальных групп), биология (сборка генома), телекоммуникации и связь (управление спутникам, проектирование телекоммуникационных сетей), информатика (кластеризация массивов данных), энергетика и коммунальное хозяйство (соединение населенных пунктов линиями электропередач и газоснабжения), электроника (проектирование топологий микросхем).

Кроме практического смысла задача имеет большую теоретическую значимость. В частности, она выступает моделью для проверки новых методов оптимизации, поскольку нахождение по-настоящему оптимального маршрута для большого количества рассматриваемых пунктов является сложной проблемой, требующей эффективных математических инструментов. Немало современных и востребованных методов дискретной оптимизации (рассмотренный нами метод ветвей и границ, метод отсечений, эвристические алгоритмы) были разработаны именно на примере задачи коммивояжера.

## АРХИТЕКТУРА РЕШЕНИЯ

### Архитектура решения Python

Выполняем функцию копирования временного решения в окончательное:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7

Выполняем функцию для нахождения минимальной стоимости ребра, имеющего конец в вершине i:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8

Выполняем функцию для нахождения второй минимальной стоимости ребра, имеющей конец в вершине i:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9

Выполняем функцию (1), принимающую в качестве аргументов нижнюю границу корневого узла, сохранение веса пути ( на текущий момент ), текущий уровень (при перемещении в дереве пространства поиска, где хранится текущее решение, которое в последствии будет скопировано в финальное решение), при условии базового случая ( достижение уровня N ), что означает охват всех узлов один раз. Проверяем, есть ли ребро от последней вершины в пути назад к первой вершине:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 10

Для любого другого уровня выполняем итерацию для всех вершин для построения рекурсивного построения дерева пространства поиска:



Рисунок 11

Рассмотрим следующую вершину на предмет схожести ( диагональная запись в матрице смежности и проверка посещения):

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 12

Вычислим нижнюю границу корневого узла для уровня 2, отличающуюся от других уровней:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13

Сложим нижнюю границу корневого узла и сохранение веса пути на текущий момент, тем самым найдем фактическую нижнюю границу для узла, к которому пришли. Если текущая нижняя граница меньше финального результата, то нам нужно будет исследовать узел дальше:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 14

Вызываем функцию (1) для следующего уровня:



Рисунок 15

В противном случае обрезаем узел, сбросив все изменения в нижней границе корневого узла и сохранении веса пути на текущий момент:



Рисунок 16

Также сбрасываем ранее посещенный массив:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 17

Выполняем функцию установления окончательного решения. Вычисляем начальную нижнюю границу для корневого узла ( ½\*(сумма первого и второго минимального)) для всех ребер. Затем инициализируем текущее решение и посещенный массив:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18

Вычисляем начальную границу:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 19

Округляем нижнюю границу до целого числа:



Рисунок 20

Мы начинаем с вершины 1, следовательно в текущем решении она равна нулю:



Рисунок 21

Вызываем функцию (1) для сохранения веса пути на текущий момент (равно нулю и уровню 1):



Рисунок 22

Выводим конечный результат:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 23

В итоге, мы получаем строку с номерами вершин, где номера соответствуют последовательности посещения вершин при оптимальном маршруте:

Изображение выглядит как текст, часы

Автоматически созданное описание

Рисунок 24

* + 1. **Алгоритм имитации отжига**

Алгоритм решения задачи о нахождении кротчайшего пути

Входные данные задаются тремя способами: 1) вводом с клавиатуры 2) случайной генерацией 3) считыванием из CSV файла

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 25

Вводим число от 1 до 3, в зависимости от нужного способа ввода данных.

После выбора вводим количество вершин матрицы:

Задаем класс алгоритма имитации отжига:

Изображение выглядит как текст, экран, снимок экрана

Автоматически созданное описание

Рисунок 26

Выполняем функцию расчета длины пути, если находимся в конце заявленного пути, возвращаемся в стартовую точку:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 27

Создаем новый случайный путь, при этом выбирая 2 случайных несовпадающих числа из массива случайных чисел величиной в количество городов для стартовых точек нового пути:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 28

Создаем заготовку под итоговую матрицу, используя при этом формулу для перепроверки ( exp(-(fy-fx)/T) = e^(-(fy-fx)/T) ):

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 29

Задаем величины:

Изображение выглядит как текст, экран, устройство, счетчик

Автоматически созданное описание

Рисунок 30

Используем первый условный маршрут для сравнения и нумерации городов:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 31

В итоге мы получаем строку с номерами вершин, где номера соответствуют последовательности посещения вершин при оптимальном маршруте.

### Архитектура решения Excel

Входные данные задаются следующей матрицей: Изображение выглядит как стол

Автоматически созданное описание

Рисунок 32

Рисунок 9

Где 1-5 – размерность заданной матрицы, вводим в матрицу расстояния между значениями.

**Описание алгоритма решения:**

Выполняем команду <Сервис / Поиск решения>. В окне <Параметры> отмечаем <Линейная модель>, <Неотрицательные значения>, <Автоматическое масштабирование>, В диалоговом окне укажите: в поле <Равной>: Минимальному значению; в поле <изменяя ячейки>: ячейки новой пустой матрицы; ячейки дополнительных переменных. Поле <Ограничения> должно иметь следующее содержание: новая матрица =целые числа, суммы каждых столбов и строк попеременно равны 1, значения в матрице ограничения по дополнительным переменным менее 3.

**Описание выходных данных:**

По итогам мы получаем следующее:

Во-первых, новую матрицу по методу ветвей и границ (с помощью сервиса «Поиск решения»):

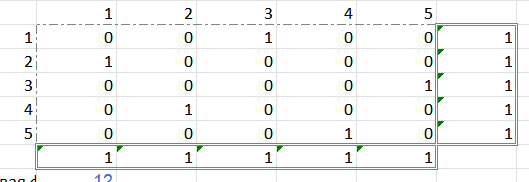


Рисунок 33

Во-вторых, целевую функцию, равную 12 (через СУММПРОИЗВ).

В-третьих, матрицу ограничений по дополнительным переменным (сервис «Поиск решения»)

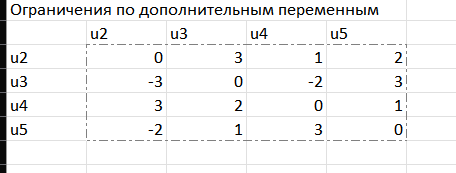


Рисунок 34

## ТЕСТИРОВАНИЕ

Проверка в excel:

Проверка 1.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 35

Проверка 2.

**Изображение выглядит как стол

Автоматически созданное описание**

Рисунок 36

Проверка 3.

**Изображение выглядит как стол

Автоматически созданное описание**

Рисунок 37

Проверка 4.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 38

Проверка 5.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 39

## ЗАКЛЮЧЕНИЕ

Задача решена двумя способами, с помощью языка программирования Python и с помощью MS Excel. Задачи универсальны и зависят от входных данных. Данные вывода выходят из условия задачи (также в MS Excel просчитывается целевая функция).

При решении задачи используются различные методы подсчета данных. Методы не влияют ни на параметры входных данных (матрица), ни на выходные данные, нужные для последующего анализа.

При решении в Python происходит оценка сходимости итерационных методов оптимизации и построение графа оптимального пути.

При решении в Excel приведен пример визуализации графа алгоритма Дейкстры:



Рисунок 40

Данная задача имеет перспективы развития визуализации представления результата в виде графиков с более понятным представлением о решении задачи.

# РЕШЕНИЕ ЗАДАЧ ПО ТЕМЕ «ТЕОРИЯ ИГР» №3-4

## ПОСТАНОВКА ЗАДАЧИ (ФИЗИЧЕСКАЯ МОДЕЛЬ)

* 1. **Решение задачи о нахождении выигрышной стратегии для антагонистической игры**

ИГРА СТРАЙКБОЛ

**Формат входных данных:**

а) Список стратегий игрока А(прячущийся):

А1: камуфляж

А2: перебежки

А3: изучение местности

б) Список стратегий игрока Б(ищущий):

Б1: звуковой локатор

Б2: тепловизор

Б3: план местности

в) матрица весовых коэффициентов:

Таблица 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Стратегии | Б1 | Б2 | Б3 | min-max |
| А1 | 90 | 10 | 50 | 10 |
| А2 | 10 | 25 | 75 | 10 |
| А3 | 75 | 50 | 20 | 20 |
| max-min | 90 | 50 | 75 |  |

Выходные данные:

а) Оптимальная чистая стратегия для игрока А - А3: изучение местности

б) Цена игры для игрока А при выборе чистой оптимальной стратегии - 20

в) Оптимальная чистая стратегия для игрока Б - Б3: план местности

г) Цена игры для игрока Б при выборе чистой оптимальной стратегии - 75

д) таблица смешанных стратегий для игрока А (указывается название стратегии и процент выделения ресурсов на нее).

Таблица 7

|  |  |
| --- | --- |
| Стратегия | |
| P1 камуфляж | 0% |
| P2 перебежки | 38% |
| P3 изучение местности | 63% |

* 1. **Решение задачи о нахождении выигрышной стратегии для биматричной игры**

ПОЛУЧЕНИЕ ПРЕМИИ ЗА ПРОЕКТ

Формат входных данных:

а) Список стратегий игрока А:

А1: молчать

А2: сказать правду

А3: сказать ложь

б) Список стратегий игрока Б:

Б1: молчать

Б2: сказать правду

Б3: сказать ложь

в) матрица весовых коэффициентов для игрока А:

Таблица 8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Матрица А | | | | |
| a | Б1 | Б2 | Б3 |  |
| А1 | 1 | 5 | 0 |  |
| А2 | 7 | 7 | 5 |  |
| А3 | 10 | 3 | 0 |  |

г) матрица весовых коэффициентов для игрока Б:

Таблица 9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Матрица Б | | | | |
| b | Б1 | Б2 | Б3 | max |
| А1 | 1 | 7 | 10 | 10 |
| А2 | 5 | 7 | 3 | 7 |
| А3 | 0 | 5 | 0 | 5 |
|  |  |  |  |  |

**Формат выходных данных:**

а) Оптимальная чистая стратегия для игрока А – А2: сказать правду

б) Цена игры для игрока А при выборе чистой оптимальной стратегии - 7

в) Оптимальная чистая стратегия для игрока Б – Б2: сказать правду

г) Цена игры для игрока Б при выборе чистой оптимальной стратегии - 7

д) Общая (суммарная) цена игры - 14

е) Количество равновесий по Нэшу - 1

ж) таблица оптимальных смешанных стратегий игрока А

Таблица 10

|  |  |
| --- | --- |
| p1 молчать | 0% |
| p2 сказать правду | 100% |
| p3 сказать ложь | 0% |

з) Цена игры для игрока А при выборе смешанной оптимальной стратегии – 5,8

и) таблица оптимальных смешанных стратегий игрока Б

Таблица 11

|  |  |
| --- | --- |
| p1 молчать | 40% |
| p2 сказать правду | 60% |
| p3 сказать ложь | 0% |

к) Цена игры для игрока Б при выборе смешанной оптимальной стратегии – 6,2

л) Общая цена игры в случае использования оптимальных стратегий - 12

* 1. **Решение задачи о принятии оптимального решения в условиях риска**

**Формат входных данных:**

а) Введите количество стратегий для компании А: 4

- Введите название 1 стратегии компании А: Продукты

- Введите название 2 стратегии компании А: Косметика

- Введите название 3 стратегии компании А: Одежда

- Введите название 4 стратегии компании А: Бытовая химия

б) Введите количество состояний природы П: 4

- Введите название 1 состояний природы П: Рецессия

- Введите название 2 состояний природы П: Стагнация

- Введите название 3 состояний природы П: Оживление

- Введите название 4 состояний природы П: Подъем

в) Введите матрицу весовых коэффициентов поэлементно:

- Введите элемент строки 1 столбца 1: 50

- Введите элемент строки 1 столбца 2: 30

- Введите элемент строки 1 столбца 3: 35

- Введите элемент строки 1 столбца 4: 45

- Введите элемент строки 2 столбца 1: 40

- Введите элемент строки 2 столбца 2: 40

- Введите элемент строки 2 столбца 3: 55

- Введите элемент строки 2 столбца 4: 40

- Введите элемент строки 3 столбца 1: 35

- Введите элемент строки 3 столбца 2: 40

- Введите элемент строки 3 столбца 3: 45

- Введите элемент строки 3 столбца 4: 55

- Введите элемент строки 4 столбца 1: 10

- Введите элемент строки 4 столбца 2: 20

- Введите элемент строки 4 столбца 3: 25

- Введите элемент строки 4 столбца 4: 35

г) Введите вектор вероятности поэлементно:

- Введите элемент строки 1 столбца 1: 0,15

- Введите элемент строки 1 столбца 2: 0,25

- Введите элемент строки 1 столбца 3: 0,5

- Введите элемент строки 1 столбца 4: 0,1

**Формат выходных данных:**

а) Оптимальная стратегия игрока А по критерию Лапласа: А2-Косметика; А3- Одежда

б) Цена игры с оптимальной стратегии по критерию Лапласа: 43.75

в) Оптимальная стратегия игрока А по критерию Баеса: А2-Косметика

г) Цена игры с оптимальной стратегии по критерию Баеса: 47.5

д) Оптимальная чистая стратегия игрока А по критерию Гермейера: А3- Одежда

е) Цена игры с оптимальной чистой стратегии по критерию Гермейера: 5.25 / 35

ж) Таблица оптимальных смешанных стратегий для игрока А по критерию Гермейера

Таблица 12

|  |  |  |  |
| --- | --- | --- | --- |
| Продукты | косметика | Одежда | Бытовая химия |
| 8% | 0% | 92% | 0% |

з) Цена игры для игрока А при выборе смешанной оптимальной стратегии по критерию Гермейера: 5.42 / 42.77

**1.4. Решение задачи о принятии оптимального решения в условиях неопределенности**

**Формат входных данных:**

а) Введите количество стратегий для компании А: 4

- Введите название 1 стратегии компании А: Продукты

- Введите название 2 стратегии компании А: Косметика

- Введите название 3 стратегии компании А: Одежда

- Введите название 4 стратегии компании А: Бытовая химия

б) Введите количество состояний природы П: 4

- Введите название 1 состояний природы П: Рецессия

- Введите название 2 состояний природы П: Стагнация

- Введите название 3 состояний природы П: Оживление

- Введите название 4 состояний природы П: Подъем

в) Введите матрицу весовых коэффициентов поэлементно:

- Введите элемент строки 1 столбца 1: 50

- Введите элемент строки 1 столбца 2: 30

- Введите элемент строки 1 столбца 3: 35

- Введите элемент строки 1 столбца 4: 45

- Введите элемент строки 2 столбца 1: 40

- Введите элемент строки 2 столбца 2: 40

- Введите элемент строки 2 столбца 3: 40

- Введите элемент строки 2 столбца 4: 40

- Введите элемент строки 3 столбца 1: 70

- Введите элемент строки 3 столбца 2: 40

- Введите элемент строки 3 столбца 3: 35

- Введите элемент строки 3 столбца 4: 50

- Введите элемент строки 4 столбца 1: 10

- Введите элемент строки 4 столбца 2: 20

- Введите элемент строки 4 столбца 3: 25

- Введите элемент строки 4 столбца 4: 35

**Формат выходных данных:**

а) Оптимальная стратегия игрока А по критерию пессимизма: А4 – Бытовая химия

б) Цена игры оптимальной стратегии по критерию пессимизма: 10

в) Оптимальная стратегия игрока А по критерию оптимизма: А1 - Продукты

г) Цена игры оптимальной стратегии по критерию оптимизма: 50

д) Оптимальная стратегия игрока А по критерию Вальда: А2 - Косметика

е) Цена игры оптимальной стратегии по критерию Вальда: 40

ж) Линейная свертка склонности к риску по критерию Гурвица

Таблица 13

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Alpha | 0 | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 | 0,6 | 0,7 | 0,8 | 0,9 | 1 |
| А1 | 30 | 32 | 34 | 36 | 38 | 40 | 42 | 44 | 46 | 48 | 50 |
| А2 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| А3 | 35 | 38,5 | 42 | 45,5 | 49 | 52,5 | 56 | 59,5 | 63 | 66,5 | 70 |
| А4 | 10 | 12,5 | 15 | 17,5 | 20 | 22,5 | 25 | 27,5 | 30 | 32,5 | 35 |
|  |  |  |  |  |  |  |  |  |  |  |  |
| max | 40 | 40 | 42 | 45,5 | 49 | 52,5 | 56 | 59,5 | 63 | 66,5 | 70 |

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

* 1. **Математическая модель антагонистической игры**

Рассмотрим игру двух лиц, интересы которых противоположны. Такие игры называют антагонистическими играми двух лиц. В этом случае выигрыш одного игрока равен проигрышу второго, и можно описать только одного из игроков.  
Предполагается, что каждый игрок может выбрать только одно из конечного множества своих действий. Выбор действия называют выбором стратегии игрока.  
Если каждый из игроков выбрал свою стратегию, то эту пару стратегий называют ситуацией игры. Следует заметить, каждый игрок знает, какую стратегию выбрал его противник, т.е. имеет полную информацию о результате выбора противника.  
Чистой стратегией игрока I является выбор одной из n строк матрицы выигрышей А, а чистой стратегией игрока II является выбор одного из столбцов этой же матрицы.

1. Проверяем, имеет ли платежная матрица седловую точку. Если да, то выписываем решение игры в чистых стратегиях.  
Считаем, что игрок I выбирает свою стратегию так, чтобы получить максимальный свой выигрыш, а игрок II выбирает свою стратегию так, чтобы минимизировать выигрыш игрока I.

Таблица 14

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Игроки | B1 | B2 | B3 | a = min(Ai) |
| A1 | 5 | 4 | 5 | 4 |
| A2 | 2 | 4 | 3 | 2 |
| A3 | 1 | 2 | 4 | 1 |
| b = max(Bi) | 5 | 4 | 5 |  |

Находим гарантированный выигрыш, определяемый нижней ценой игры a = max(ai) = 4, которая указывает на максимальную чистую стратегию A1.  
Верхняя цена игры b = min(bj) = 4.  
Седловая точка (1, 2) указывает решение на пару альтернатив (A1,B2). Цена игры равна 4.  
**2. Проверяем платежную матрицу на доминирующие строки и доминирующие столбцы**.  
Иногда на основании простого рассмотрения матрицы игры можно сказать, что некоторые чистые стратегии могут войти в оптимальную смешанную стратегию лишь с нулевой вероятностью.  
Говорят, что i-я стратегия 1-го игрока доминирует его k-ю стратегию, если aij ≥ akj для всех j Э N и хотя бы для одного j aij > akj. В этом случае говорят также, что i-я стратегия (или строка) – доминирующая, k-я – доминируемая.  
Говорят, что j-я стратегия 2-го игрока доминирует его l-ю стратегию, если для всех j Э M aij ≤ ail и хотя бы для одного i aij < ail. В этом случае j-ю стратегию (столбец) называют доминирующей, l-ю – доминируемой.  
Стратегия A1 доминирует над стратегией A2 (все элементы строки 1 больше или равны значениям 2-ой строки), следовательно, исключаем 2-ую строку матрицы. Вероятность p2 = 0.

Таблица 15

|  |  |  |
| --- | --- | --- |
| 5 | 4 | 5 |
| 1 | 2 | 4 |

С позиции проигрышей игрока В стратегия B1 доминирует над стратегией B3 (все элементы столбца 1 меньше элементов столбца 3), следовательно, исключаем 3-й столбец матрицы. Вероятность q3 = 0.

Таблица 16

|  |  |
| --- | --- |
| 5 | 4 |
| 1 | 2 |

Мы свели игру 3 x 3 к игре 2 x 2.  
**3. Находим решение игры в смешанных стратегиях**.  
Математические модели пары двойственных задач линейного программирования можно записать так:  
найти минимум функции F(x) при ограничениях (для игрока II):  
5x1+x2 ≥1  
4x1+2x2 ≥1  
F(x)=x1+x2 →min  
найти максимум функции Z(y) при ограничениях (для игрока I):  
5y1+4y2 ≤1  
y1+2y2 ≤1  
Z(y)=y1+y2 →max  
Решим прямую задачу линейного программирования симплексным методом, с использованием симплексной таблицы.  
Определим максимальное значение целевой функции Z(Y) = y1+y2 при следующих условиях-ограничений.  
5y1+4y2≤1  
y1+2y2≤1  
Для построения первого опорного плана систему неравенств приведем к системе уравнений путем введения дополнительных переменных (переход к канонической форме).  
5y1+4y2+y3 =1  
y1+2y2+y4 =1  
Решим систему уравнений относительно базисных переменных: y3, y4  
Полагая, что свободные переменные равны 0, получим первый опорный план:  
Y0 = (0,0,1,1)

Таблица 17

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Базис | B | y1 | y2 | y3 | y4 |
| y3 | 1 | 5 | 4 | 1 | 0 |
| y4 | 1 | 1 | 2 | 0 | 1 |
| Z(Y0) | 0 | -1 | -1 | 0 | 0 |

Переходим к основному алгоритму симплекс-метода.  
**Итерация №0**.  
Текущий опорный план не оптимален, так как в индексной строке находятся отрицательные коэффициенты.  
В качестве ведущего выберем столбец, соответствующий переменной y2, так как это наибольший коэффициент по модулю.  
Вычислим значения Di по строкам как частное от деления: bi / ai2  
и из них выберем наименьшее:  
min(1:4,1:2)= 1/4  
Следовательно, 1-ая строка является ведущей.  
Разрешающий элемент равен (4) и находится на пересечении ведущего столбца и ведущей строки.

Таблица 18

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | B | y1 | y2 | y3 | y4 | min |
| y3 | 1 | 5 | 4 | 1 | 0 | 1/4 |
| y4 | 1 | 1 | 2 | 0 | 1 | 1/2 |
| Z(Y1) | 0 | -1 | -1 | 0 | 0 |  |

Формируем следующую часть симплексной таблицы. Вместо переменной y3 в план 1 войдет переменная y2.  
Получаем новую симплекс-таблицу:

Таблица 19

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Базис | B | y1 | y2 | y3 | y4 |
| y2 | 1/4 | 5/4 | 1 | 1/4 | 0 |
| y4 | 1/2 | -3/2 | 0 | -1/2 | 1 |
| Z(Y1) | 1/4 | 1/4 | 0 | 1/4 | 0 |

Конец итераций: индексная строка не содержит отрицательных элементов - найден оптимальный план  
Среди значений индексной строки нет отрицательных. Поэтому эта таблица определяет оптимальный план задачи.  
Окончательный вариант симплекс-таблицы:

Таблица 20

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Базис | B | y1 | y2 | y3 | y4 |
| y2 | 1/4 | 5/4 | 1 | 1/4 | 0 |
| y4 | 1/2 | -3/2 | 0 | -1/2 | 1 |
| Z(Y2) | 1/4 | 1/4 | 0 | 1/4 | 0 |

Оптимальный план можно записать так:  
y1 =0,y2 = 1/4  
Z(Y)=1\*0+1\*1/4 = 1/4  
Используя последнюю итерацию прямой задачи найдем, оптимальный план двойственной задачи.  
x1=1/4,x2=0  
Это же решение можно получить, применив теоремы двойственности.  
Из теоремы двойственности следует, что X = C\*A-1.  
Составим матрицу A из компонентов векторов, входящих в оптимальный базис.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A = (A2, A4) = | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 4 | 0 | | 2 | 1 | |  | |  |

Определив обратную матрицу D = А-1 через алгебраические дополнения, получим:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D = A-1 = | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 1/4 | 0 | | -1/2 | 1 | |  | |  |

Как видно из последнего плана симплексной таблицы, обратная матрица A-1 расположена в столбцах дополнительных переменных.  
Тогда X = C\*A-1 =

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (1, 0) x | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 1/4 | 0 | | -1/2 | 1 | |  | | = (1/4;0) |

Оптимальный план двойственной задачи равен:  
x1 = 1/4,x2 =0  
F(X)=1\*1/4+1\*0= 1/4  
Цена игры будет равна g = 1/F(x), а вероятности применения стратегий игроков:  
qi =g\*yi;pi =g\*xi.  
Цена игры: g=1: 1/4 =4

p1 =4\*1/4 =1  
p2 =4\*0=0  
Оптимальная смешанная стратегия игрока I: P=(1;0)  
q1 =4\*0=0  
q2 =4\*1/4 =1  
Оптимальная смешанная стратегия игрока II: Q=(0;1); Цена игры: v=4  
4. Проверим правильность решения игры с помощью критерия оптимальности стратегии.  
∑aijqj ≤v  
∑aijpi ≥v  
M(P1;Q)=(5\*0)+(4\*1)=4=v  
M(P2;Q)=(1\*0)+(2\*1)=2≤v  
M(P;Q1)=(5\*1)+(1\*0)=5≥v  
M(P;Q2)=(4\*1)+(2\*0)=4=v  
Все неравенства выполняются как равенства или строгие неравенства, следовательно, решение игры найдено верно.  
Поскольку из исходной матрицы были удалены строки и столбцы, то найденные векторы вероятности можно записать в виде:  
P(1,0,0)  
Q(0,1,0)

* 1. **Математическая модель биматричной игры**

Решение:  
В каждом столбце матрицы A найдем максимальный элемент. Эти элементы подчеркнуты в матрице A. Их положение соответствует приемлемым ситуациям 1-го игрока, когда второй игрок выбрал стратегию j соответственно.  
Затем в каждой строке матрицы B выберем наибольший элемент. Эти элементы подчеркнуты в матрице B. Их положение будет определять приемлемые ситуации 2-го игрока, когда первый игрок выбрал стратегию i соответственно.  
Платежная матрица игрока А:

Таблица 21

|  |  |  |
| --- | --- | --- |
| 3 | 2 | **5** |
| **4** | **3** | 2 |
| 1 | **3** | 4 |

Позиции максимумов в столбцах матрицы А: (2,1), (2,2), (3,2), (1,3)  
Платежная матрица игрока B:

Таблица 22

|  |  |  |
| --- | --- | --- |
| 1 | 2 | **4** |
| 2 | 3 | **5** |
| 3 | **4** | 2 |

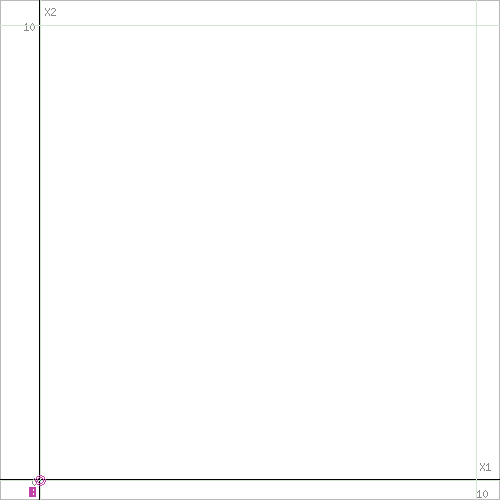
Позиции максимумов в строках матрицы В: (1,3), (2,3), (3,2)  
Пересечение этих двух множеств: (3;2), (1;3),  
Таким образом, найдены 2 равновесные ситуации по Нэшу (3;2), (1;3). Эти ситуации оказались оптимальные по Парето для обоих игроков.  
В равновесной ситуации (3,2) игрок 1 выигрывает 3 единиц, а игрок 2 - 4 единицы.  
В равновесной ситуации (1,3) игрок 1 выигрывает 5 единиц, а игрок 2 - 4 единицы.  
Множество всех реализуемых векторов выигрышей для рассматриваемо игры имеет вид, изображенный на рисунке.  


Рисунок 1

Как видим, имеется только одна ситуация, оптимальная по Парето: H1=, H2=  
Вычислим цены игр для матричных игр: va=211/12, vb=23/4. Следовательно, функция полезности по Нэшу примет вид:  
U=(H1-211/12)(H2-23/4)

* 1. **Математическая модель природные игры с рисками**

Исходные данные:

Таблица 23

|  |  |  |
| --- | --- | --- |
| 5 | 4 | 3 |
| 2 | 1 | 4 |
| 3 | 2 | 5 |

**Критерий Байеса**.

По критерию Байеса за оптимальные принимается та стратегия (чистая) Ai, при которой максимизируется средний выигрыш a или минимизируется средний риск r. Считаем значения ∑(aijpj)

∑(a1,jpj)=5\*0.33+4\*0.33+3\*0.33=3.96  
∑(a2,jpj)=2\*0.33+1\*0.33+4\*0.33=2.31  
∑(a3,jpj) = 3\*0.33 + 2\*0.33 + 5\*0.33 = 3.3

Таблица 24

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | ∑(aijpj) |
| A1 | 1.65 | 1.32 | 0.99 | 3.96 |
| A2 | 0.66 | 0.33 | 1.32 | 2.31 |
| A3 | 0.99 | 0.66 | 1.65 | 3.3 |
| pj | 0.33 | 0.33 | 0.33 |  |

Выбираем из (3.96; 2.31; 3.3) максимальный элемент max=3.96  
Вывод: выбираем стратегию N=1.  
**Критерий Вальда**.

По критерию Вальда за оптимальную принимается чистая стратегия, которая в наихудших условиях гарантирует максимальный выигрыш, т.е.  
a = max(min aij)  
Критерий Вальда ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации.

Таблица 25

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | min(aij) |
| A1 | 5 | 4 | 3 | 3 |
| A2 | 2 | 1 | 4 | 1 |
| A3 | 3 | 2 | 5 | 2 |

Выбираем из (3; 1; 2) максимальный элемент max=3  
Вывод: выбираем стратегию N=1.  
**Критерий Севиджа**.  
Критерий минимального риска Севиджа рекомендует выбирать в качестве оптимальной стратегии ту, при которой величина максимального риска минимизируется в наихудших условиях, т.е. обеспечивается:  
a = min(max rij)  
Критерий Сэвиджа ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации.  
Находим матрицу рисков.  
Риск – мера несоответствия между разными возможными результатами принятия определенных стратегий. Максимальный выигрыш в j-м столбце bj = max(aij) характеризует благоприятность состояния природы.  
1. Рассчитываем 1-й столбец матрицы рисков.  
r11 =5-5=0; r21 =5-2=3; r31 =5-3=2;

1. Рассчитываем 2-й столбец матрицы рисков.  
   r12 =4-4=0; r22 =4-1=3; r32 =4-2=2;

3. Рассчитываем 3-й столбец матрицы рисков.  
r13 = 5 - 3 = 2; r23 = 5 - 4 = 1; r33 = 5 - 5 = 0;

Таблица 26

|  |  |  |  |
| --- | --- | --- | --- |
| Ai | П1 | П2 | П3 |
| A1 | 0 | 0 | 2 |
| A2 | 3 | 3 | 1 |
| A3 | 2 | 2 | 0 |

Результаты вычислений оформим в виде таблицы.

Таблица 27

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | max(aij) |
| A1 | 0 | 0 | 2 | 2 |
| A2 | 3 | 3 | 1 | 3 |
| A3 | 2 | 2 | 0 | 2 |

Выбираем из (2; 3; 2) минимальный элемент min=2  
Вывод: выбираем стратегию N=1.

Проведение идеального эксперимента.

В крайнем правом столбце рассчитаем средний риск.

Таблица 28

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | ri |
| A1 | 0 | 0 | 2 | 0.66 |
| A2 | 3 | 3 | 1 | 2.31 |
| A3 | 2 | 2 | 0 | 1.32 |

Минимальное значение средних рисков равно 0.66. Следовательно, выше этой цены планирование эксперимента становится нецелесообразным.  
**Критерий Гурвица**.

Критерий Гурвица является критерием пессимизма - оптимизма. За оптимальную принимается та стратегия, для которой выполняется соотношение:  
max(si)  
где si =ymin(aij)+(1-y)max(aij)

При y = 1 получим критерий Вальде, при y = 0 получим – оптимистический критерий (максимакс).

Критерий Гурвица учитывает возможность как наихудшего, так и наилучшего для человека поведения природы. Как выбирается y? Чем хуже последствия ошибочных решений, тем больше желание застраховаться от ошибок, тем y ближе к 1.

Рассчитываем si.

s1 =0.5\*3+(1-0.5)\*5=4  
s2 =0.5\*1+(1-0.5)\*4=2.5  
s3 = 0.5\*2+(1-0.5)\*5 = 3.5

Таблица 29

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | min(aij) | max(aij) | y min(aij) + (1-y)max(aij) |
| A1 | 5 | 4 | 3 | 3 | 5 | 4 |
| A2 | 2 | 1 | 4 | 1 | 4 | 2.5 |
| A3 | 3 | 2 | 5 | 2 | 5 | 3.5 |

Выбираем из (4; 2.5; 3.5) максимальный элемент max=4  
Вывод: выбираем стратегию N=1.  
Таким образом, в результате решения статистической игры по различным критериям чаще других рекомендовалась стратегия A1.

* 1. **Природная игра с минимальным риском (неопределенные условия)**

Исходные данные:

Таблица 30

|  |  |  |
| --- | --- | --- |
| 5 | 4 | 3 |
| 2 | 1 | 4 |
| 3 | 2 | 5 |

Поскольку необходимо минимизировать затраты, то модифицируем матрицу умножением всех элементов на (-1) и затем сложением их с максимальным элементом матрицы (5) так, чтобы матрица не содержала бы отрицательных элементов. Тем самым сводим решение к поиску максимальной функции.

Таблица 31

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 2 |
| 3 | 4 | 1 |
| 2 | 3 | 0 |

**Критерий Байеса**.

По критерию Байеса за оптимальные принимается та стратегия (чистая) Ai, при которой максимизируется средний выигрыш a или минимизируется средний риск r.  
Считаем значения ∑(aijpj)  
∑(a1,jpj)=0\*0.33+1\*0.33+2\*0.33=0.99  
∑(a2,jpj)=3\*0.33+4\*0.33+1\*0.33=2.64  
∑(a3,jpj) = 2\*0.33 + 3\*0.33 + 0\*0.33 = 1.65

Таблица 32

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | ∑(aijpj) |
| A1 | 0 | 0.33 | 0.66 | 0.99 |
| A2 | 0.99 | 1.32 | 0.33 | 2.64 |
| A3 | 0.66 | 0.99 | 0 | 1.65 |
| pj | 0.33 | 0.33 | 0.33 |  |

Выбираем из (0.99; 2.64; 1.65) максимальный элемент max=2.64  
Вывод: выбираем стратегию N=2.  
**Критерий Лапласа**.

Если вероятности состояний природы правдоподобны, для их оценки используют принцип недостаточного основания Лапласа, согласно которого все состояния природы полагаются равновероятными, т.е.:  
q1 = q2 = ... = qn = 1/n.  
qi = 1/3

Таблица 33

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | ∑(aij) |
| A1 | 0 | 0.333 | 0.667 | 1 |
| A2 | 1 | 1.333 | 0.333 | 2.667 |
| A3 | 0.667 | 1 | 0 | 1.667 |
| pj | 0.333 | 0.333 | 0.333 |  |

Выбираем из (1; 2.67; 1.67) максимальный элемент max=2.67  
Вывод: выбираем стратегию N=2.  
**Критерий Вальда**.

По критерию Вальда за оптимальную принимается чистая стратегия, которая в наихудших условиях гарантирует максимальный выигрыш, т.е.  
a = max(min aij)

Критерий Вальда ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации.

Таблица 34

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | min(aij) |
| A1 | 0 | 1 | 2 | 0 |
| A2 | 3 | 4 | 1 | 1 |
| A3 | 2 | 3 | 0 | 0 |

Выбираем из (0; 1; 0) максимальный элемент max=1  
Вывод: выбираем стратегию N=2.  
**Критерий Севиджа**.

Критерий минимального риска Севиджа рекомендует выбирать в качестве оптимальной стратегии ту, при которой величина максимального риска минимизируется в наихудших условиях, т.е. обеспечивается:  
a = min(max rij)  
Критерий Сэвиджа ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации.  
Находим матрицу рисков.

**Риск** – мера несоответствия между разными возможными результатами принятия определенных стратегий. Максимальный выигрыш в j-м столбце bj = max(aij) характеризует благоприятность состояния природы.  
1. Рассчитываем 1-й столбец матрицы рисков.

r11 =3-0=3; r21 =3-3=0; r31 =3-2=1;

2. Рассчитываем 2-й столбец матрицы рисков.

r12 =4-1=3; r22 =4-4=0; r32 =4-3=1;

3. Рассчитываем 3-й столбец матрицы рисков.

r13 = 2 - 2 = 0; r23 = 2 - 1 = 1; r33 = 2 - 0 = 2;

Таблица 35

|  |  |  |  |
| --- | --- | --- | --- |
| Ai | П1 | П2 | П3 |
| A1 | 3 | 3 | 0 |
| A2 | 0 | 0 | 1 |
| A3 | 1 | 1 | 2 |

Результаты вычислений оформим в виде таблицы.

Таблица 36

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | max(aij) |
| A1 | 3 | 3 | 0 | 3 |
| A2 | 0 | 0 | 1 | 1 |
| A3 | 1 | 1 | 2 | 2 |

Выбираем из (3; 1; 2) минимальный элемент min=1  
Вывод: выбираем стратегию N=2.

**Проведение идеального эксперимента**.

В крайнем правом столбце рассчитаем средний риск.

Таблица 37

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | ri |
| A1 | 3 | 3 | 0 | 1.98 |
| A2 | 0 | 0 | 1 | 0.33 |
| A3 | 1 | 1 | 2 | 1.32 |

Минимальное значение средних рисков равно 0.33. Следовательно, выше этой

цены планирование эксперимента становится нецелесообразным.  
**Критерий Гурвица**.

Критерий Гурвица является критерием пессимизма - оптимизма. За оптимальную принимается та стратегия, для которой выполняется соотношение:  
max(si)  
где si = y min(aij) + (1-y)max(aij)

При y = 1 получим критерий Вальде, при y = 0 получим – оптимистический критерий (максимакс).

Критерий Гурвица учитывает возможность как наихудшего, так и наилучшего для человека поведения природы. Как выбирается y? Чем хуже последствия ошибочных решений, тем больше желание застраховаться от ошибок, тем y ближе к 1.  
Рассчитываем si.  
s1 =0.5\*0+(1-0.5)\*2=1  
s2 =0.5\*1+(1-0.5)\*4=2.5  
s3 = 0.5\*0+(1-0.5)\*3 = 1.5

Таблица 38

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ai | П1 | П2 | П3 | min(aij) | max(aij) | y min(aij) + (1-y)max(aij) |
| A1 | 0 | 1 | 2 | 0 | 2 | 1 |
| A2 | 3 | 4 | 1 | 1 | 4 | 2.5 |
| A3 | 2 | 3 | 0 | 0 | 3 | 1.5 |

Выбираем из (1; 2.5; 1.5) максимальный элемент max=2.5

Вывод: выбираем стратегию N=2.

Таким образом, в результате решения статистической игры по различным критериям чаще других рекомендовалась стратегия A2.

## АЛГОРИТМЫ

* 1. **Алгоритм решения антагонистической игры в Excel**

**Параметры входных данных:**

Входными данными при решении данной задачи является искомая матрица для решения.

**Алгоритм решения:**

Для начала находится целевая функция с максимумом (с помощью сервиса Поиск решений с ограничением). Далее с помощью целевой функции определяется цена игры (1/целевую функцию). Затем с помощью целевой функции сервисом Поиск решения создается вспомогательный вектор. Далее с помощью формулы умножения вспомогательного вектора на цену игры определяет вектор решения второго игрока.

Далее аналогично с помощью сервиса Поиск решений находится целевая функция с минимумом. Определяется цена игры аналогичной формулой. Сервисом Поиск решений находится вспомогательный вектор, с помощью которого формулой произведения цены игры на вспомогательный вектор определяется вектор решения первого игрока.

**Параметры выходных данных:**

Выходными данными из данной матрицы с помощью сервиса Поиск решений мы имеем две целевые функций, с помощью самостоятельного решения – векторы решения первого и второго игроков.

Алгоритм решения антагонистической игры в Excel

**Параметры входных данных:**

Входными данными при решении данной задачи являются две искомые матрицы для решения.

**Алгоритм решения:**

Для начала находится целевая функция с максимумом (с помощью сервиса Поиск решений с ограничениями). С помощью целевой функции определяются переменные и вспомогательные матрицы. Далее с помощью вспомогательной матрицы находятся цены игр игроков (с помощью функции Excel СУММ().

Действия выполняются для обеих матриц.

**Параметры выходных данных:**

Выходными данными из данной матрицы с помощью сервиса Поиск решений мы имеем целевую функцию матриц и далее самостоятельно определяем с помощью формулы цены игр двух игроков.

1. Алгоритм решения природной игры в Excel с рисками

**Параметры входных данных:**

Входными данными при решении данной задачи является искомая матрица для решения.

**Алгоритм решения:**

Решение данной игры производится по четырем критериям (Байеса, Вальда, Севиджа, Гурвица) с использованием первоначальной матрицы и соответствующих формул. По критериям Байеса, Севиджа и Гурвица составляются на основе формул вспомогательные матрицы. По Вальду вспомогательная матрица не нужна, достаточно вектора. В итоге, на основе вспомогательный данных с помощью функции Excel МАКС() находим результат по четырем критериям (максимально ожидаемый выигрыш)

**Параметры выходных данных:**

Выходными данными из данной матрицы с помощью формул критериев и функции МАКС() считаются результаты четырех критериев.

1. Алгоритм решения природной игры с минимизацией рисков (в условиях неопределенности) Excel

**Параметры входных данных:**

Входными данными при решении данной задачи является искомая матрица для решения.

**Алгоритм решения:**

Решение данной игры производится по четырем критериям (Байеса, Вальда, Севиджа, Гурвица) с использованием первоначальной матрицы и соответствующих формул. По критериям Байеса, Севиджа и Гурвица составляются на основе формул вспомогательные матрицы. По Вальду вспомогательная матрица не нужна, достаточно вектора. В итоге, на основе вспомогательный данных с помощью функции Excel МАКС() находим результат по четырем критериям (минимальный риск).

**Параметры выходных данных:**

Выходными данными из данной матрицы с помощью формул критериев и функции МАКС() считаются результаты четырех критериев.

1. Алгоритм решения биматричных игр в Python

**Описание входных данных:** искомая матрица, при вводе 1 данные матрицы заполняются вручную, а при вводе 2 – автоматически генерируются

**Описание работы алгоритма:**

Алгоритм работает благодаря библиотекам «numpy» и «nashpy». Для каждого из игроков, в зависимости от выбранного типа ввода данных, выбирается стратегия, коэффициент (строка и колонка матрицы ). Затем, благодаря «random.randint», матрица каждого из игроков заполняется. После получения заполненных матриц обоих игроков, благодаря «nash.Game», находится оптимальная стратегия для обоих игроков, цена для обоих игроков, общая цена игры и количество равновесий. В заключении, если количество элементов «len» решения больше одного, находится таблица смежных стратегий игроков. Затем для обоих игроков находится стратегия, наиболее оптимальная из найденных. После этого выводится цена игры для обоих игроков при выборе смешанной оптимальной стратегии, из чего следует ответ.

**Описание выходных данных:**

Выходными данными являются два вектора оптимальной цены для каждого игрока.

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

**Варианты использования алгоритма решения биматричных игр**

**Python:**

**Ввод данных:**

Для начала необходимо ввести количество стратегий:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2

Затем вводится вариант ввода данных матрицы:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3

1. При помощи клавиатуры: последовательно вводим стратегию и коэффициенты для обоих игроков

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4

1. Случайные числа: автоматический ввод случайно сгенерированных данных

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5

**Алгоритм решения:**

После выбора стратегий и коэффициентов для обоих игроков, используя библиотеку «nash», матрица каждого из игроков используется для нахождения искомых решение ( равновесие ).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6

Затем, выводятся следующие показатели: оптимальная стратегия и цена для обоих игроков, общая цена игры и искомое решение ( равновесие ).

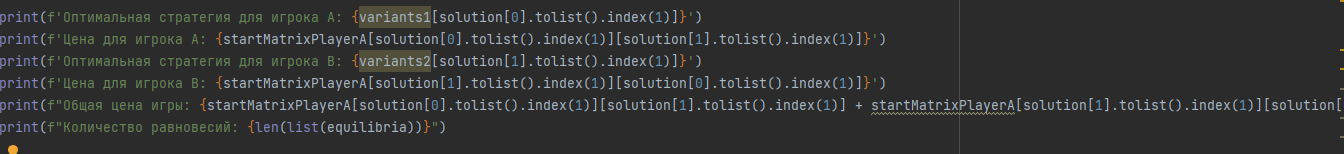


Рисунок 7

В случае, если находится несколько равновесий, будет выбрано наиболее оптимальное решение из предложенных

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8

Выходные данные:

В итоге имеем оптимальную стратегию и цену для каждого игрока, общую цену, оптимальную цену и количество равновесий. Для проверки: количество равновесий на выходе всегда должно быть равно 1.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9

**Excel:**

**Ввод данных:**

Формат входных данных:

а) Список стратегий игрока А:

А1: молчать

А2: сказать правду

А3: сказать ложь

б) Список стратегий игрока Б:

Б1: молчать

Б2: сказать правду

Б3: сказать ложь

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 10

**Алгоритм решения:**

Создание матрицы весовых коэффициентов, благодаря функции «Поиск решения»

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 11

Выходные данные:

Таблица наиболее подходящей стратегии и цена игры для каждого игрока и общая цена игры

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 12

*Изображение выглядит как стол

Автоматически созданное описание*

Рисунок 13

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 14

**

Рисунок 15

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 16

**Применение биматричных игр:**

Биматричные игры могут применяться в следующих играх и стратегиях.

1. «БОРЬБА ЗА РЫНКИ».

Небольшая фирма (игрок А) намерена сбыть партию товара на одном из двух рынков, контролируемых другой, более крупной фирмой (игрок В). Для этого фирма A готова предпринять на одном из рынков соответствующие приготовления (например, развернуть рекламную компанию). Господствующая на рынках фирма B может попытаться воспрепятствовать этому, приняв на одном из рынков предупредительные меры (разумеется, в рамках закона). Не встречая противодействия на рынке, фирма A захватывает его; при наличии препятствий — терпит поражение. Будем считать для определенности, что проникновение фирмы A на первый рынок более выгодно для нее, нежели проникновение на второй. Естественно также считать, что и борьба за первый рынок потребует вложения больших средств. Например, победа фирмы A на первом рынке принесет ей вдвое больший выигрыш, чем победа на втором, но зато и поражение при попытке освоиться на первом рынке полностью разорит ее, избавив фирму B от конкурента. Что же касается второго рынка, то при поражении фирмы A ее потери будут не столь разорительны, но и победа принесет немного. Таким образом, у фирмы A две стратегии: Α1 — «выбор первого рынка», Α2 — «выбор второго рынка». Такие же стратегии и у фирмы B: Β1 — «выбор первого рынка», B2 – «выбор второго рынка».

1. «НЕФТЕДОБЫЧА».

Рассмотрим две нефтедобывающие страны, назовем их, скажем, A и В. Эти две страны могут кооперироваться (К), договариваясь об объемах ежедневной добычи нефти, «заморозив», например, добычу на уровне 2 млн. баррелей в день. С другой стороны, страны могут действовать «не кооперативно» (Н), добывая, например, по 4 млн. баррелей нефти в день. Прибыли стран в зависимости от их объемов добычи нефти следующие.

1. «ЯСТРЕБ—ГОЛУБЬ».

Предположим, что в некоторой популяции соперничающие индивидуумы в общении между собой применяют только две стратегии – стратегию Ястреба и стратегию Голубя. Эти названия используются здесь в том смысле, в каком их обычно применяют к людям, и, вообще говоря, никак не связаны с особенностями биологии соответствующих птиц (в действительности голуби– довольно агрессивные птицы). Ястребы всегда дерутся неистово и безудержно, отступая лишь при серьезных ранениях. Голуби же ограничиваются угрозами, с достоинством соблюдая все условности, и никогда не наносят противнику повреждений. Если Ястреб сталкивается с Голубем, то Голубь быстро убегает, оставаясь, таким образом, невредимым. Если Ястреб дерется с Ястребом, то драка продолжается до тех пор, пока один из соперников не получит серьезных повреждений или не будет убит. Если же Голубь встречается с Голубем, то ни один из них не страдает. Голуби долго выступают друг перед другом, принимая разные позы, пока один из них не устанет или не решит, что ему не стоит продолжать противостояние, а лучше отступить. Будем исходить из предположения, что индивидуум не может заранее узнать, с кем ему предстоит драться (с Ястребом или с Голубем), и решить, какую стратегию из двух указанных (стратегию Ястреба или стратегию Голубя) он выберет для себя. Индивидуум обнаруживает это только в момент начала поединка и, следовательно, опытом прошлых встреч с определенными индивидуумами воспользоваться не может, так как попросту не помнит о них.

1. БИМАТРИЧНЫЕ ИГРЫ 2x2. СИТУАЦИЯ РАВНОВЕСИЯ.

Здесь мы рассмотрим случай, когда у каждого из игроков имеется ровно две стратегии, то есть случай: m = n = 2. Поэтому выпишем приведенные выше формулы для вычисления средних выигрышей H1 и H2 игроков A и B именно для такого случая.

* 1. **Варианты использования алгоритма решения антагонистической игры**

**Excel:**

**Входные данные:**

Входными данными при решении данной задачи является искомая матрица для решения.

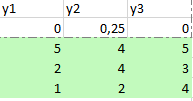


Рисунок 17

**Алгоритм решения**

Для начала находится целевая функция с максимумом (с помощью сервиса Поиск решений с ограничением).



Рисунок 18

Далее с помощью целевой функции определяется цена игры (1/целевую функцию).



Рисунок 19

Затем с помощью целевой функции сервисом Поиск решения создается вспомогательный вектор.



Рисунок 20

Далее с помощью формулы умножения вспомогательного вектора на цену игры определяет вектор решения второго игрока.



Рисунок 21

Далее аналогично с помощью сервиса Поиск решений находится целевая функция с минимумом. Определяется цена игры аналогичной формулой. Сервисом Поиск решений находится вспомогательный вектор, с помощью которого формулой произведения цены игры на вспомогательный вектор определяется вектор решения первого игрока.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 22

**Выходные данные:**

Выходными данными из данной матрицы с помощью сервиса Поиск решений мы имеем две целевые функций, с помощью самостоятельного решения – векторы решения первого и второго игроков.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 23

**

Рисунок 24

**Применение антагонистических игр**

Антагонистические игры используются в:

1. ИГРА «СРАВНЕНИЕ МОНЕТ».

Каждый из двух игроков одновременно и независимо друг от друга накрывает рукой монету вверх гербом (эта сторона монеты по-другому называется «Орел») или вверх числом (эта сторона монеты по-другому называется «Решка»).

1. ИГРА «ПОСТАВКА ТОВАРОВ».

На каждой из двух баз ассортиментный минимум составляет один и тот же набор из n видов товаров. Каждая база должна поставить в свой магазин только одни из этих n видов товаров.

Магазины, назовем их «магазин A» и «магазин B», конкурируют между собой. Один и тот же вид товара продается в обоих магазинах по одной и той же цене. Однако товар, поставляемый в «магазин B», более высокого качества. Если «магазин A» завезет с базы товар i-го вида, а «магазин B» завезет с базы товар j-го вида, отличный от товара i-го вида (то есть i ≠ j), то товар i-го вида в «магазине A» будет пользоваться спросом, и «магазин A» получит прибыль Ci у.е.

Если же в «магазин A» и «магазин B» завезены товары одинакового вида (то есть i = j), то товар i-го вида в «магазине A» не будет пользоваться спросом, так как такой же товар, но более высокого качества, продается в «магазине B» по такой же цене. В этом случае (i=j) «магазин A» получит убытки Dj у.е. по транспортировке, хранению и, возможно, порче товара i-го вида в этом магазине. Требуется составить платежную матрицу игры.

1. РЕШЕНИЕ ИГРЫ В ЧИСТЫХ СТРАТЕГИЯХ. СИТУАЦИИ РАВНОВЕСИЯ. СЕДЛОВЫЕ ТОЧКИ.

Рассмотрим матричную игру, представленную матрицей выигрышей m x n, где число строк равно числу m, а число столбцов равно числу n. Применим принцип получения максимального гарантированного результата при наихудших условиях. Игрок 1 стремится принять такую стратегию, которая должна обеспечить максимальный проигрыш игрока 2. Соответственно игрок 2 стремится принять стратегию, обеспечивающую минимальный выигрыш игрока 1.

* 1. **Варианты использования алгоритма решения игр с природой**

**Excel:**

**Входные данные:**

Входными данными при решении данной задачи является искомая матрица для решения

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 25

**Алгоритм решения:**

Решение данной игры производится по четырем критериям (Байеса, Вальда, Севиджа, Гурвица) с использованием первоначальной матрицы и соответствующих формул. По критериям Байеса, Севиджа и Гурвица составляются на основе формул вспомогательные матрицы.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 26

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 27

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 28

По Вальду вспомогательная матрица не нужна, достаточно вектора.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 29

В итоге, на основе вспомогательный данных с помощью функции Excel МАКС() находим результат по четырем критериям (максимально ожидаемый выигрыш)

Критерий Байеса:



Рисунок 30

Критерий Вальда:



Рисунок 31

Критерий Севиджа:



Рисунок 32

Критерий Гурвинца:



Рисунок 33

**Выходные данные:**

Выходными данными из данной матрицы с помощью формул критериев и функции МАКС() считаются результаты четырех критериев.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 34

* 1. **С минимизацией рисков (в условиях неопределенности)**

**Excel:**

**Входные данные:**

Входными данными при решении данной задачи является искомая матрица для решения.

**Изображение выглядит как стол

Автоматически созданное описание**

Рисунок 35

**Алгоритм решения:**

Решение данной игры производится по четырем критериям (Байеса, Вальда, Севиджа, Гурвица, Лапласа) с использованием первоначальной матрицы и соответствующих формул.

По критериям Байеса, Севиджа и Гурвица составляются на основе формул вспомогательные матрицы. По Вальду вспомогательная матрица не нужна, достаточно вектора.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 36

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 37

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 38

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 39

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 40

В итоге, на основе вспомогательный данных с помощью функции Excel МАКС() находим результат по пяти критериям (минимальный риск).

Критерий Байеса:



Рисунок 41

Критерий Лапласа:

****

Рисунок 42

Критерий Вальда:



Рисунок 43

Критерий Севиджа:

****

Рисунок 44

Критерий Гурвица:

****

Рисунок 45

**Выходные данные:**

Выходными данными из данной матрицы с помощью формул критериев и функции МАКС() считаются результаты пяти критериев.

**Изображение выглядит как стол

Автоматически созданное описание**

Рисунок 46

* 1. **Применение игр с природой**

Игры с природой применяются в следующих случаях:

1. ДОМИНИРОВАНИЕ СТРАТЕГИЙ В ИГРЕ С ПРИРОДОЙ

В играх «с природой» можно и нужно пользоваться принципом доминирования стратегий игрока A (игрока – ЛПР). Но доминирование стратегий в игре с природой имеет определенную специфику.

1. КРИТЕРИИ ВЫБОРА ОПТИМАЛЬНЫХ РЕШЕНИЙ В УСЛОВИЯХ ПОЛНОЙ НЕОПРЕДЕЛЕННОСТИ

Рассмотрим игру «с природой», в которой вероятности pj состояний природы Qj неизвестны, и отсутствует всякая возможность получения о них какой-либо статистической информации. То есть мы находимся в состоянии полной неопределенности, связанной с отсутствием информации о вероятностях состояний природы. Неопределенность, связанную с отсутствием информации о вероятностях состояний среды (природы), называют «безнадежной» или «дурной». В случаях неопределенности такого вида («безнадежной» неопределенности) для выбора наилучших решений в играх «с природой» применяются следующие классические критерии.

1. ВЫБОР ОПТИМАЛЬНОЙ СТРАТЕГИИ ПО МАКСИМИННОМУ КРИТЕРИЮ ВАЛЬДА

Допустим, что предприятие готовится к переходу на новые виды продукции. При этом возможны четыре решения A1, A2, A3, A4. Каждому из решений соответствует определенный вид выпуска продукции или их сочетание. Результаты принятых решений существенно зависят от экономической ситуации, которая неопределенна.

## АРХИТЕКТУРА РЕШЕНИЯ

Архитектура решения антагонистической и биматричной задач выполняются с помощью языка программирования Python:

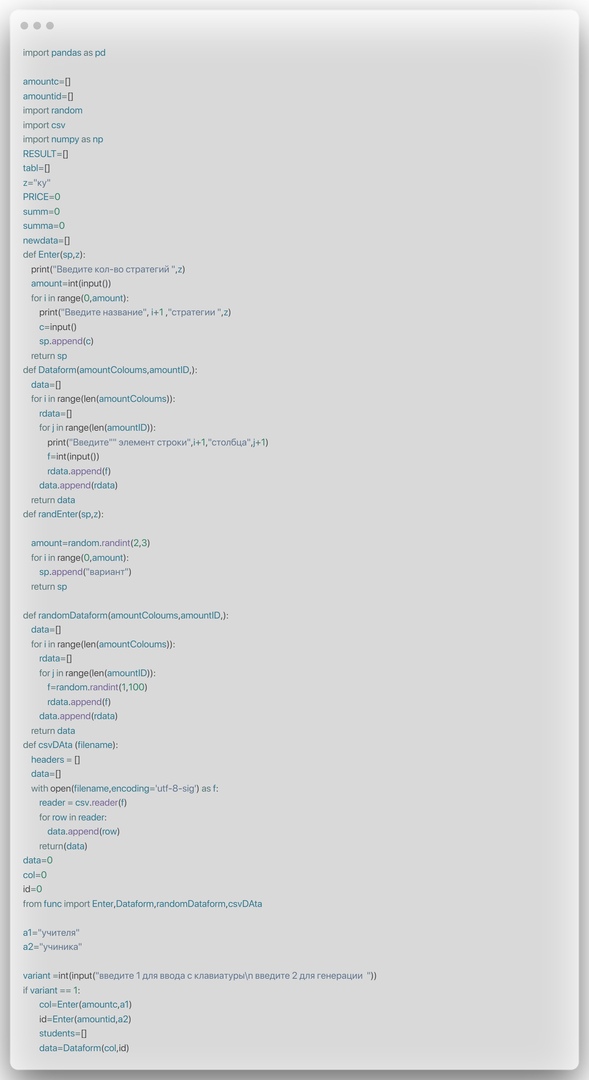
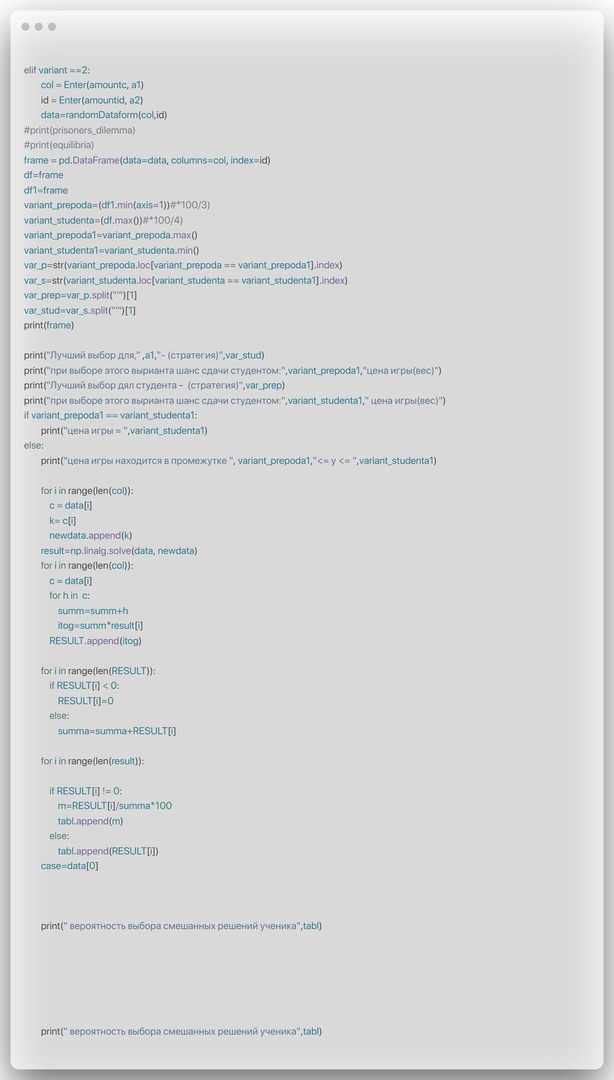


Рисунок 47

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 48

Решение природной задачи (в условиях неопределенности и с рисками) выполнялось в программе Excel:

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 49 - рисунок решения природной задачи с рисками

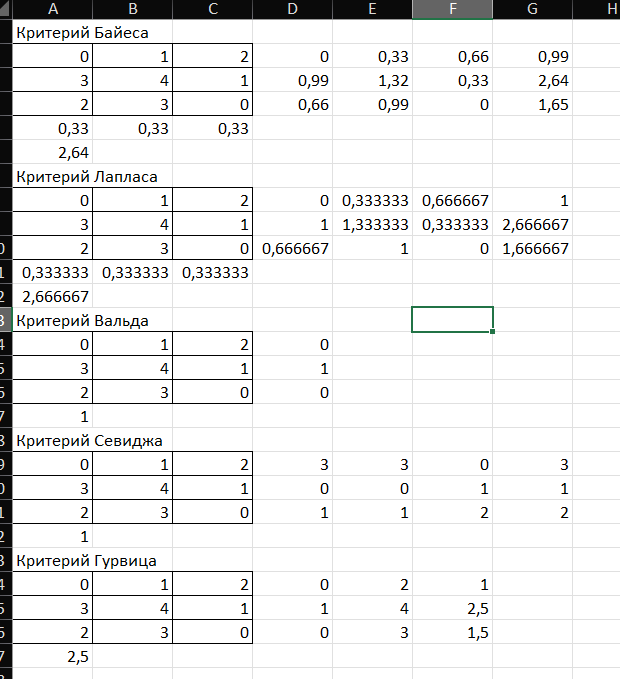


Рисунок 50 - рисунок решение природной задачи в условиях неопределенности

## ТЕСТИРОВАНИЕ

Тестирование антагонистической и биматричной задач было выполнено в программе Excel.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 51 - тестирование антагонистической игры в Excel

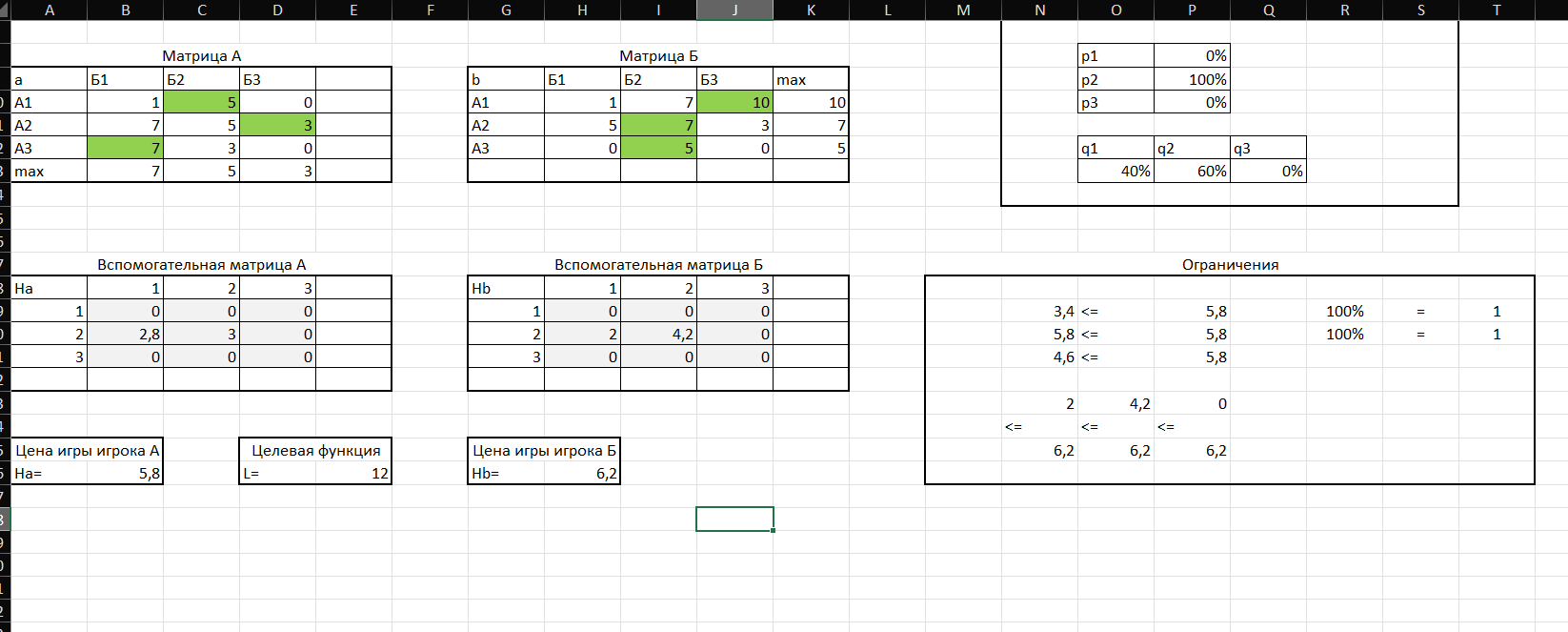


Рисунок 52 - рисунок Тестирование биматричной игры в Excel

Природные игры (с рисками и в условиях неопределенности) были выполнены в программе Excel и протестированы на основе математической модели задачи.

## ЗАКЛЮЧЕНИЕ

Таким образом, решения игр (антагонистических, биматричных, природных) были выполнены в различных программах (Python, Excel, ручным способом). В различных условиях находились и различные коэффициенты, нужные для решения данных задач. Задачи решены с помощью универсального способа (а значит – входные данные легко изменить). Основными входными данными были матрица или несколько матриц. На выходе мы получали цену игр, целевые функции матриц и искомые векторы или матрицы для наиболее удачной игры в выдвинутых условиях.

# РЕШЕНИЕ ЗАДАЧ ПО ТЕМЕ «МЕТОД НАИМЕНЬШИХ КВАДРАТОВ» №5

## ФИЗИЧЕСКАЯ МОДЕЛЬ

**Задание 1**

Новому работнику государственной компании поступило первое задание. Необходимо купить товар у трех разных поставщиков из одного региона, в рамках программы поддержки малого бизнеса в регионах. Товар очень популярен, но цены (х) на его приобретение в разных регионах разнятся, при этом бюджет компании не безграничен, из-за чего работнику необходимо выбрать подходящий и недорогостоящий вариант. Работнику была предоставлена сводка, показывающая цены в трех различных регионах:

Таблица 1 - Исходные данные задачи

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

**Задание 2**

После выполнения описанной выше задачи, начальник отдела поручил работнику выполнить прогноз роста прибыли от товара каждого из регионов, представленных в сводке.

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Для построения линейной аппроксимации с выбросом точек будут использоваться следующие формулы:

1. **Уравнение регрессии:**

\widehat{y}=ax+b

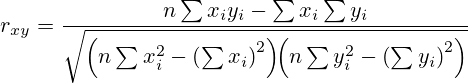
1. **Коэффициент a:**

a&=\frac{\sum x_i \sum y_i- n\sum x_iy_i}{\left(\sum x_i\right)^2-n\sum x_i^2}

1. **Коэффициент b:**

b&=\frac{\sum x_i \sum x_iy_i-\sum x_i^2\sum y_i}{\left(\sum x_i\right)^2-n\sum x_i^2}

1. **Коэффициент линейной парной корреляции:**



1. **Коэффициент детерминации:**

R^2=r_{xy}^2

1. **Средняя ошибка аппроксимации:**

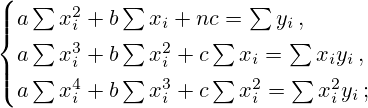
\overline{A}=\dfrac{1}{n}\sum\left|\dfrac{y_i-\widehat{y}_i}{y_i}\right|\cdot100\%

Для квадратичной аппроксимации методом наименьших квадратов будут использоваться в работе следующие формулы:

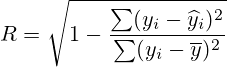
1. **Уравнение регрессии:**

**\widehat{y}=ax^2+bx+c**

1. **Система уравнений для нахождения коэффициентов a, b и c:**

****

1. **Коэффициент корреляции:**

**,  
где  
\overline{y}= \dfrac{1}{n}\sum y_i**

1. **Коэффициент детерминации:**

**R^2**

1. **Средняя ошибка аппроксимации:**

**\overline{A}=\dfrac{1}{n}\sum\left|\dfrac{y_i-\widehat{y}_i}{y_i}\right|\cdot100\%**

Все формулы требуются для нахождения ошибки аппроксимации с последующим удалением выбросов.

**Пример:**

**Входные данные:**

1. Значения X: 83 71 64 69 69 64 68 59 81 91 57 65 58 62
2. Значение Y: 183 168 171 178 176 172 165 158 183 182 163 175 164 175
3. Линейная регрессия: y=0.6233x+129.5721
4. Коэффициент линейной парной корреляции: 0.7883
5. Коэффициент детерминации: 0.6214
6. Средняя ошибка аппроксимации, %: 2.4459 %
7. Квадратичная регрессия: y=−0.0168x2+3.0867x+41.3745
8. Коэффициент корреляции: 0.8164
9. Коэффициент детерминации: 0.6665
10. Средняя ошибка аппроксимации, %: 2.0026 %

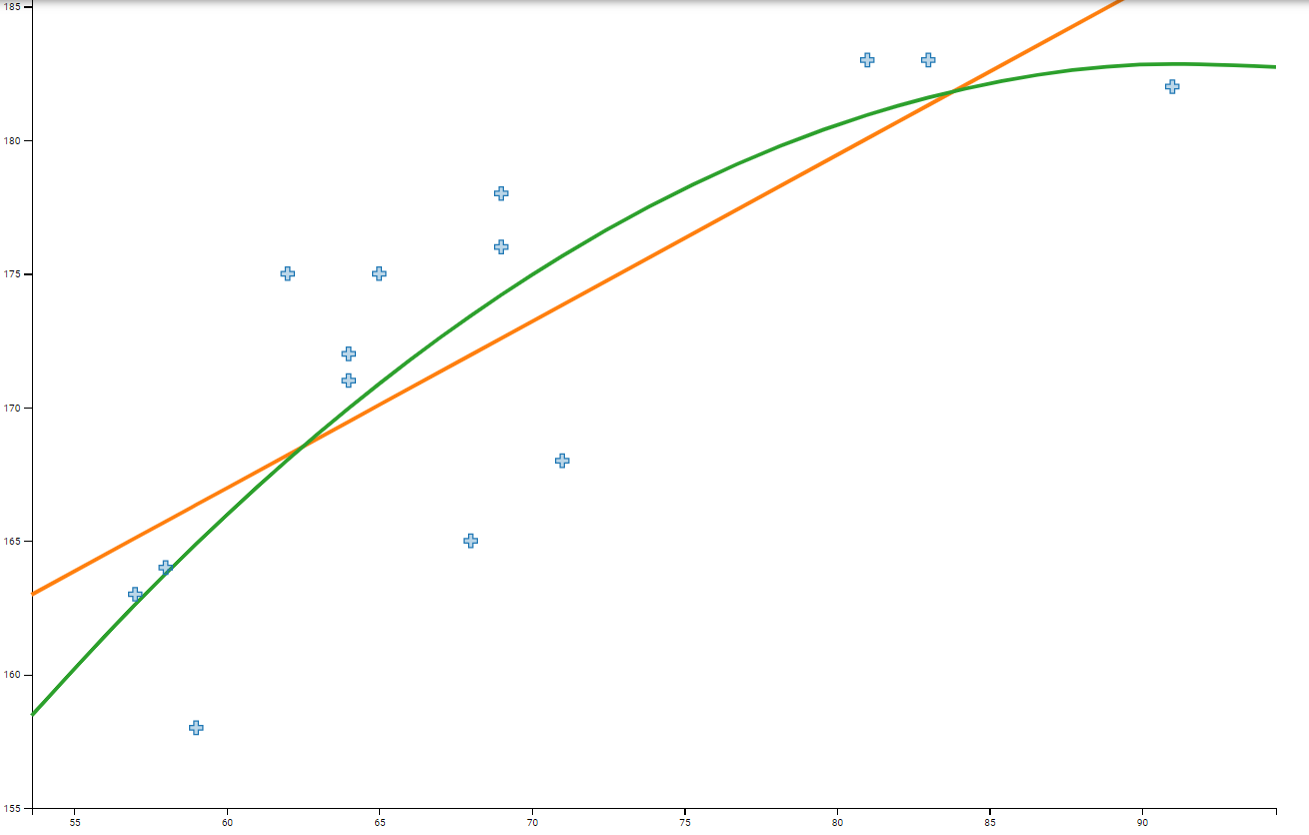


Рисунок 1

**Результат:**

Таблица 2

| i | x | y | Линейная регрессия | Квадратичная регрессия |
| --- | --- | --- | --- | --- |
|  | 53.6 |  | 162.9809 | 158.4548 |
| 1 | 57 | 163 | 165.1002 | 162.6187 |
| 2 | 58 | 164 | 165.7235 | 163.7694 |
| 3 | 59 | 158 | 166.3468 | 164.8863 |
| 4 | 62 | 175 | 168.2167 | 168.0351 |
| 5 | 64 | 171 | 169.4633 | 169.9660 |
| 6 | 64 | 172 | 169.4633 | 169.9660 |
| 7 | 65 | 175 | 170.0866 | 170.8809 |
| 8 | 68 | 165 | 171.9565 | 173.4237 |
| 9 | 69 | 178 | 172.5797 | 174.2039 |

### 

* + - 1. **АЛГОРИТМЫ РЕШЕНИЯ**

## Входные данные

Параметрами входных данных является массив X, критерии оптимизации и искомое направление (к минимуму или максимуму).

## Алгоритм решения

1. Возьмем нужный массив, составим и решим функции в зависимости от направления к минимуму и максимуму.
2. Преобразуем функцию в задачу максимизации.
3. Построим область допустимых решений ОДР.
4. Подвергнем координаты каждой точки плоскости Ox1x2 преобразованиям.
5. Находим множество Парето в зависимости от точки утопии или критериев.
6. Находим во множестве Парето точку, расположенную ближе всех к точке утопии (критериям)
7. Найдем ее координаты.
8. Решаем систему уравнений, состоящую из прямой Паррето и прямой, проходящий через точку пункта 6.
9. Найдем из системы линейных уравнений соответствующие значения x1, x2.
10. Построим лепестковые диаграммы
11. Вывод результата.

## Выходные данные

Параметрами выходных значений можно считать множество точек Паррето, оптимальное решение (то есть вывод идеальной точки и сверстки критериев) и диаграммы.

1. **ТЕСТИРОВАНИЕ**

Приемо-сдаточным тестированием являются дополнительные построения при визуализации графиков.

### ВХОДНЫЕ ДАННЫЕ

Параметрами входных данных является массив X, критерии оптимизации и искомое направление (к минимуму или максимуму).

### АЛГОРИТМ РЕШЕНИЯ

Алгоритм решения повторяется за исключением пункта 11 и 12.

1. Возьмем нужный массив, составим и решим функции в зависимости от направления к минимуму и максимуму.
2. Преобразуем функцию в задачу максимизации.
3. Построим область допустимых решений ОДР.
4. Подвергнем координаты каждой точки плоскости Ox1x2 преобразованиям.
5. Находим множество Парето в зависимости от точки утопии или критериев.
6. Находим во множестве Парето точку, расположенную ближе всех к точке утопии (критериям)
7. Найдем ее координаты.
8. Решаем систему уравнений, состоящую из прямой Паррето и прямой, проходящий через точку пункта 6.
9. Найдем из системы линейных уравнений соответствующие значения x1, x2.
10. Построим лепестковые диаграммы
11. Построим доверительные интервалы
12. Вывод результата.

Описание выходных данных

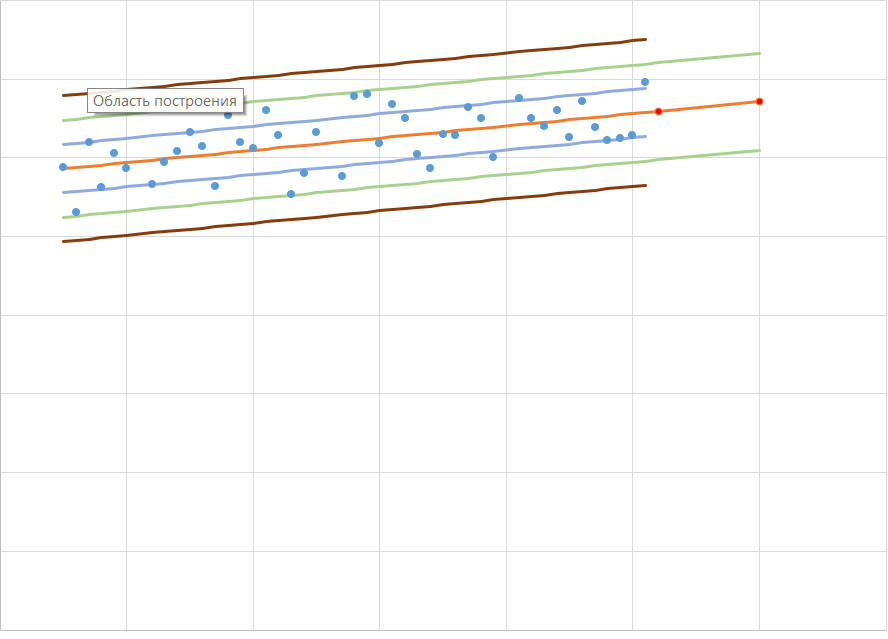
Выходными данными будут являться графики

Рисунок 2 - Линейная аппроксимация

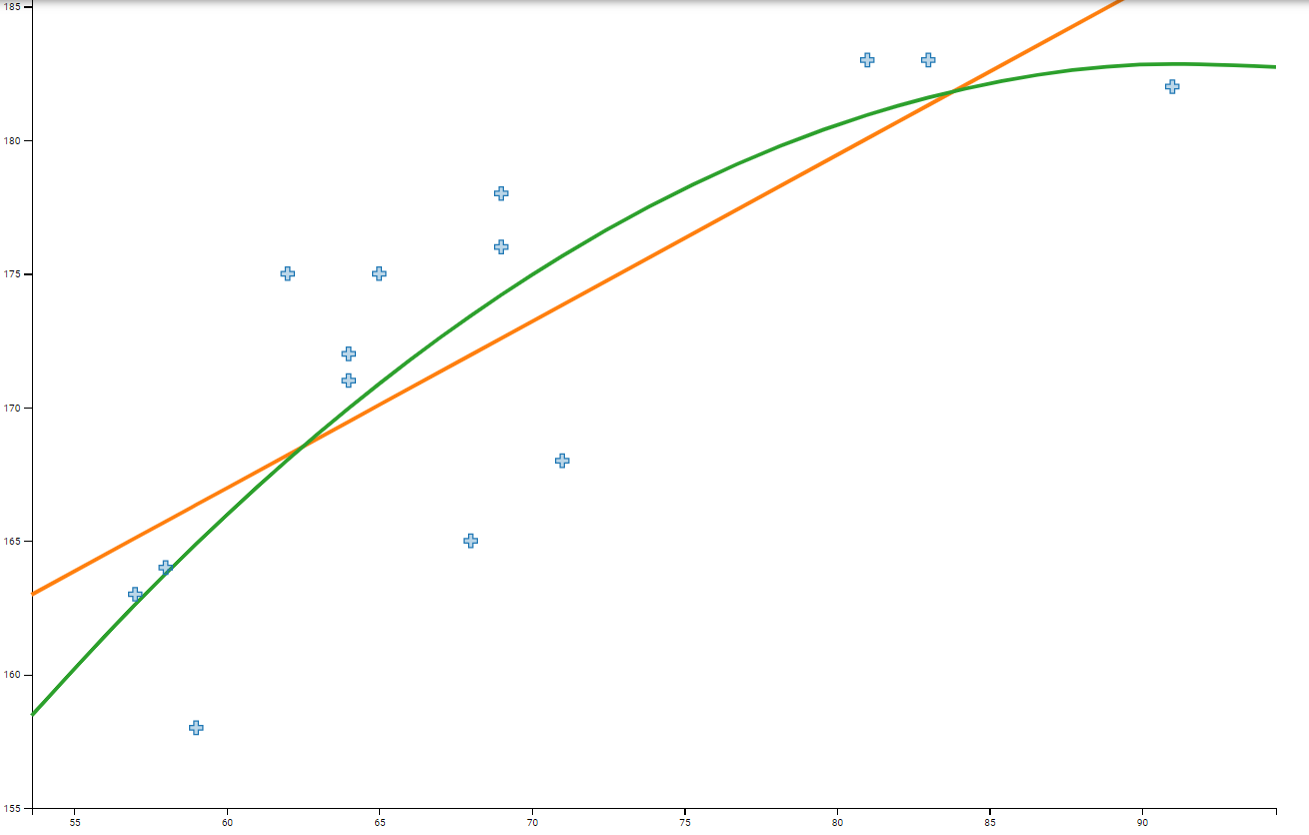


Рисунок 3 - Полиномиальная регрессия

1. **ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ**

В этой части работы мы рассмотрим пользовательскую инструкция, а именно, какой порядок действия у пользователя для использования рассматриваемой в нашей работе системы. Так как мы рассматриваем использование системы с помощью языка программирования Python, то данная инструкция будет на основе его.

1. Для начала, пользователь запускает IDE;
2. Далее для решения непосредственно задачи берется готовый код, созданный нами, и запускается;
3. Вводятся данные:
   1. Ввод данных вручную:
      1. Вводится в строку через пробел значение X;
      2. Вводится в строку через пробел значение Y;
4. Через код считаются все данные;
5. Выводится пользователю результат:
   1. Массив точек
   2. Дисперсия
   3. Аппроксимирующая функция
6. **АРХИТЕКТУРА РЕШЕНИЯ**

Архитектура решения Python

Используем проверку системной переменной, чтобы исключить работу скрипта при импорте в другой файл и вызываем исполнительную функцию:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4

Функция для аппроксимации массива точек прямой, основанная на методе наименьших квадратов:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5

Функция для аппроксимации квадратичной функции:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6

Функция, которая находит коэффициенты для вывода формулы квадратичной функции:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7

Главная функция, исполняющая все алгоритмы и выводящая графики и решенияИзображение выглядит как текст

Автоматически созданное описание*Рисунок 8*

Архитектура решения Excel

В основе данных лежит выборка о погоде: Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 9*

Далее отфильтруем данные по формуле и внесем в таблицу:

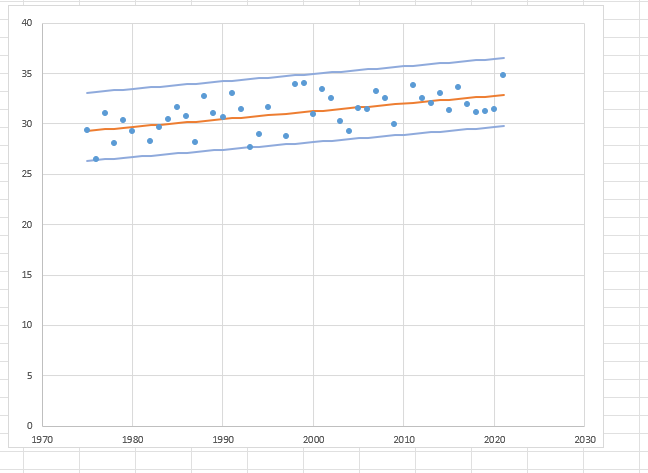
Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 10*

Данные фильтруются с помощью формул математической статистики в Excel.

Далее построим точечную диаграмму отфильтрованных данных:



*Рисунок 11*

На основе полученных данных составим таблицу прогнозирования и построим линии тренда (на основе отфильтрованных данных):



*Рисунок 12*

## ТЕСТИРОВАНИЕ

**1 проход:**

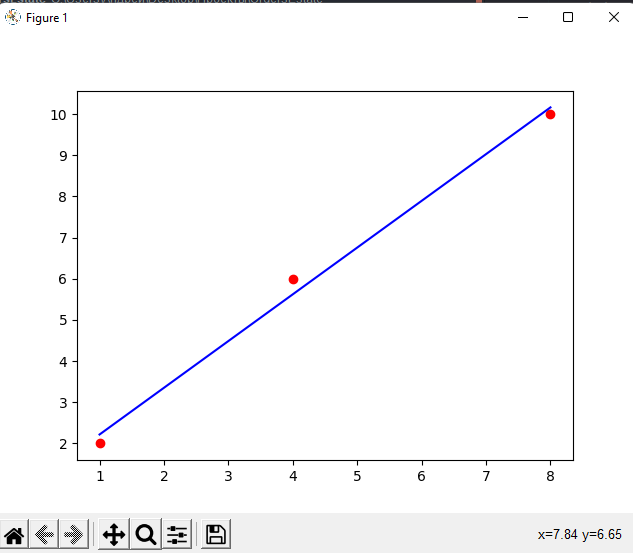


Рисунок 13

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

**Уравнение прямой, проходящей через эти точки:** y = 1.14\*x + 4.49

**Дисперсия:** 10.138888888888888

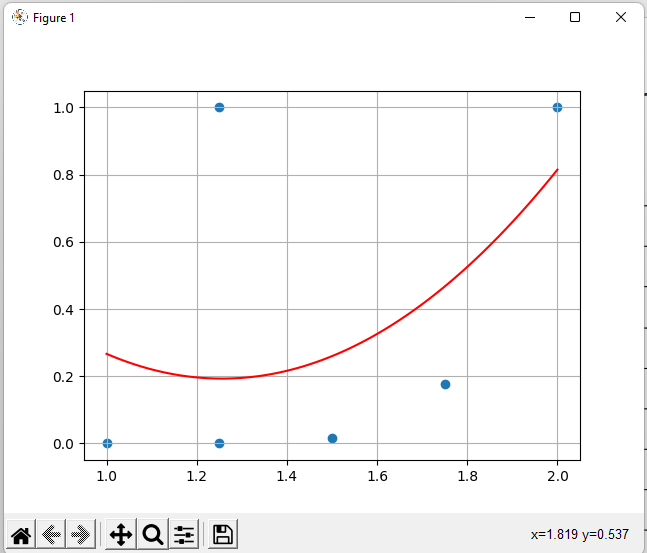


Рисунок 14

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

**2 проход:**

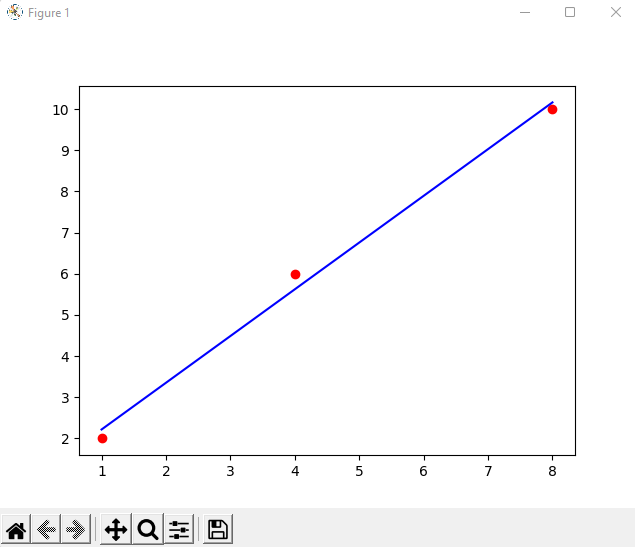


Рисунок 15

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

**Уравнение прямой, проходящей через эти точки:** y = 1.14\*x + 4.49

**Дисперсия:** 10.138888888888888

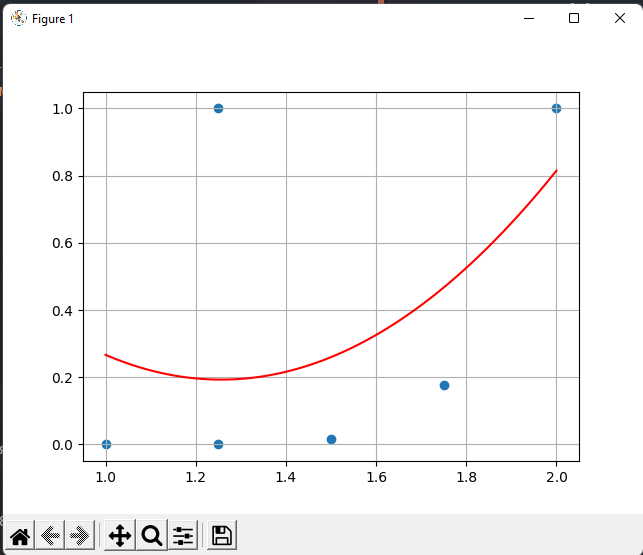


Рисунок 16

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

**3 проход:**

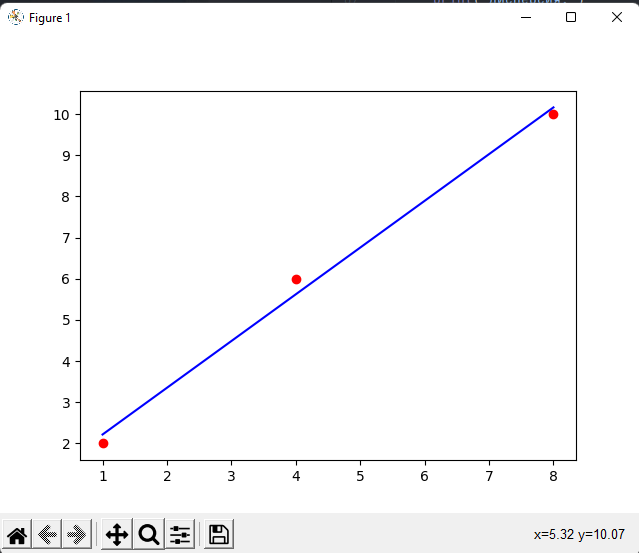


Рисунок 17

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

Уравнение прямой, проходящей через эти точки: y = 1.14\*x + 4.49

Дисперсия: 10.138888888888888

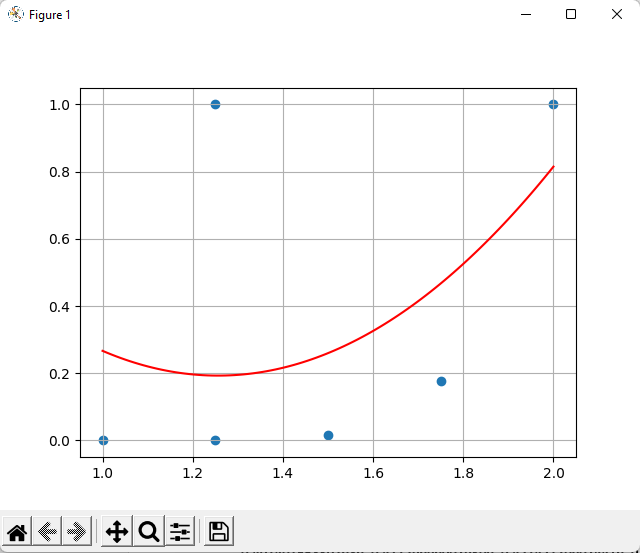


Рисунок 18

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

**4 проход:**

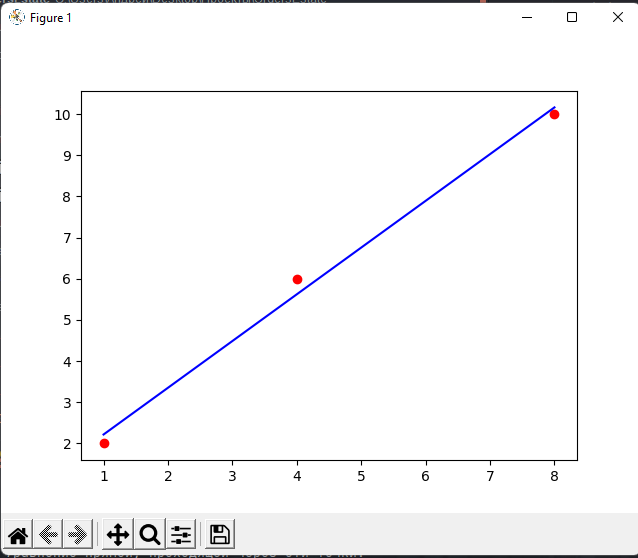


Рисунок 19

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

**Уравнение прямой, проходящей через эти точки:** y = 1.14\*x + 4.49

**Дисперсия:** 10.138888888888888

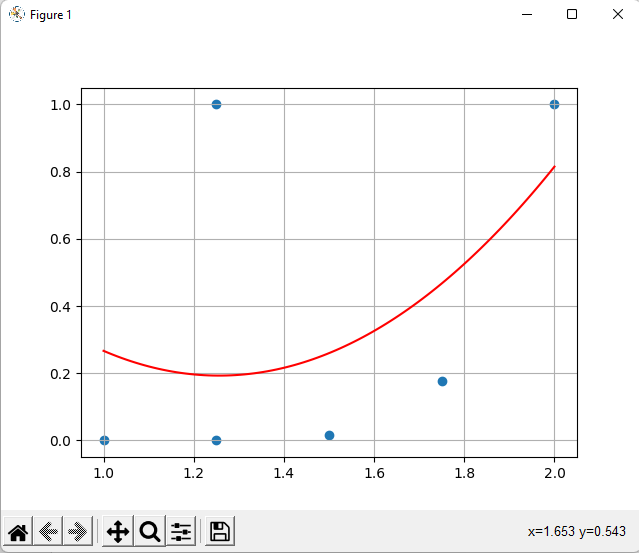


Рисунок 20

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

**5 проход:**

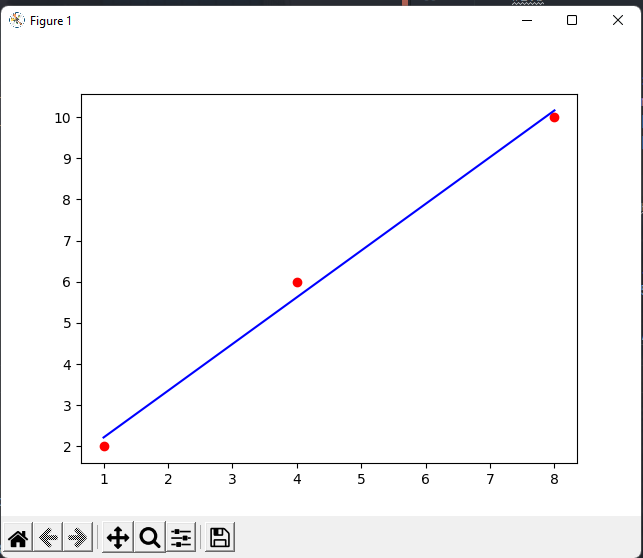


Рисунок 21

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

Уравнение прямой, проходящей через эти точки: y = 1.14\*x + 4.49

Дисперсия: 10.138888888888888

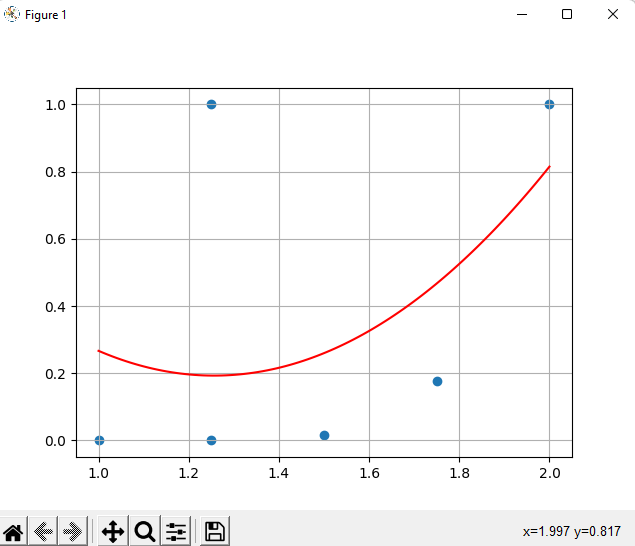


Рисунок 22

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

**6 проход:**

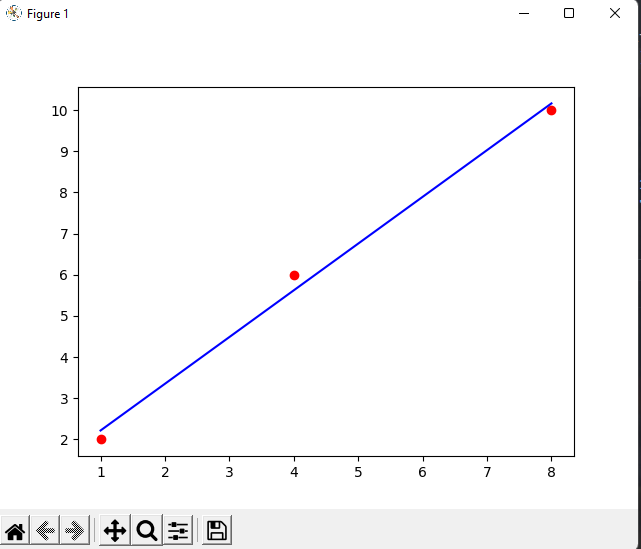


Рисунок 22

[[ 1. 2. 2.21621622]

[ 4. 6. 5.62162162]

[ 8. 10. 10.16216216]]

**Уравнение прямой, проходящей через эти точки:** y = 1.14\*x + 4.49

**Дисперсия:** 10.138888888888888

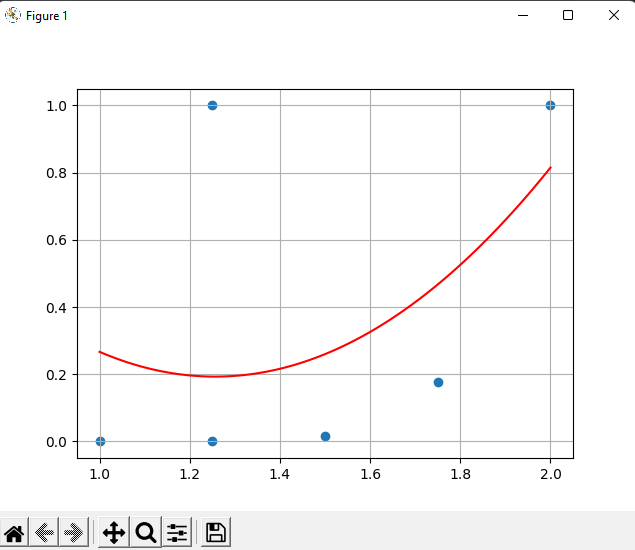


Рисунок 23

**Задание 2.а**

**Массив точек:**

[1, 1.25, 1.25, 1.5, 1.75, 2] [0.0, 0.000244140625, 1, 0.015625, 0.177978515625, 1] [0.266555786132813, 0.260855780707466, 0.255385119505604, 0.250143802527226, 0.245131829772333, 0.240349201240925, 0.235795916933002, 0.231471976848564, 0.227377380987611, 0.223512129350142, 0.219876221936159, 0.216469658745660, 0.213292439778646, 0.210344565035117, 0.207626034515072, 0.205136848218513, 0.202877006145438, 0.200846508295849, 0.199045354669744, 0.197473545267124, 0.196131080087989, 0.195017959132339, 0.194134182400173, 0.193479749891492, 0.193054661606296, 0.192858917544586, 0.192892517706360, 0.193155462091619, 0.193647750700363, 0.194369383532591, 0.195320360588304, 0.196500681867502, 0.197910347370185, 0.199549357096353, 0.201417711046006, 0.203515409219144, 0.205842451615766, 0.208398838235873, 0.211184569079465, 0.214199644146543, 0.217444063437104, 0.220917826951151, 0.224620934688683, 0.228553386649699, 0.232715182834201, 0.237106323242187, 0.241726807873657, 0.246576636728614, 0.251655809807054, 0.256964327108979, 0.262502188634390, 0.268269394383285, 0.274265944355665, 0.280491838551530, 0.286947076970879, 0.293631659613715, 0.300545586480034, 0.307688857569838, 0.315061472883126, 0.322663432419902, 0.330494736180160, 0.338555384163904, 0.346845376371133, 0.355364712801847, 0.364113393456043, 0.373091418333727, 0.382298787434896, 0.391735500759548, 0.401401558307686, 0.411296960079309, 0.421421706074416, 0.431775796293008, 0.442359230735085, 0.453172009400647, 0.464214132289693, 0.475485599402226, 0.486986410738242, 0.498716566297744, 0.510676066080729, 0.522864910087200, 0.535283098317157, 0.547930630770598, 0.560807507447523, 0.573913728347933, 0.587249293471828, 0.600814202819208, 0.614608456390075, 0.628632054184424, 0.642884996202258, 0.657367282443578, 0.672078912908383, 0.687019887596672, 0.702190206508446, 0.717589869643705, 0.733218877002448, 0.749077228584676, 0.765164924390390, 0.781481964419589, 0.798028348672272, 0.814804077148440]

1.0 0.266555786132813 1.0101010101010102 0.260855780707466 1.02020202020202 0.255385119505604

y = 1.12390136718690\*x^2 + (-2.82345581054568\*x) + (1.96611022949180)

**Задание 2.б**

Дисперсия: 0.45739753544330597

## ЗАКЛЮЧЕНИЕ

В заключение мы получаем систему нахождения линейной и квадратичной аппроксимации для заданных массивов.

Можно считать, что для заданной физической модели подходит этот тип решения, так как он определяет регрессию массивов для сравнения между несколькими массивами или выборками, чем и являются входные данные в физической постановке задачи.

Данное решение имеет перспективу роста и развития, например, можно улучшить количество входных массивов, т.е. увеличить его для наиболее быстрого нахождения решения задачи.

# ЭКСПЕРТНЫЕ ОЦЕНКИ №6

## ФИЗИЧЕСКАЯ МОДЕЛЬ

Житель Подмосковья решил переехать в Москву. Для себя он отметил 10 предложений, предварительно выбрав наиболее важные критерии, на которые он будет опираться при выборе окончательного варианта. В таблице представлены данные по данным критериям на каждую из квартир

Таблица 1 - задача

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | Стоимость | Площадь | Время до метро | Год постройки | Этаж | Потолок | Залог | Предыдущих собственников | Стоимость коммунальных платежей |
| 1 | 35000 | 30 | 15 | 2010 | 6 | 2.7 | 10000 | 2 | 9 |
| 2 | 47000 | 39 | 10 | 2015 | 7 | 2.5 | 15000 | 7 | 6 |
| 3 | 42500 | 35 | 12 | 2009 | 5 | 3 | 13000 | 5 | 8 |
| 4 | 65000 | 43 | 10 | 2017 | 8 | 2.7 | 30000 | 8 | 3 |
| 5 | 38000 | 35 | 10 | 2013 | 5 | 2.8 | 12000 | 5 | 4 |
| 6 | 50500 | 44 | 15 | 2008 | 6 | 2.5 | 20000 | 2 | 5 |
| 7 | 48000 | 35 | 10 | 2006 | 7 | 3 | 14000 | 7 | 3 |
| 8 | 58000 | 42 | 10 | 2010 | 8 | 2.8 | 28000 | 8 | 6 |
| 9 | 54500 | 39 | 13 | 2018 | 7 | 2.7 | 22000 | 7 | 4 |
| 10 | 45000 | 40 | 17 | 2013 | 5 | 3 | 17000 | 1 | 6 |

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

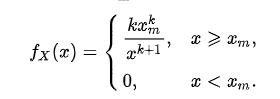
Точка x ∗ ∈ X называется оптимальной по Парето, если не существует такой точки x ∈ X, для которой выполнено неравенство f(x) ≥ f(x ∗).

В случае одного критерия (m = 1) оптимальная по Парето точка превращается в точку максимума функции f1. Отсюда следует, что понятие Парето-оптимальной точки представляет собой прямое обобщение точки максимума скалярной числовой функции на случай векторного критерия.

Тогда формула для расчета распределения Парето выглядит следующим образом:

C:\Users\202486\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\8DFA916D.tmp

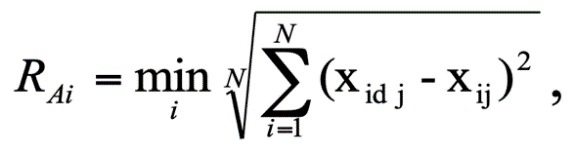
Тогда говорят, что Х имеет распределение Парето. Плотность распределения Парето имеет вид:



Для поиска единственного оптимального решения сведем задачу к однокритериальной и воспользуемся методом линейной свертки по критериям. Для этого необходимо присвоить каждому критерию показатель значимости в процентах из ста.

Необходимо также учесть, что среди рассматриваемых критериев оптимальными значениями могут быть как максимальное среди имеющихся показателей критериев, так и минимальное. В случае, если оптимальным значением является минимальное, необходимо умножить коэффициент a на -1.

В основу метода «идеальной точки» положен расчет расстояния в многомерном пространстве критериев между точкой, соответствующей идеальной альтернативе, и точкой, соответствующей рассматриваемой альтернативе. Идеальной называется такая альтернатива, которая имеет наилучшие значения всех критериев. Но наиболее приемлемой считается альтернатива, у которой расстояние от «идеальной точки» минимально:



## АЛГОРИТМЫ РЕШЕНИЯ

### Параметры входных данных

Массивы характеристик объектов, список критериев оптимизации, направление оптимизации и весовые коэффициенты характеристик (для линейной сверстки критериев).

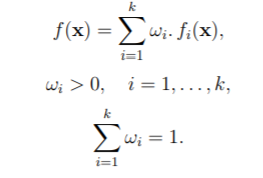
### Алгоритм решения

1. Формулирование задачи так, чтобы ее можно было решить с помощью итеративного алгоритма;

2. Сведение к однокритериальной задаче методом скаляризации

C:\Users\202486\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\8705D68F.tmp

3. Получение различных точек Парето-фронта

4. Возьмем линейных функционал метода взвешенной суммы:

5. Найдем значения c, при котором линия ω · f T = c касается границы множества допустимых значений целевых функций (f(Ω)).

6. Для линейной свертки необходимо присвоить каждому критерию критерий значимости.

7. Для того, чтобы перевести данные показатели в процентное выражение, необходимо сложить приведенные выше критерии и поделить каждый показатель на полученную сумму;

8. Для вычисления коэффициента каждого из вариантов в каждом критерии необходимо найти максимальное значение по критерию и поделить каждый из критериев на найденный максимум по критерию;

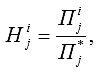
9. Для вычисления оптимального решения необходимо произвести суммирование произведений коэффициента из вариантов и процентного выражения критерия значимости;

10. Значения минимальные следует умножить на -1;

11. Максимальное значение из вычисленных сумм является оптимальным решением;

12. Для алгоритма метода «идеальной точки» определяется «идеальная точка», т.е. лучшие параметры, с которыми сравниваются параметры рассматриваемых проектов.

13. Поскольку параметры проектов могут быть разными по масштабу, определяются их нормированные значения:



14. Определяются «расстояния» между проектами;

15. Затем проекты ранжируются по возрастанию расстояния до «идеальной точки» R\*(i);

16. Лучшим считается проект, у которого расстояние до «идеальной точки» является минимальным.

### Параметры выходных данных

Набор точек Парето, диаграмма значений, оптимальное решение, диаграмма линейной свертки критериев, идеальная точка.

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

Чтобы получить ответ для желаемого типа экспертных оценок пользователь должен:

Выбрать тип ввода данных.

Ввести «-» для случайного заполнения массива характеристик.

Сначала пользователь указывает количество объектов (n).

Затем он указывает количество характеристик (m).

Далее пользователь указывает нижнюю границу для случайного заполнения.

Затем - верхнюю границу для случайного заполнения.

Потом пользователь указывает список критериев через запятую, по которым осуществляется оптимизация. (Например: Цена, Количество).

После, пользователь указывает направление оптимизации для критериев. Он вводит "max" для направления к максимуму или "min" для направления к минимуму.

Затем пользователь вводит весовые коэффициенты характеристик, которые необходимы для метода линейной свертки. Важно, чтобы сумма коэффициентов давала единицу.

Далее, пользователю выводится датафрейм объектов и их характеристик (размера n\*m), список критериев для оптимизации и направление оптимизации.

Ввести «+» для ручного заполнения массива характеристик.

Сначала пользователь указывает количество объектов (n).

Затем пользователь вводит имена этих объектов.

Потом он указывает количество характеристик (m).

Затем пользователь вводит имена характеристик.

Дальше, он заполняет через запятую, без пробелов, значения характеристик объектов построчно. (Пример для первой характеристики: 19,23,45,56)

Потом пользователь указывает список критериев через запятую, по которым осуществляется оптимизация. (Например: Цена, Количество).

Далее, он указывает порядковые номера критериев через запятую, по которым осуществляется оптимизация. Это необходимо для решения методом идеальной точки.

После, пользователь указывает направление оптимизации для критериев. Он вводит "max" для направления к максимуму или "min" для направления к минимуму.

Затем пользователь вводит весовые коэффициенты характеристик, которые необходимы для метода линейной свертки. Важно, чтобы сумма коэффициентов давала единицу.

Далее, пользователю выводится датафрейм объектов и их характеристик (размера n\*m), список критериев для оптимизации и направление оптимизации.

После проделанных действий пользователь может активировать любой алгоритм на выбор из представленных:

⦁ «Поиск Парето оптимального решения».

⦁ «Линейная свертка критериев».

⦁ «Метод идеальной точки».

⦁ В зависимости от выбранного алгоритма пользователь получает ответ на задачу:

⦁ «Поиск Парето оптимального решения».

⦁ Набор точек поверхности Парето.

⦁ Диаграмма значений.

⦁ Лепестковая диаграмма с указанием Парето оптимального множества решений.

⦁ «Линейная свертка критериев».

⦁ Оптимальное значение.

⦁ Диаграмма линейной свертки критериев.

⦁ Набор точек поверхности Парето.

⦁ «Метод идеальной точки».

⦁ Оптимальное значение.

⦁ Лепестковая диаграмма с указанием Парето оптимального множества решений.

⦁ Набор точек поверхности Парето (идеальная точка).

## АРХИТЕКТУРА РЕШЕНИЯ

Архитектура решения поиска Паретооптимального решения

Функции считывания информации

Метод data\_entry

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная vibor – выбор варианта заполнения точек (поддерживает тип данных str)

⦁ Переменная number\_object – количество объектов (поддерживает тип данных int)

⦁ Переменная number\_char – количество характеристик (поддерживает тип данных int)

⦁ Переменная dd – нижняя граница максимального числа для рандомного заполнения

⦁ Переменная cc – верхняя граница максимального числа для рандомного заполнения

⦁ Переменная matrr – матрица весов (поддерживает тип данных dict)

⦁ Переменная df –

⦁ Переменная code –

⦁ Переменная entered\_list\_a – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных list)

⦁ Переменная name\_object – содержит данные из переменной entered\_list\_a, переведенных в формат int (поддерживает тип данных list)

⦁ Переменная criteria – cписок критериев (поддерживает тип данных list)

⦁ Переменная direction – направление оптимизации для критериев (поддерживает тип данных dict)

⦁ Переменная weight – весовые коэффициенты характеристик (поддерживает тип данных dict)

⦁ Переменная name\_char – название характеристик (поддерживает тип данных list)

⦁ Переменная matrr\_1 – значения характеристик объектов (поддерживает тип данных dict)

⦁ Переменная entered\_list – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных str)

⦁ Переменная ent\_list – содержит переменную entered\_list, с замененными символами кавычек (поддерживает тип данных list)

⦁ Переменная e\_list – переводит в новый тип списка (поддерживает тип данных list)

⦁ Переменная criteria\_num – список критериев (поддерживает тип данных list)

Функции обработки информации

Затрагиваемые в ходе работы переменные:

Метод pareto\_solution

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная mask – список критериев (поддерживает тип данных list)

⦁ Переменная ax – значения для диаграммы (поддерживает тип данных list)

⦁ Переменная categories – все значения датафрейма без индексов и заголовков колонок (поддерживает тип данных list)

⦁ Переменная matr\_PM – матрица точек (поддерживает тип данных list)

⦁ Переменная fig – значения графика (поддерживает тип данных list)

⦁ Переменная angularaxis – углы (поддерживает тип данных dict)

Функции вывода информации

Вывод текстовой и числовой информации осуществлен посредством функции print, расположенной в следующих методах:

- pareto\_solution

Архитектура решения линейной свертки критериев

Функции считывания информации

Метод data\_entry

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная vibor – выбор варианта заполнения точек (поддерживает тип данных str)

⦁ Переменная number\_object – количество объектов (поддерживает тип данных int)

⦁ Переменная number\_char – количество характеристик (поддерживает тип данных int)

⦁ Переменная dd – нижняя граница максимального числа для рандомного заполнения

⦁ Переменная cc – верхняя граница максимального числа для рандомного заполнения

⦁ Переменная matrr – матрица весов (поддерживает тип данных dict)

⦁ Переменная df –

⦁ Переменная code –

⦁ Переменная entered\_list\_a – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных list)

⦁ Переменная name\_object – содержит данные из переменной entered\_list\_a, переведенных в формат int (поддерживает тип данных list)

⦁ Переменная criteria – cписок критериев (поддерживает тип данных list)

⦁ Переменная direction – направление оптимизации для критериев (поддерживает тип данных dict)

⦁ Переменная weight – весовые коэффициенты характеристик (поддерживает тип данных dict)

⦁ Переменная name\_char – название характеристик (поддерживает тип данных list)

⦁ Переменная matrr\_1 – значения характеристик объектов (поддерживает тип данных dict)

⦁ Переменная entered\_list – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных str)

⦁ Переменная ent\_list – содержит переменную entered\_list, с замененными символами кавычек (поддерживает тип данных list)

⦁ Переменная e\_list – переводит в новый тип списка (поддерживает тип данных list)

⦁ Переменная criteria\_num – список критериев (поддерживает тип данных list)

Функции обработки информации

Метод linear\_convolution

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная coords – список критериев (поддерживает тип данных list)

⦁ Переменная coords\_final – список, внутри которого списки данных характеристик конкретного объекта (поддерживает тип данных list)

⦁ Переменная line\_coords – вектор координат (поддерживает тип данных list)

⦁ Переменная new\_weight – новые веса (поддерживает тип данных dict)

⦁ Переменная weight\_all – список значений словаря из new\_weight (поддерживает тип данных list)

⦁ Переменная svertka – первый этап выполнения линейной свертки по критериям (поддерживает тип данных list)

⦁ Переменная svertka\_final – линейная свертка по критериям (поддерживает тип данных list)

⦁ Переменная line\_sv – сумма объектов вектора (поддерживает тип данных float)

⦁ Переменная li – округление значения из line\_sv (поддерживает тип данных int)

⦁ Переменная opti – объект, который подходит для оптимального значения (поддерживает тип данных int)

⦁ Переменная optimal\_name – имя объекта, который подходит для оптимального значения (поддерживает тип данных str)

⦁ Переменная DD – словарь данных для диаграммы свертки (поддерживает тип данных dict)

⦁ Переменная DD1 – содержит значения из name\_object (поддерживает тип данных list)

⦁ Переменная new\_coords – список новых координат (поддерживает тип данных list)

⦁ Переменная n\_coords – делает значения координат отрицательными (поддерживает тип данных list)

⦁ Переменная new\_coords\_final – итоговое значение списка координат (поддерживает тип данных list)

⦁ Переменная alpha1 – параметры верстки (поддерживает тип данных list)

⦁ Переменная alpha2 – параметры верстки (поддерживает тип данных list)

⦁ Переменная line\_step – округленные до 3-х символов после запятой, значения из переменной alp (поддерживает тип данных list)

⦁ Переменная step1 –значения из переменной line\_step (поддерживает тип данных list)

⦁ Переменная Final – транспонированный массив значений из переменной step1 (поддерживает тип данных array)

⦁ Переменная maxlist – максимальные значения по каждому критерию (поддерживает тип данных list)

⦁ Переменная true\_points – значения характеристик объектов (поддерживает тип данных list)

⦁ Переменная point – значения индекса точки (поддерживает тип данные float)

⦁ Переменная LLL – индексы нужных объектов (поддерживает тип данных list)

⦁ Переменная Names – наименования нужных объектов (поддерживает тип данных list)

⦁ Переменная VALUES – значения матрицы весов (поддерживает тип данных list)

⦁ Переменная NEW\_VALUES – содержит отсортированные значения матрицы весов для пандаса (поддерживает тип данных list)

⦁ Переменная itog\_points\_pareto – набор точек поверхности Паретто (поддерживает тип данных list)

⦁ Переменная new\_df – датафрейм, содержащий значения из переменной itog\_points\_pareto (поддерживает тип данных dataframe)

⦁ Переменная final\_points – точки для графика (поддерживает тип данных list)

⦁ Переменная data – делаем словарь, где плюс - стратегия, значение - строчка линейной свертки (поддерживает тип данных dict)

⦁ Переменная dff – датафрейм, содержащий значения из переменной data (поддерживает тип данных dataframe)

Функции вывода информации

Вывод текстовой и числовой информации осуществлен посредством функции print, расположенной в следующих методах:

- linear\_convolution

Архитектура решения метода идеальных точек

Функции считывания информации

Метод data\_entry

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная vibor – выбор варианта заполнения точек (поддерживает тип данных str)

⦁ Переменная number\_object – количество объектов (поддерживает тип данных int)

⦁ Переменная number\_char – количество характеристик (поддерживает тип данных int)

⦁ Переменная dd – нижняя граница максимального числа для рандомного заполнения

⦁ Переменная cc – верхняя граница максимального числа для рандомного заполнения

⦁ Переменная matrr – матрица весов (поддерживает тип данных dict)

⦁ Переменная df –

⦁ Переменная code –

⦁ Переменная entered\_list\_a – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных list)

⦁ Переменная name\_object – содержит данные из переменной entered\_list\_a, переведенных в формат int (поддерживает тип данных list)

⦁ Переменная criteria – cписок критериев (поддерживает тип данных list)

⦁ Переменная direction – направление оптимизации для критериев (поддерживает тип данных dict)

⦁ Переменная weight – весовые коэффициенты характеристик (поддерживает тип данных dict)

⦁ Переменная name\_char – название характеристик (поддерживает тип данных list)

⦁ Переменная matrr\_1 – значения характеристик объектов (поддерживает тип данных dict)

⦁ Переменная entered\_list – список критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных str)

⦁ Переменная ent\_list – содержит переменную entered\_list, с замененными символами кавычек (поддерживает тип данных list)

⦁ Переменная e\_list – переводит в новый тип списка (поддерживает тип данных list)

⦁ Переменная criteria\_num – список критериев (поддерживает тип данных list)

Функции обработки информации

Метод ideal\_dot

Входные параметры:

Нет входных параметров

Выходные параметры:

Нет выходных параметров

Затрагиваемые в ходе работы переменные:

⦁ Переменная criteria\_num – содержит данные из переменной name\_object, переведенных в формат int (поддерживает тип данных list)

⦁ Переменная direct – укажите направление оптимизации для критериев (поддерживает тип данных dict)

⦁ Переменная entered\_list\_a– порядковые номера критериев через запятую, по которым осуществляется оптимизация (поддерживает тип данных str)

⦁ Переменная categories – все значения датафрейма без индексов и заголовков колонок (поддерживает тип данных list)

⦁ Переменная matr\_PM – матрица точек (поддерживает тип данных list)

⦁ Переменная trash – выбросы (поддерживает тип данных array)

⦁ Переменная trtr – транспонированное значение переменной trash (поддерживает тип данных array)

⦁ Переменная ideal\_dot – формирование идеальной точки (поддерживает тип данных list)

⦁ Переменная clear\_trash – значения по используемым критериям (поддерживает тип данных list)

⦁ Переменная dot – точка из переменной trash (поддерживает тип данных float)

⦁ Переменная clear\_dot – очищенные значения из dot с только нужными критериями (поддерживает тип данных list)

⦁ Переменная distances – интервал точек (поддерживает тип данных list)

⦁ Переменная final\_distances – финальный интервал точек (поддерживает тип данных list)

⦁ Переменная k – сумма всех объектов из переменной distances (поддерживает тип данных float)

⦁ Переменная minim – оптимальное значение (поддерживает тип данных float)

⦁ Переменная mask – накладываем маску (поддерживает тип данных dataframe)

⦁ Переменная pareto\_list– список всех точек по Паретто (поддерживает тип данных list)

⦁ Переменная fig – значения графика (поддерживает тип данных dataframe)

⦁ Переменная angularaxis – углы (поддерживает тип данных dict)

Функции вывода информации

Вывод текстовой и числовой информации осуществлен посредством функции print, расположенной в следующих методах:

- ideal\_dot

## ТЕСТИРОВАНИЕ

Протестировав код методом белого ящика (по кейсу, представленному в физической модели) получились следующие результаты

Таблица 2

|  |  |
| --- | --- |
| Параметры | Код |
| Скорость работы кода | 10 секунд < |
| Наличие дефектов | Отсутствуют |
| Верификация | Полное соответствие с поставленной задачей |
| Легкость восприятия для пользователя | Средне |
| Вариативность использования | Высокая |

## ЗАКЛЮЧЕНИЕ

Таким образом, решение данной задачи сводится к тому, что имея некоторые критерии для анализа и выбора оптимального решения задачи (на примере автомобилей), с помощью интерфейса можно найти оптимальное значение, идеальную точку.

В решении использовались разные методы решения, основой которой служил поиск множества точек Парето, линейной свертки и метода идеальной точки.

У данной задачи можно найти перспективы развития – более точный результат оптимального решения, ускорение работы системы при увеличении вводимых в систему данных.

# ЭКСПЕРТНЫЕ ОЦЕНКИ №7

## ФИЗИЧЕСКАЯ МОДЕЛЬ

Задание. Произвести экспертную оценку технических параметров холодильников по степени значимости их для потребителей, а также проверить отношение клиентов к заведениям быстрого питания на станции метро Рязанский проспект:

Исходные данные:

Технические параметры холодильников:

1. Объем камеры м3.

2. Количество камер шт.

3. Энергопотребление.

4. Наличие системы Hoy Frost.

5. Дизайн.

6. Бренд.

Число факторов n= 6

Число экспертов m= 4

Число опрошенных респондентов фастфуда = 5

число компаний фастфуда = 5

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Этап 1. Создание экспертной комиссии.  
В экспертную группу вошло 4 эксперта.

Этап 2. Сбор мнений специалистов путем анкетного опроса.  
Оценку степени значимости технических параметров холодильника для потребителей эксперты производят путем присвоения им рангового номера. Фактору, которому эксперт дает наивысшую оценку, присваивается ранг 1. Если эксперт признает несколько факторов равнозначными, то им присваивается одинаковый ранговый номер. На основе данных анкетного опроса составляется сводная матрица рангов.

Этап 3. Составление сводной матрицы рангов.  
Таблица 1  
**Матрица рангов**

Таблица 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Технические параметры холодильника** | **Единица измерения** | **Эксперты** | | | |
| **1** | **2** | **3** | **4** |
| 1 | Объем камеры (x1) | м3 | 3 | 4 | 5 | 4 |
| 2 | Бренд (x2) | - | 5 | 6 | 3 | 4 |
| 3 | Количество камер (x3) | Шт. | 4 | 3 | 1 | 3 |
| 4 | Наличие систем Hoy Frost (x4) | - | 1 | 1 | 3 | 1 |
| 5 | Энергопотребление (x5) | Вт. | 6 | 5 | 6 | 6 |
| 6 | Дизайн (x6) | - | 2 | 2 | 2 | 2 |

Так как в матрице имеются связанные ранги (одинаковый ранговый номер) в оценках 3-его и 4-ого экспертов, произведем их переформирование. Переформирование рангов производиться без изменения мнения эксперта, то есть между ранговыми номерами должны сохраниться соответствующие соотношения (больше, меньше или равно). Также не рекомендуется ставить ранг выше 1 и ниже значения равного количеству параметров (в данном случае n= 6).  
Переформирование рангов производится в табл. 2, 3  
*Таблица 2*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номера мест в упорядоченном ряду | 1 | 2 | 3 | 4 | 5 | 6 |
| Расположение факторов по оценке эксперта | 1(x3) | 2(x6) | 3(x2) | 3(x4) | 5(x1) | 6(x5) |
| Новые ранги | 1 | 2 | 3,5 | 3,5 | 5 | 6 |

*Таблица 3*

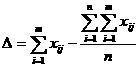
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номера мест в упорядоченном ряду | 1 | 2 | 3 | 4 | 5 | 6 |
| Расположение факторов по оценке эксперта | 1(x4) | 2(x6) | 3(x3) | 4(x1) | 4(x2) | 6(x5) |
| Новые ранги | 1 | 2 | 3 | 4,5 | 4,5 | 6 |

На основании переформирования рангов строится новая матрица рангов (табл. 4).

Матрица рангов:

Таблица 4

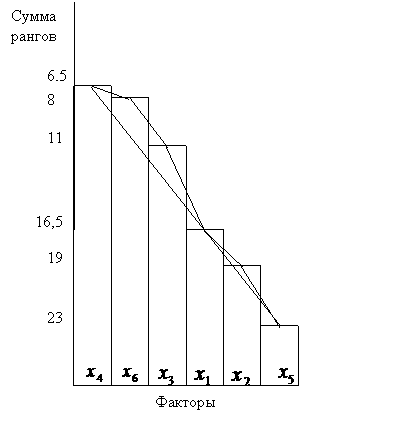
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Эксперты** **Факторы** | **1** | **2** | **3** | **4** | **Сумма рангов** | **Δ** |  |
| x1 | 3 | 4 | 5 | 4,5 | 16,5 | 2,5 | 6,25 |
| x2 | 5 | 6 | 3,5 | 4,5 | 19 | 5 | 25 |
| x3 | 4 | 3 | 1 | 3 | 11 | -3 | 9 |
| x4 | 1 | 1 | 3,5 | 1 | 6,5 | -7,5 | 56,25 |
| x5 | 6 | 5 | 6 | 6 | 23 | 9 | 81 |
| x6 | 2 | 2 | 2 | 2 | 8 | -6 | 36 |
| Σ | 21 | 21 | 21 | 21 | 84 | S = 213,5 | |

.  
Проверка правильности составления матрицы на основе исчисления контрольной суммы:  
.  
Сумма по столбцам матрицы равны между собой и контрольной суммы, значит матрица составлена правильно.

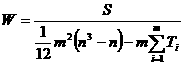
Этап 4. Анализ значимости исследуемых факторов.  
В данном примере факторы по значимости распределились следующим образом (табл. 5).  
*Таблица 5 - Расположение факторов по значимости*

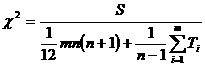
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Факторы | x4 | x6 | x3 | x1 | x2 | x5 |
| **Сумма рангов** | 6,5 | 8 | 11 | 16,5 | 19 | 23 |

Для наглядности полученных результатов оценок факторов построим гистограмму и полигон распределения сумм рангов по степени их значимости для потребителей (рис. 1), проведем классификацию факторов по сумме рангов.

  
Рис. 1. Гистограмма и полигон распределения сумм рангов

Гистограмма (рис. 1) позволяет сделать следующие выводы:  
1. Наибольшее значение для потребителей имеют фактор x4 (наличие систем Hoy Frost) и фактор x6 (дизайн).  
2. Вторая группа по значимости включает в себя один фактор x3 (количество камер).  
3. Третья группа включает в себя факторы x1 (объем камеры), x2 (бренд), x5 (энергопотребление).  
Итак, важнейшими для потребителей являются следующие технические параметры холодильников: x4, x6, x3.

Этап 5. Оценка средней степени согласованности мнений всех экспертов.  
Воспользуемся коэффициентом конкордации для случая, когда имеются связанные ранги (одинаковые значения рангов в оценках одного эксперта)  
,  
где , Li - число связок (видов повторяющихся элементов) в оценках i-го эксперта, tl - количество элементов в l-й связке для i-го эксперта (количество повторяющихся элементов).  
S = 213,5 (см. табл. 6.4), n = 6, m = 4,  
 (в оценках 3-его эксперта одна связка, повторяется ранг «3,5» 2 раза),  
, (в оценках 4-ого эксперта одна связка, повторяется ранг «4,5» 2 раза),  
, (если нет связанных рангов, то Ti равно нулю)  
.  
W = 0.774 говорит о наличии высокой степени согласованности мнений экспертов. На высокую степень согласованности мнений экспертов указывает и полигон распределения сумм рангов (рис. 6.1). Ломанная и прямая линии близко расположены друг к другу.

Этап 6. Оценка значимости коэффициента конкордации.  
Для этой цели исчислим критерий согласования Пирсона: , .  
Вычисленный  сравним с табличным значением для числа степеней свободы K = n-1 = 6-1 = 5 и при заданном уровне значимости α = 0,05.  
Так как  расчетный 15,471 >  табличного = 11,07, то W = 0.744 - величина не случайная, а потому полученные результаты по оценке качественных свойств холодильников по степени их значимости для потребителей имеют смысл и могут использоваться в дальнейших исследованиях.

Этап 7. Подготовка решения экспертной комиссии.  
Была поставлена задача выяснить, какие технические параметры холодильников в наибольшей степени интересуют потребителей.  
В результате проведенного исследования на основе экспертных оценок выяснилось, что важнейшими качественными свойствами являются: наличие систем, дизайн и количество камер. Следовательно, качественное совершенствование холодильников должны идти по пути улучшения этих свойств (исходя из требований НТП).  
Себестоимость, цены, доплата к ценам на изделия улучшенного качества и скидки с цен на холодильники должны строиться с учетом вышеперечисленных качественных свойств.  
На основе получения суммы рангов (табл. 4) можно вычислить показатели весомости рассмотренных технических параметров холодильников для потребителя с тем, чтобы их можно было учитывать при оценке технического уровня холодильников. Для этого произведем следующие вычисления. Сначала по каждому параметру вычислим величины, обратные сумме рангов, то есть  
 ; ;  
; ; .  
Это делается для того, чтобы привести в соответствие содержание сумм рангов коэффициентам весомости. Расположим полученные числа по мере убывания, сложим их, взвесим каждое число в полученной сумме, которую примем равной 1 (табл. 6).

*Таблица 6*

|  |  |  |
| --- | --- | --- |
| **Технические параметры** | **Величины, обратные сумме рангов** | **Коэффициенты весомости параметров** |
| x4  x6  x3  x1  x2  x5 | 0,15  0,12  0,09  0,06  0,05  0,04 | 0,29  0,23  0,18  0,12  0,10  0,08 |

Затем введем таблицу экспертных оценок проектов внедрения холодильников в производство.

Упорядочим оценки каждого эксперта по возрастанию. Затем найдем медиану, т.е. среднее тех значений, которые стоят на m = 12 / 2 = 6 и 7 местах.

*Таблица 7*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| 1 | 8 | 7 | 5 | 6 | 8 | 5 | 8 | 8 |
| 2 | 7 | 6 | 5 | 4 | 8 | 4 | 7 | 8 |
| 3 | 6 | 6 | 4 | 3 | 8 | 4 | 7 | 8 |
| 4 | 6 | 4 | 4 | 3 | 8 | 4 | 7 | 8 |
| 5 | 6 | 4 | 3 | 3 | 8 | 4 | 6 | 7 |
| 6 | 5 | 3 | 3 | 2.5 | 8 | 4 | 6 | 7 |
| 7 | 5 | 3 | 3 | 2 | 7 | 4 | 6 | 7 |
| 8 | 5 | 2 | 3 | 2 | 6 | 4 | 6 | 7 |
| 9 | 5 | 1 | 2.5 | 2 | 5 | 2 | 5 | 7 |
| 10 | 5 | 1 | 2 | 2 | 5 | 2 | 3 | 7 |
| 11 | 1 | 1 | 2 | 1 | 3 | 1 | 2 | 6 |
| 12 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 5 |
| Медианы рангов | 5 | 3 | 3 | 2.25 | 7.5 | 4 | 6 | 7 |

Итак, ранжировка по медианам рангов имеет вид:

*Таблица 8*

|  |  |  |
| --- | --- | --- |
| N | Медиана | Новый ранг |
| A | 5 | 5 |
| B | 3 | 2.5 |
| C | 3 | 2.5 |
| D | 2.25 | 1 |
| E | 7.5 | 8 |
| F | 4 | 4 |
| G | 6 | 6 |
| H | 7 | 7 |

D < B, C < F < A < G < H < E  
Здесь запись типа D < B означает, что проект D предшествует проекту B (т.е. проект D лучше проекта B).  
Поскольку некоторые проекты получили одинаковую сумму баллов, то по рассматриваемому методу они эквивалентны, а потому объединены в группу - класс эквивалентности.

В соответствии с определением медианы Кемени следует ввести в рассмотрение функцию

С(А) = ∑ D(Ai ,A) = D(A2 ,A)+D(A4 ,A)+D(A5 ,A)+D(A8 ,A)+D(A9 ,A)

## АЛГОРИТМЫ РЕШЕНИЯ

Этап 1. Создание экспертной комиссии.

Число факторов n, Число экспертов m.

Этап 2. Сбор мнений специалистов путем анкетного опроса.

Оценку степени значимости параметров эксперты производят путем присвоения им рангового номера. Фактору, которому эксперт дает наивысшую оценку, присваивается ранг 1. Если эксперт признает несколько факторов равнозначными, то им присваивается одинаковый ранговый номер. На основе данных анкетного опроса составляется сводная матрица рангов.

Этап 3. Составление сводной матрицы рангов.

Если в матрице имеются связанные ранги (одинаковый ранговый номер) в оценках m-го эксперта, произведем их переформирование. Переформирование рангов производиться без изменения мнения эксперта, то есть между ранговыми номерами должны сохраниться соответствующие соотношения (больше, меньше или равно). Также не рекомендуется ставить ранг выше 1 и ниже значения равного количеству параметров. Переформирование рангов производится в табл.

На основании переформирования рангов строится новая матрица рангов.

Проверка правильности составления матрицы на основе исчисления контрольной суммы.

Сумма по столбцам матрицы равны между собой и контрольной суммы, значит, матрица составлена правильно.

Этап 4. Анализ значимости исследуемых факторов.

В данном примере факторы по значимости распределились следующим образом (табл.).

Расположение факторов по значимости

Этап 5. Оценка средней степени согласованности мнений всех экспертов.

Воспользуемся коэффициентом конкордации для случая, когда имеются связанные ранги (одинаковые значения рангов в оценках одного эксперта).

Li - число связок (видов повторяющихся элементов) в оценках i-го эксперта, tl - количество элементов в l-й связке для i-го эксперта (количество повторяющихся элементов).

W говорит о наличии слабой степени согласованности мнений экспертов.

Этап 6. Подготовка решения экспертной комиссии.

На основе получения суммы рангов (табл.) можно вычислить показатели весомости рассмотренных параметров. Матрицу опроса преобразуем в матрицу преобразованных рангов по формуле sij = xmax - xij.

Метод медиан рангов.

Алгоритм метода медиан рангов:

* Упорядочивание оценок каждого эксперта по возрастанию.
* Нахождение медианы, mj.
* Построение новой ранжировки.
* Выбор проектов из принципа - чем меньше ранг, чем лучше проект.

С помощью расстояния Кемени находят итоговое мнение комиссии экспертов. Пусть А1 , А2 , А3 ,…, Ар - ответы р экспертов, представленные в виде бинарных отношений. Для их усреднения используют т.н. медиану Кемени

Arg min ∑ D (Ai ,A) ,

где Arg min - то или те значения А, при которых достигает минимума указанная сумма расстояний Кемени от ответов экспертов до текущей переменной А, по которой и проводится минимизация. Таким образом,

∑ D (Ai ,A) = D (A1 ,A) + D (A2 ,A) + D (A3 ,A) +…+ D (Aр ,A) .

Кроме медианы Кемени, используют среднее по Кемени, в котором вместо D (Ai ,A) стоит D2 (Ai ,A) .

Медиана Кемени - частный случай определения эмпирического среднего в пространствах нечисловой природы [1]. Для нее справедлив закон больших чисел, т.е. эмпирическое среднее приближается при росте числа составляющих (т.е. р - числа слагаемых в сумме), к теоретическому среднему:

Arg min ∑ D (Ai ,A) → Arg min М D (A1 , A) .

Здесь М - символ математического ожидания. Предполагается, что ответы р экспертов А1 , А2 , А3 ,…, А р есть основания рассматривать как независимые одинаково распределенные случайные элементы (т.е. как случайную выборку) в соответствующем пространстве произвольной природы, например, в пространстве упорядочений или отношений эквивалентности.

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ

**Бинарные отношения:**

Запустим код. После запуска в консоли должны появиться следующие данные



Рисунок 1

Получаем массив ранговых оценок экспертов.

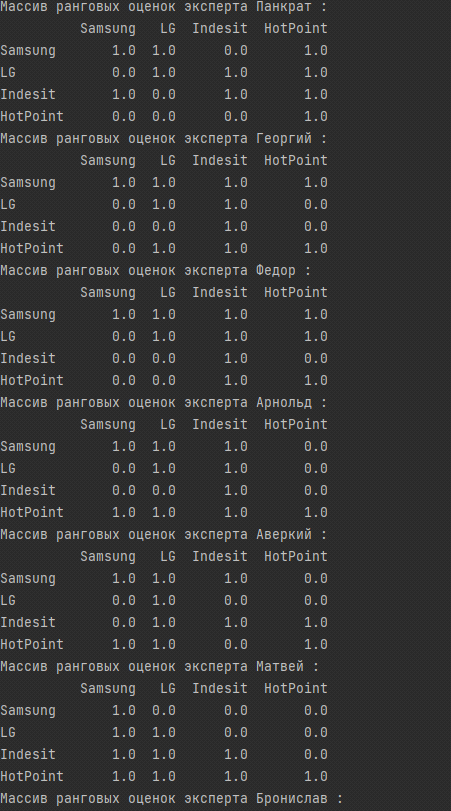


Рисунок 2

Получаем медиану кемени, далее получаем значение целевой функции:



Рисунок 3

**Экспертные оценки:**

При запуске кода изначально получаем варианты оценок респондентов:

Математическое ожидание:

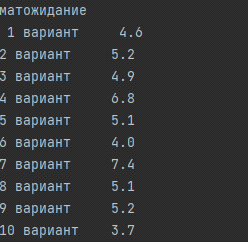


Рисунок 4

Дисперсия:

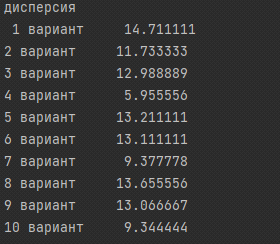


Рисунок 5

Стандартное отклонение:

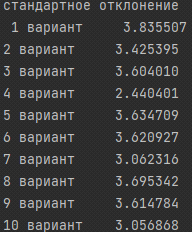


Рисунок 6

Среднеквадратичное отклонение:

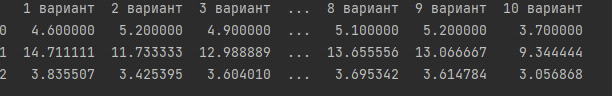


Рисунок 7

Доверительный интервал:

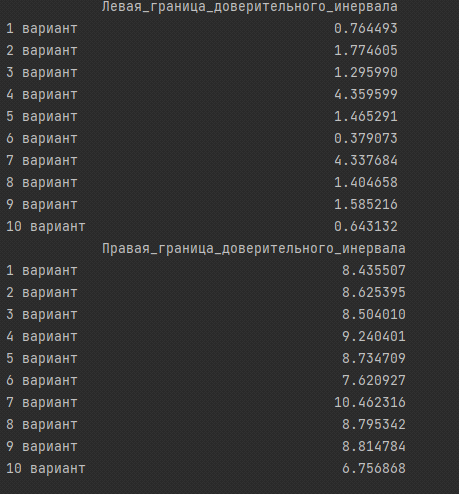


Рисунок 8

Гистограмма вариантов:

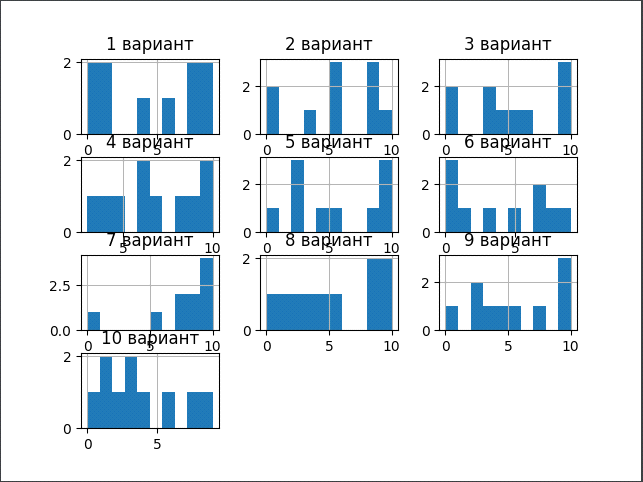


Рисунок 9

**Ранговые оценки**

При запуске получаем оценки респондентов:



Рисунок 10

Медиана оценок:

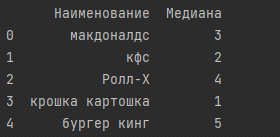


Рисунок 11

Коэффициент компетентности респондентов:

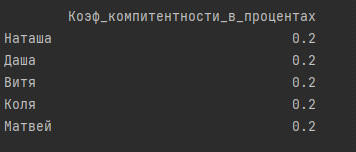


Рисунок 12

Средневзвешенные ранги:

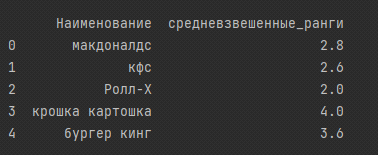


Рисунок 13

Гистограмма вариантов:

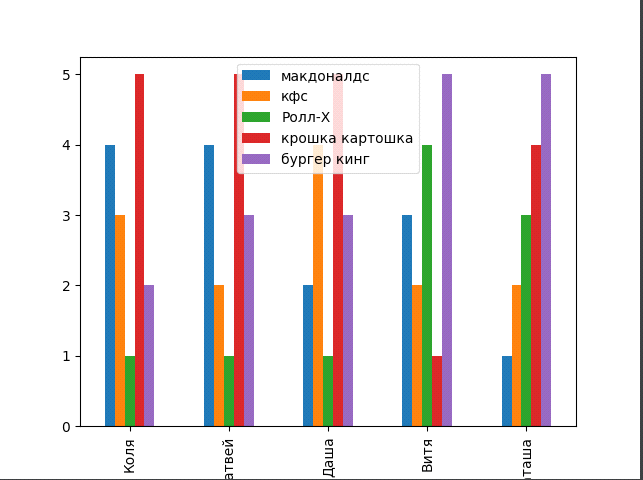


Рисунок 14

## АРХИТЕКТУРА РЕШЕНИЯ

Файл binary\_relationships.py

Импортируем библиотеки:

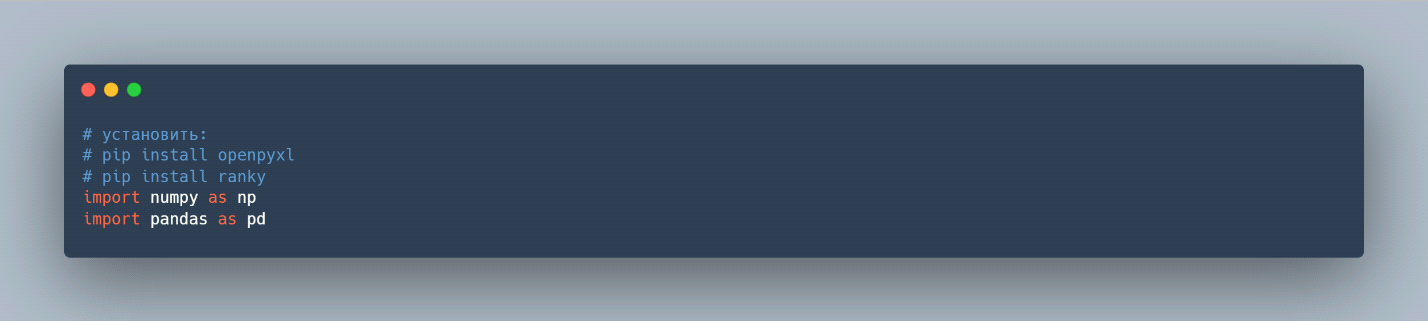


Рисунок 15

Читаем excel файл:

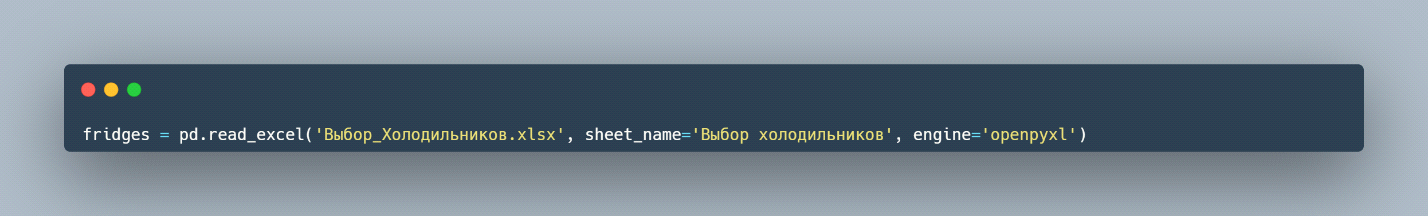


Рисунок 16

Считываем данные колонок таблицы:



Рисунок 17

Генерация матриц бинарных отношений для всех рассматриваемых экспертов:

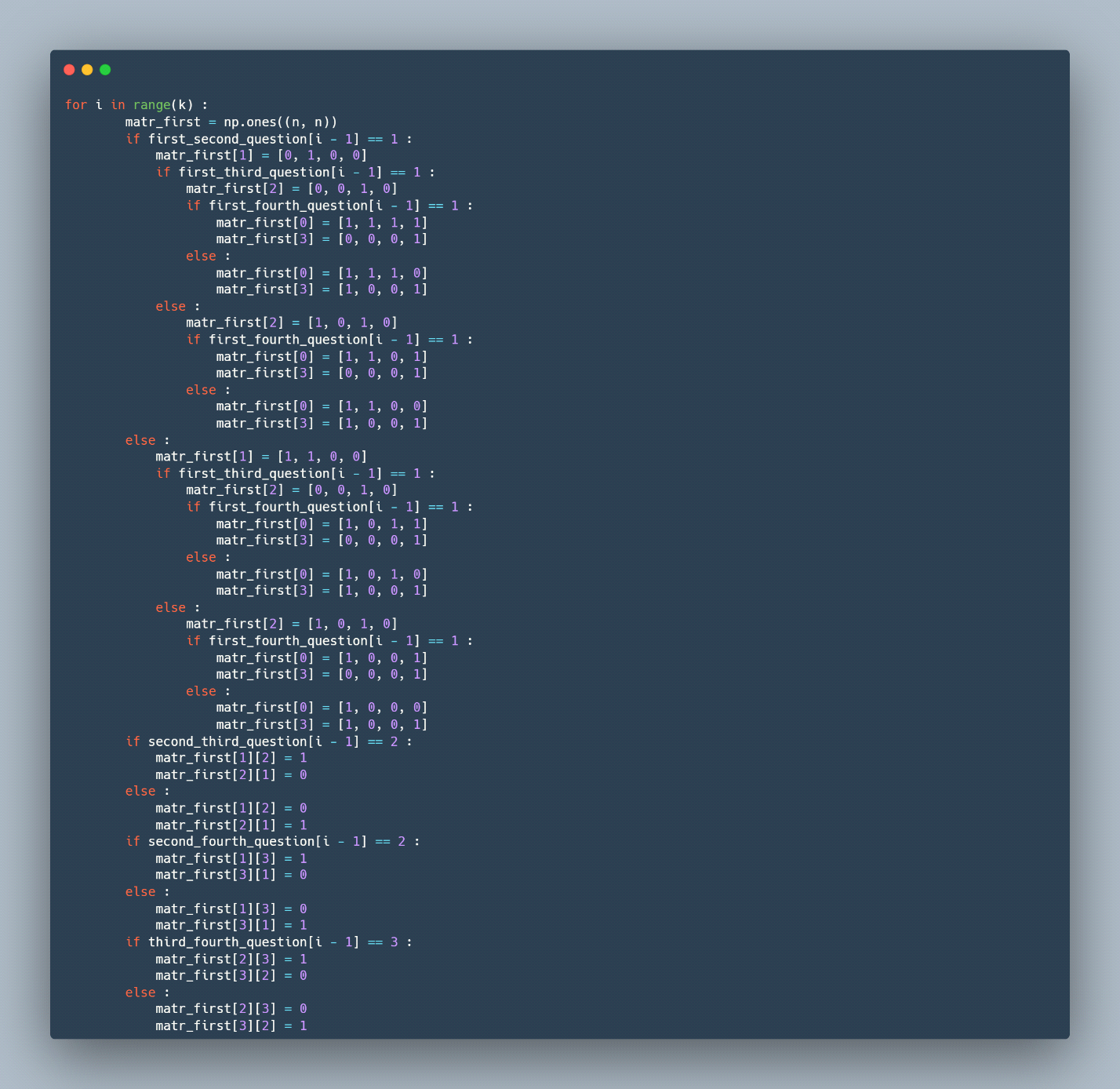


Рисунок 18

Задаем названия колонок и строк в датафрейме:

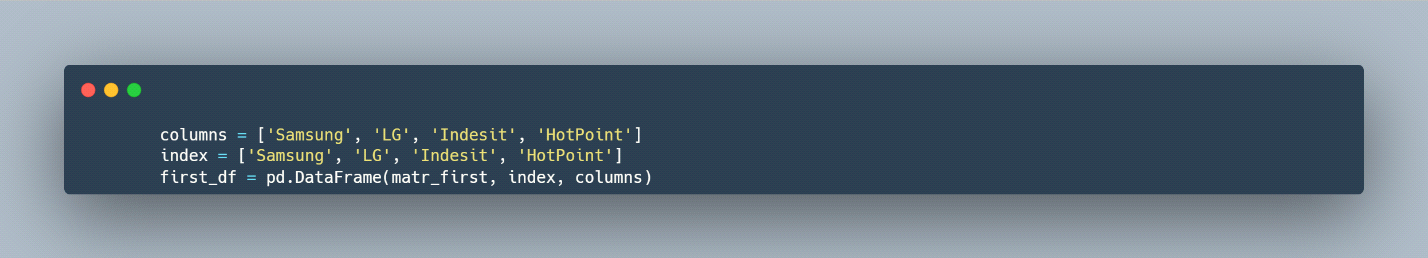


Рисунок 19

Формирование матрицы Кемени:



Рисунок 20

Расчет расстояние между матрицами бинарных отношение и расчет значения целевой функции:



Рисунок 21

Файл expert\_points.py

Импорт необходимых библиотек:

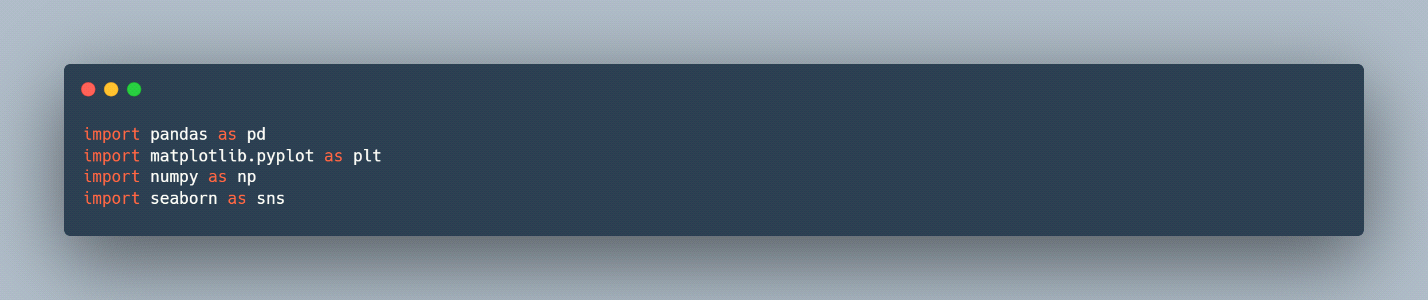


Рисунок 22

Считывание файла с данными и создание пустых фреймов:

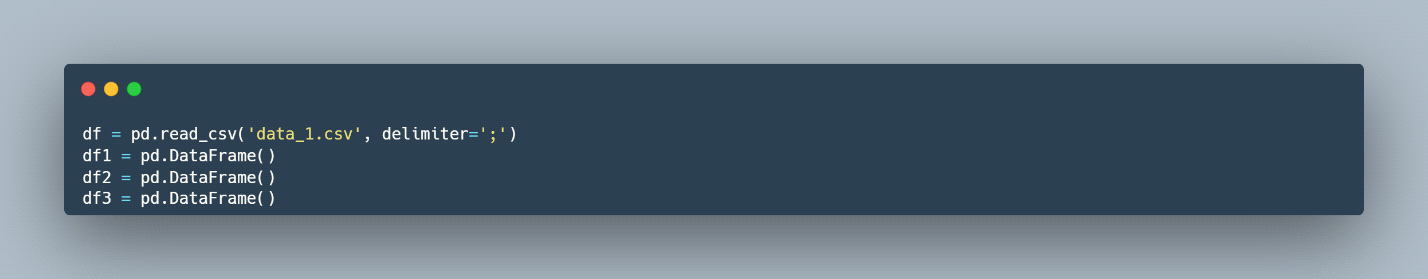


Рисунок 23

Расчет необходимых значений, а также добавление в созданные фреймы значений, которые впоследствии участвуют в построении диаграммы:



Рисунок 24

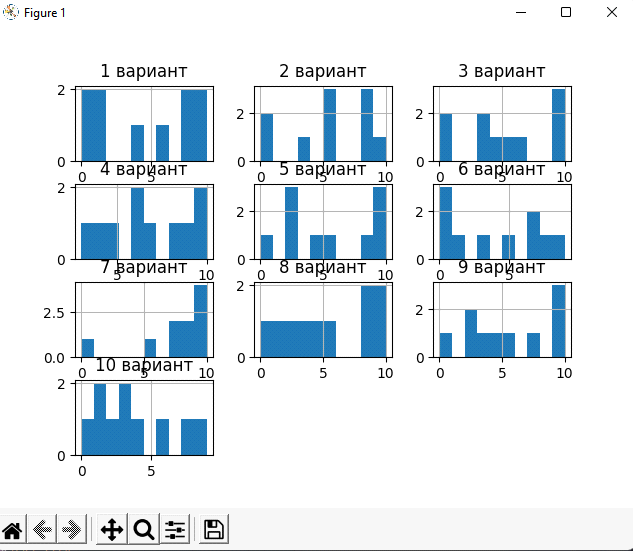


Рисунок 25

Файл rank\_points.py

Импорт необходимых библиотек:



Рисунок 26

Считывание файла с данными:



Рисунок 27

Сортировка фрейма и создание пустых фреймов:

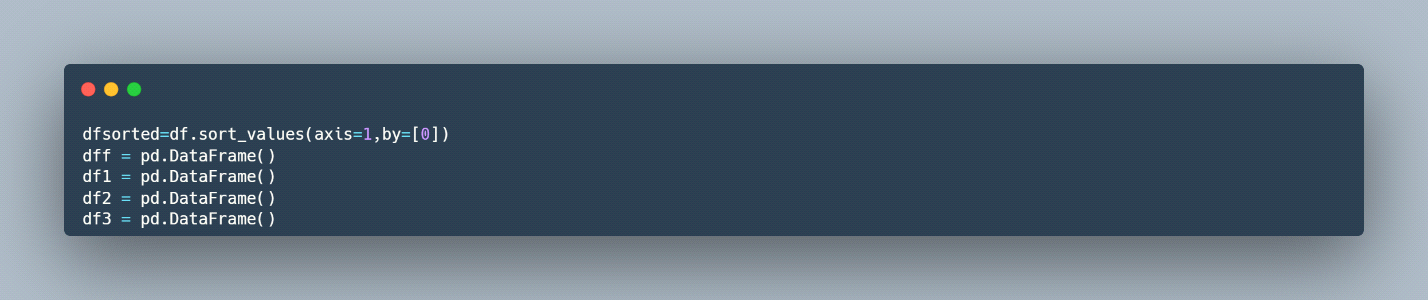


Рисунок 28

Расчет значений и создание новых столбцов во фрейме:

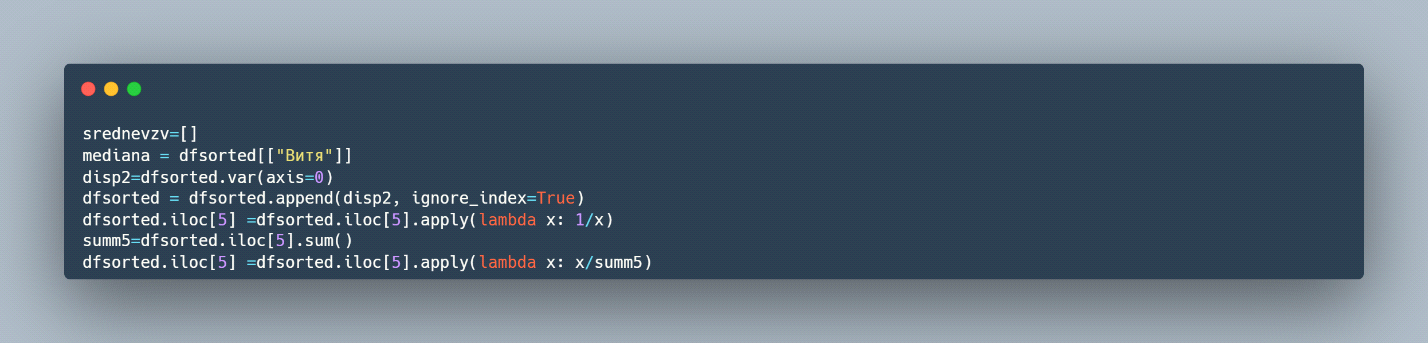


Рисунок 29

Действия с фреймами и построение гистограммы, а также ее вывод:

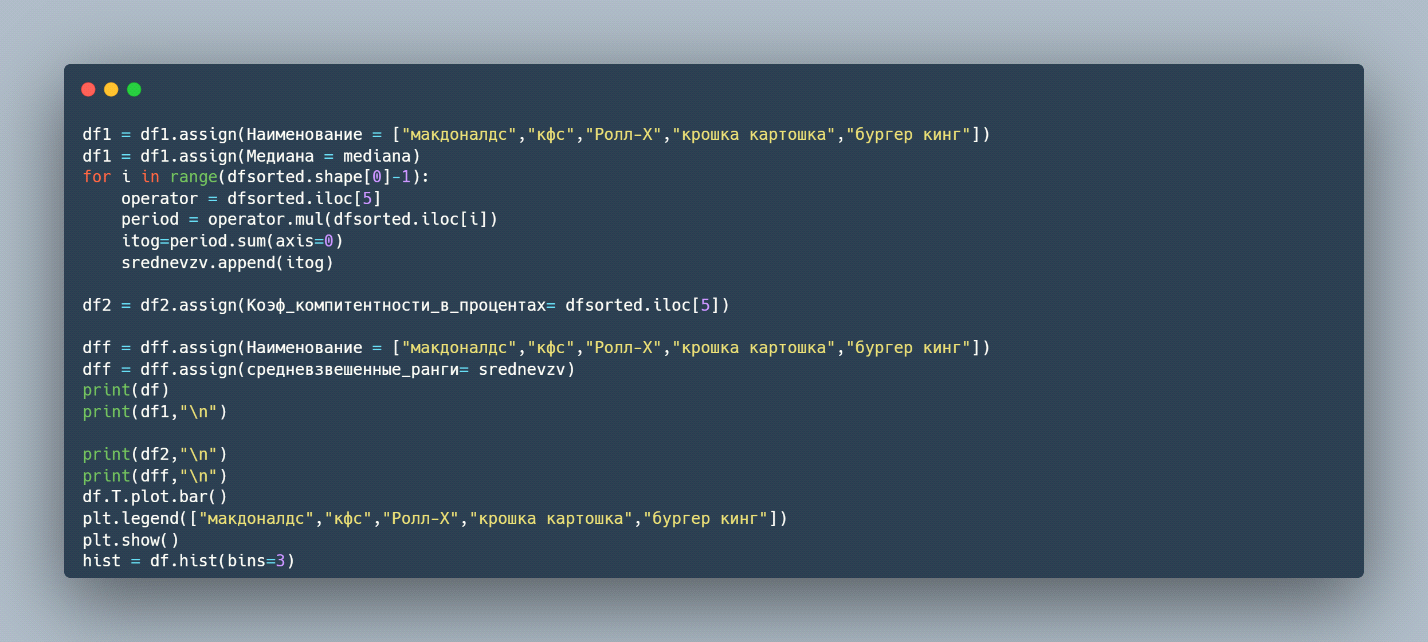


Рисунок 30

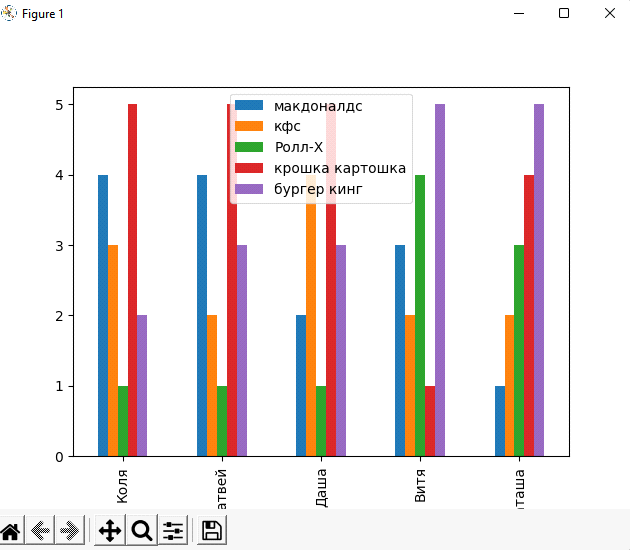


Рисунок 31

## ТЕСТИРОВАНИЕ

Для выбора наилучшей марки холодильника был проведен опрос 54 респондентов. В качестве характеристик-критериев холодильников были выбраны вместительность, дизайн, кол-во камер и экономичность. Выбирали респонденты из 4 марок холодильников: Samsung, LG, Hotpoint-Ariston и Indesit. Код вывел следующие результаты:

Метод 10-бальных вопросов

[https://docs.google.com/forms/d/1pk2vaGiMPyRjsGeHtGgi2J\_9z01T3Xak2Asxj3PN2UE/edit](https://vk.com/away.php?utf=1&to=https%3A%2F%2Fdocs.google.com%2Fforms%2Fd%2F1pk2vaGiMPyRjsGeHtGgi2J_9z01T3Xak2Asxj3PN2UE%2Fedit)

В ходе опроса респонденты ответили оценкой по 10-бальной шкале на 20 вопросов. По результатам опроса выявлены средние значения:

Таблица 9

|  |  |
| --- | --- |
| Вместительность - важность | 9 |
| Дизайн - важность | 6 |
| Кол-во камер - важность | 7 |
| Экономичность - важность | 8 |
| Samsung - вместительность | 7 |
| Samsung - дизайн | 6 |
| Samsung - кол-во камер | 8 |
| Samsung - экономичность | 8 |
| Indesit - вместительность | 6 |
| Indesit - дизайн | 5 |
| Indesit - кол-во камер | 7 |
| Indesit - экономичность | 6 |
| LG - вместительность | 8 |
| LG - дизайн | 7 |
| LG - кол-во камер | 6 |
| LG - экономичность | 9 |
| Hotpoint-Ariston - вместительность | 7 |
| Hotpoint-Ariston - дизайн | 9 |
| Hotpoint-Ariston - кол-во камер | 7 |
| Hotpoint-Ariston - экономичность | 8 |

По итогам опроса:

* Самая важная характеристика – вместительность
* Самый вместительная марка холодильника – LG
* Марка холодильника лучшая по дизайну – Hotpoint - Ariston
* Марка холодильника с лучшим кол-вом камер – Samsung
* Самая экономичная марка холодильника – LG
* Лучшая марка холодильника (среднее по всем характеристикам ) – Hotpoint – Ariston

Метод ранговых вопросов:

[https://docs.google.com/forms/d/1ojvViYbbSKSVFQvMeWCnfw1itlFHwpLgr9Oa0xCywcA/edit](https://vk.com/away.php?utf=1&to=https%3A%2F%2Fdocs.google.com%2Fforms%2Fd%2F1ojvViYbbSKSVFQvMeWCnfw1itlFHwpLgr9Oa0xCywcA%2Fedit)

В ходе опроса респонденты расставили марки холодильников по различным местам в топах по различным характеристикам. По результатам опроса выявлены средние значения:

Таблица 10

|  |  |
| --- | --- |
| Топ холодильников по вместительности | 1. LG  2. Hotpoint - Ariston  3. Samsung  4. Indesit |
| Топ холодильников по дизайну | 1. Hotpoint - Ariston  2. LG  3. Samsung  4. Indesit |
| Топ холодильников по кол-ву камер | 1. Samsung  2. Hotpoint - Ariston  3. Indesit  4. LG |
| Топ холодильников по экономичности | 1. LG  2. Hotpoint - Ariston  3. Samsung  4. Indesit |

* Лучшая марка холодильника по итогам опроса – Hotpoint – Ariston

Метод бинарных вопросов:

<https://docs.google.com/forms/d/1hVWyQ890nh_-KNDb-4YKIbuPgJCyO9m9D9V2KqtFagA/edit>

В ходе опроса респонденты сравнивали марки холодильников между собой. По результатам опроса выявлены результаты:

Таблица 11

|  |  |
| --- | --- |
| Samsung или LG? | Победитель: LG |
| Samsung или Indesit? | Победитель: Samsung |
| Samsung или HotPoint? | Победитель: Hotpoint |
| LG или Indesit? | Победитель: LG |
| LG или HotPoint? | Победитель: Hotpoint |
| Indesit или HotPoint? | Победитель: Hotpoint |

* Лучшая марка холодильника по итогам опроса – Hotpoint – Ariston

Таким образом, проведя опрос более 50 респондентов тремя способами выяснилось, что наилучшая марка холодильника – Hotpoin – Ariston

Также, код показал себя с лучшей стороны:

Таблица 12

|  |  |
| --- | --- |
| Параметры | Код |
| Скорость работы кода | 10 секунд < |
| Наличие дефектов | Отсутствуют |
| Верификация | Полное соответствие с поставленной задачей |
| Легкость восприятия для пользователя | Средне |
| Вариативность использования | Высокая |

## ЗАКЛЮЧЕНИЕ

Само экспертное оценивание — процедура получения оценки проблемы на основе мнения специалистов (экспертов) с целью последующего принятия решения (выбора).

В случаях чрезвычайной сложности проблемы, ее новизны, недостаточности имеющейся информации, невозможности математической формализации процесса решения приходится обращаться к рекомендациям компетентных специалистов, прекрасно знающих проблему, — к экспертам. Их решение задачи, аргументация, формирование количественных оценок, обработка последних формальными методами получили название метода экспертных оценок.

Отсюда следует вывод, что данное решение анализа экспертных оценок помогает принимать даже самые непростые решения, в том числе в условиях бизнеса для рационального подхода к его развитию.

Данное решение имеет три способа математического анализа экспертных оценок для принятия наиболее точного и взвешенного решения, однако существуют и другие алгоритмы, и методы анализа оценок, которые в работе не применялись, следовательно, решение имеет путь развития и расширения.

# СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ №8

## ФИЗИЧЕСКАЯ МОДЕЛЬ

В качестве рассматриваемой организации возьмем McDonald’s. Этот ресторан быстрого питания является самым популярным в России. Ежесуточно каждую точку посещают около 1000 человек. Рассмотрим один день одного пункта «Макдональдс». Возьмем время работы его с 6 утра до 12 ночи, количество касс – 10 шт., максимальная очередь в кассу 7 человек. Следует определить, сколько касс самообслуживания потребуется, чтобы обслужить всех клиентов.

Данные будут представлены в таблице:

Таблица 1 – исходные данные задачи

|  |  |  |
| --- | --- | --- |
| Временной промежуток | Количество клиентов | Время обслуживания одного клиента |
| 6:00-12:00 | 200 | 2 |
| 12:00-18:00 | 350 | 2 |
| 18:00-00:00 | 450 | 2 |

## МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Теория массового обслуживания является разделом математической экономики, который занимается анализом процессов в системах обслуживания, производства и управления, в которых однородные действия повторяются многократно (предприятия торговли, транспортные системы, автоматические линии производства). Предмет у теории массового обслуживания – установленные зависимости между основными характеристиками системы обслуживания, а именно:

1. Число каналов обслуживания;
2. Характер входного потока заявок, которые необходимо обслужить;
3. Производительность отдельно взятого канала.

Можно сказать, что все задачи теории массового обслуживания носят оптимизационный характер и все они направлены на определения такого варианта работы системы, при котором будет обеспечен минимум суммарных затрат от простоев каналов обслуживания, потерь времени и ресурсов на обслуживания.

Рассмотрим непосредственно задачи теории массового обслуживания:

1. Построение математической модели системы и ее основных характеристик;
2. Установление зависимости эффективности работы системы;
3. Решение оптимизационных задач, связанных с функционированием системы;
4. Рациональное построение системы массового обслуживания.

Структура системы массового обслуживания:

1. Заявки на обслуживание;
2. Каналы обслуживания;

Так как большое количество заявок образует потоки, которые ждут своей очереди до непосредственного их выполнения, а после возможного начала их обслуживания образуется поток обслуживания в каналах, далее происходит формирование выходящего потока заявок, в теории массового обслуживания существует термин «Интенсивность потока покупателей», который обозначает основную характеристику потока заявок на обслуживание.

Интенсивность потока покупателей рассчитывается по следующей формуле:

Интенсивность потока покупателей = Длительность обслуживания - Случайная величина, подчиненная показательному закону распределения

(Поток обслуживаний является простейшим пуассоновским потоком событий. Размер очереди допускает нахождение в ней N заявок.)

Входной поток заявок рассчитывается с помощью интенсивности входного потока заявок, ниже представлена формула:

N – количество заявок, пришедшее за период времени,

T - средний интервал времени между поступлением заявок.

Выходной поток заявок непосредственно связан с потоком обслуживания в канале, подчиняется показательному закону распределения:

- интенсивность потока обслуживания, т.е. среднее число заявок, обслуживаемых в единицу времени

tобсл – случайная величина, время обслуживания

Интенсивность потока находится по формуле:

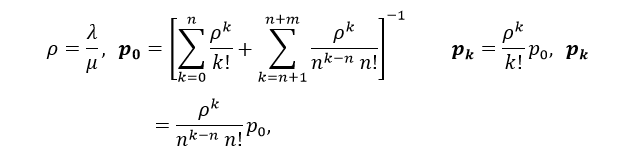
Также при анализе случайных процессов выгоднее использовать граф, а именно схематическое изображение возможных состояний систем массового обслуживания. Рассмотрим граф на примере нашего ресторана быстрого питания:



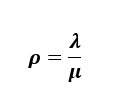
Рисунок 1

Система тут может находиться в одном из этих трех состояний, а именно в одном из этих трех каналов 1 человек на кассе самообслуживания, 5 человека на кассе самообслуживания, 10 человек на кассе самообслуживания. Переход из одного канала в другой происходит под воздействием потока интенсивности. Переход – задержка системы в данном состоянии.

Вероятности нахождения системы в каждом из одного калана находится по формуле:

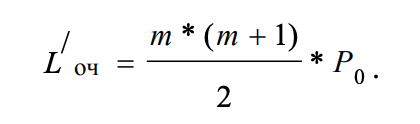


Показатель загруженности касс находится по формуле:

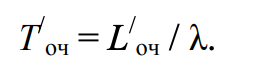


При этом, если получается что p < n, то система является недогруженной, что, безусловно, является выгодным для клиентов, но при этом невыгодной для работников предприятия, а если p > n, то присутствует баланс в работе, допустимые показатели по загруженности, есть вариант, когда p > n, тогда система может быть перегруженной.

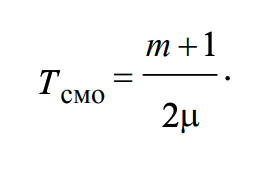
Среднее число заявок, стоящих в очереди на обслуживании (Lоч):



Среднее время ожидания обслуживания:



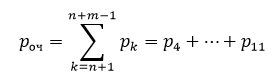
Время пребывания заявок в системе (среднее время пребывания в очереди и среднее время обслуживания) вычисляется по формуле:



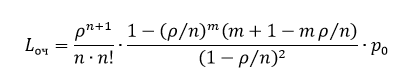
Помимо этих формул, есть еще необходимые отдельно для клиентов и отдельно для работников заведения формулы, к примеру:

Для клиентов важно знать:

1. Вероятность встать в очередь возможно рассчитать по формуле:



1. Приблизительная длина очереди:



1. Приблизительное время ожидания в очереди:



Для работников организации важно знать:

1. Среднее количество занятых приборов:



1. Абсолютная пропускная способность:



1. Скорость обслуживания заявок от клиентов:



1. Коэффициент простоя:



## АЛГОРИТМЫ РЕШЕНИЯ

### Параметры входных данных

Массивы данных по объекту Макдоналдс: временной промежуток, количество потребителей, время приготовления продукта, количество точек выдачи заказа, максимально-допустимая очередь.

### Алгоритм решения

1. Просчитаем интенсивность входного потока заявок: количество заявок на каждое время суток поделим на интервал времени.

Утро (с 9:00 до 12:00): 0,666666667 чел/мин

День (с 12:00 до 17:00): 0,666666667 чел/мин

Вечер (с 17:00 до 22:00): 0,666666667 чел/мин

2. Просчитаем интенсивность выходного потока заявок от одного прибора: количество обработанных одним прибор в единицу времени:

Утро (с 9:00 до 12:00): 0,666666667 чел/мин

День (с 12:00 до 17:00): 0,666666667 чел/мин

Вечер (с 17:00 до 22:00): 0,666666667 чел/мин

3. Просчитаем показатель нагруженности – интенсивность входного потока заявок работоспособность прибора (интенсивность выходного потока заявок от одного прибора):

|  |  |
| --- | --- |
| Утро (с 9:00 до 12:00) | 1 |
| День (с 12:00 до 17:00) | 1 |
| Вечер (с 17:00 до 22:00) | 1 |

Система сбалансирована во все части дня.

4. Просчитаем числовые характеристики системы для клиентов:

Изображение выглядит как текст

Автоматически созданное описание

4.1 Просчитаем величину вероятности отказа клиенту в обслуживании.

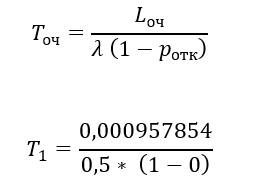
4.2 Просчитаем вероятность встать в очередь (сумма вероятности всех очередей).

4.3 Просчитаем среднюю длину очереди.

Изображение выглядит как стол

Автоматически созданное описание

4.4 Просчитаем среднее время ожидания по формуле Литтла.



5. Просчитаем числовые характеристики системы для владельцев:

5.1 Просчитаем абсолютную пропускную способность.

Изображение выглядит как стол

Автоматически созданное описание

5.3 Просчитаем относительную пропускную способность.

Изображение выглядит как текст

Автоматически созданное описание

5.4 Просчитаем среднее количество занятых приборов.

Изображение выглядит как стол

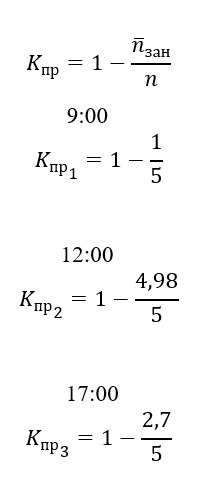
Автоматически созданное описание

5.5 Просчитаем коэффициент простоя

### Параметры выходных данных

Количество точек выдачи заказа для минимизации очереди.

Коэффициент простоя высчитывается по следующей формуле:



Утро (с 9:00 до 12:00): 20%

День (с 12:00 до 17:00): 0%

Вечер (с 17:00 до 22:00): 52%

## ВАРИАНТЫ ИСПОЛЬЗОВАНИЯ

Запустим код. После запуска в консоли должны появиться следующие данные:

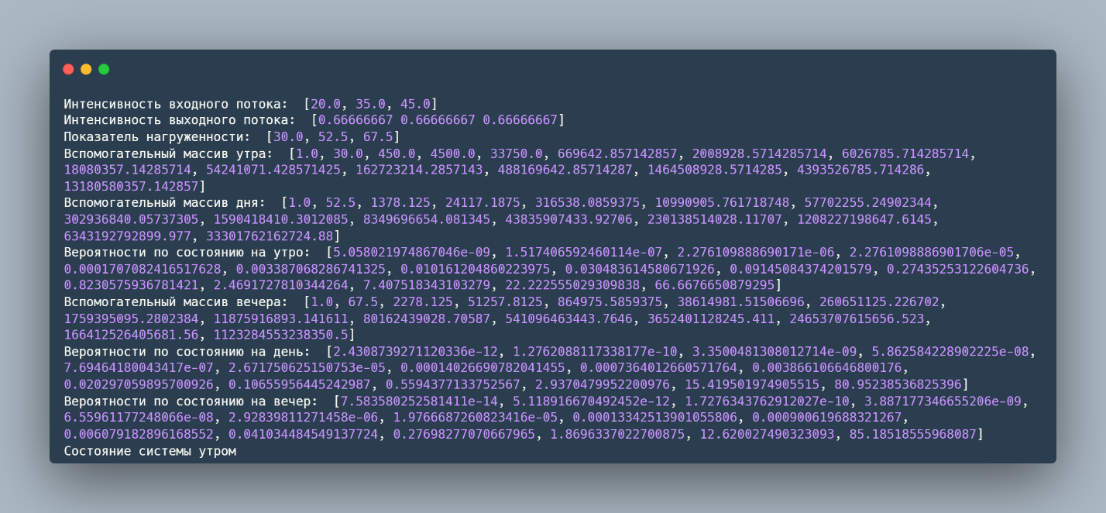


Рисунок 2

После должно быть выведено состояние системы утро:

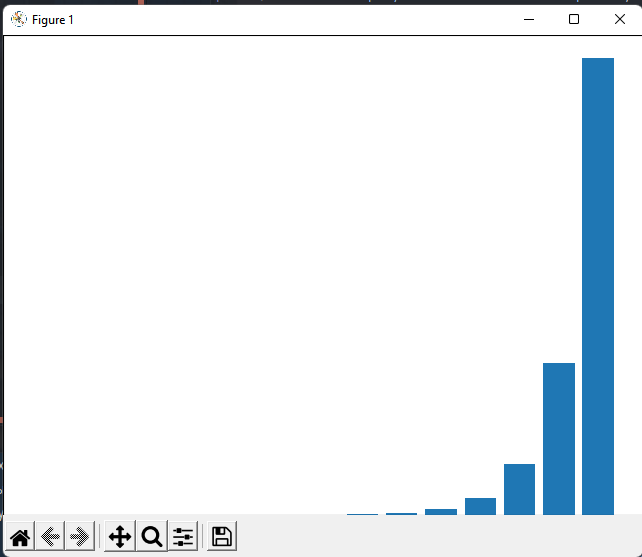


Рисунок 3

Состояние системы днем:



Рисунок 4

Состояние системы вечером:

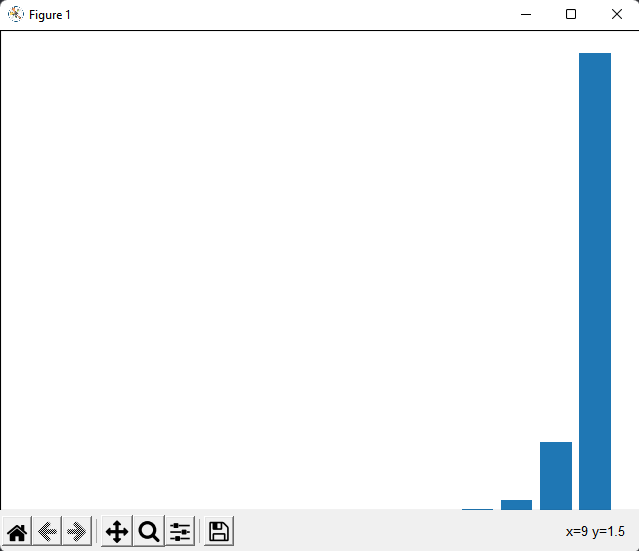


Рисунок 5

Далее должны быть выведены характеристики для клиента:

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 6

И основные характеристики для владельца:

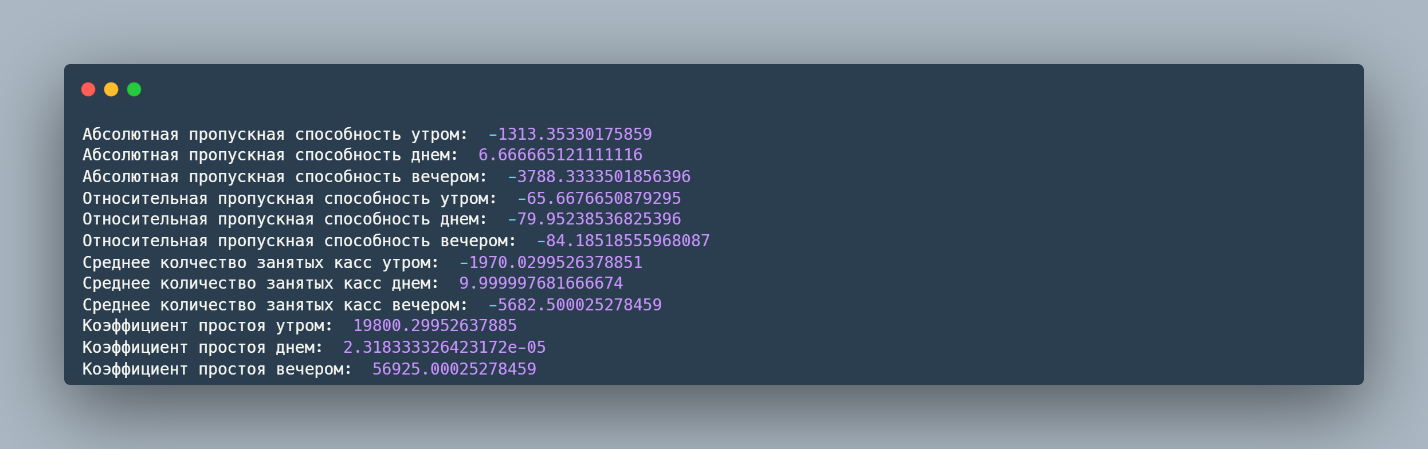


Рисунок 7

## АРХИТЕКТУРА РЕШЕНИЯ

Импорт библиотек:

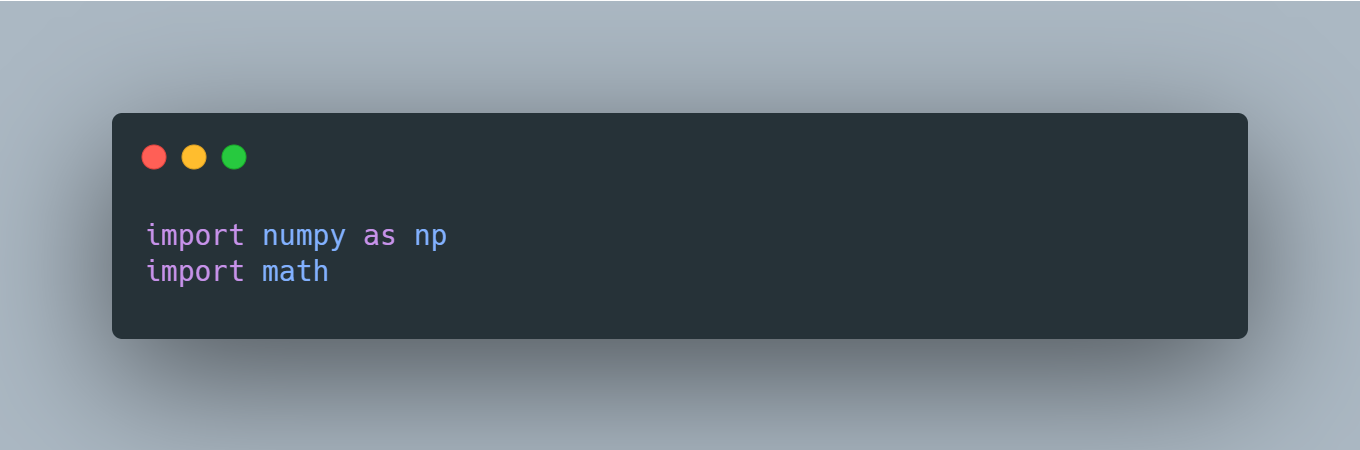


Рисунок 8

Задание данных для работы:



Рисунок 9

Преобразование данных для работы:

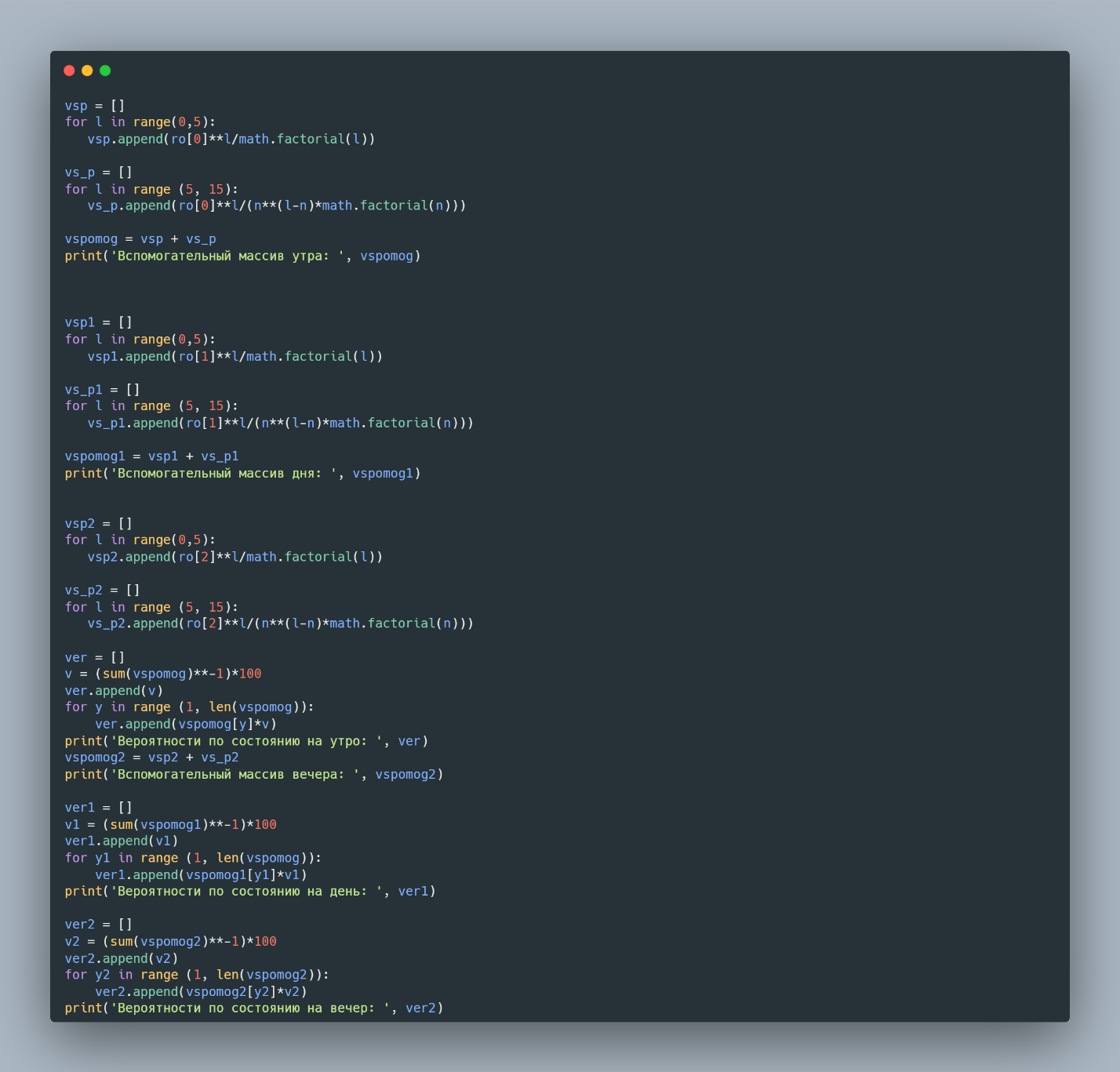


Рисунок 10

Построение графиков:



Рисунок 11

Характеристики для клиента:



Рисунок 12

Характеристики для владельца:



Рисунок 13

## ТЕСТИРОВАНИЕ

Для тестирования была выбрана сеть быстрого питания Макдональдс.

С помощью кода были выявлены следующие показатели:

1. Интенсивность входного потока
2. Интенсивность выходного потока
3. Показатель загруженности
4. Вспомогательный массив утра
5. Вспомогательный массив дня
6. Вероятности по состоянию на утро
7. Вспомогательный массив вечера
8. Вероятности по состоянию на день
9. Вероятности по состоянию на вечер
10. Состояние системы утром
11. Состояние системы вечером

Благодаря этому было выявлено, что вероятность отказа системы крайне мала.

В ходе тестирования были выявлены следующие результаты:

|  |  |
| --- | --- |
| Параметры | Код |
| Скорость работы кода | 10 секунд < |
| Наличие дефектов | Отсутствуют |
| Верификация | Полное соответствие с поставленной задачей |
| Легкость восприятия для пользователя | Высокая |
| Вариативность использования | Высокая |

## ЗАКЛЮЧЕНИЕ

В ходе решения была выполнена задача по аналитике данных «Макдональдс» с помощью ручного решения (на основе математической модели) и с помощью решения Python (визуальный пользовательский интерфейс).

В ходе решения было проанализирована возможность расширения точки потребления и необходимость в этом расширении. То есть данное решение служит нахождением оптимального решения для бизнеса для минимизации издержек и максимизации прибыли. Данное решение может оценить, теряет ли в нынешней ситуации любой магазин (производитель) прибыль и позволяет делать выводы о дальнейших перспективах развития.

Данное решение имеет развитие в качестве визуализации решения (графики, линейчатые диаграммы, точечные диаграммы) для более наглядного вывода о работе и упрощения анализа данных.