**Федеральное государственное образовательное**

**бюджетное учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ**

**ФЕДЕРАЦИИ»**

**(Финансовый университет)**

**Факультет**

**информационных технологий и анализа больших данных**

**Кафедра «Бизнес-информатика»**

**Домашнее задание № 1**

«Задачи линейного программирования: транспортная задача.»

Студенты группы БИ20-8:

Луканина Полина

Аверкин Никита

Филимонова Арина

Совин Владимир

Горшков Георгий

Киселева Евгения

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

### Оглавление

[Оглавление 2](#_Toc97216721)

[1. Постановка задачи (физическая модель) 5](#_Toc97216722)

[2. Математическая модель 6](#_Toc97216723)

[3. Алгоритмы 7](#_Toc97216724)

[3.1. Алгоритм 1 7](#_Toc97216725)

[3.1.1. Описание входных данных 7](#_Toc97216726)

[3.1.2. Описание алгоритма решения 7](#_Toc97216727)

[3.1.3. Описание выходных данных 7](#_Toc97216728)

[3.2. Алгоритм 2 7](#_Toc97216729)

[3.2.1. Описание входных данных 7](#_Toc97216730)

[3.2.2. Описание алгоритма решения 7](#_Toc97216731)

[3.2.3. Описание выходных данных 8](#_Toc97216732)

[3.3. Алгоритм 3 8](#_Toc97216733)

[3.3.1. Описание входных данных 8](#_Toc97216734)

[3.3.2. Описание алгоритма решения 8](#_Toc97216735)

[3.3.3. Описание выходных данных 9](#_Toc97216736)

[4. Варианты использования системы 9](#_Toc97216737)

[4.1. ВИ 1 9](#_Toc97216738)

[4.2. ВИ 2 10](#_Toc97216739)

[4.3. ВИ 3 10](#_Toc97216740)

[5. Архитектура решения 12](#_Toc97216741)

[5.1. Методы считывания информации 12](#_Toc97216742)

[5.2. Методы обработки информации 13](#_Toc97216743)

[5.3. Методы вывода информации 14](#_Toc97216744)

[6. Тестирование 15](#_Toc97216745)

[6.1. Результаты тестирования первого дата сета: 16](#_Toc97216746)

[6.1.1. Проверка кода Python и визуализации: 16](#_Toc97216747)

[6.1.2. Проверка кода методом Excel: 16](#_Toc97216748)

[6.1.3. Проверка кода в онлайн-калькуляторе: 17](#_Toc97216749)

[6.2. Результаты тестирования второго дата сета: 17](#_Toc97216750)

[6.2.1. Проверка кода Python и визуализация: 17](#_Toc97216751)

[6.2.2. Проверка кода методом Excel: 18](#_Toc97216752)

[6.2.3. Проверка кода в онлайн-калькуляторе: 18](#_Toc97216753)

[6.3. Результаты тестирования третьего дата сета: 19](#_Toc97216754)

[6.3.1. Проверка кода Python и визуализация: 19](#_Toc97216755)

[6.3.2. Проверка кода методом Excel: 19](#_Toc97216756)

[6.3.3. Проверка кода в онлайн-калькуляторе: 20](#_Toc97216757)

[6.4. Результаты тестирования четвёртого дата сета: 20](#_Toc97216758)

[6.4.1. Проверка кода Python и визуализации: 20](#_Toc97216759)

[6.4.2. Проверка кода методом Excel: 21](#_Toc97216760)

[6.4.3. Проверка кода в онлайн-калькуляторе: 21](#_Toc97216761)

[6.5. Результаты тестирования пятого дата сета: 22](#_Toc97216762)

[6.5.1. Проверка кода Python и визуализации: 22](#_Toc97216763)

[6.5.2. Проверка кода методом Excel: 23](#_Toc97216764)

[6.5.3. Проверка кода в онлайн-калькуляторе: 23](#_Toc97216765)

[6.6. Результаты тестирования шестого датасета: 24](#_Toc97216766)

[6.6.1. Проверка кода Python: 24](#_Toc97216767)

[6.6.2. Проверка кода методом Excel: 25](#_Toc97216768)

[6.6.3. Проверка кода в онлайн-калькуляторе: 25](#_Toc97216769)

[6.7. Результаты тестирования седьмого дата сета: 26](#_Toc97216770)

[6.7.1. Проверка кода Python: 26](#_Toc97216771)

[6.7.2. Проверка кода методом Excel: 26](#_Toc97216772)

[6.7.3. Проверка кода в онлайн-калькуляторе: 27](#_Toc97216773)

[7. Заключение 29](#_Toc97216774)

# Постановка задачи (физическая модель)

Имеется сеть железных дорог, на которой расположены 3 пункта отправления вагонов с углем и 4 станции приема через которые уголь попадает на ТЭЦ. Известны затраты на перевозку одного вагона угля от i-ой до j-ой станции. Так же известны объемы угля, которые хранятся в каждом пункте отправления и объемы, которые может вместить в себя каждая из станций приема. Необходимо составить оптимальный план доставки угля из пункта A пункт B с минимальными затратами на перевозку.

Объемы запасов угля в пунктах отправления (в кол-ве вагонов): 300, 250 и 200

Объемы угля, которые могут в себя вместить пункты приема (в кол-ве вагонов): 220, 150, 250 и 180 соответственно.

Стоимость перевозки одного вагона угля с I станции до хранилища j указаны в правых верхних углах соответствующих клеток транспортной матрице.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| заказы  запасы | | *B*1 | | *B*2 | | *B*3 | | *B*4 | |
| 220 | | 150 | | 250 | | 180 | |
| *A*1 | 300 |  | 400 |  | 500 |  | 300 |  | 600 |
|  | |  | |  | |  | |
| *A*2 | 250 |  | 700 |  | 200 |  | 100 |  | 500 |
|  | |  | |  | |  | |
| *A*3 | 200 |  | 600 |  | 100 |  | 400 |  | 200 |
|  | |  | |  | |  | |

Транспортная матрица 1

# Математическая модель

Далее для того, чтобы приступить к дальнейшему решению транспортной задачи, проверим условие на сбалансированность:

Поскольку *ai = 300 + 250 + 200 = 750 ≠* *bj* = 220 + 150 + 250 + 180 = 800, то тип транспортной задачи является открытым, что свидетельствует нам о том, что придется сводить ее к закрытой путем добавления фиктивного дополнительного условия (дополнительный пункт приема или отправления).

Искомые переменные будут располагаться в матрице Xij – объем поставляемого угля, перевозимого с i-ой станции в j-ое хранилище:

x x x x 

x x x x 

x x x x 

Теперь запишем и найдем целевую функцию:

3 4

*W*min  *cij xij*

*i*1 *j*1

При следующих ограничениях:

n

1.  *xij ≤* *bj;*

*j*1

m

1.  *xij =* *ai.*

*i*1

Таким образом, получим оптимальный план перевозок Xij, где целевая функция Wmin – минимальный затраты на перевозку угля, при выполнении всех поставленных условий.

# Алгоритмы

У нас есть 3 алгоритма решения: 1 алгоритм реализован в python, 2 алгоритм реализован в python- визуализация и 3 алгоритм решения с помощью Excel.

## Алгоритм 1

Решение задачи с помощью python. Запускаем программу и выбираем способ ввода исходных данных.

### Описание входных данных

### Для ручного ввода нужно ввести "1",

### Для заполнения матрицы значений случайными числами вводим "2"

### Для ввода значений с помощью файла csv вводим “3”.

Но в данном примере мы рассмотрим только первый способ ввода данных. На рисунке 1 мы видим строчку с «way = input…», которая позволяет нам выбрать тип вводимых данных. В данном случае мы выбрали ручной ввод данных (1 на клавиатуре). С помощью второго абзаца рисунка 1 мы сможем сформировать размер матрицы (кол-во столбцов и строк в ней). В третьей части рисунка (с момента использования функции matrix) мы задали цикл для последовательного ввода значений и заполнения матрицы исходных данных.

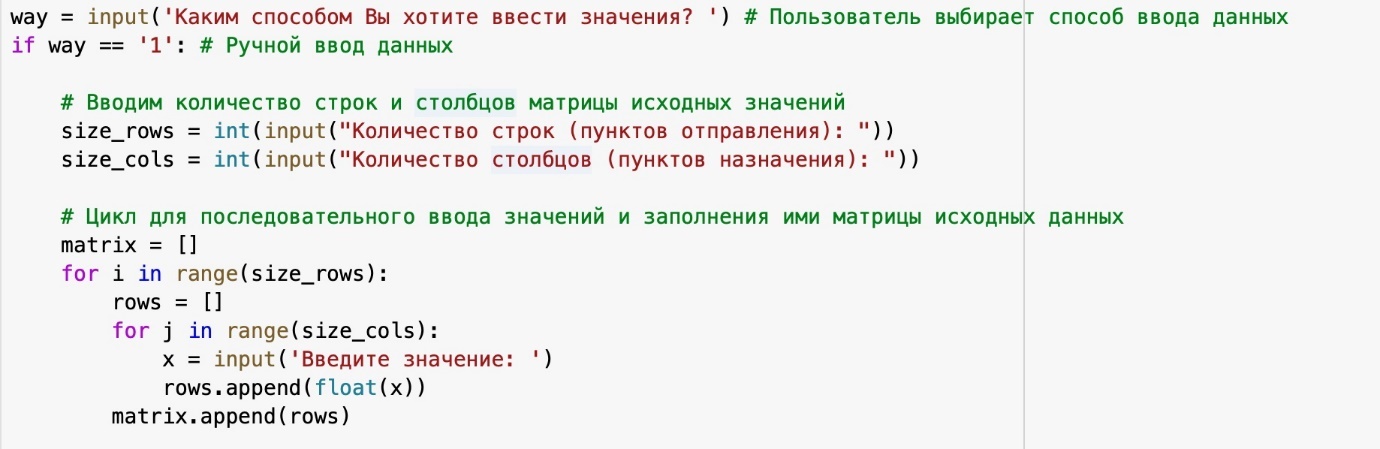


Рисунок 1

Следующим этапом мы вводим в задачу потребности и производительность и обозначаем их как max\_company\_1 и max\_factory\_2. (см. Рисунок 2)

Разбиваем строку потребностей на список и переводим значения в int. То же самое делаем и с производительностью.

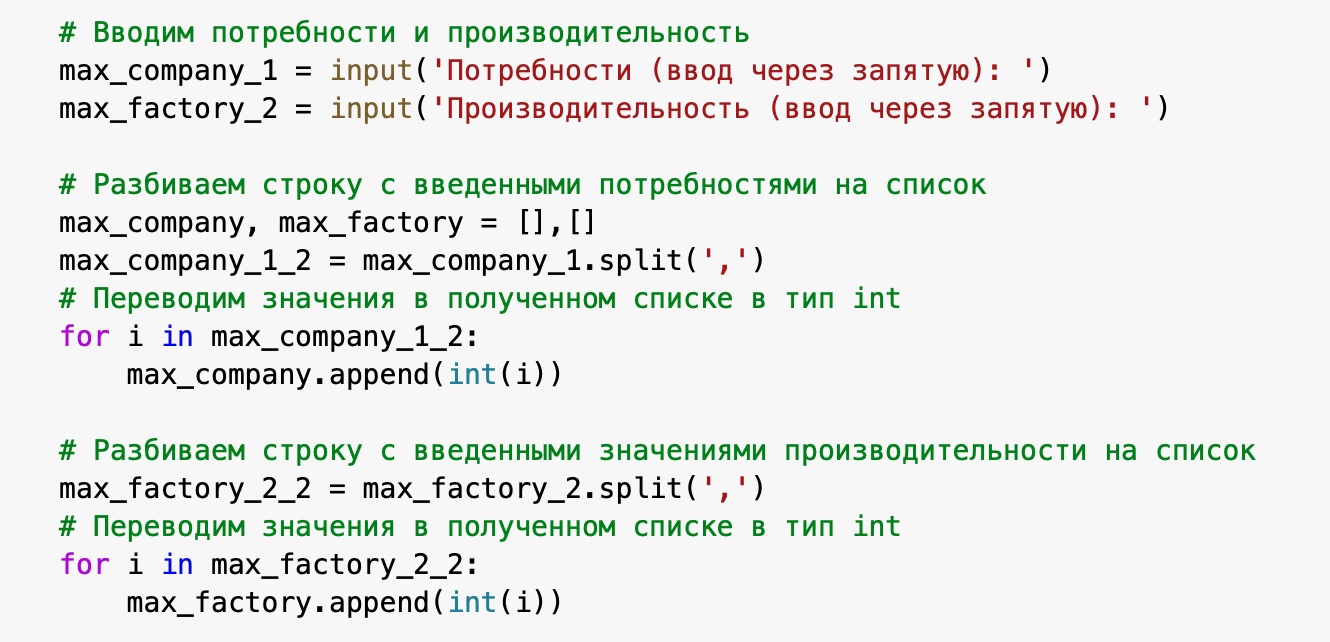
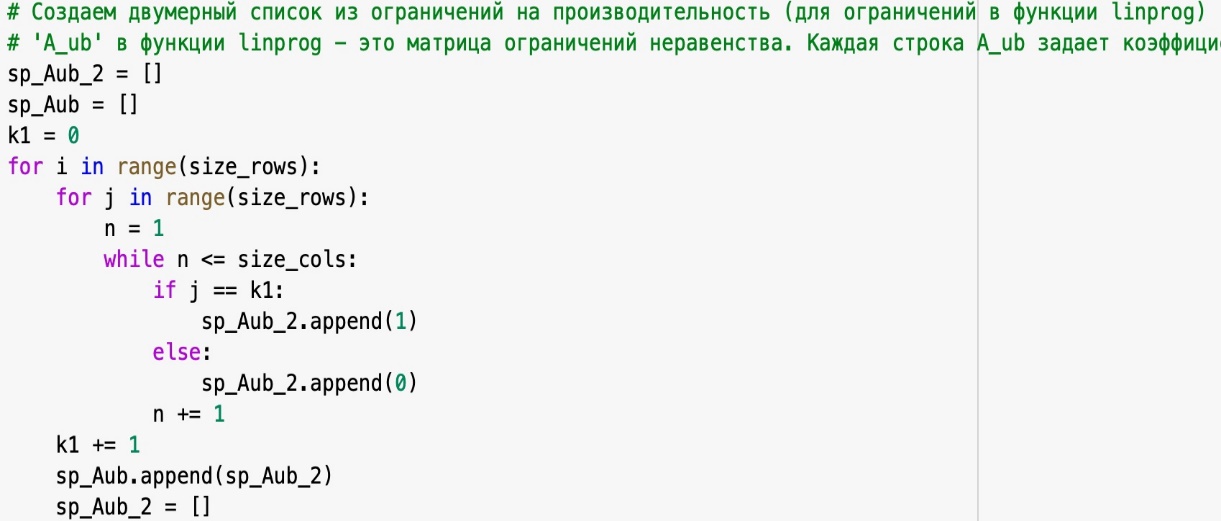
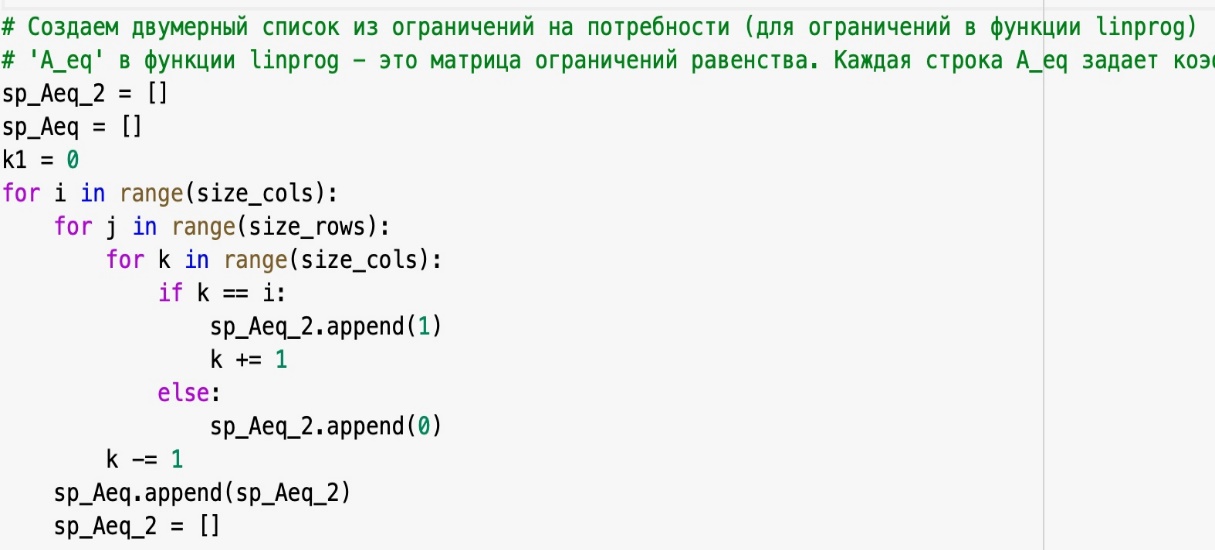


Рисунок 2

Остальные способы ввода информации сделаны аналогично данному, поэтому описывать их не имеет смысла.

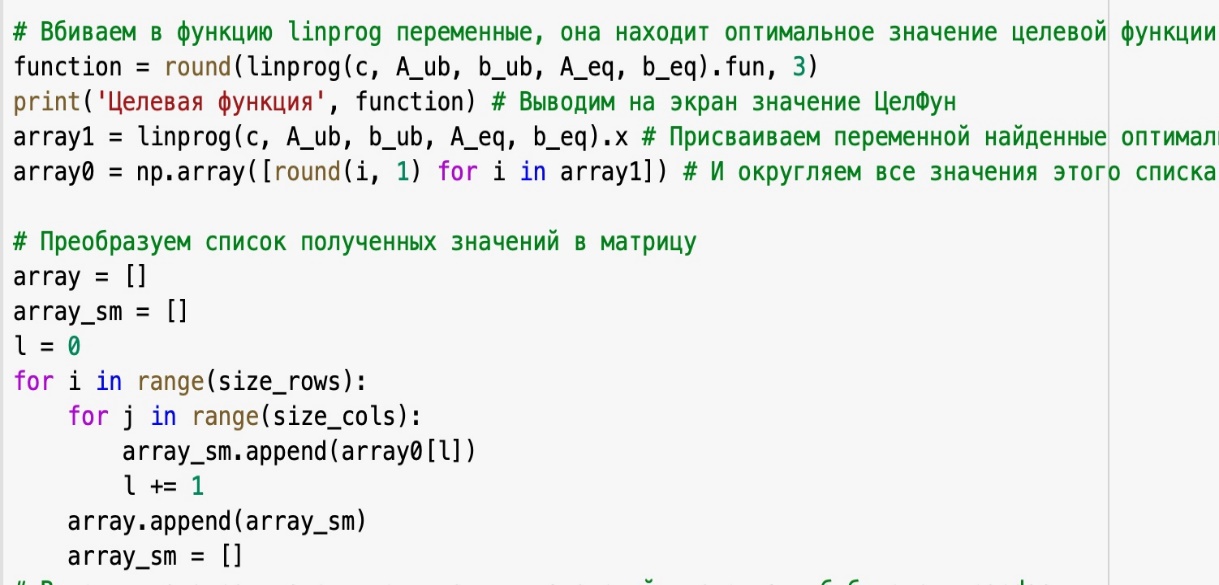
### Описание алгоритма решения

В разделе описывается последовательность действий, совершаемых алгоритмом для получения целевой функции и искомую матрицу.



Скриншот 1 и 2

На скриншотах выше показан скрипт действий по созданию матрицы с ограничениями на производительность и потребность для передачи данных в функцию linprog.

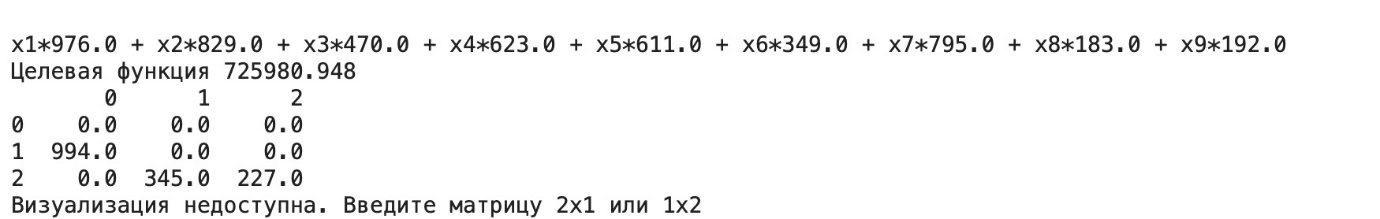


Скриншот 3

С помощью функции linprog мы вычислили целевую функцию, которую мы перевели в искомую матрицу.

### Описание выходных данных

Пользователь на выходе получает следующую информацию: целевая функция в обобщённом виде и искомая матрица.



Скриншот 4

## Алгоритм 2

Второй алгоритм — это алгоритм визуализации и начало кода полностью идентично первому алгоритму.

### Описание входных данных

### Для ручного ввода нужно ввести "1",

### Для заполнения матрицы значений случайными числами вводим "2"

### Для ввода значений с помощью файла csv вводим “3”.

Но в данном примере мы рассмотрим только первый способ ввода данных. На рисунке 1 мы видим строчку с «way = input…», которая позволяет нам выбрать тип вводимых данных. В данном случае мы выбрали ручной ввод данных (1 на клавиатуре). С помощью второго абзаца рисунка 1 мы сможешь сформировать размер матрицы (кол-во столбцов и строк в ней). В третьей части рисунка (с момента использования функции matrix) мы задали цикл для последовательного ввода значений и заполнения матрицы исходных данных.

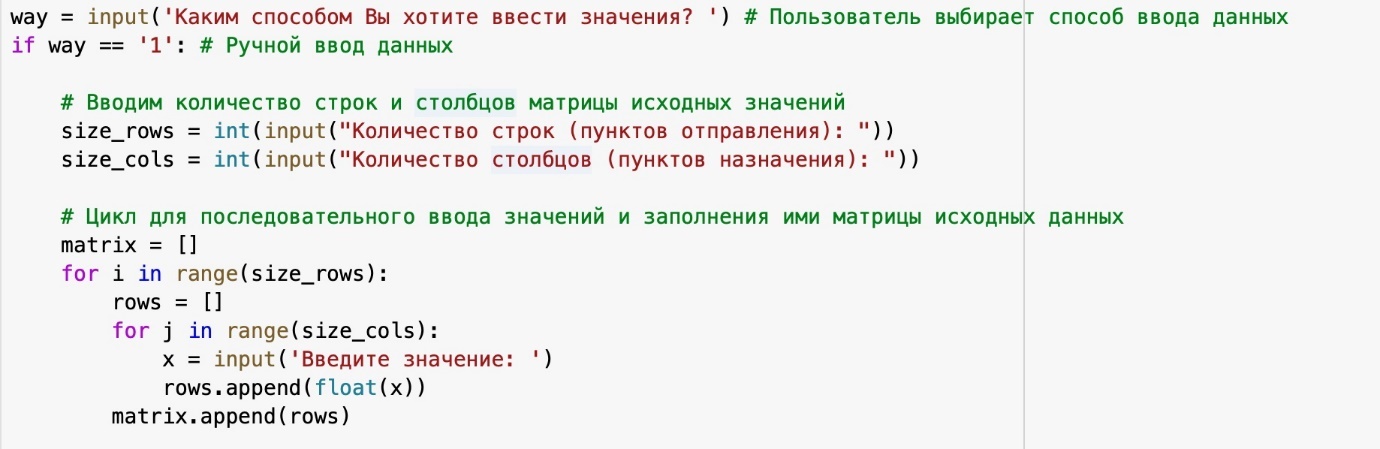


Рисунок 3

Следующим этапом мы вводим в задачу потребности и производительность и обозначаем их как max\_company\_1 и max\_factory\_2. (см. Рисунок 2)

Разбиваем строку потребностей на список и переводим значения в int. То же самое делаем и с производительностью.

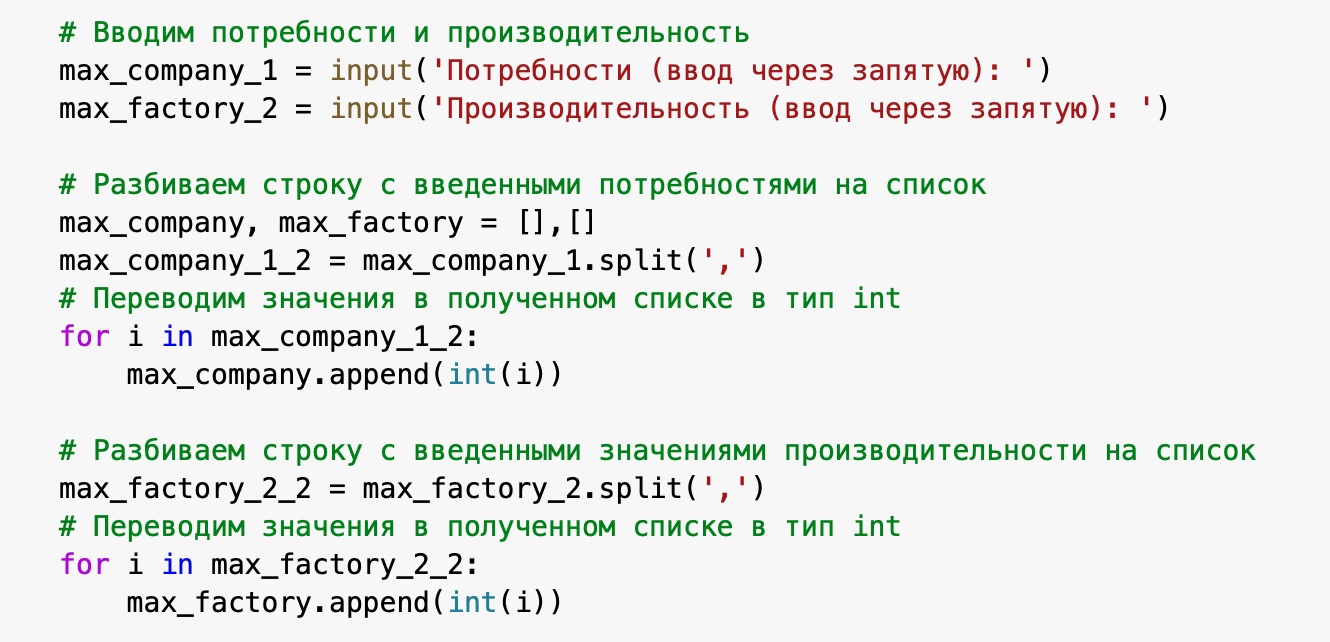
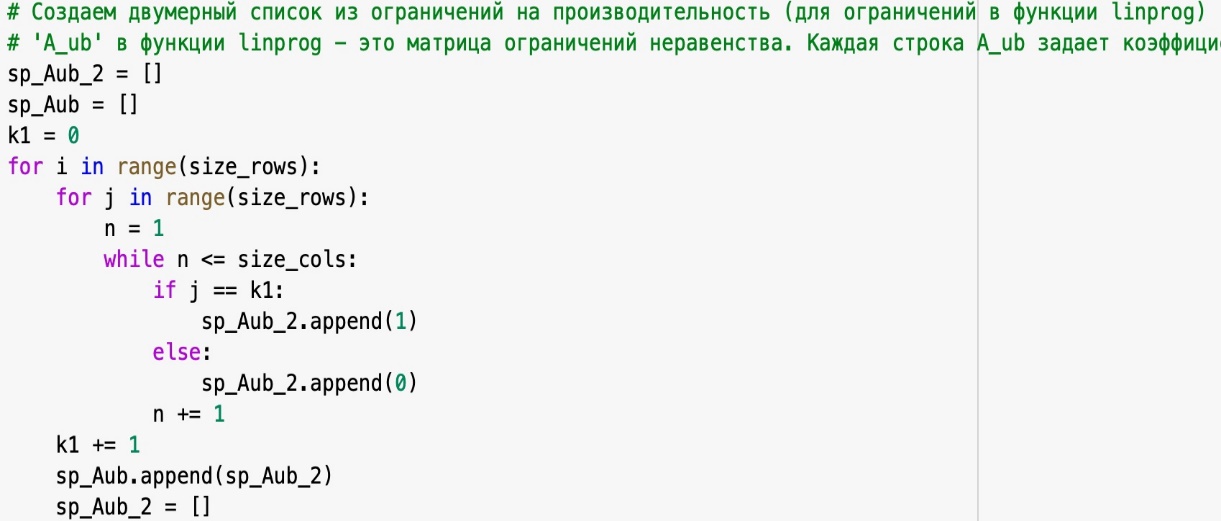
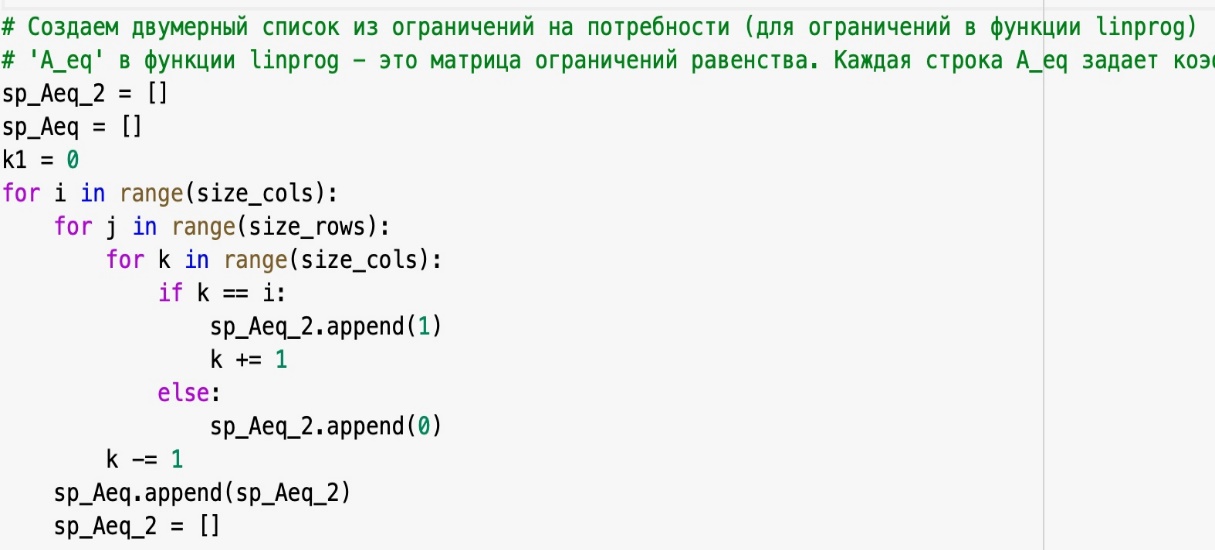


Рисунок 4

Остальные способы ввода информации сделаны аналогично данному, поэтому описывать их не имеет смысла.

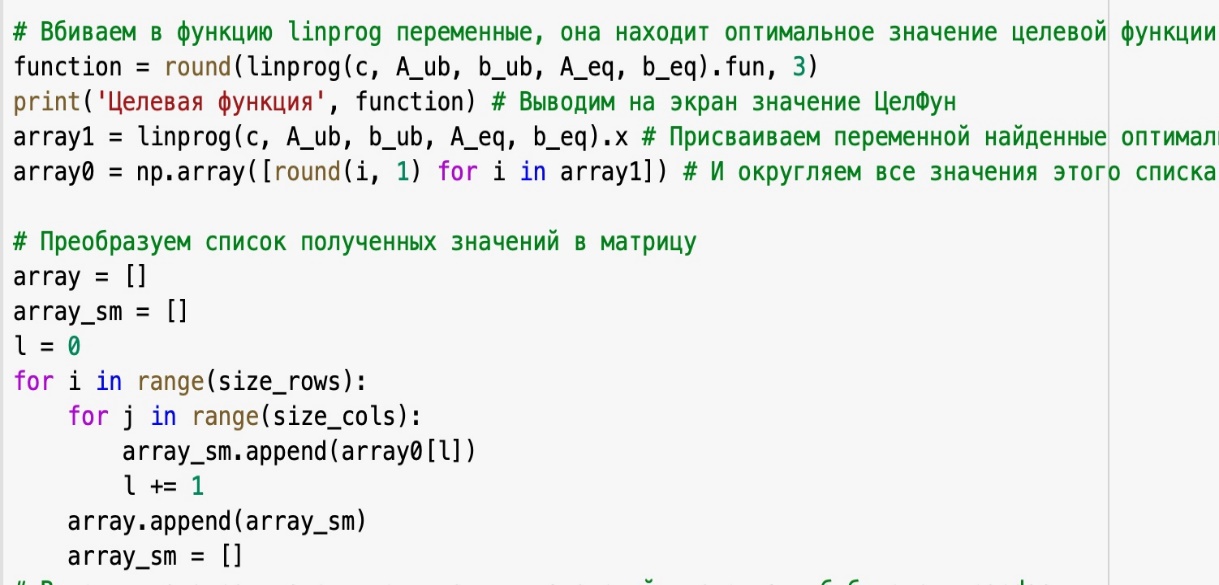
### Описание алгоритма решения

В разделе описывается последовательность действий, совершаемых алгоритмом для получения целевой функции и искомую матрицу.



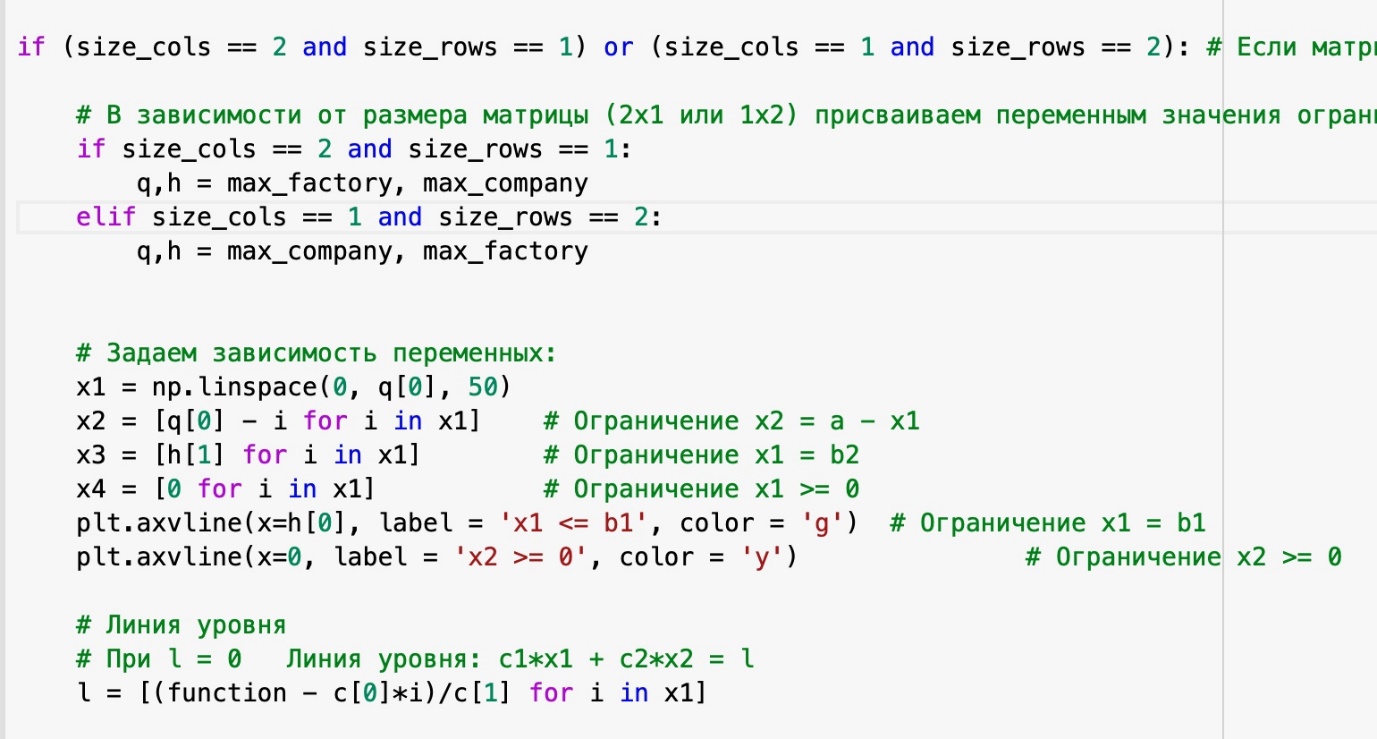
Скриншот 1 и 2

На скриншотах выше показан скрипт действий по созданию матрицы с ограничениями на производительность и потребность для передачи данных в функцию linprog.



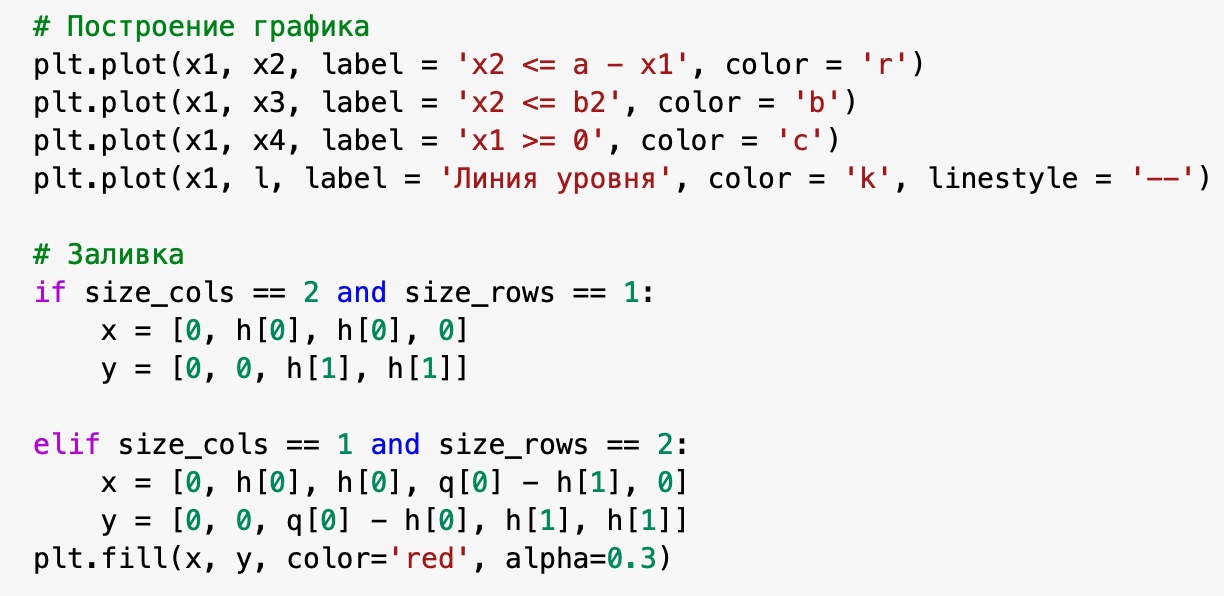
Скриншот 3

С помощью функции linprog мы вычислили целевую функцию, которую мы перевели в искомую матрицу. Изначально в условии мы прописываем, что в матрице не может быть более 2-х переменных. То есть наша матрица не может быть более чем 2х1 или 1х2.Мы вносим на график линию уровня и зависимость переменных. (см. скриншот 4)



Скриншот 4

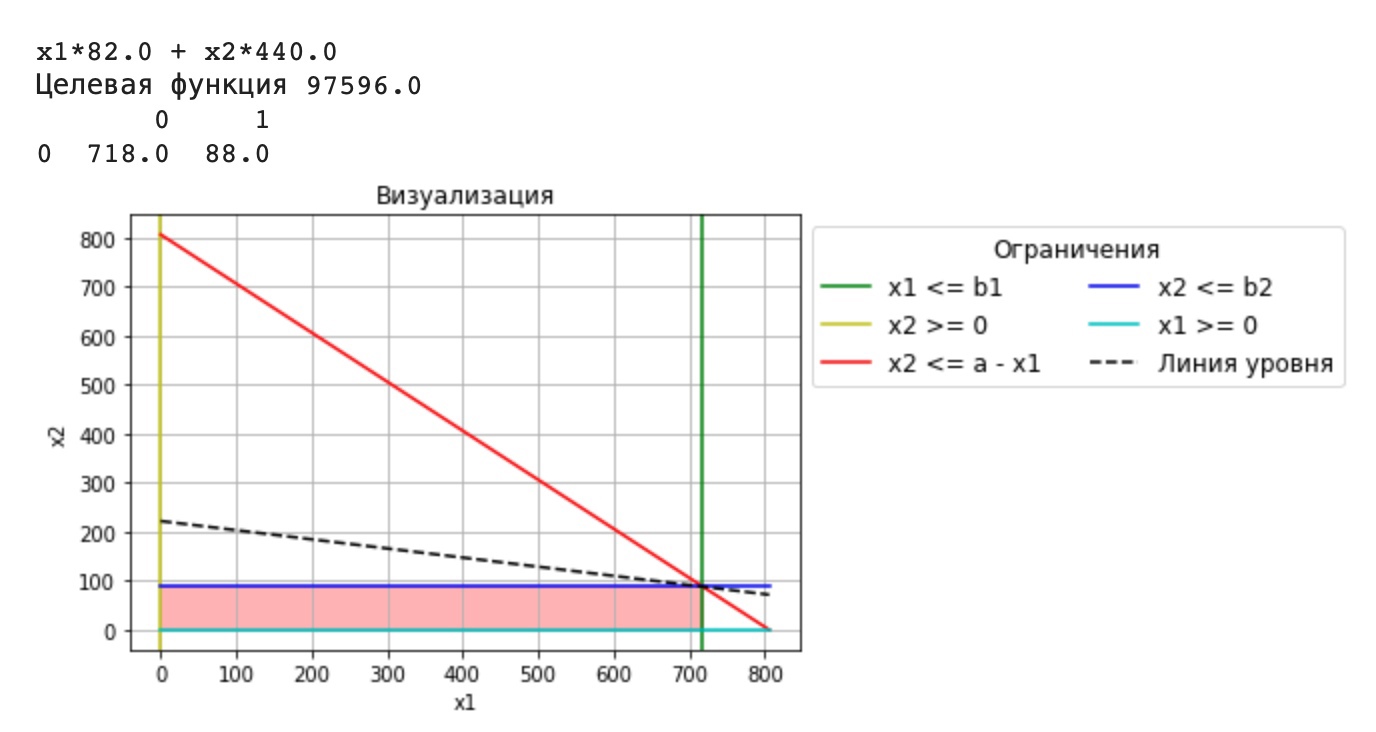
С помощью функции plt.plot мы строим график и производим заливку интересующей нас области для наглядности. (См. скриншот 5)



Скриншот 5

### Описание выходных данных

На выходе мы получаем график с целевой функцией, с ограничениями и линией уровня, значением целевой функции и искомой матрицей.



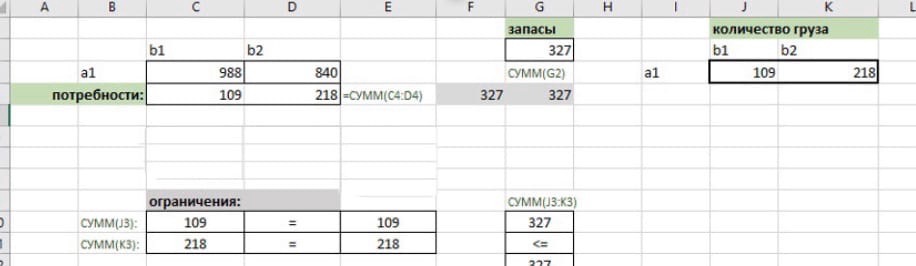
Визуализация 1

## Алгоритм 3

Решение в экселе.

### Описание входных данных

В соответствии с требованиями и условиями, которые предоставил нам заказчик составляем матрицу (таблицу) в экселе со стоимостями, соответствующими каждому заводу/складу. Далее вводим данные о потребностях/запасах и производительности. Сверяем суммы всех запасов с суммами всех производительностей и в зависимости от схождений значений понимаем какой тип задачи (открытый или закрытый).



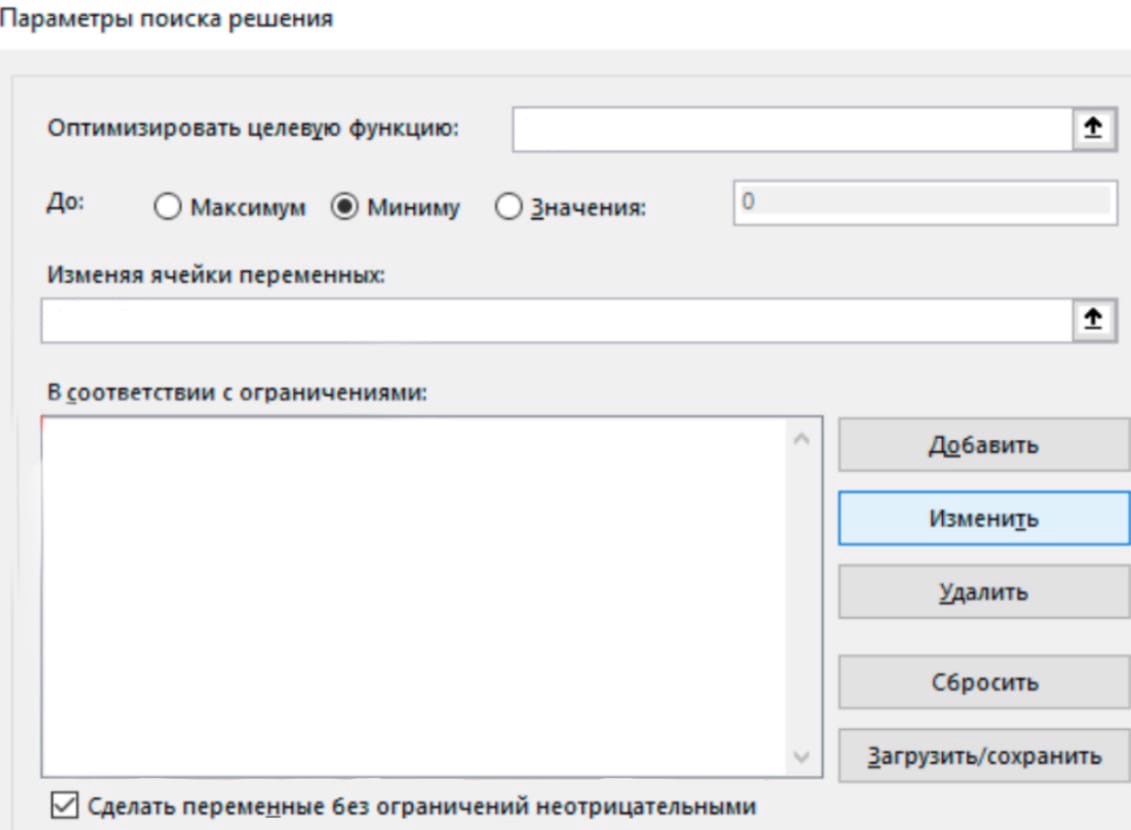
эксель 1

### Описание алгоритма решения

* Составляем искомую матрицу, которая будет заполнена количеством отправленного груза из соответствующего завода в соответствующий склад.
* Создаем целевую функцию – формула суммпроизв (ячейки стоимостей, ячейки искомой матрицы с кол-ом груза)
* Прописываем ограничения, основываясь на искомую матрицу. Общее количество груза соответствующего завода не должно превышать его производительность, а общее кол-во груза, отправленного в соответствующую мастерскую, должно равняться её потребности.
* Через инструмент «Поиск решений» заполняем ячейку с целевой функцией, изменяя значения в искомой матрице с количеством груза, чтобы искомая стремилась к минимуму.
* Добавляем 2 ограничения, прописанных ранее и выполняем поиск решения симплекс методом. Изображение выглядит как стол

  Автоматически созданное описание

эксель 2



Поиск решения 1

### Описание выходных данных

На выходе получаем в целевой функции стоимость в рублях, в искомой матрице количество груза в штуках.

Изображение выглядит как стол

Автоматически созданное описание

эксель 3

В конечном итоге мы имеем целевую функцию и исходную матрицу, которую вы можете увидеть на скриншоте 8.

# Варианты использования системы

В нашей системе есть три варианта использования.

## ВИ 1

Данный вариант использования включает в себя ручной ввод данных с клавиатуры. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «1».

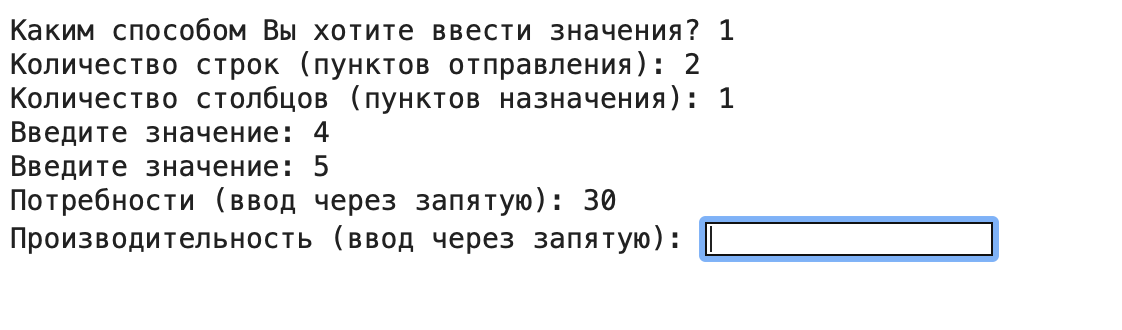


Рисунок 5

В графах «количество строк» и «количество столбцов» указывается общее количество пунктов отправления и пунктов назначения.

В графах «потребности» и «производительность» данные вводятся через запятую. Сверху вниз для графы «Производительность», слева направо для графы «Потребность».

Далее последуют области для заполнения нужных данных, стоит учитывать, что в графу «введите значения» вводятся данные из матрицы обязательно построчно слева направо.

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальное решение.

## ВИ 2

Данный вариант использования включает в себя ввод случайных данных. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «2».

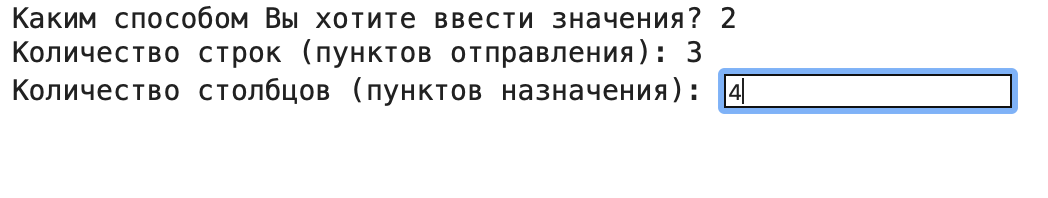


Рисунок 6

В графах «количество строк» и «количество столбцов» указывается общее количество пунктов отправления и пунктов назначения.

Далее последуют области для заполнения нужных данных, стоит учитывать, что в графу «введите значения» вводятся данные из матрицы обязательно построчно слева направо.

После этого от пользователя требуется лишь нажатие клавиши «Enter» и на экране выведется оптимальное решение.

## ВИ 3

Данный вариант использования включает в себя ввод данных с помощью файла csv. Для того, чтобы его активировать в графу «Каким способом вы хотите ввести значения?» надо ввести цифру «3».

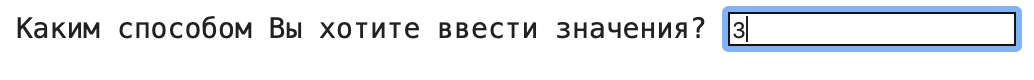
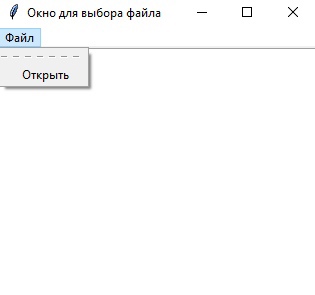


Рисунок 7

После активации данного варианта использования пользователю откроется окно. В зависимости от вашего программного обеспечения физический вид окна может отличаться. При взаимодействии через Windows с левой стороны окна будет активная кнопка «файл», на которую пользователь должен нажать, а далее выбрать нужный файл.



CSV 1

# Архитектура решения

В данном разделе будут описаны 3 созданных метода описываются для решения задачи.

## Методы считывания информации

Здесь будут описываться все методы, которые отвечают за получение программой информации, в нашей программе существует три метода ввода информации, а именно вручную, через автоматическое заполнение данных случайными числами, а также с помощью прикрепления файла csv.

Для каждого метода (функции) необходимо указать следующую информацию:

* **Первым алгоритмом** является ввод данных вручную. Такой метод принимает следующие входные данные: количество строк, количество столбцов, матричные значения, потребности и производительность. Выходными данными являются данные матричных значений, потребности и производительности. Затрагиваемые в ходе работы переменные: size\_rows, size\_cols, matrix\_t, rows, size\_cols, max\_company\_1, max\_factory\_2, max\_company\_1\_2, max\_factory\_2\_2, max\_company, max\_factory.
* **Второй метод** считывания информации – это заполнение данных случайными значениями. Входные данные представляют собой количество строк и столбцов, зато выходными данными являются данные матричных значений, потребности и производительность. Затрагиваемые в ходе работы переменные: size\_rows, size\_cols, matrix\_t, rows, x, max\_company, max\_factory.
* **Третий метод** представляет собой считывание данных с файла формата csv. Входными данными являются данные файла, которые в последствии обрабатываются. Выходными данными являются данные матричных значений, потребности и производительность. Затрагиваемые в ходе работы переменные: size\_rows, size\_cols, matrix\_t, rows, max\_company, max\_factory, max\_company\_1\_2.

## Методы обработки информации

В данном разделе описываются все методы обработки введенной пользователем информации и получение решения.

Первичные методы обработки информации зависят от того, каким способом водятся данные.

1. Для ввода данных соответствуют следующие блоки:

Первый блок - цикл для последовательного ввода значений и заполнения ими матрицы исходных данных. Входными данными являются количество строк, количество столбцов и матричные значения. Выходными данными является матрица. Затрагиваемые переменные: size\_rows, size\_cols, matrix\_t, rows, x.

Второй блок – преобразование данных для дальнейших операций. Входные данные представляют собой потребность и производительность. Выходными данными являются числовые значения потребности и производительности. Затрагиваемые переменные: max\_company, max\_factory, max\_company\_1\_2.

1. Для получения данных с помощью генератора случайных чисел не выполняется никаких особенных действий на первоначальном этапе по обработке информации.
2. Для получения данных с помощью файла формата csv соответствует следующая функция – считывание данных. Входными данными являются данные файла. Выходными данными являются преобразованные данные потребностей, производительности и матрицы. Затрагиваемыми переменными являются: self, filename, row, max\_company, max\_factory, max\_company\_1\_2.

Далее все сводится к единому алгоритму, который состоит из следующего метода:

Расчет целевой функции и оптимальной матрицы. Входными данными является матрица. Выходными данными значение целевой функции, матрица оптимальных значений. Затрагиваемые переменные: matrix, function\_text, sp\_Aub, sp\_Aub\_2, size\_rows, size\_cols, sp\_Aeq, sp\_Aeq\_2, b\_ub, A\_ub, function, array, array\_sm, array1, array0.

## Методы вывода информации

В разделе описываются все методы(функции) отвечающие за вывод информации пользователю, такие как график и конечные значения.

1. Первый метод – это вывод значений, которые обрабатываются для решения задачи. Входными параметрами являются полученные при обработке информации данные, такие как целевая функция, формула, по которой считается целевая функция, а также матрица. Выходными данными также является формула целевой функции, целевая функция и матрица. Затрагиваемые переменные в данном методе: function, function\_text, array.
2. Второй метод – это визуализация данных. Входными данными являются ограничения, линия уровня, целевая функция. Выходными данными у нас будет график, в котором отображается ограничения, линия уровня. Затрагиваемые переменные: x1, x2, x3, x4, l, x, y.

# Тестирование

Таблица 1. Матрицы 1х2 и 2х1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Входные данные** | **Python** | **Визуализация** | **Excel** | **Онлайн-калькулятор** |
| Потреб: 970, 408  Произв: 1378 | 577870 | Рисунок 1 | 577870 | 577870 |
| Потреб: 109, 218  Произв: 327 | 290812 | Рисунок 2 | 290812 | 290812 |
| Потреб: 437, 150  Произв: 587 | 280755 | Рисунок 3 | 280755 | 280755 |
| Потреб: 474  Произв: 248, 387 | 175278 | Рисунок 4 | 175278 | 175278 |
| Потреб: 521  Произв: 301, 368 | 135102 | Рисунок 5 | 135102 | 135102 |

## 6.1. Результаты тестирования первого дата сета:

### 6.1.1. Проверка кода Python и визуализации:

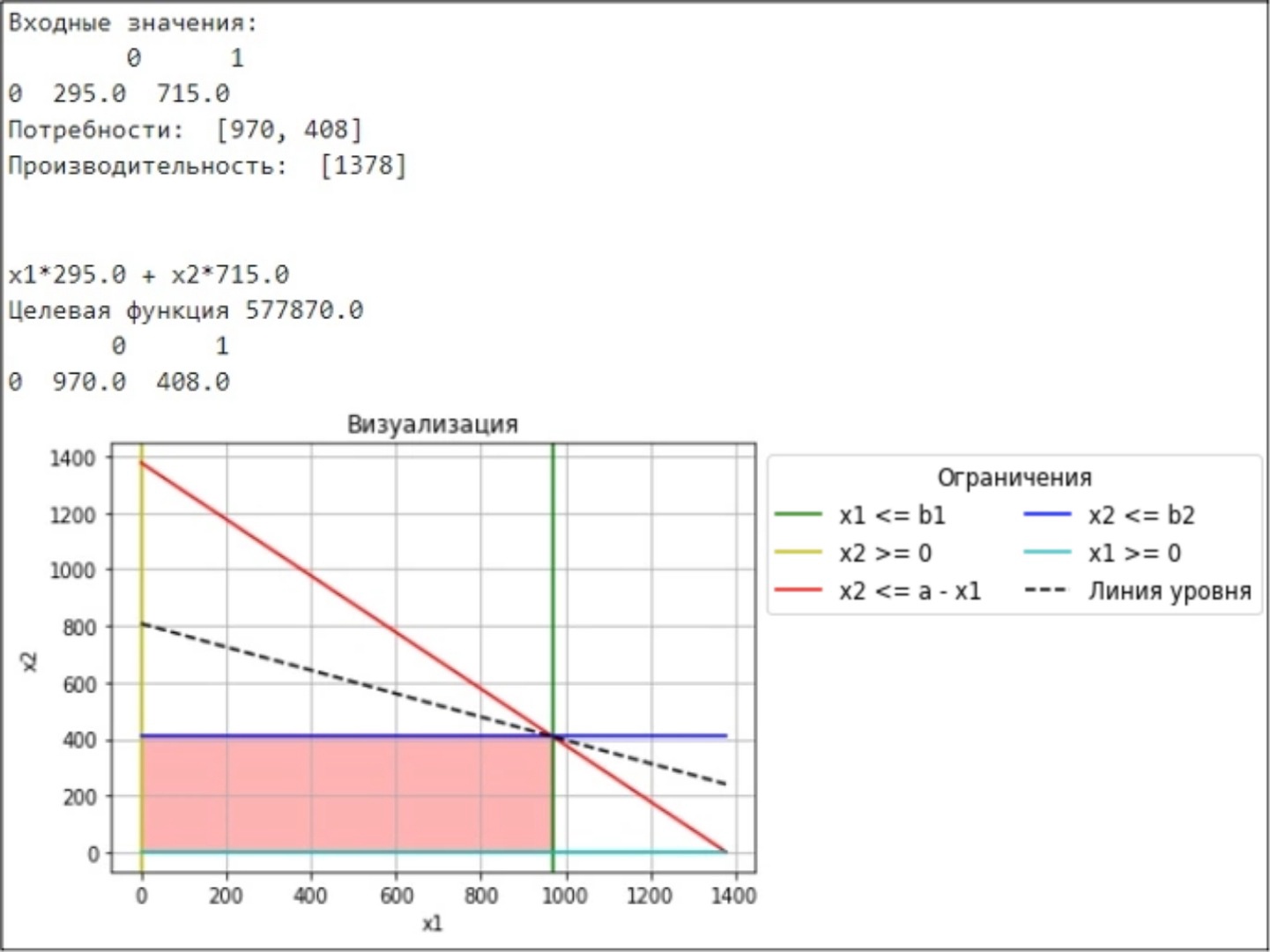


Рисунок 8

### 6.1.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 4

### 6.1.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 1

## 6.2. Результаты тестирования второго дата сета:

### 6.2.1. Проверка кода Python и визуализация:

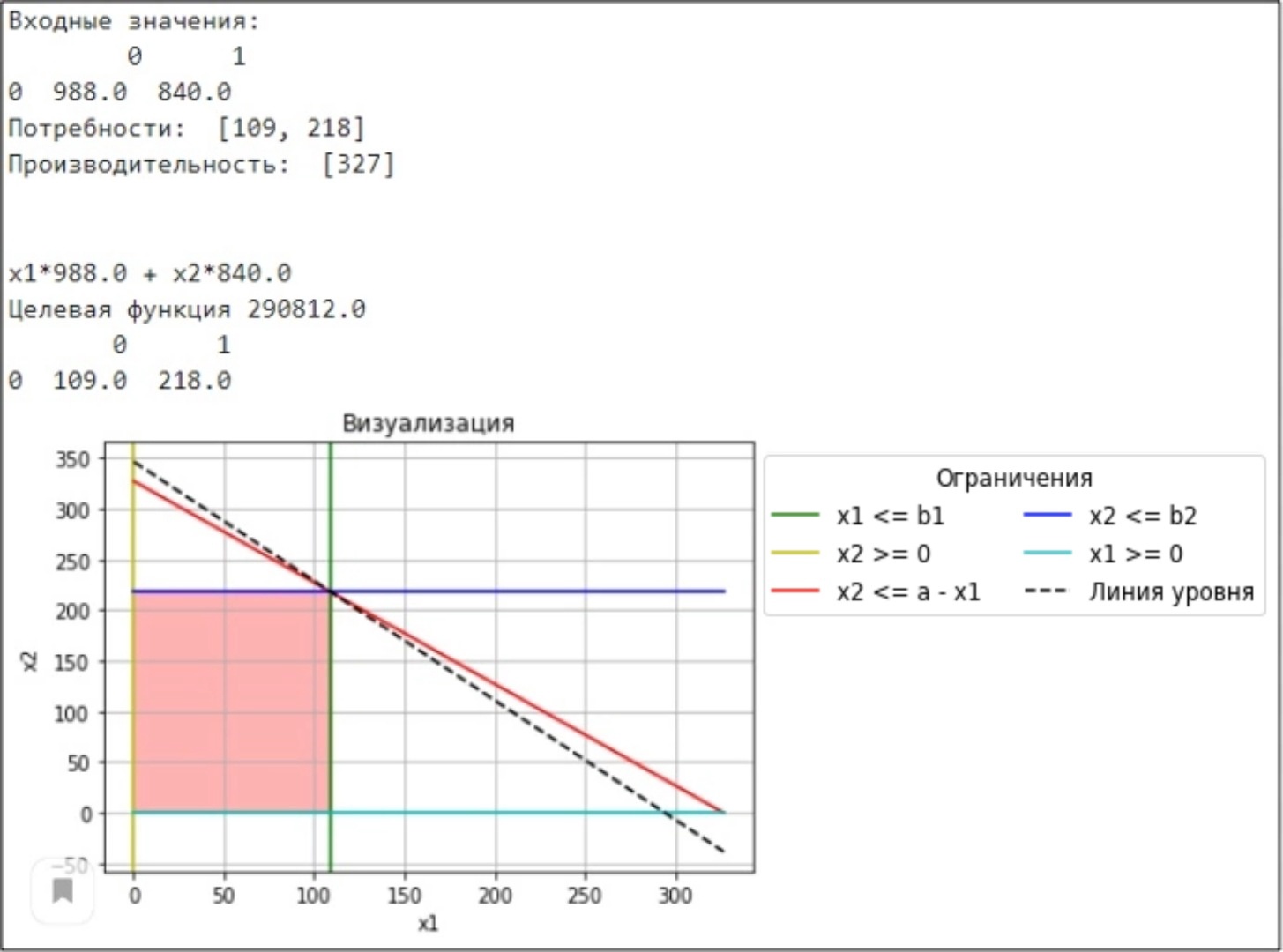


Рисунок 9

### 6.2.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 5

### 6.2.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 2

## 6.3. Результаты тестирования третьего дата сета:

### 6.3.1. Проверка кода Python и визуализация:

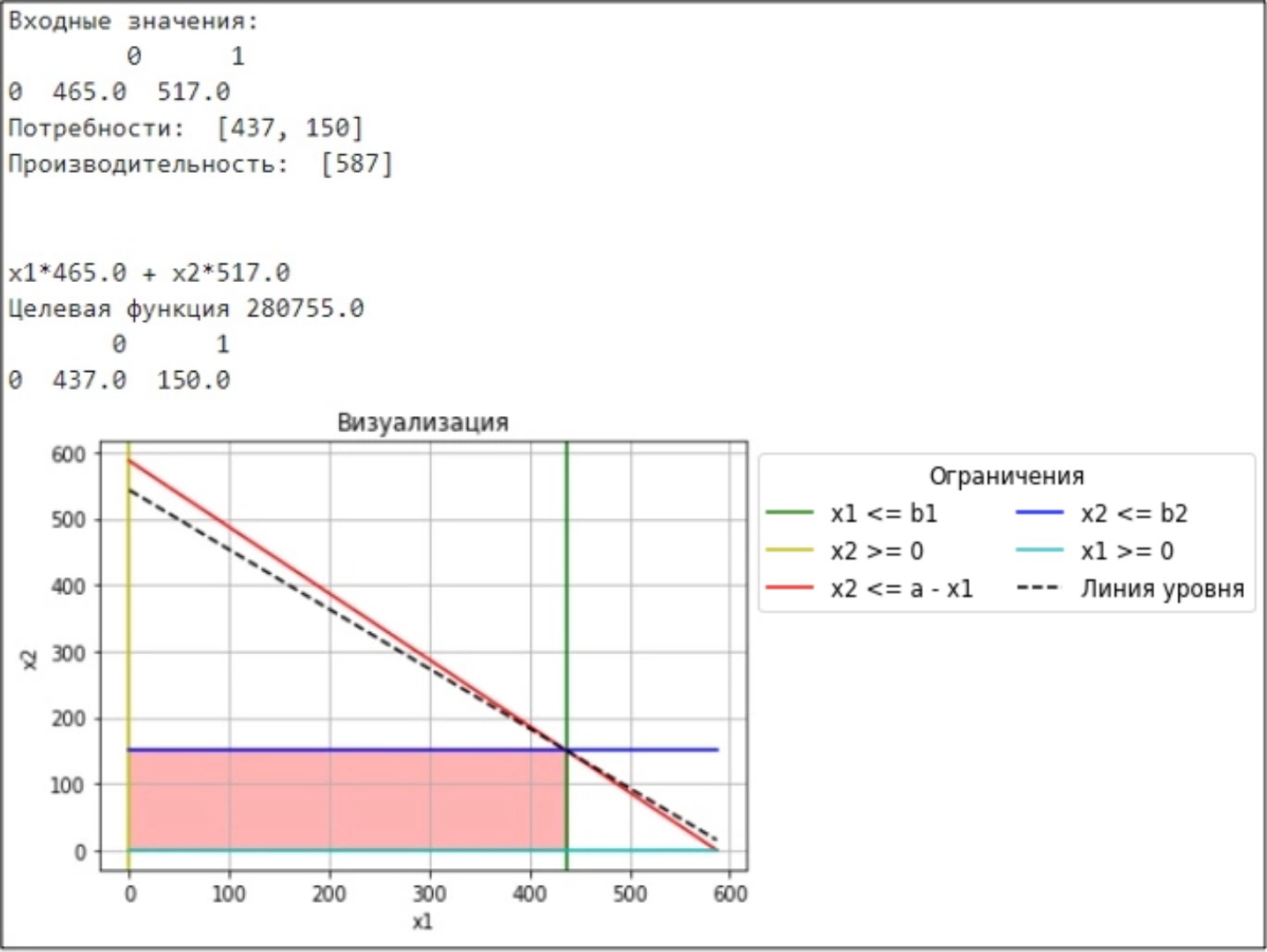


Рисунок 10

### 6.3.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 6

### 6.3.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 3

## 6.4. Результаты тестирования четвёртого дата сета:

### 6.4.1. Проверка кода Python и визуализации:

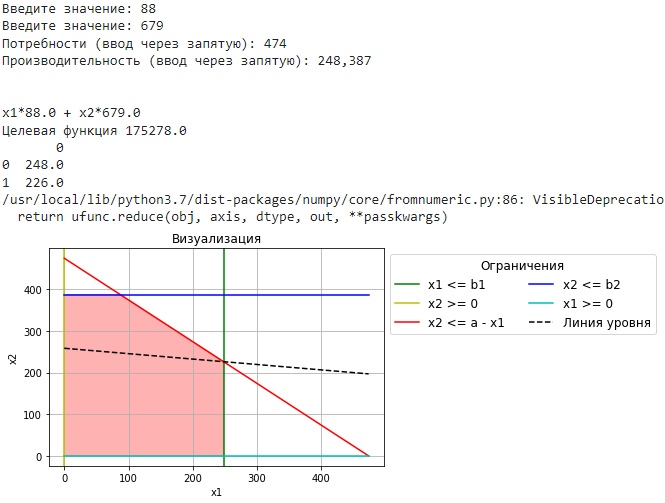


Рисунок 11

### 6.4.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 7

### 6.4.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 4

## 6.5. Результаты тестирования пятого дата сета:

### 6.5.1. Проверка кода Python и визуализации:

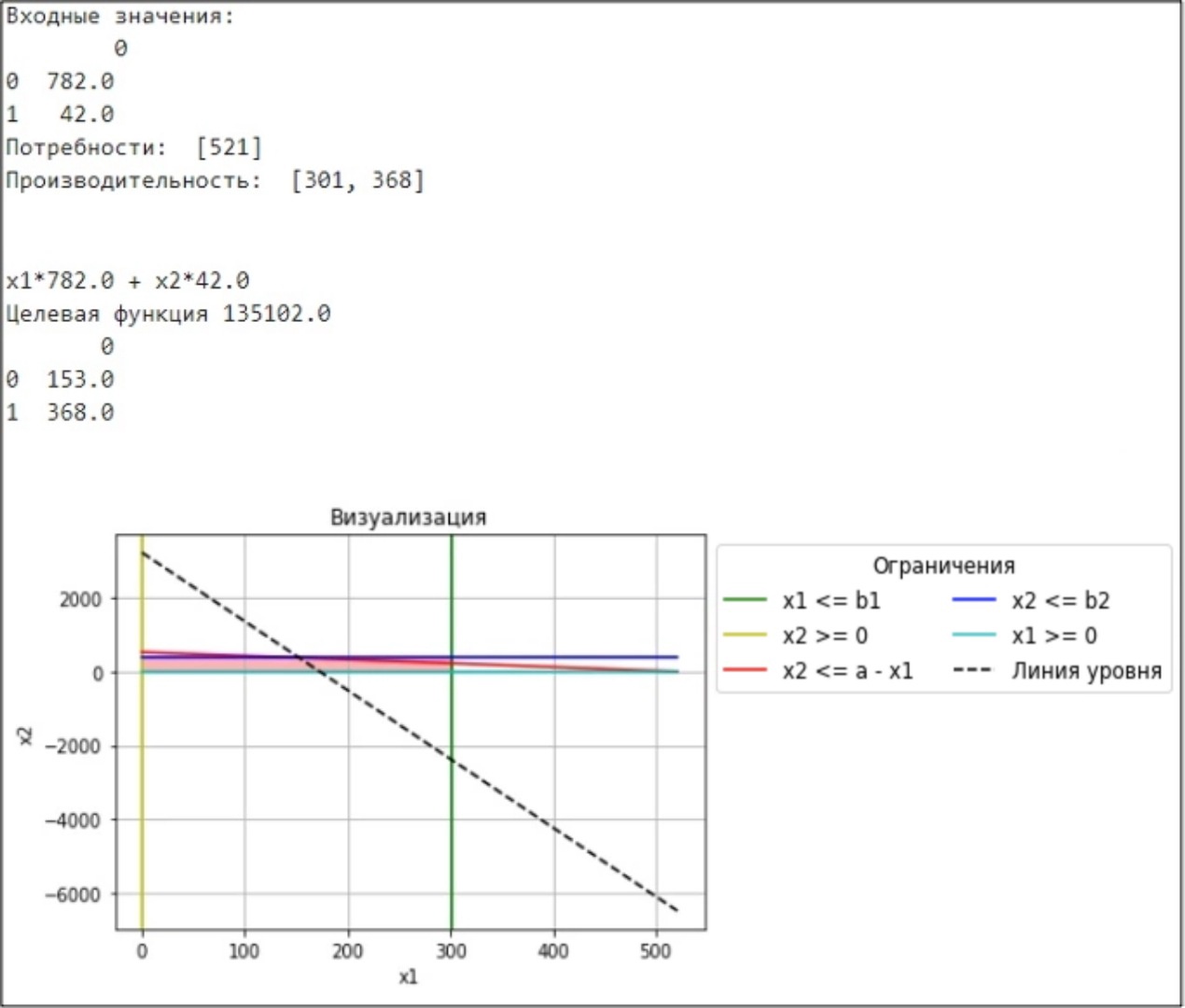


Рисунок 12

### 6.5.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 8

### 6.5.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 5

Таблица 2. Матрицы 3х4 и 4х3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Входные данные** | **Python** | **Матрица и целевая функция** | **Excel** | **Онлайн-калькулятор** |
| Потреб: 197, 1000, 309  Произв: 914, 1044, 756, 323 | 132739 | Рисунок 6 | 132739 | 132739 |
| Потреб: 654, 591, 822, 579  Произв: 122, 2234, 2359 | 751420 | Рисунок 7 | 751420 | 751420 |

## 6.6. Результаты тестирования шестого дата сета:

### 6.6.1. Проверка кода Python:

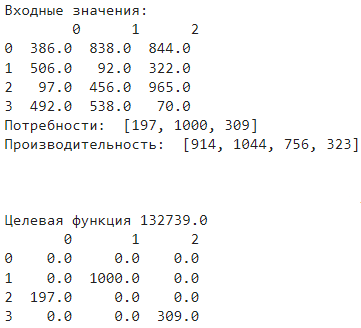


Рисунок 13

### 6.6.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 9

### 6.6.3. Проверка кода в онлайн-калькуляторе:

Изображение выглядит как текст

Автоматически созданное описание

онлайн калькулятор 6

## 6.7. Результаты тестирования седьмого дата сета:

### 6.7.1. Проверка кода Python:

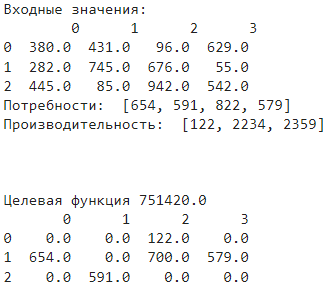


Рисунок 14

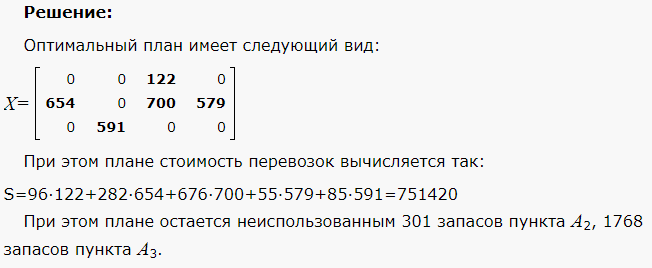
### 6.7.2. Проверка кода методом Excel:

Изображение выглядит как стол

Автоматически созданное описание

эксель 10

### 6.7.3. Проверка кода в онлайн-калькуляторе:



онлайн калькулятор 7

Для подсчёта времени на выполнение алгоритмов будет использовано время, затраченное на ввод необходимых данных в предназначенные поля и расчёт данных самой программой с дальнейшим выводом их пользователю на экран.

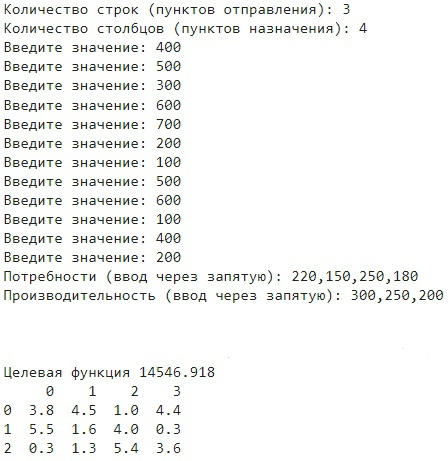
Таблица 3. Время выполнения алгоритмов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Входные данные** | **Python** | **Матрица и целевая функция** | **Excel** | **Онлайн-калькулятор** |
| Дата сет №1 | ~10-30 секунд | ~10-30 секунд | ~1-1,5 минуты | ~12 секунд |
| Дата сет №2 | ~10-30 секунд | ~10-30 секунд | ~1-1,5 минуты | ~20–40 секунд |
| Дата сет №3 | ~10-30 секунд | ~10-30 секунд | ~1-1,5 минуты | ~20–40 секунд |
| Дата сет №4 | ~10-30 секунд | ~10-30 секунд | ~1-1,5 минуты | ~20–40 секунд |
| Дата сет №5 | ~10-30 секунд | ~10-30 секунд | ~1-1,5 минуты | ~20–40 секунд |
| Дата сет №6 | ~10-30 секунд | ~10-30 секунд | ~1,5-2 минут | ~30–50 секунд |
| Дата сет №7 | ~10-40 секунд | ~10-40 секунд | ~1,5-2 минут | ~30–50 секунд |

Из этого следует, что предпочтительным алгоритмом является использование Python и онлайн-калькулятора в связи с их точностью и быстротой выполняемых команд. Погрешности в выводим данных между разными алгоритмами не выявлено.

# Заключение

Наш предпочтительный алгоритм, а именно программный код в Питоне, решает поставленную задачу. На основании тестирования данного алгоритма, можно сделать вывод о том, что Python выводит самое оптимальное решение достаточно быстро. Ниже представлено решение, поставленной задачи:



Значения 1

Теперь сравним несколько алгоритмов по критериям: эффективности, скорости использования алгоритма, простоты использования, надежность в разрезе человеческого фактора и точность предоставленного решения.

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Python | Excel | Визуализация |
| эффективность | высокая | высокая | средняя |
| скорость использования алгоритма | высокая | низкая | высокая |
| простота использования | высокая | низкая | высокая |
| надежность (человеческий фактор) | высокая | низкая | высокая |
| точность | высокая | высокая | высокая |

Для заказчика самым оптимальным вариантом будет первый алгоритм — это программа в Python. Поскольку исходя из подобранных критериев, она удовлетворяет всем требования. Такой алгоритм прост в использование, работа с программой не отнимет много времени. Более того, программа сведена к минимуму от воздействия человеческого фактора - то есть ошибок при решении, все это делает компьютер. Ну и конечно же самое главное — это точность ответа и эффективность данного алгоритма.

Одной из перспектив развития этого алгоритма - является усовершенствование его, для решения так же производственной задачи. Также алгоритм может быть усовершенствован с точки зрения наглядности, например, выдавать пошаговое решение.

Более того, разработанный нами код поможет заказчику минимизировать затраты при любых транспортировках, будь то внешние или внутренние перевозки.