ГУАП

КАФЕДРА № 43

ОТЧЁТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| д-р техн. наук, доцент |  |  |  | С. И. Колесникова |
| должность, уч. Степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ОТЧЁТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 | | | | | |
| проблема многокритериального выбора. маи и маи+ | | | | | |
| по дисциплине: СИСТЕМНЫЙ АНАЛИЗ | | | | | |
|  | | | | | |
| РАБОТУ ВЫПОЛНИЛ | | | | | |
| СТУДЕНТ ГР. | 4330М |  |  |  | А.А. Кинько |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

**Цель работы**

Целью работы является освоение средств моделирования текстовых (содержательных) слабоструктурированных задач многокритериального выбора.

**Текст задания**

Согласно варианту №8, разработать программу, моделирующую алгоритм поиска «лучшего» решения слабоструктурированной задачи решения проблемы выбора поставщика продукции:

Были отобраны пять альтернатив (варианты поставщиков): , ИП Калинин – посредник, город Екатеринбург; ООО «Сильва» оптовый посредник, город Нижний Тагил; ООО «Эксперт» - посредник, город Тюмень; ИП Малинин – посредник, город Новосибирск; ООО «ALLO» - оптовый посредник, город Тобольск, соответственно.

Каждая альтернатива (поставщик) оценивается по совокупности критериев:  
 C1. Цена  
 С2. Партионность и скидки  
 С3. Надежность исполнителя (репутация)  
 С4. Расстояние от склада поставщика до склада предприятия  
 С5. Транспортные расходы (для оптового посредника ниже, в 2 раза)  
 С6. Сроки поставки  
 С7. Место расположения поставщика

**Ход работы**

Оценим важность критериев с помощью шкалы относительной важности, где 1 – равная важность сравниваемых элементов иерархии; 9 – очень значительное превосходство элемента. Внесем данные в таблицу важности критериев.

*Таблица 1. Оценки важности критериев*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | C1 | C2 | C3 | C4 | C5 | C6 | C7 |
| С1 | 1 | 1/3 | 1/2 | 2 | 1/2 | 1/3 | 4/3 |
| С2 | 3 | 1 | 3/2 | 2 | 3/2 | 1/2 | 2 |
| С3 | 2 | 2/3 | 1 | 3 | 2/3 | 1/2 | 2/3 |
| С4 | 1/2 | 1/2 | 1/3 | 1 | 2/3 | 2/3 | 3/2 |
| С5 | 2 | 2/3 | 3/2 | 3/2 | 1 | 2/3 | 3/2 |
| С6 | 3 | 2 | 2 | 3/2 | 3/2 | 1 | 2/3 |
| С7 | 3/4 | 1/2 | 3/2 | 3/2 | 2/3 | 3/2 | 1 |

Обозначим формируемую на каждом этапе МАИ матрицу парных сравнений (МПС) альтернатив , где , – компоненты весового вектора – количество сравниваемых альтернатив.

Получим локальные приоритеты по каждому из критериев, нормировав среднее геометрическое каждой из строк матрицы.

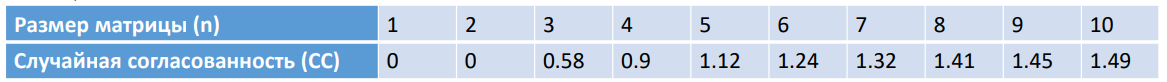
*Таблица 2. Локальные приоритеты критериев*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | C1 | C2 | C3 | C4 | C5 | C6 | C7 | Сумма |
| Произведение | 6 | 0.0005 | 300 | 0.007 | 3 | 2.5 | 21 | - |
|  | 1.35 | 0.282 | 2.587 | 0.437 | 1.201 | 1.165 | 1.661 | 8.683 |
| Локальный приоритет | 0.155 | 0.032 | 0.298 | 0.05 | 0.138 | 0.134 | 0.191 | 1 |

Для того, чтобы продолжать расчеты, необходимо проверить таблицы данных на согласованность. Для этого произведем расчет максимального собственного числа матрицы:

Далее необходимо найти индекс согласованности суждений по матрице:

Определив индекс случайной согласованности () по таблице согласованности (рис. 1), определим индекс отношений согласованности. Если он больше , то суждения эксперта, на основе которых заполнена исследуемая матрица, сильно рассогласованы, следует заполнить матрицу заново.

**

*Рисунок 1. Таблица согласованности Т. Саати*

Аналогичные операции проделаем и с таблицами альтернатив по всем критериям.

После этого необходимо составить общую таблицу (таб. 3) весов критериев и альтернатив, заполнив их значениями вектора .

*Таблица 3. Веса критериев и альтернатив*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | C1 | C2 | C3 | C4 | C5 | C6 | C7 |
|  |  |  |  |  |  |  |  |
| П1 |  |  |  |  |  |  |  |
| П2 |  |  |  |  |  |  |  |
| П3 |  |  |  |  |  |  |  |
| П4 |  |  |  |  |  |  |  |
| П5 |  |  |  |  |  |  |  |

Наконец, вычисляется вес альтернативы и выбирается наибольшее значение и соответствующая альтернатива:

Метод МАИ+ заменяет последний шаг алгоритма. Вместо этого необходимо:

1. Найти всевозможные векторы локальных ВКА (уровень 1) по каждому критерию и нормализовать по формуле: , где – номер критерия, ( – количество критериев)
2. Сформировать обобщенную матрицу(локальные ВКА уровня 2) относительно всей совокупности мер относительной важности признаков в соответствии с формулами:

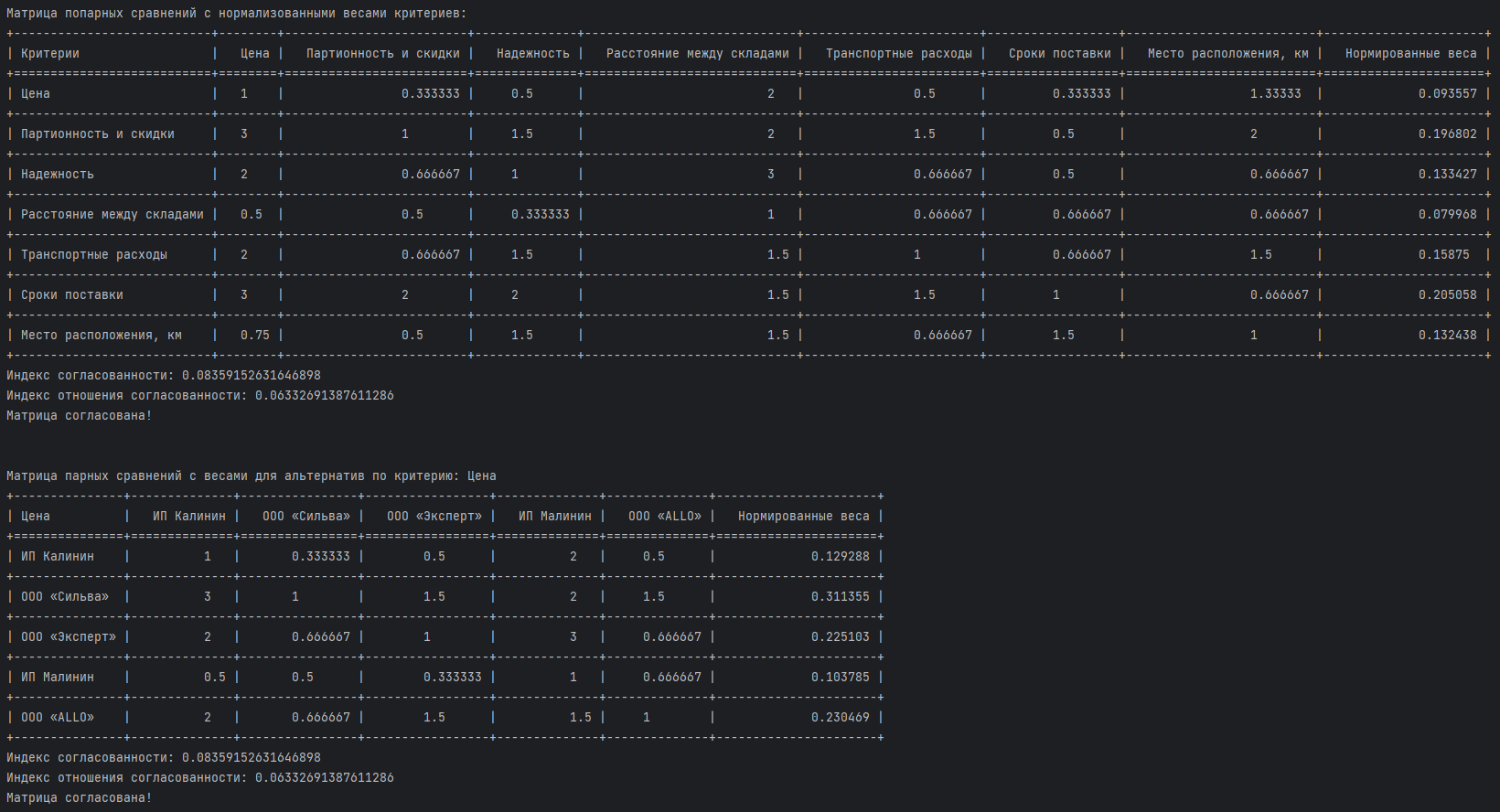
– вес критерия .

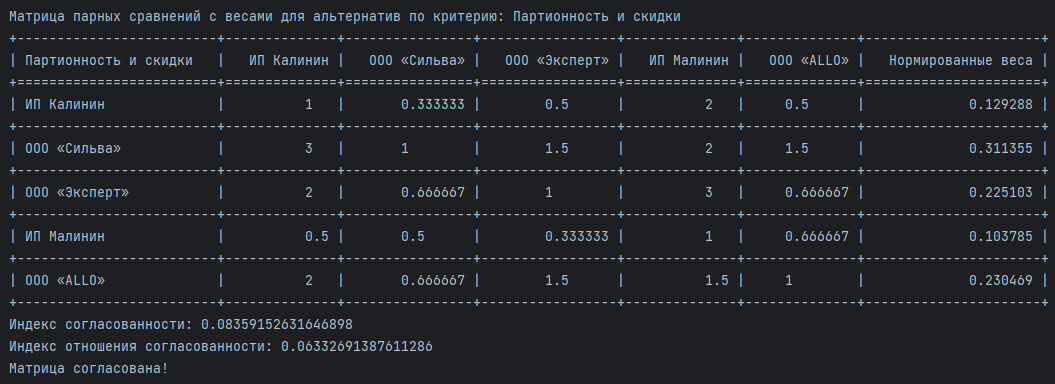
1. Произвести нормализацию, аналогично пункту 2.
2. Получить глобальные значения ВКА по одной формуле:

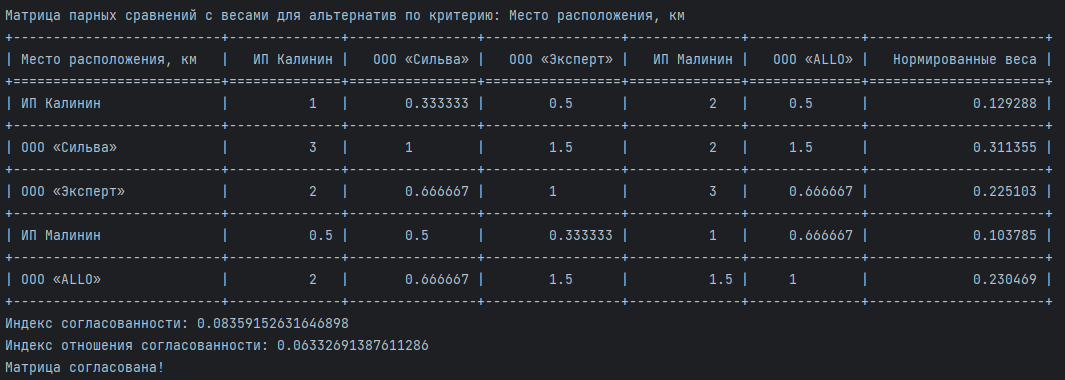
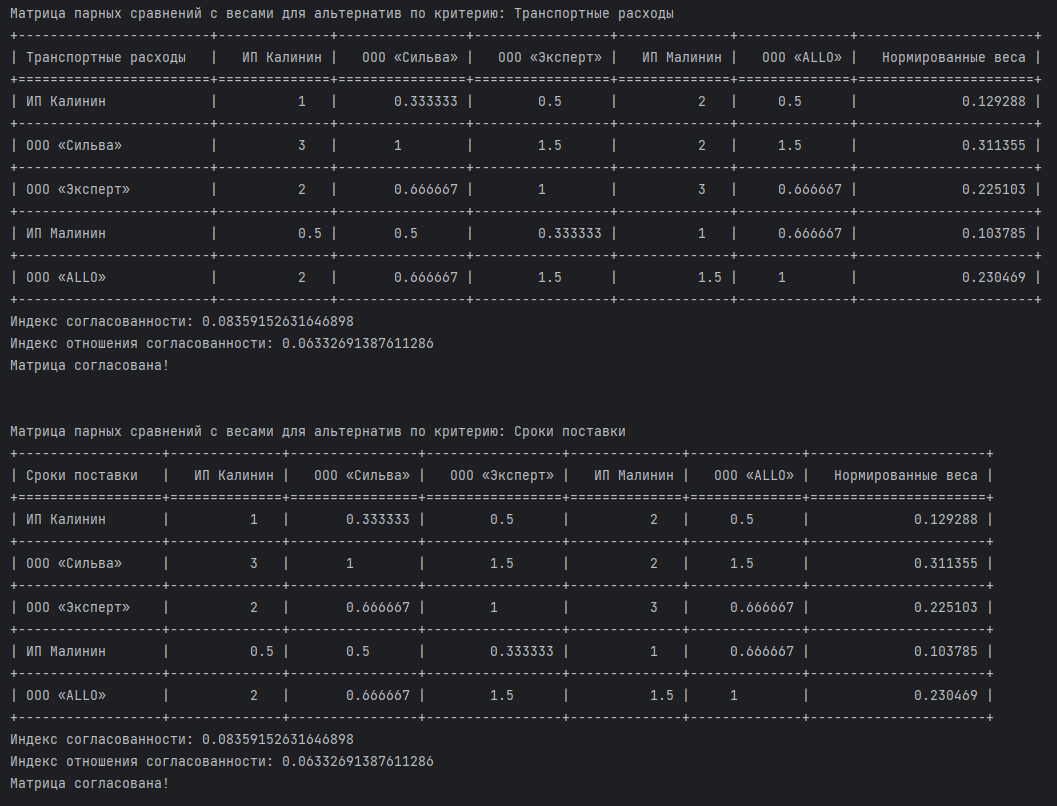
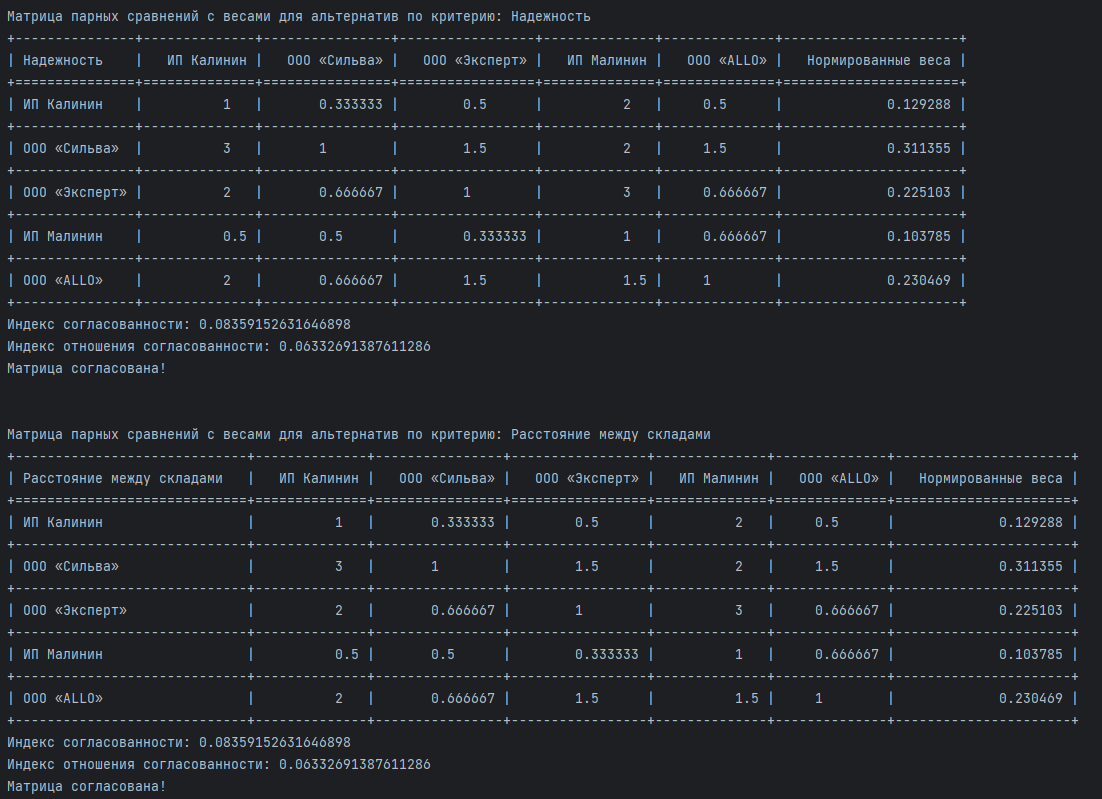
, где – количество альтернатив.

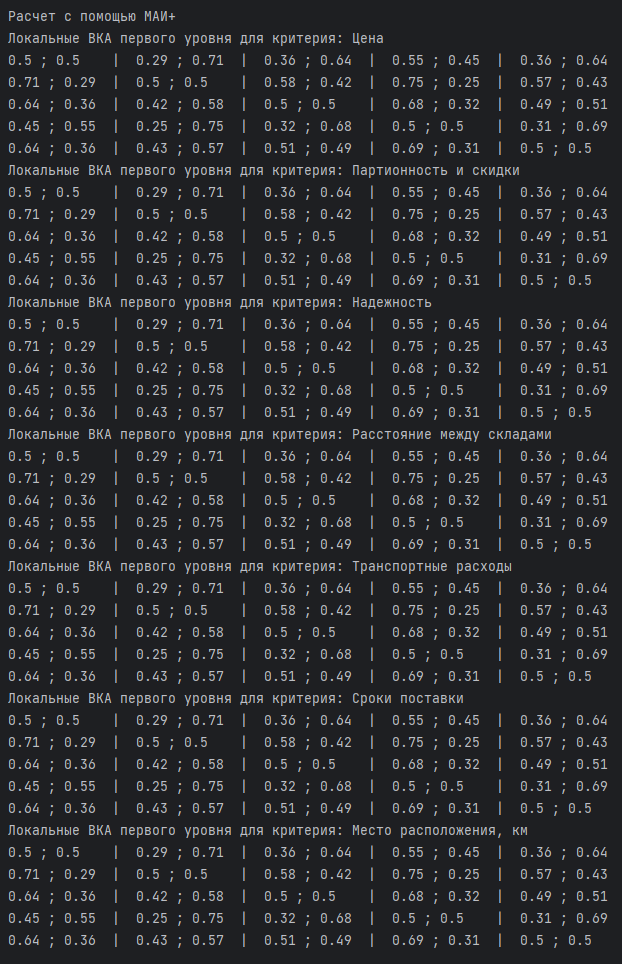
1. Нормализовать веса, найти наилучшую альтернативу в соответствии с полученными весами, записать ответ.

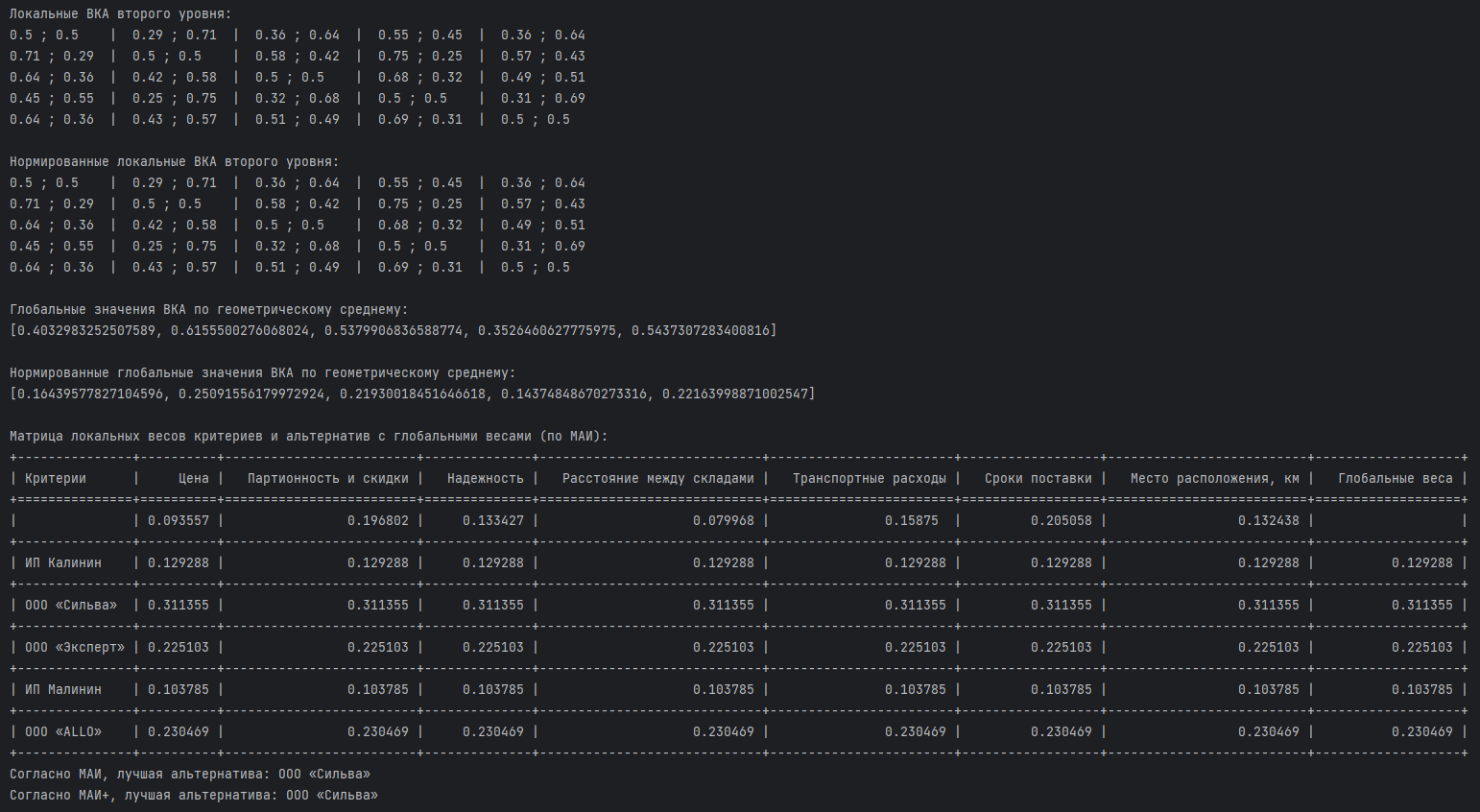
Для автоматизации всех операций был написан скрипт Python (приложение А), результат выполнения которого представлен ниже.











**Выводы**

В ходе выполнения лабораторной работы были получены теоретические знания об алгоритме МАИ и МАИ+, а также разработана программа, моделирующая алгоритм поиска «лучшего» решения слабоструктурированной задачи решения проблемы выбора поставщика продукции. Так, для варианта №8, для задачи с поиском «лучшего» поставщика, и МАИ, и МАИ+ определили таковой компанию ООО «Сильва».

**Приложение А.**

Листинг скрипта Python (3.10):

# Вариант №8.  
import fractions  
import io  
import math  
  
from tabulate import tabulate  
  
# C1. Цена  
# C2. Партионность и скидки  
# C3. Надежность  
# C4. Расстояние между складами  
# C5. Транспортные расходы  
# C6. Сроки поставки  
# C7. Место расположения, км  
  
import numpy  
from unicodedata import decimal  
  
  
def calculate\_weights(criteria\_table\_path):  
 criteria\_table = []  
 weights = []  
  
 with open(criteria\_table\_path) as table:  
 for line in table:  
 current\_criteria = [float(fractions.Fraction(x)) for x in line.rsplit()]  
 criteria\_table.append(current\_criteria)  
 weights.append(  
 pow(numpy.prod([float(fractions.Fraction(x)) for x in current\_criteria]), 1 / len(current\_criteria)))  
 sum\_weights = sum(weights)  
 normal\_weights = [x / sum\_weights for x in weights]  
 return criteria\_table, normal\_weights  
  
  
def create\_weight\_alternative\_table(cr\_table, criteria):  
 total\_weights = [cr\_table]  
 for i in range(len(criteria[0])):  
 new\_line = []  
 for j in range(len(criteria)):  
 new\_line.append(criteria[j][i])  
 total\_weights.append(new\_line)  
 return total\_weights  
  
  
def calculate\_total\_coefs(tl\_weights):  
 return [sum([tl\_weights[0][j] \* tl\_weights[i][j] for j in range(len(tl\_weights[0]))])  
 for i in range(1, len(tl\_weights))]  
  
  
def is\_consistence(table, weights):  
 if len(weights) <= 2:  
 return True  
 consistence\_random = [0, 0, 0, 0.58, 0.9, 1.12, 1.24, 1.32, 1.41, 1.45, 1.49]  
 sum\_column = [sum([table[j][i] for j in range(len(table))]) for i in range(len(table))]  
 lambda\_max = sum([sum\_column[i] \* weights[i] for i in range(len(table))])  
 consistence\_index = (lambda\_max - len(table)) / (len(table) - 1)  
 print('Индекс согласованности:', consistence\_index)  
 consistence\_ratio\_index = consistence\_index / consistence\_random[len(table)]  
 print('Индекс отношения согласованности:', consistence\_ratio\_index)  
 return consistence\_ratio\_index < 0.1  
  
def calculate\_mai\_plus(cr\_weights, al\_weights, names):  
 locals\_table = []  
 for table in al\_weights:  
 locals = []  
 for i in range(len(al\_weights[0])):  
 local\_row = []  
 for j in range(len(al\_weights[0])):  
 local\_row.append([table[i] / (table[i] + table[j]), table[j] / (table[i] + table[j])])  
 locals.append(local\_row)  
 locals\_table.append(locals)  
  
 for i in range(len(locals\_table)):  
 print("Локальные ВКА первого уровня для критерия:", names[i])  
 for local in locals\_table[i]:  
 test = [" ; ".join([str(round(x, 2)) for x in elem]) for elem in local]  
 print("\t | \t".join(test))  
  
 vka\_2 = []  
 for i in range(len(locals\_table[0])):  
 temp\_row = []  
 for j in range(len(locals\_table[0])):  
 elem\_x, elem\_y = 0, 0  
 for crit in range(len(cr\_weights)):  
 elem\_x += cr\_weights[crit] \* locals\_table[crit][i][j][0]  
 elem\_y += cr\_weights[crit] \* locals\_table[crit][i][j][1]  
 temp\_row.append([elem\_x, elem\_y])  
 vka\_2.append(temp\_row)  
  
 print("\nЛокальные ВКА второго уровня:")  
 for local in vka\_2:  
 test = [" ; ".join([str(round(x, 2)) for x in elem]) for elem in local]  
 print("\t | \t".join(test))  
  
 vka\_2 = [[[elem[0] / (elem[0] + elem[1]), elem[1] / (elem[0] + elem[1])] for elem in row] for row in vka\_2]  
 globs = [pow(math.prod([x[0] for x in row]), 1 / len(vka\_2)) for row in vka\_2]  
  
 print("\nНормированные локальные ВКА второго уровня:")  
 for local in vka\_2:  
 test = [" ; ".join([str(round(x, 2)) for x in elem]) for elem in local]  
 print("\t | \t".join(test))  
  
 print("\nГлобальные значения ВКА по геометрическому среднему:")  
 print(globs)  
 globs = [elem / sum(globs) for elem in globs]  
 print("\nНормированные глобальные значения ВКА по геометрическому среднему:")  
 print(globs)  
 return globs  
  
  
def solve(criteria\_table\_path, number\_of\_criterias):  
 criteria\_names = []  
 with io.open("tables/criteria\_names.txt", encoding='utf-8') as names\_file:  
 for line in names\_file:  
 criteria\_names.append("".join(line.rsplit('\n')))  
 names\_for\_table\_criteria = ['Критерии'] + criteria\_names + ['Нормированные веса']  
  
 alternative\_names = []  
 with io.open("tables/alternative\_names.txt", encoding='utf-8') as names\_file:  
 for line in names\_file:  
 alternative\_names.append("".join(line.rsplit('\n')))  
  
 print("Матрица попарных сравнений с нормализованными весами критериев:")  
 cr\_table, n\_weights = calculate\_weights(criteria\_table\_path)  
 cr\_table\_tabulate = []  
 for i in range(len(cr\_table)):  
 temp\_line = [criteria\_names[i]]  
 temp\_line += [str(x) for x in cr\_table[i]]  
 temp\_line += [str(n\_weights[i])]  
 cr\_table\_tabulate.append(temp\_line)  
 print(tabulate(cr\_table\_tabulate, names\_for\_table\_criteria, 'grid'))  
 if not is\_consistence(cr\_table, n\_weights):  
 print("Матрица не согласована! Невозможно применить МАИ")  
 return  
 else:  
 print("Матрица согласована!\n\n")  
  
 alternative\_tables = []  
 alternative\_weights = []  
 for i in range(number\_of\_criterias):  
 a\_table, a\_weights = calculate\_weights("tables/alternative\_c" + str(i + 1) + ".txt")  
 alternative\_tables.append(a\_table)  
 alternative\_weights.append(a\_weights)  
 names\_for\_alternative = [criteria\_names[i]] + alternative\_names + ['Нормированные веса']  
 a\_table\_tabulate = []  
 for j in range(len(a\_table)):  
 temp\_line = [alternative\_names[j]]  
 temp\_line += [str(x) for x in a\_table[j]]  
 temp\_line += [str(a\_weights[j])]  
 a\_table\_tabulate.append(temp\_line)  
 print("Матрица парных сравнений с весами для альтернатив по критерию:", criteria\_names[i])  
 print(tabulate(a\_table\_tabulate, names\_for\_alternative, 'grid'))  
 if not is\_consistence(cr\_table, n\_weights):  
 print("Матрица не согласована! Невозможно применить МАИ")  
 return  
 else:  
 print("Матрица согласована!\n\n")  
  
 print("Расчет с помощью МАИ+")  
  
 globs = calculate\_mai\_plus(n\_weights, alternative\_weights, criteria\_names)  
 total = create\_weight\_alternative\_table(n\_weights, alternative\_weights)  
  
 result = calculate\_total\_coefs(total)  
  
 print("\nМатрица локальных весов критериев и альтернатив с глобальными весами (по МАИ):")  
 alternative\_names.insert(0, "")  
 names\_for\_table\_criteria.insert(-1, "Глобальные веса")  
 for i in range(len(total)):  
 if i != 0:  
 total[i] += [result[i - 1]]  
 total[i] = [alternative\_names[i]] + [str(elem) for elem in total[i]]  
 print(tabulate(total, names\_for\_table\_criteria, 'grid'))  
 print("Согласно МАИ, лучшая альтернатива:", alternative\_names[result.index(max(result)) + 1])  
  
 print("Согласно МАИ+, лучшая альтернатива:", alternative\_names[globs.index(max(globs)) + 1])  
  
  
solve("tables/criteria\_table.txt", 7)

Таблицы альтернатив «alternative\_c1.txt» и прочие:

1 1/3 1/2 2 1/2  
3 1 3/2 2 3/2  
2 2/3 1 3 2/3  
1/2 1/2 1/3 1 2/3  
2 2/3 3/2 3/2 1

Таблица названий альтернатив «alternative\_names.txt» и критериев «criteria\_names.txt» соответственно:

ИП Калинин  
ООО «Сильва»  
ООО «Эксперт»  
ИП Малинин  
ООО «ALLO»

Цена  
Партионность и скидки  
Надежность  
Расстояние между складами  
Транспортные расходы  
Сроки поставки  
Место расположения, км

Таблица критериев «criteria\_table»:

1 1/3 1/2 2 1/2 1/3 4/3  
3 1 3/2 2 3/2 1/2 2  
2 2/3 1 3 2/3 1/2 2/3  
1/2 1/2 1/3 1 2/3 2/3 2/3  
2 2/3 3/2 3/2 1 2/3 3/2  
3 2 2 3/2 3/2 1 2/3  
3/4 1/2 3/2 3/2 2/3 3/2 1