



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА – Российский технологический университет»

**РТУ МИРЭА**

Институт информационных технологий

Кафедра вычислительной техники

## КУРСОВАЯ РАБОТА

По дисциплине

«Теория принятия решений»

(наименование дисциплины)

Тема курсовой работы

Методы многокритериальной оптимизации и линейного

(наименование темы)

программирования «Выбор оптимального варианта телевизора»

Студент группы

ИКБО-04-20

(учебная группа)

Хан Анастасия Александровна

(Фамилия Имя Отчество)

*A. Хан*  
(подпись студента)

Руководитель курсовой работы

доцент каф. ВТ Сорокин А.Б.

(Должность, звание, ученая степень)

*Сорокин*  
(подпись руководителя)

Консультант

(Должность, звание, ученая степень)

(подпись консультанта)

Работа представлена к защите «09» 06 2022 г.

Допущен к защите «09» 06 2022 г.

Москва 2022 г.

*отличено*  
*Сд*



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА – Российский технологический университет»

**РТУ МИРЭА**

Институт информационных технологий

Кафедра вычислительной техники

Утверждаю

Заведующий кафедрой

Платонова О.В.

ФИО

Подпись

« 17 » февраля 2022г.

**ЗАДАНИЕ**

На выполнение курсовой работы  
по дисциплине «Теория принятия решений»

Студент Хан Анастасия Александровна Группа ИКБО-04-20

Тема Методы многокритериальной оптимизации и линейного программирования  
«Выбор оптимального варианта телевизора»

**Исходные данные:**

1. Описания исходных данных для многокритериальной оптимизации: 10 альтернатив и 6 критериев.

2. Линейное программирование по вариантам.

**Перечень вопросов, подлежащих разработке, и обязательного графического материала:**

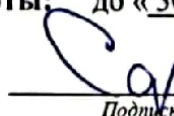
1. Реализовать расчет и консольное приложение:

1.1. Для многокритериальной оптимизации: Парето множество и его сужение, метод Электро II и анализ иерархий;

1.2. Для линейного программирования: графический метод, симплекс метод, двойственную (3 теоремы) и транспортную задачу.

**Срок представления к защите курсовой работы:** до « 30 » мая 2022 г.

Задание на курсовую работу выдал

  
Подпись

( Сорокин А.Б.)

ФИО консультанта

« 17 » февраля 2022 г.

Задание на курсовую работу получил

  
Подпись

( Хан А.А.)

ФИО исполнителя

« 17 » февраля 2022 г.

Москва 2022г.

## ОТЗЫВ

на курсовую работу

по дисциплине «Теория принятия решений»

Студент Хан Анастасия Александровна группа ИКБО-04-20  
(ФИО студента) (Группа)

### Характеристика курсовой работы

Критерий	Да	Нет	Не полностью
1. Соответствие содержания курсовой работы указанной теме	+		
2. Соответствие курсовой работы заданию	+		
3. Соответствие рекомендациям по оформлению текста, таблиц, рисунков и пр.	+		
4. Полнота выполнения всех пунктов задания	+		
5. Логичность и системность содержания курсовой работы	+		
6. Отсутствие фактических грубых ошибок	+		

Замечаний:

нет.

Рекомендуемая оценка:

отлично



доцент каф. ВТ Сорокин А.Б.

(Подпись руководителя)

(ФИО руководителя)

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	7
1 ОПИСАНИЕ АЛГОРИТМА МЕТОДА ПАРЕТО .....	8
1.1 Постановка задачи .....	8
1.2 Расчетная часть по множеству Парето .....	8
1.3 Сужение по Парето .....	10
1.4 Метод верхних и нижних критериев .....	11
1.5 Метод субоптимизации .....	12
1.6 Метод лексикографической оптимизации .....	14
2 ОПИСАНИЕ АЛГОРИТМА МЕТОДА ЭЛЕКТРА II .....	16
2.1 Постановка задачи .....	16
2.2 Выполнение первого этапа .....	16
2.2.1 Составление таблицы критериев для оценки .....	17
2.2.2 Расчет всех пар проектов .....	18
2.3 Выполнение второго этапа .....	34
3 ОПИСАНИЕ АЛГОРИТМА МЕТОДА МАИ .....	36
3.1 Постановка задачи .....	36
3.2 Выполнение первого этапа .....	37
3.3 Выполнение второго этапа .....	41
3.4 Выполнение третьего этапа .....	45
3.5 Ручной счет .....	46
3.5.1 Первый этап .....	46
3.5.2 Второй этап .....	47
3.5.3 Третий этап .....	66

3.6 Программная реализация .....	67
4 ОПИСАНИЕ АЛГОРИТМА ГРАФИЧЕСКОГО МЕТОДА .....	68
4.1 Постановка задачи.....	68
4.2 Выполнение работы .....	68
4.2.1 Первый этап .....	68
4.2.2 Второй этап .....	69
5 ОПИСАНИЕ АЛГОРИТМА СИМПЛЕКС МЕТОДА.....	71
5.1 Постановка задачи.....	71
5.2 Расчетная часть.....	72
5.3 Программная реализация .....	75
6 ОПИСАНИЕ АЛГОРИТМА ДВОЙСТВЕННОЙ ЗАДАЧИ .....	77
6.1 Постановка задачи.....	77
6.2 Расчетная часть.....	77
6.2.1 Первая теорема двойственности.....	77
6.2.2 Вторая теорема двойственности.....	80
6.2.3 Третья теорема двойственности .....	82
6.3 Программная реализация .....	89
7 ОПИСАНИЕ АЛГОРИТМА ТРАНСПОРТНОЙ ЗАДАЧИ.....	91
7.1 Постановка задачи.....	91
7.2 Расчетная часть.....	91
7.2.1 Метод северо-западного угла.....	92
7.2.2 Метод минимальной стоимости .....	95
7.2.3 Метод потенциалов .....	98
7.3 Программная реализация .....	107
ЗАКЛЮЧЕНИЕ .....	108

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ .....	109
ПРИЛОЖЕНИЯ.....	110

## ВВЕДЕНИЕ

Теория принятия решений – это особая область исследований, оперирующая математическими, статистическими, экономическими, психологическими и управленческими терминами, для закономерностей выбора людьми путей принятия решений и разрешений проблем и способов достижения поставленных целей.

Ядром служит отдельная область – принятие решения в условиях неопределенности, то есть в таких случаях, когда результат выбора неизвестен. Неопределенность может быть стохастической, поведенческой, природной и априорной. Выбор в условиях неопределенности основывается на неизвестном множестве итогов.

С учетом разных вариантов поведения, каждый из которых может привести к нескольким результатам, рациональный подход должен выявлять все возможные итоги, устанавливать их ценность и вероятность, и указывать на основе их совокупности общую ожидаемую ценность. Это сокращает негативное воздействие на принятие решения эффекта неопределенности.

Несложно заметить, что теория принятия решений таит в себе огромное количество полезной информации, изучение которой позволит намного глубже вникнуть в поведенческую психологию. Она определяет нормы поведения для человека, принимающего решение. Но теория вовсе не диктует поведение человека. Она лишь помогает ему, обеспечивает методологией, позволяющей принимать решения.

По мере роста сложности проблем ослабевает способность человека неформально обрабатывать информацию, основываясь на собственных суждениях. Именно здесь теория принятия решений проявляет себя во всей красе, предлагая преимущества перед любым другим аналитическим подходом к решению проблем. Она включает в себя множество субъективных аспектов проблем, что особенно важно при принятии решений в индивидуальном порядке.

# **1 ОПИСАНИЕ АЛГОРИТМА МЕТОДА ПАРЕТО**

Выбор множества Парето-оптимальных решений представляет собой отбор перспективных альтернатив, из которых затем отбирается одна лучшая альтернатива.

Множество Парето представляет собой множество альтернатив, обладающих следующим свойством: любая из альтернатив, входящих во множество Парето, хотя бы по одному критерию лучше любой другой альтернативы, входящей в это множество [1].

Все альтернативы попарно сравниваются друг с другом по всем критериям. Если при сравнении каких-либо альтернатив, не лучше другой ни по одному критерию, то ее можно исключить из рассмотрения. Исключенную альтернативу не требуется сравнивать с другими альтернативами, так как она неперспективна [2].

## **1.1 Постановка задачи**

Выбрать телевизор с использованием метода Парето.

## **1.2 Расчетная часть по множеству Парето**

Приведу пример выбора телевизора на российском рынке с использованием Парето-оптимального множества решений. Проанализировав информацию на сайте «Технопарк» (<https://www.technopark.ru/>) — одной из известных российских компаний, специализирующаяся на продаже крупной бытовой техники и цифровой электроники, имеющая множество магазинов по всей стране, были выделены варианты решений (альтернативы) и их критерии, и сведены в Таблице 1.2.



Таблица 1.2 - Альтернативы и их критерии

		Модель и фирма	Критерии					
			Цена, тыс.руб (-)	Разрешение экрана, К (+)	Год (-)	Гарантия, мес (+)	Вес, кг (-)	Диагональ, см (+)
1	A1	4K UHD OLED Smart TV - LG	180	4	2021	12	24	145
2	A2	4K UHD QLED Smart TV - Samsung	95	4	2020	12	12	247
3	A3	8K UHD QLED Smart TV - Samsung	200	8	2021	12	40	165
4	A4	8K UHD NanoCell Smart TV - LG	80	8	2020	24	17	140
5	A5	4K UHD LED Smart TV - Sony	140	4	2021	12	22	165
6	A6	8K UHD Neo QLED Smart TV - Sony	170	8	2021	24	32	191
7	A7	4K UHD Neo QLED Smart TV - Samsung	999	4	2020	24	75	247
8	A8	8K UHD LED Smart TV - Sony	200	8	2020	24	32	191
9	A9	8K UHD Neo QLED Smart TV - Samsung	999	8	2021	24	54	216

Продолжение Таблицы 1.2

<b>10</b>	<b>A10</b>	4K UHD NanoCell Smart TV - LG	130	4	2020	12	37	127
-----------	------------	----------------------------------------	-----	---	------	----	----	-----

Было определено, что оптимизация по Парето использует отношение Парето-доминирования, которое отдаёт предпочтение одному объекту перед другим только в том случае, когда первый объект по всем критериям не хуже второго, и, хотя бы по одному из них лучше [1]. При истинности этого условия первый объект считается доминирующим, а второй - доминируемым. Два объекта, для которых предпочтение хотя бы, по одному критерию расходится, считаются несравнимыми.

### 1.3 Сужение по Парето

Сравним попарно все альтернативы и сведём их в Таблице 1.3.

Таблица 1.3 – Сравнение альтернатив

		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>
<b>1</b>	<b>A1</b>	X	X	X	X	X	X	X	X	X	X
<b>2</b>	<b>A2</b>	A2	X	X	X	A2	X	X	X	X	A2
<b>3</b>	<b>A3</b>	H	H	X	X	X	X	X	X	X	X
<b>4</b>	<b>A4</b>	H	H	H	X	X	X	X	X	X	A4
<b>5</b>	<b>A5</b>	A5	H	H	H	X	X	X	X	X	X
<b>6</b>	<b>A6</b>	H	H	A6	H	H	X	X	X	X	X
<b>7</b>	<b>A7</b>	H	H	H	H	H	H	X	X	X	X
<b>8</b>	<b>A8</b>	H	H	A8	H	H	H	H	X	X	X
<b>9</b>	<b>A9</b>	H	H	H	H	H	H	H	H	X	X
<b>10</b>	<b>A10</b>	H	H	H	H	H	H	H	H	H	X

Выделяются Парето-оптимальные варианты  $p = \{2,4,5,6,8\}$ .

Результат выполнения программы представлен на Рисунке 1.3.1 и Рисунке 1.3.2.

```

C:\Users\Anastasia\source\repos\prrrr1\prrrr1\bin\Debug\prrrr1.exe
Выберите способ задания таблицы:
1 - из файла
2 - случайные значения для N альтернатив
>> 1
№ price per year garant weight diag
1 180 4 2021 12 24 145
2 95 4 2020 12 12 247
3 200 8 2021 12 40 165
4 80 8 2020 24 17 140
5 140 4 2021 12 22 165
6 170 8 2021 24 32 191
7 999 4 2020 24 75 247
8 200 8 2020 24 32 191
9 999 8 2021 24 54 216
10 130 4 2020 12 37 127

ПОПАРНОЕ СРАВНЕНИЕ ВСЕХ АЛЬТЕРНАТИВ.
1 2 3 4 5 6 7 8 9 10
1 x x x x x x x x x
x 2 2 x x x x x x x
x 3 н н x x x x x x x
x 4 н н н x x x x x x
x 5 5 2 н н x x x x x
x 6 н н 6 н н x x x x
x 7 н н н н н н x x x
x 8 н н 8 н н н н x x
x 9 н н н н н н н н x
x 10 н 2 н 4 н н н н н

```

Рисунок 1.3.1 – Реализация оптимизации по Парето

№	price	per	year	garant	weight	diag
2	95	4	2020	12	12	247
4	80	8	2020	24	17	140
5	140	4	2021	12	22	165
6	170	8	2021	24	32	191
8	200	8	2020	24	32	191

Рисунок 1.3.2 – Парето-оптимальные варианты

## 1.4 Метод верхних и нижних критериев

Установим для приведенного примера верхнюю границу: цена не более 180 тыс. руб., год 2021, вес не более 40 кг.

Согласно данным условиям Таблицы 1.2 трансформируется в Таблицу 1.4.

Таблица 1.4.1 - Варианты, удовлетворяющие дополнительным ограничениям

	Модель и фирма	Критерии					
		Цена, тыс.руб (-)	Разрешение экрана, К (+)	Год (-)	Гарантия, мес (+)	Вес, кг (-)	Диагональ, см (+)
1	4K UHD OLED Smart TV - LG	180	4	2021	12	24	145

Продолжение Таблицы 1.4.1

3	8K UHD QLED Smart TV – Samsung	200	8	2021	12	40	165
6	8K UHD Neo QLED Smart TV - Sony	170	8	2021	24	32	191

Варианты, удовлетворяющие этим дополнительным ограничениям: {1, 3, 6}; из них оптимальными по Парето является вариант 6.

Сравним попарно все альтернативы и сведем их в Таблице 1.4.2.

Таблица 1.4.2 – Сравнение альтернатив

	1	3	6
1	X	X	X
3	H	X	X
6	H	A6	X

Результат выполнения программы представлен на Рисунке 1.4.

```

УКАЗАНИЕ ВЕРХНИХ \ НИЖНИХ ГРАНИЦ КРИТЕРИЕВ.

Задайте значения границ критериев (или введите def, чтобы пропустить критерий):

Минимальная цена: 170
Максимальное разрешение: 8
Минимальный год: 2021
Максимальная гарантия: 24
Минимальная вес: 32
Максимальная диагональ: 191

№   price   per   year   garant   weight   diag
6   170      8     2021    24      32      191

```

Рисунок 1.4 – Реализация метода верхних/нижних границ критериев

## 1.5 Метод субоптимизации

Субоптимизацию производят следующим образом: выделяют один из критериев, а по всем остальным критериям назначают нижние границы [2]. Оптимальным при этом считается исход, максимизирующий выделенный критерий на множестве исходов, оценки которых по остальным критериям не ниже назначенных.

Пусть в примере главным критерием выступает цена; ограничения: диагональ не более 165 см; год 2021; вес не более 40 кг. Отбросим варианты, которые не удовлетворяют данным ограничениям и составим Таблицу 1.5.1.

Таблица 1.5.1 - Варианты, удовлетворяющие дополнительным ограничениям

	Модель и фирма	Критерии					
		Цена, тыс.руб (-)	Разрешение экрана, К (+)	Год (-)	Гарантия, мес (+)	Вес, кг (-)	Диагональ, см (+)
<b>1</b>	4K UHD OLED Smart TV - LG	180	4	2021	12	24	145
<b>3</b>	8K UHD QLED Smart TV – Samsung	200	8	2021	12	40	165
<b>5</b>	4K UHD LED Smart TV - Sony	140	4	2021	12	22	165

Сравним попарно все альтернативы и сведем их в Таблице 1.5.2.

Таблица 1.5.2 – Сравнение альтернатив

	<b>1</b>	<b>3</b>	<b>5</b>	<b>6</b>
<b>1</b>	X	X	X	X
<b>3</b>	H	X	X	X
<b>5</b>	A5	H	X	X

Из табл. 6 видно, остаются варианты {1, 3, 5}. Из них минимальную цену имеет вариант 5. Этот вариант и будет оптимальным.

Результат выполнения программы представлен на Рисунке 1.5.

```
СУБОПТИМИЗАЦИЯ.
Укажите главный критерий (1 - цена, 2 - разрешение, 3 - год, 4 - гарантия, 5 - вес, 6 - диагональ):
>> 1

Максимальное разрешение: 8
Минимальный год: 2021
Максимальная гарантия: 12
Минимальная вес: 40
Максимальная диагональ: 165

Выбор альтернатив по критериям:

№   price   per   year   garant   weight   diag
6   170      8     2021    24      32      191
8   200      8     2020    24      32      191

Выбор по главному критерию:

№   price   per   year   garant   weight   diag
6   170      8     2021    24      32      191
```

Рисунок 1.5 – Реализация метода субоптимизации

С помощью метода субоптимизации задача многокритериальной оптимизации превращается в задачу скалярной оптимизации на суженном допустимом множестве. Выделение одного из критериев, а также указание нижних границ для остальных критериев основано на дополнительной информации, получаемой от ЛПР. Следовательно, окончательное решение здесь также имеет субъективный характер.

## 1.6 Метод лексикографической оптимизации

Лексикографическая оптимизация основана на упорядочении критериев по их относительной важности. После этого процедуру нахождения оптимального решения проводят следующим образом. На первом шаге отбирают исходы, которые имеют максимальную оценку по важнейшему критерию. Если такой исход единственный, то его и считают оптимальным. Если же таких исходов несколько, то среди них отбирают те, которые имеют максимальную оценку по следующему за важнейшим критерием [2]. В результате такой процедуры всегда остается (по крайней мере, в случае конечного множества исходов) единственный исход — он и будет оптимальным.

Упорядочим критерии в примере по относительной важности, например, следующим образом: важнейший критерий – диагональ, следующий за ним по важности – разрешение. Максимальное значение по критерию «Диагональ» имеют варианты {2, 7, 9}. Далее сравниваем эти варианты по критерию «Разрешение», остается вариант 9.

Таким образом, наглядно проявляется недостаток лексикографической оптимизации — фактический учет одного (важнейшего) критерия. Например, в последнем случае в качестве оптимального выступает вариант 9, который имеет более низкую оценку по критерию «Диагональ».

Результат программы представлен на Рисунке 1.6.

```
ЛЕКСИКОГРАФИЧЕСКАЯ ОПТИМИЗАЦИЯ.  
Укажите главный критерий (1 - цена, 2 - разрешение, 3 - год, 4 - гарантия, 5 - вес, 6 - диагональ):  
>> 1  
  
№   price   per   year   garant   weight   diag  
4    80      8     2020    24      17      140
```

**Рисунок 1.6 – Реализация метода лексикографической оптимизации**

Код реализации представлен в Приложении А.

## **2 ОПИСАНИЕ АЛГОРИТМА МЕТОДА ЭЛЕКТРА II**

Основу методологии решающих правил основанных на порогах чувствительности составляют методы класса ЭЛЕКТРА, которые были разработана коллективом французских ученых, возглавляемым профессором Б. Руа. В настоящее время разработан ряд методов семейства ЭЛЕКТРА.

ЭЛЕКТРА I позволяет из множества вариантов исключить неэффективные варианты. В основе данного метода лежит попарное сравнение отдельных вариантов.

ЭЛЕКТРА II служит для упорядочения индифферентных классов вариантов.

ЭЛЕКТРА III отличается от метода ЭЛЕКТРА II способом задания порогов чувствительности.

В данных подходах принято различать 2 этапа:

1. Этап разработки, на котором строятся один или несколько индексов попарного сравнения альтернатив.
2. Этап исследования, на котором построенные индексы используются для ранжирования (или классификации) заданного множества альтернатив.

### **2.1 Постановка задачи**

Выбрать телевизор с использованием метода ЭЛЕКТРА II.

### **2.2 Выполнение первого этапа**

Этап разработки, на котором строятся один или несколько индексов попарного сравнения альтернатив.



## 2.2.1 Составление таблицы критериев для оценки

На первом этапе определяется множество решений и для каждого из N критериев определяется вес – число, характеризующее важность соответствующего критерия [1]. Вес критерия можно установить множеством способов (ранжированием, экспертных оценок и т.д.).

Таблица критериев, по которым будут оценивать варианты представлена на Таблице 2.2.1.1.

*Таблица 2.2.1.1 - Таблица критериев для оценки проектов*

Критерии	Вес	Шкала	Код	Стремление
Цена	5	До 95	15	Min
		96 – 170	10	
		171 – 200	5	
		201 – 999	1	
Разрешение	5	Высокое (8)	10	Max
		Низкое (4)	5	
Год	4	2020	8	Min
		2021	4	
Гарантия	5	24 мес	10	Max
		12 мес	5	
Вес	2	12 – 23	8	Min
		24 – 54	4	
		55 - 75	2	
Диагональ	4	127 – 145	12	Max
		146 – 165	8	
		166 – 191	4	
		192 - 247	2	

Далее составляем таблицу оценок вариантов решений, которая представлена на Таблице 2.2.1.2.

*Таблица 2.2.1.2 - Таблица оценок вариантов по критериям*

	Цена	Разрешение	Год	Гарантия	Вес телевизора	Диагональ
1	LG	5	5	4	5	4

Продолжение Таблицы 2.2.1.2

2	Samsung	10	5	8	5	8	2
3	Samsung	5	10	4	5	4	8
4	LG	15	10	8	10	8	12
5	Sony	10	5	4	5	8	8
6	Sony	10	10	4	10	4	4
7	Samsung	1	5	8	10	2	2
8	Sony	5	10	8	10	4	4
9	Samsung	1	10	4	10	4	2
10	LG	10	5	8	5	4	12
Вс		5	5	4	5	2	4
Стремление		min	max	min	max	min	max

## 2.2.2 Расчет всех пар проектов

Рассматриваем все пары проектов  $i$  и  $j$ . Если по какому-либо критерию  $i$ -ый проект лучше, чем  $j$ -ый, то соответствующий критерию вес прибавляется к  $P_{ij}$  (эти баллы символизируют выбор «За»), в противном случае — к  $N_{ij}$  (эти баллы символизируют выбор «Против»). То же самое справедливо для  $j$ -го проекта: если  $j$ -ый проект оказывается лучше, чем  $i$ -ый, то соответствующий критерию вес прибавляется к  $P_{ji}$ , в противном случае — к  $N_{ji}$  (обратите внимание на порядок следования индексов  $j$  и  $i$  у  $P$  и  $N$ ). Если повстречалось одинаковое для  $i$ -го и для  $j$ -го проектов значение критерия, то оно пропускается. Затем, когда по паре  $i$  и  $j$  рассмотрены все критерии, находятся отношения  $D_{ij} = \frac{P_{ij}}{N_{ij}}$  и  $D_{ji} = \frac{P_{ji}}{N_{ji}}$ . Значения  $D \leq 1$  отбрасываются. Заметим, что  $D_{ji} = \frac{1}{D_{ij}}$  (и наоборот), таким образом, вычисления можно несколько упростить [2].

$$P_{12} = 5 + 0 + 4 + 0 + 2 + 4 = 15$$

$$N_{12} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{12} = \frac{P_{12}}{N_{12}} = \frac{15}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{21} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{21} = 5 + 0 + 4 + 0 + 2 + 4 = 15$$

$$D_{21} = \frac{P_{21}}{N_{21}} = \frac{0}{15} = 0 < 1 - \text{отбрасываем}$$

$$P_{13} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{13} = 0 + 5 + 0 + 0 + 0 + 0 = 5$$

$$D_{13} = \frac{P_{13}}{N_{13}} = \frac{4}{5} = 0,8 < 1 - \text{отбрасываем}$$

$$P_{31} = 0 + 5 + 0 + 0 + 0 + 0 = 5$$

$$N_{31} = 0 + 0 + 0 + 0 + 0 + 0 = 4$$

$$D_{31} = \frac{P_{31}}{N_{31}} = \frac{5}{4} = 1,25 > 1 - \text{принимаем}$$

$$P_{14} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$N_{14} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$D_{14} = \frac{P_{14}}{N_{14}} = \frac{11}{10} = 1,1 > 1 - \text{принимаем}$$

$$P_{41} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$N_{41} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$D_{41} = \frac{P_{41}}{N_{41}} = \frac{10}{11} = 0,9 < 1 - \text{отбрасываем}$$

$$P_{15} = 5 + 0 + 0 + 0 + 2 + 4 = 11$$

$$N_{15} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{15} = \frac{P_{15}}{N_{15}} = \frac{11}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{51} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{51} = 5 + 0 + 0 + 0 + 2 + 4 = 11$$

$$D_{51} = \frac{P_{51}}{N_{51}} = \frac{0}{11} = 0 < 1 - \text{отбрасываем}$$

$$P_{16} = 5 + 0 + 0 + 0 + 4 = 9$$

$$N_{16} = 5 + 0 + 5 + 0 + 0 = 10$$

$$D_{16} = \frac{P_{16}}{N_{16}} = \frac{9}{10} = 0,9 < 1 - \text{отбрасываем}$$

$$P_{61} = 5 + 0 + 5 + 0 + 0 = 10$$

$$N_{61} = 5 + 0 + 0 + 0 + 4 = 9$$

$$D_{61} = \frac{P_{61}}{N_{61}} = \frac{10}{9} = 1,11 > 1 - \text{принимаем}$$

$$P_{17} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$N_{17} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$D_{17} = \frac{P_{17}}{N_{17}} = \frac{8}{12} = 0,6 < 1 - \text{отбрасываем}$$

$$P_{71} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$N_{71} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$D_{71} = \frac{P_{71}}{N_{71}} = \frac{12}{8} = 1,5 > 1 - \text{принимаем}$$

$$P_{18} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$N_{18} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$D_{18} = \frac{P_{18}}{N_{18}} = \frac{8}{10} = 0,8 < 1 - \text{отбрасываем}$$

$$P_{81} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$N_{81} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$D_{81} = \frac{P_{81}}{N_{81}} = \frac{10}{8} = 1,25 > 1 - \text{принимаем}$$

$$P_{19} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{19} = 5 + 5 + 0 + 5 + 0 + 0 = 15$$

$$D_{19} = \frac{P_{19}}{N_{19}} = \frac{4}{15} = 0,26 < 1 - \text{отбрасываем}$$

$$P_{91} = 5 + 5 + 0 + 5 + 0 + 0 = 15$$

$$N_{91} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{91} = \frac{P_{41}}{N_{41}} = \frac{15}{4} = 3,75 > 1 - \text{принимаем}$$

$$P_{110} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$N_{110} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{110} = \frac{P_{110}}{N_{110}} = \frac{9}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{101} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{101} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$D_{101} = \frac{P_{101}}{N_{101}} = \frac{0}{9} = 0 < 1 - \text{отбрасываем}$$

$$P_{23} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{23} = 5 + 5 + 4 + 0 + 2 + 4 = 20$$

$$D_{23} = \frac{P_{23}}{N_{23}} = \frac{0}{20} = 0 < 1 - \text{отбрасываем}$$

$$P_{32} = 5 + 5 + 4 + 0 + 2 + 4 = 20$$

$$N_{32} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{32} = \frac{P_{32}}{N_{32}} = \frac{20}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{24} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$N_{24} = 0 + 5 + 0 + 5 + 0 + 4 = 14$$

$$D_{24} = \frac{P_{24}}{N_{24}} = \frac{5}{14} = 0,35 < 1 - \text{отбрасываем}$$

$$P_{42} = 0 + 5 + 0 + 5 + 0 + 4 = 14$$

$$N_{42} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$D_{42} = \frac{P_{42}}{N_{42}} = \frac{14}{5} = 2,8 > 1 - \text{принимаем}$$

$$P_{25} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$N_{25} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{25} = \frac{P_{25}}{N_{25}} = \frac{4}{4} = 1 = 1 - \text{отбрасываем}$$

$$P_{52} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{52} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$D_{52} = \frac{P_{52}}{N_{52}} = \frac{4}{4} = 1 = 1 - \text{отбрасываем}$$

$$P_{26} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{26} = 0 + 5 + 4 + 5 + 2 + 4 = 20$$

$$D_{26} = \frac{P_{26}}{N_{26}} = \frac{0}{20} = 0 < 1 - \text{отбрасываем}$$

$$P_{62} = 0 + 5 + 4 + 5 + 2 + 4 = 20$$

$$N_{62} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{62} = \frac{P_{62}}{N_{62}} = \frac{20}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{27} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{27} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$D_{27} = \frac{P_{27}}{N_{27}} = \frac{0}{12} = 0 < 1 - \text{отбрасываем}$$

$$P_{72} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$N_{72} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{72} = \frac{P_{72}}{N_{72}} = \frac{12}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{28} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{28} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$D_{28} = \frac{P_{28}}{N_{28}} = \frac{0}{17} = 0 < 1 - \text{отбрасываем}$$

$$P_{82} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$N_{82} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{82} = \frac{P_{82}}{N_{82}} = \frac{17}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{29} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{29} = 5 + 5 + 4 + 5 + 2 + 0 = 21$$

$$D_{29} = \frac{P_{29}}{N_{29}} = \frac{0}{21} = 0 < 1 - \text{отбрасываем}$$

$$P_{92} = 5 + 5 + 4 + 5 + 2 + 0 = 21$$

$$N_{92} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{92} = \frac{P_{92}}{N_{92}} = \frac{21}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{210} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{210} = 0 + 0 + 0 + 0 + 2 + 4 = 6$$

$$D_{210} = \frac{P_{210}}{N_{210}} = \frac{0}{6} = 0 < 1 - \text{отбрасываем}$$

$$P_{102} = 0 + 0 + 0 + 0 + 2 + 4 = 6$$

$$N_{102} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{102} = \frac{P_{102}}{N_{102}} = \frac{6}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{34} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$N_{34} = 0 + 0 + 0 + 5 + 0 + 4 = 9$$

$$D_{34} = \frac{P_{34}}{N_{34}} = \frac{11}{9} = 1,22 > 1 - \text{принимаем}$$

$$P_{43} = 0 + 0 + 0 + 5 + 0 + 4 = 9$$

$$N_{43} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$D_{43} = \frac{P_{43}}{N_{43}} = \frac{9}{11} = 0,8 < 1 - \text{отбрасываем}$$

$$P_{35} = 5 + 5 + 0 + 0 + 2 + 0 = 12$$

$$N_{35} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$D_{35} = \frac{P_{35}}{N_{35}} = \frac{12}{0} = \infty > 1 - \text{присваиваем}$$

$$P_{53} = 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$N_{53} = 5 + 5 + 0 + 0 + 2 + 0 = 12$$

$$D_{53} = \frac{P_{53}}{N_{53}} = \frac{0}{12} = 0 < 1 - \text{отбрасываем}$$

$$P_{36} = 5 + 0 + 0 + 0 + 0 + 4 = 9$$

$$N_{36} = 0 + 0 + 0 + 5 + 0 + 0 = 5$$

$$D_{36} = \frac{P_{36}}{N_{36}} = \frac{9}{5} = 1,8 > 1 - \text{принимаем}$$

$$P_{63} = 0 + 0 + 0 + 5 + 0 + 0 = 5$$

$$N_{63} = 5 + 0 + 0 + 0 + 0 + 4 = 9$$

$$D_{63} = \frac{P_{63}}{N_{63}} = \frac{5}{9} = 0,5 < 1 - \text{отбрасываем}$$

$$P_{37} = 0 + 5 + 4 + 0 + 0 + 4 = 13$$

$$N_{37} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$D_{37} = \frac{P_{37}}{N_{37}} = \frac{13}{12} = 1,08 > 1 - \text{принимаем}$$

$$P_{73} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$N_{73} = 0 + 5 + 4 + 0 + 0 + 4 = 13$$

$$D_{73} = \frac{P_{73}}{N_{73}} = \frac{12}{13} = 0,9 < 1 - \text{отбрасываем}$$

$$P_{38} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$



$$N_{38} = 0 + 0 + 0 + 5 + 0 + 0 = 5$$

$$D_{38} = \frac{P_{38}}{N_{38}} = \frac{8}{5} = 1,6 > 1 - \text{принимаем}$$

$$P_{83} = 0 + 0 + 0 + 5 + 0 + 0 = 5$$

$$N_{83} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$D_{83} = \frac{P_{83}}{N_{83}} = \frac{5}{8} = 0,625 < 1 - \text{отбрасываем}$$

$$P_{39} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{39} = 5 + 0 + 0 + 5 + 0 + 0 = 10$$

$$D_{39} = \frac{P_{39}}{N_{39}} = \frac{4}{10} = 0,4 < 1 - \text{отбрасываем}$$

$$P_{93} = 5 + 0 + 0 + 5 + 0 + 0 = 10$$

$$N_{93} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{93} = \frac{P_{93}}{N_{93}} = \frac{10}{4} = 2,5 > 1 - \text{принимаем}$$

$$P_{310} = 5 + 5 + 4 + 0 + 0 + 0 = 14$$

$$N_{310} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{310} = \frac{P_{310}}{N_{310}} = \frac{14}{4} = 3,5 > 1 - \text{принимаем}$$

$$P_{103} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{103} = 5 + 5 + 4 + 0 + 0 + 0 = 14$$

$$D_{103} = \frac{P_{103}}{N_{103}} = \frac{4}{14} = 0,3 < 1 - \text{отбрасываем}$$

$$P_{45} = 0 + 5 + 0 + 5 + 0 + 4 = 14$$

$$N_{45} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$D_{45} = \frac{P_{45}}{N_{45}} = \frac{14}{9} = 1,56 > 1 - \text{принимаем}$$

$$P_{54} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$N_{54} = 0 + 5 + 0 + 5 + 0 + 4 = 14$$

$$D_{54} = \frac{P_{54}}{N_{54}} = \frac{9}{14} = 0,64 < 1 - \text{отбрасываем}$$

$$P_{46} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{46} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$D_{46} = \frac{P_{46}}{N_{46}} = \frac{4}{11} = 0,36 < 1 - \text{отбрасываем}$$

$$P_{64} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$N_{64} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{64} = \frac{P_{64}}{N_{64}} = \frac{11}{4} = 2,75 > 1 - \text{принимаем}$$

$$P_{47} = 0 + 5 + 0 + 0 + 0 + 4 = 9$$

$$N_{47} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$D_{47} = \frac{P_{47}}{N_{47}} = \frac{9}{7} = 1,29 > 1 - \text{принимаем}$$

$$P_{74} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$N_{74} = 0 + 5 + 0 + 0 + 0 + 4 = 9$$

$$D_{74} = \frac{P_{74}}{N_{74}} = \frac{7}{9} = 0,7 < 1 - \text{отбрасываем}$$

$$P_{48} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{48} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$D_{48} = \frac{P_{48}}{N_{48}} = \frac{4}{7} = 0,57 < 1 - \text{отбрасываем}$$

$$P_{84} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$N_{84} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{84} = \frac{P_{84}}{N_{84}} = \frac{7}{4} = 1,75 > 1 - \text{принимаем}$$

$$P_{49} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{49} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$D_{49} = \frac{P_{49}}{N_{49}} = \frac{4}{11} = 0,36 < 1 - \text{отбрасываем}$$

$$P_{94} = 5 + 0 + 4 + 0 + 2 + 0 = 11$$

$$N_{94} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{94} = \frac{P_{94}}{N_{94}} = \frac{11}{4} = 2,75 > 1 - \text{принимаем}$$

$$P_{410} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$N_{410} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$D_{410} = \frac{P_{410}}{N_{410}} = \frac{10}{7} = 1,43 > 1 - \text{принимаем}$$

$$P_{104} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$N_{104} = 0 + 5 + 0 + 5 + 0 + 0 = 10$$

$$D_{104} = \frac{P_{104}}{N_{104}} = \frac{7}{10} = 0,7 < 1 - \text{отбрасываем}$$

$$P_{56} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{56} = 0 + 5 + 0 + 5 + 2 + 0 = 12$$

$$D_{56} = \frac{P_{56}}{N_{56}} = \frac{4}{12} = 0,3 < 1 - \text{отбрасываем}$$

$$P_{65} = 0 + 5 + 0 + 5 + 2 + 0 = 12$$

$$N_{65} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{65} = \frac{P_{65}}{N_{65}} = \frac{12}{4} = 3 > 1 - \text{принимаем}$$

$$P_{57} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$N_{57} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$D_{57} = \frac{P_{57}}{N_{57}} = \frac{8}{12} = 0,6 < 1 - \text{отбрасываем}$$

$$P_{75} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$N_{75} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$D_{75} = \frac{P_{75}}{N_{75}} = \frac{12}{8} = 1,5 > 1 - \text{принимаем}$$

$$P_{58} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$N_{58} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$D_{58} = \frac{P_{58}}{N_{58}} = \frac{8}{17} = 0,47 < 1 - \text{отбрасываем}$$

$$P_{85} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$N_{85} = 0 + 0 + 4 + 0 + 0 + 4 = 8$$

$$D_{85} = \frac{P_{85}}{N_{85}} = \frac{17}{8} = 2,12 > 1 - \text{принимаем}$$

$$P_{59} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{59} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$D_{59} = \frac{P_{59}}{N_{59}} = \frac{4}{17} = 0,23 < 1 - \text{отбрасываем}$$

$$P_{95} = 5 + 5 + 0 + 5 + 2 + 0 = 17$$

$$N_{95} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{95} = \frac{P_{95}}{N_{95}} = \frac{17}{4} = 4,25 > 1 - \text{принимаем}$$

$$P_{510} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$N_{510} = 0 + 0 + 0 + 0 + 2 + 4 = 6$$

$$D_{510} = \frac{P_{510}}{N_{510}} = \frac{4}{6} = 0,6 < 1 - \text{отбрасываем}$$

$$P_{105} = 0 + 0 + 0 + 0 + 2 + 4 = 6$$

$$N_{105} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$D_{105} = \frac{P_{105}}{N_{105}} = \frac{6}{4} = 1,5 > 1 - \text{принимаем}$$

$$P_{67} = 0 + 5 + 4 + 0 + 0 + 4 = 13$$

$$N_{67} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$D_{67} = \frac{P_{67}}{N_{67}} = \frac{13}{7} = 1,86 > 1 - \text{принимаем}$$

$$P_{76} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$N_{76} = 0 + 5 + 4 + 0 + 0 + 4 = 13$$

$$D_{76} = \frac{P_{76}}{N_{76}} = \frac{7}{13} = 0,54 < 1 - \text{отбрасываем}$$

$$P_{68} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$N_{68} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$D_{68} = \frac{P_{68}}{N_{68}} = \frac{4}{5} = 0,8 < 1 - \text{отбрасываем}$$

$$P_{86} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$N_{86} = 0 + 0 + 4 + 0 + 0 + 0 = 4$$

$$D_{86} = \frac{P_{86}}{N_{86}} = \frac{5}{4} = 1,25 > 1 - \text{принимаем}$$

$$P_{69} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{69} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$D_{69} = \frac{P_{69}}{N_{69}} = \frac{4}{5} = 0,8 < 1 - \text{отбрасываем}$$

$$P_{96} = 5 + 0 + 0 + 0 + 0 + 0 = 5$$

$$N_{96} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{96} = \frac{P_{96}}{N_{96}} = \frac{5}{4} = 1,25 > 1 - \text{принимаем}$$

$$P_{610} = 0 + 5 + 4 + 5 + 0 + 0 = 14$$

$$N_{610} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{610} = \frac{P_{610}}{N_{610}} = \frac{14}{4} = 3,5 > 1 - \text{принимаем}$$

$$P_{106} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{106} = 0 + 5 + 4 + 5 + 0 + 0 = 14$$

$$D_{106} = \frac{P_{106}}{N_{106}} = \frac{4}{14} = 0,29 < 1 - \text{отбрасываем}$$

$$P_{78} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$N_{78} = 0 + 5 + 0 + 0 + 0 + 4 = 9$$

$$D_{78} = \frac{P_{78}}{N_{78}} = \frac{7}{9} = 0,7 < 1 - \text{отбрасываем}$$

$$P_{87} = 0 + 5 + 0 + 0 + 0 + 4 = 9$$

$$N_{87} = 5 + 0 + 0 + 0 + 2 + 0 = 7$$

$$D_{87} = \frac{P_{87}}{N_{87}} = \frac{9}{7} = 1,29 > 1 - \text{принимаем}$$

$$P_{79} = 0 + 0 + 0 + 0 + 2 + 0 = 2$$

$$N_{79} = 0 + 5 + 4 + 0 + 0 + 0 = 9$$

$$D_{79} = \frac{P_{79}}{N_{79}} = \frac{2}{9} = 0,2 < 1 - \text{отбрасываем}$$

$$P_{97} = 0 + 5 + 4 + 0 + 0 + 0 = 9$$

$$N_{97} = 0 + 0 + 0 + 0 + 2 + 0 = 2$$

$$D_{97} = \frac{P_{97}}{N_{97}} = \frac{9}{2} = 4,5 > 1 - \text{принимаем}$$

$$P_{710} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$N_{710} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{710} = \frac{P_{710}}{N_{710}} = \frac{12}{4} = 3 > 1 - \text{принимаем}$$

$$P_{107} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{107} = 5 + 0 + 0 + 5 + 2 + 0 = 12$$

$$D_{107} = \frac{P_{107}}{N_{107}} = \frac{4}{12} = 0,3 < 1 - \text{отбрасываем}$$

$$P_{89} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{89} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$D_{89} = \frac{P_{89}}{N_{89}} = \frac{4}{9} = 0,4 < 1 - \text{отбрасываем}$$

$$P_{98} = 5 + 0 + 4 + 0 + 0 + 0 = 9$$

$$N_{98} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{98} = \frac{P_{98}}{N_{98}} = \frac{9}{4} = 2,25 > 1 - \text{принимаем}$$

$$P_{810} = 5 + 5 + 0 + 5 + 0 + 0 = 15$$

$$N_{810} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{810} = \frac{P_{810}}{N_{810}} = \frac{15}{4} = 3,75 > 1 - \text{принимаем}$$

$$P_{108} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{108} = 5 + 5 + 0 + 5 + 0 + 0 = 15$$

$$D_{108} = \frac{P_{108}}{N_{108}} = \frac{4}{15} = 0,26 < 1 - \text{отбрасываем}$$

$$P_{910} = 5 + 5 + 4 + 5 + 0 + 0 = 19$$

$$N_{910} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$D_{910} = \frac{P_{910}}{N_{910}} = \frac{19}{4} = 4,75 > 1 - \text{принимаем}$$

$$P_{109} = 0 + 0 + 0 + 0 + 0 + 4 = 4$$

$$N_{109} = 5 + 5 + 4 + 5 + 0 + 0 = 19$$

$$D_{109} = \frac{P_{109}}{N_{109}} = \frac{4}{19} = 0,21 < 1 - \text{отбрасываем}$$

Составляем матрицу, внося вычисленные (и принятые) значения  $D$ . Матрица имеет смысл предпочтений проектов между собой. Для нашего примера матрица представлена в Таблице 2.2.2.

Таблица 2.2.2 - Полная матрица предпочтений проектов, составленная методом Электра.

	1	2	3	4	5	6	7	8	9	10
1	X	-	-	1,1	-	-	-	-	-	-
2	$\infty$	X	$\infty$	-	-	$\infty$	$\infty$	$\infty$	-	$\infty$
3	1,25	-	X	1,22	$\infty$	1,8	1,08	1,6	-	3,5
4	-	2,8	-	X	1,56	-	1,29	-	-	1,43
5	-	-	-	-	X	-	-	-	-	-
6	1,11	-	-	2,75	3	X	1,86	-	-	3,5
7	1,5	-	-	-	1,5	-	X	-	-	3
8	1,25	-	-	1,75	2,12	1,25	1,29	X	-	3,75
9	3,75	-	2,5	2,75	4,25	1,25	4,5	2,25	X	3,75
10	$\infty$	-	-	-	1,5	-	-	-	-	X

Результат выполнения программы представлен на Рисунке 2.2.2.1.

```

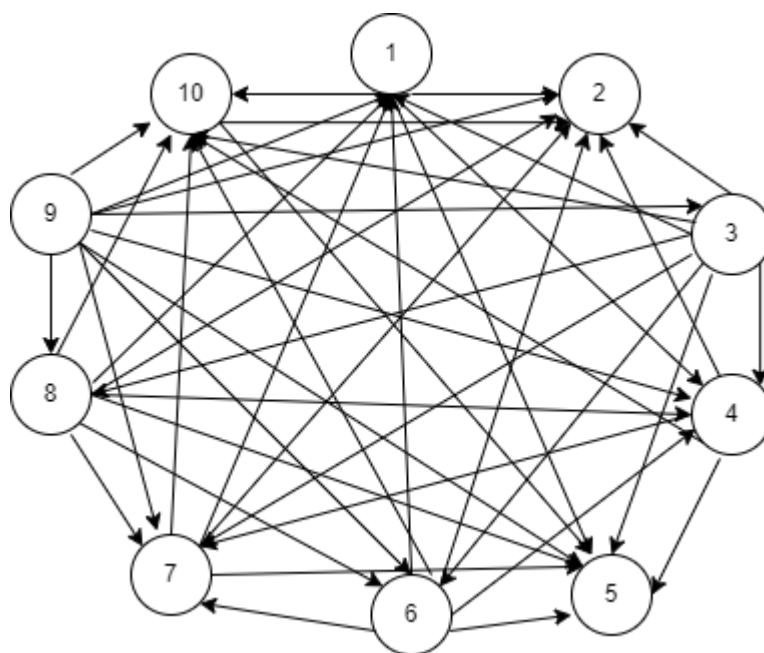
>> 1
СПИСОК АЛЬТЕРНАТИВ:
№   Price   Per   Year   Garant   Weight   Diag
1    5        5     4      5        4       12
2   10        5     8      5        8        2
3    5       10     4      5        4        8
4   15       10     8     10        8       12
5   10        5     4      5        8        8
6   10       10     4     10        4        4
7    1        5     8     10        2        2
8    5       10     8     10        4        4
9    1       10     4     10        4        2
10  10        5     8      5        4       12

МАТРИЦА ПРЕДПОЧТЕНИЙ:
   1    2    3    4    5    6    7    8    9   10
1  0    0    0    1,1  0    0    0    0    0    0
2  0    0    0    0    0    0    0    0    0    0
3  1,25  0    0    1,22  0    1,8  1,08  1,6  0    3,5
4  0    2,8  0    0    1,56  0    1,29  0    0    1,43
5  0    0    0    0    0    0    0    0    0    0
6  1,11  0    0    2,75  3    0    1,86  0    0    3,5
7  1,5    0    0    0    1,5  0    0    0    0    3
8  1,25  0    0    1,75  2,12  1,25  1,29  0    0    3,75
9  3,75  0    2,5  2,75  4,25  1,25  4,5  2,25  0    4,75
10 0    0    0    0    1,5  0    0    0    0    0
  
```

Рисунок 2.2.2.1 – Реализация матрицы предпочтений

По матрице строится граф предпочтений, который представлен на Рисунке 2.2.2.2.





**Рисунок 2.2.2.2 - Вид графа предпочтений**

Программная реализация графа предпочтений представлена на Рисунке 2.2.2.3.

```

УРОВНИ ГРАФА ПРЕДПОЧТЕНИЙ:

Уровень: 1; 1 3 4 5 7 8 10
[ 1 ] 1 0 1 1 1 0 1 1 0 1
[ 3 ] 1 0 1 1 1 2 1 1 0 2
[ 4 ] 1 2 1 1 1 2 1 1 0 2
[ 5 ] 1 2 1 1 1 2 1 1 0 2
[ 7 ] 1 2 1 1 1 2 1 1 0 2
[ 8 ] 1 2 1 1 2 2 1 1 0 2
[ 10 ] 1 2 1 1 2 2 1 1 0 2
Уровень: 2; 2 5 6 10
[ 2 ] 1 2 1 1 2 2 1 1 0 2
[ 5 ] 1 2 1 1 2 2 1 1 0 2
[ 6 ] 1 2 1 3 3 2 3 1 0 3
[ 10 ] 1 2 1 3 3 2 3 1 0 3
Уровень: 3; 4 5 7 10
[ 4 ] 1 4 1 3 3 2 3 1 0 3
[ 5 ] 1 4 1 3 3 2 3 1 0 3
[ 7 ] 1 4 1 3 3 2 3 1 0 4
[ 10 ] 1 4 1 3 3 2 3 1 0 4
Уровень: 4; 2 10
[ 2 ] 1 4 1 3 3 2 3 1 0 4
[ 10 ] 1 4 1 3 3 2 3 1 0 4
Уровень: 5;

```

**Рисунок 2.2.2.3 – Программная реализация графа предпочтений**

## 2.3 Выполнение второго этапа

Этап исследования, на котором построенные индексы используются для ранжирования (или классификации) заданного множества альтернатив.

Назначим порог отбора предпочтений  $C = 1,76$  (это соответствует тому, что мы попробуем учесть только более сильные связи в графе, не отвлекаясь на малозначимые расхождения в проектах). Таким образом, матрица разрежается. В ней остаются только самые сильные связи, которые представлены в Таблице 2.3.

Таблица 2.3 - Матрица предпочтений проектов, при пороге  $C=1,76$

	1	2	3	4	5	6	7	8	9	10
1	X	-	-	-	-	-	-	-	-	-
2	$\infty$	X	$\infty$	-	-	$\infty$	$\infty$	$\infty$	-	$\infty$
3	-	-	X	-	-	1,8	-	-	-	3,5
4	-	2,8	-	X	-	-	-	-	-	-
5	-	-	-	-	X	-	-	-	-	-
6	-	-	-	2,75	3	X	1,86	-	-	3,5
7	-	-	-	-	-	-	X	-	-	3
8	-	-	-	-	2,12	-	-	X	-	3,75
9	-	-	2,5	2,75	4,25	-	4,5	2,25	X	4,75
10	$\infty$	-	-	-	-	-	-	-	-	X

Результат выполнения программы представлен на Рисунке 2.3.1

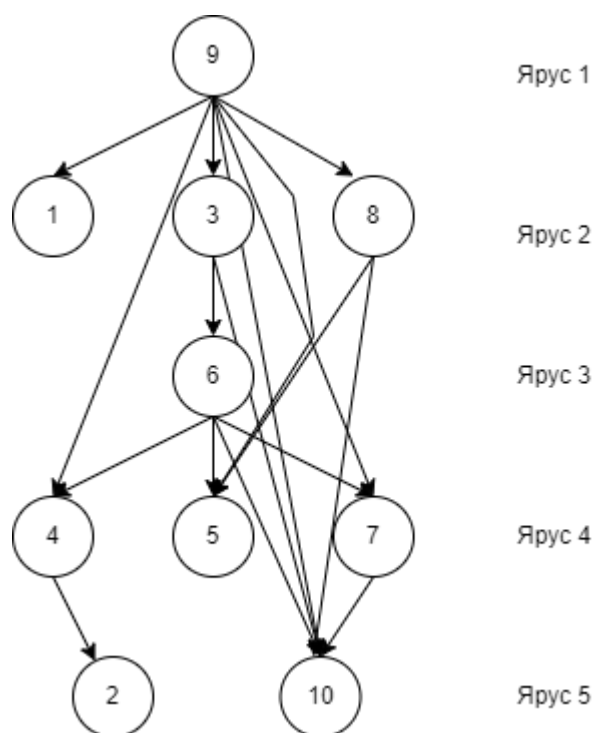
МАТРИЦА ПРЕДПОЧТЕНИЙ С ПОРОГОМ ОТБОРА ПРЕДПОЧТЕНИЙ

Введите порог отбора предпочтений: 1,76

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	1,8	0	0	0	3,5
4	0	2,8	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	2,75	3	0	1,86	0	0	3,5
7	0	0	0	0	0	0	0	0	0	3
8	0	0	0	0	2,12	0	0	0	0	3,75
9	3,75	0	2,5	2,75	4,25	0	4,5	2,25	0	4,75
10	0	0	0	0	0	0	0	0	0	0

Рисунок 2.3.1 – Реализация матрицы предпочтений с порогом отбора

Снова строим граф по матрице предпочтений, но только уже с учетом порога. Граф представлен на Рисунке 2.3.2.



**Рисунок 2.3.2 - Вид графа предпочтений для случая порога принятия решений**

Программная реализация графа предпочтений представлена на Рисунке 2.3.3.

```

Уровни графа:
Уровень 1: 9
Уровень 2: 1 3 8
Уровень 3: 6
Уровень 4: 4 5 7
Уровень 5: 2 10
  
```

**Рисунок 2.3.3 – Программная реализация графа предпочтений**

Код реализации представлен в Приложении Б.

Решение говорит нам о том, что лучший проект - 9. На второе, третье и четвертое место делят проекты – 1, 3, 8, пятое место - проект 6, на шестое, седьмое, восьмое место делят проекты – 4, 5, 7, а девятое и десятое место делят – 2 и 10 проекты.

### **3 ОПИСАНИЕ АЛГОРИТМА МЕТОДА МАИ**

Метод анализа иерархий (МАИ) является замкнутой логической конструкцией, которая обеспечивает с помощью простых и хорошо обоснованных правил, решение задач МКО, включающих как качественные, так и количественные факторы, причем количественные факторы могут иметь разную размерность. Метод основан на декомпозиции задачи и представлении ее в виде иерархической структуры, что позволяет включить в иерархию все имеющиеся у лица, принимающего решение знания по решаемой проблеме и последующей обработке суждений. В результате может быть выявлена относительная степень взаимодействия элементов в иерархии, которые затем выражаются численно. МАИ включает процедуры синтеза множественных суждений, получения приоритетности критериев и нахождения альтернативных решений [1].

Весь процесс решения подвергается проверке и переосмыслению на каждом этапе, что позволяет проводить оценку качества полученного решения. Решение многокритериального выбора основано на трех основных этапах:

- Первый этап – представление системы критериев (целей) в виде иерархической структуры.
- Второй этап – оценки приоритетов (весов) критериев с учётом их места в иерархии относительной важности.
- Третий этап – определение лучшей альтернативы по значениям её характеристик и важности критериев.

#### **3.1 Постановка задачи**

Выбрать телевизор с использованием метода анализа иерархий в программе MPRIORITY 1.0.

## 3.2 Выполнение первого этапа

Создаем новый проект (Рисунок 3.2.1).

Создание нового проекта

Параметры нового проекта

Цель проекта	Число уровней в иерархии:
ВЫБОР TV	3
Комментарии	Максимальное число элементов:
Задача "Выбрать телевизор"	5

Да Отменить

Рисунок 3.2.1 – Создание нового проекта

Редактируем второй уровень «критерии» (Рисунок 3.2.2 – Рисунок 3.2.6).

Редактирование элемента

Редактирование текущего элемента

Наименование элемента: КРИТЕР\_1

Комментарии

Цена

Да Отмена

Рисунок 3.2.2 – Редактирование второго уровня «КРИТЕР\_1»

Редактирование элемента

Редактирование текущего элемента

Наименование элемента: КРИТЕР\_2

Комментарии

Год

Да Отмена

Рисунок 3.2.3 – Редактирование второго уровня «КРИТЕР\_2»

#### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: КРИТЕР\_3

Комментарии

Гарантия

Да Отмена

Рисунок 3.2.4 – Редактирование второго уровня «КРИТЕР\_3»

#### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: КРИТЕР\_4

Комментарии

Вес

Да Отмена

Рисунок 3.2.5 – Редактирование второго уровня «КРИТЕР\_4»

#### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: КРИТЕР\_5

Комментарии

Диагональ

Да Отмена

Рисунок 3.2.6 – Редактирование второго уровня «КРИТЕР\_5»

Редактируем третий уровень «альтернативы» (Рисунок 3.2.7 – Рисунок 3.2.11).

### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: АЛЬТЕРН\_1

Комментарии

LG QLED

Да Отмена

Рисунок 3.2.7 – Редактирование третьего уровня «АЛЬТЕРН\_1»

### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: АЛЬТЕРН\_2

Комментарии

SAMSUNG NEO QLED

Да Отмена

Рисунок 3.2.8 – Редактирование третьего уровня «АЛЬТЕРН\_2»

### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: АЛЬТЕРН\_3

Комментарии

SAMSUNG UHD QLED

Да Отмена

Рисунок 3.2.9 – Редактирование третьего уровня «АЛЬТЕРН\_3»

### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: АЛЬТЕРН\_4

Комментарии

SONY UHD LED

Да Отмена

Рисунок 3.2.10 – Редактирование третьего уровня «АЛЬТЕРН\_4»

### Редактирование элемента

Редактирование текущего элемента

Наименование элемента: АЛЬТЕРН\_5

Комментарии

LG LED

Да Отмена

Рисунок 3.2.11 – Редактирование третьего уровня «АЛЬТЕРН\_5»

Получаем граф иерархий (Рисунок 3.2.12).

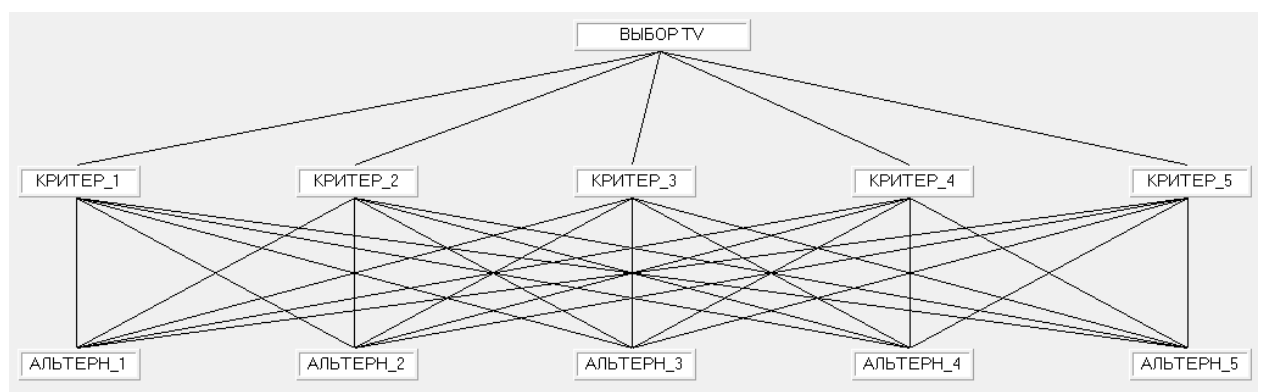


Рисунок 3.2.12 – Граф иерархий



### 3.3 Выполнение второго этапа

На этом этапе мы выбираем первый уровень и строим матрицу парных сравнений. При работе с матрицей используется качественная шкала, которая позволяет провести субъективно парные сравнения. Она представляет собой шкалу от 1 до 9, с распределением интенсивности относительной важности.

Из этого, мы получаем обратно симметричную матрицу для парного сравнения критериев (Рисунок 3.3.1).

Работа эксперта

Производим попарные сравнения относительно объекта  
ВЫБОР TV

		1.	2.	3.	4.	5.	Приоритет
1.	КРИТЕР_1	1	3	3	5	7	0,4655
2.	КРИТЕР_2	1/3	1	1	3	5	0,2032
3.	КРИТЕР_3	1/3	1	1	3	5	0,2032
4.	КРИТЕР_4	1/5	1/3	1/3	1	3	0,0857
5.	КРИТЕР_5	1/7	1/5	1/5	1/3	1	0,0421

СЗ: 5,1263    Применить  
ИС: 0,0315    Закреть  
ОС: 0,0282    Отмена    Исследовать

Рисунок 3.3.1 - Обратно симметричная матрица для критериев

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.

Исследуем матрицу (Рисунок 3.3.2).

Улучшение согласованности исходных данных

		1.	2.	3.	4.	5.	Сумма
1.	КРИТЕР_1	0	0,7	0,7	0,43	4,05	5,9
2.	КРИТЕР_2	3,43	0	0	0,62	0,17	4,23
3.	КРИТЕР_3	3,43	0	0	0,62	0,17	4,23
4.	КРИТЕР_4	5,18	3,42	3,42	0	0,96	12,99
5.	КРИТЕР_5	7,09	5,2	5,2	3,49	0	20,99

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закреть

Рисунок 3.3.2 – Матрица исследована

КРИТЕР\_1 – Цена (Рисунок 3.3.3).

Работа эксперта

Производим попарные сравнения относительно объекта  
КРИТЕР\_1

		1.	2.	3.	4.	5.	Приоритет
1.	АЛЬТЕРН_1	1	7	3	5	1/3	0,2667
2.	АЛЬТЕРН_2	1/7	1	1/5	1/3	1/9	0,0332
3.	АЛЬТЕРН_3	1/3	5	1	1/3	1/5	0,0844
4.	АЛЬТЕРН_4	1/5	3	3	1	1/7	0,0998
5.	АЛЬТЕРН_5	3	9	5	7	1	0,5156

СЗ: 5,4936  
ИС: 0,1234  
ОС: 0,1101

Применить  
Закреть  
Отмена

Исследовать

Рисунок 3.3.3 - Обратнo симметричная матрица для КРИТЕР\_1

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.

Исследуем матрицу (Рисунок 3.3.4).

Улучшение согласованности исходных данных

		1.	2.	3.	4.	5.	Сумма
1.	АЛЬТЕРН_1	0	1,01	0,15	2,32	3,51	7,02
2.	АЛЬТЕРН_2	7,12	0	5,39	3,33	9,06	24,91
3.	АЛЬТЕРН_3	3,31	2,46	0	3,84	5,16	14,78
4.	АЛЬТЕРН_4	5,37	0	1,81	0	7,19	14,38
5.	АЛЬТЕРН_5	1,06	6,49	1,1	1,83	0	10,5

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закреть

Рисунок 3.3.4 – Матрица исследована

КРИТЕР\_2 – Год (Рисунок 3.3.5).

Работа эксперта

Производим попарные сравнения относительно объекта  
КРИТЕР\_2

		1.	2.	3.	4.	5.	Приоритет
1.	АЛЬТЕРН_1	1	3	5	7	5	0,47
2.	АЛЬТЕРН_2	1/3	1	5	7	5	0,3028
3.	АЛЬТЕРН_3	1/5	1/5	1	5	1	0,0973
4.	АЛЬТЕРН_4	1/7	1/7	1/5	1	1/5	0,0323
5.	АЛЬТЕРН_5	1/5	1/5	1	5	1	0,0973

СЗ: 5,4161  
ИС: 0,104  
ОС: 0,0928

Применить  
Закреть  
Отмена

Исследовать

Рисунок 3.3.5 - Обратнo симметричная матрица для КРИТЕР\_2

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.

Исследуем матрицу (Рисунок 3.3.6).

Улучшение согласованности исходных данных

		1.	2.	3.	4.	5.	Сумма
1.	АЛЬТЕРН_1	0	1,44	0,17	7,5	0,17	9,3
2.	АЛЬТЕРН_2	3,64	0	1,88	2,34	1,88	9,77
3.	АЛЬТЕРН_3	5,2	5,32	0	1,99	0	12,52
4.	АЛЬТЕРН_4	7,06	7,1	5,33	0	5,33	24,84
5.	АЛЬТЕРН_5	5,2	5,32	0	1,99	0	12,52

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закрывать

Рисунок 3.3.6 – Матрица исследована

КРИТЕР\_3 – Гарантия (Рисунок 3.3.7).

Работа эксперта

Производим попарные сравнения относительно объекта

КРИТЕР\_3

		1.	2.	3.	4.	5.	Приоритет
1.	АЛЬТЕРН_1	1	3	1/5	3	5	0,1963
2.	АЛЬТЕРН_2	1/3	1	1/7	1/3	3	0,0688
3.	АЛЬТЕРН_3	5	7	1	7	9	0,5899
4.	АЛЬТЕРН_4	1/3	3	1/7	1	3	0,1067
5.	АЛЬТЕРН_5	1/5	1/3	1/9	1/3	1	0,038

СЗ: 5,326    Применить

ИС: 0,0815    Закрывать

ОС: 0,0727    Отмена

Исследовать

Рисунок 3.3.7 - Обратно симметричная матрица для КРИТЕР\_3

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.

Исследуем матрицу (Рисунок 3.3.8).

		1.	2.	3.	4.	5.	Сумма
1.	АЛЬТЕРН_1	0	0,14	5,33	1,16	0,15	6,79
2.	АЛЬТЕРН_2	3,35	0	7,11	3,64	1,19	15,3
3.	АЛЬТЕРН_3	1,99	1,57	0	1,47	6,49	11,53
4.	АЛЬТЕРН_4	3,54	1,44	7,18	0	0,19	12,36
5.	АЛЬТЕРН_5	5,19	3,55	9,06	3,35	0	21,16

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закреть

**Рисунок 3.3.8 – Матрица исследована**  
КРИТЕР\_4 – Вес (Рисунок 3.3.9).

Производим попарные сравнения относительно объекта

КРИТЕР\_4

		1.	2.	3.	4.	5.	Приоритет
1.	АЛЬТЕРН_1	1	3	1/5	3	7	0,2051
2.	АЛЬТЕРН_2	1/3	1	1/7	3	5	0,1155
3.	АЛЬТЕРН_3	5	7	1	7	9	0,5764
4.	АЛЬТЕРН_4	1/3	1/3	1/7	1	5	0,0744
5.	АЛЬТЕРН_5	1/7	1/5	1/9	1/5	1	0,0283

СЗ: 5,4761    Применить

ИС: 0,119    Закреть

ОС: 0,1062    Отмена

Исследовать

**Рисунок 3.3.9 - Обратно симметричная матрица для КРИТЕР\_4**  
Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.  
Исследуем матрицу (Рисунок 3.3.10).

		1.	2.	3.	4.	5.	Сумма
1.	АЛЬТЕРН_1	0	1,22	5,35	0,24	0,23	7,06
2.	АЛЬТЕРН_2	3,56	0	7,2	1,44	0,92	13,13
3.	АЛЬТЕРН_3	2,19	2,01	0	0,74	11,33	16,27
4.	АЛЬТЕРН_4	3,36	3,64	7,12	0	2,37	16,51
5.	АЛЬТЕРН_5	7,13	5,24	9,04	5,38	0	26,81

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закреть

**Рисунок 3.3.10 – Матрица исследована**

КРИТЕР\_5 – Диагональ (Рисунок 3.3.11).

Работа эксперта

Производим попарные сравнения относительно объекта  
КРИТЕР\_5

		1.	2.	3.	4.	5.	Приоритет
1.	АЛЬТЕРН_1	1	5	7	3	1/3	0,2774
2.	АЛЬТЕРН_2	1/5	1	5	1/3	1/5	0,0792
3.	АЛЬТЕРН_3	1/7	1/5	1	1/7	1/9	0,0292
4.	АЛЬТЕРН_4	1/3	3	7	1	1/3	0,1614
5.	АЛЬТЕРН_5	3	5	9	3	1	0,4526

СЗ: 5,3683    Применить  
 ИС: 0,092    Закрывать  
 ОС: 0,0822    Отмена    Исследовать

Рисунок 3.3.11 - Обратно симметричная матрица для КРИТЕР\_5

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «выбор смесителя» согласована.

Исследуем матрицу (Рисунок 3.3.12).

Улучшение согласованности исходных данных

		1.	2.	3.	4.	5.	Сумма
1.	АЛЬТЕРН_1	0	1,5	2,49	1,28	3,61	8,88
2.	АЛЬТЕРН_2	5,28	0	2,28	3,49	5,17	16,23
3.	АЛЬТЕРН_3	7,1	5,36	0	7,18	9,06	28,71
4.	АЛЬТЕРН_4	3,58	0,96	1,47	0	3,35	9,37
5.	АЛЬТЕРН_5	1,36	0,71	6,49	0,19	0	8,76

Список нарушений условия транзитивности

Информация о нарушении транзитивности -- отсутствует!

Закрывать

Рисунок 3.3.12 – Матрица исследована

### 3.4 Выполнение третьего этапа

Вывод результата (Рисунок 3.4.1).



**Рисунок 3.4.1 – Итоговый результат**

## **3.5 Ручной счет**

### **3.5.1 Первый этап**

Этот этап предусматривает представление проблемы в виде иерархии или сети. В простейшем случае, иерархия строится, начиная с цели, которая помещается в вершину иерархии. Через промежуточные уровни, на которых располагаются критерии и от которых зависят последующие уровни, к самому низкому уровню, который содержит перечень альтернатив.

#### **Критерии:**

- К 1 - цена;
- К 2 - год;
- К 3 - гарантия;
- К 4 – вес;
- К 5 – диагональ.

#### **Альтернативы:**

- А 1 – LG QLED;
- А 2 – SAMSUNG NEO QLED;
- А 3 – SAMSUNG UHD QLED;

- А 4 – SONY UHD LED;
- А 5 – LG LED;

### 3.5.2 Второй этап

После иерархического представления задачи необходимо установить приоритеты критериев и оценить каждую из альтернатив по критериям, определив наиболее важную их них.

В методе анализа иерархий элементы сравниваются попарно по отношению к их влиянию на общую для них характеристику. Парные сравнения приводят к записи характеристик сравнений в виде квадратной таблицы чисел, которая называется матрицей.

#### Синтез приоритетов

После построения иерархии и определения величин парных субъективных суждений следует этап, на котором иерархическая декомпозиция и относительные суждения объединяются для получения осмысленного решения многокритериальной задачи принятия решений.

Из групп парных сравнений формируется набор локальных критериев, которые выражают относительное влияние элементов на элемент, расположенный на уровне выше [2].

Составим обратно симметричную матрицу для парного сравнения критериев (Таблица 3.5.2.1).

Таблица 3.5.2.1 - Обратно симметричная матрицу для парного сравнения критериев

Цель	К 1	К 2	К 3	К 4	К 5	$V_i$	$W_{2i}$
К 1	1	3	3	5	7	3,15982	0,465581
К 2	1/3	1	1	3	5	1,37973	0,203295
К 3	1/3	1	1	3	5	1,37973	0,203295
К 4	1/5	1/3	1/3	1	3	0,581811	0,0857265
К 5	1/7	1/5	1/5	1/3	1	0,285738	0,0421018
$\Sigma V_i$						6,786829	

Для определения относительной ценности каждого элемента необходимо найти геометрическое среднее и с этой целью перемножить 5-ть элементов каждой строки и из полученного результата извлечь корни 5-й степени (размерность матрицы  $n=5$ ).

$$V_1 = (1 * 3 * 3 * 5 * 7)^{\frac{1}{5}} = 3,15982$$

$$V_2 = \left(\frac{1}{3} * 1 * 1 * 3 * 5\right)^{\frac{1}{5}} = 1,37973$$

$$V_3 = \left(\frac{1}{3} * 1 * 1 * 3 * 5\right)^{\frac{1}{5}} = 1,37973$$

$$V_4 = \left(\frac{1}{5} * \frac{1}{3} * \frac{1}{3} * 1 * 3\right)^{\frac{1}{5}} = 0,581811$$

$$V_5 = \left(\frac{1}{7} * \frac{1}{5} * \frac{1}{5} * \frac{1}{3} * 1\right)^{\frac{1}{5}} = 0,285738$$

Нормирующий коэффициент  $\sum V_i$ :

$$\begin{aligned} \sum V_i &= V_1 + V_2 + V_3 + V_4 + V_5 = \\ &= 3,15982 + 1,37973 + 1,37973 + 0,581811 + 0,285738 = \\ &= 6,786829 \end{aligned}$$

Важность приоритетов:

$$W_{21} = \frac{3,15982}{6,786829} = 0,465581$$

$$W_{22} = \frac{1,37973}{6,786829} = 0,203295$$

$$W_{23} = \frac{1,37973}{6,786829} = 0,203295$$

$$W_{24} = \frac{0,581811}{6,786829} = 0,0857265$$

$$W_{25} = \frac{0,285738}{6,786829} = 0,0421018$$



Вектор приоритетов:

$$W_{2i} = (0,47; 0,2; 0,2; 0,09; 0,04)$$

### КРИТЕР\_1 – Цена

Составим обратно симметричную матрицу для КРИТЕР\_1 (Таблица 3.5.2.2).

Таблица 3.5.2.2 - Обратно симметричная матрица для КРИТЕР\_1

К1	A1	A2	A3	A4	A5	V <sub>K1Y</sub>	W <sub>зK1Y</sub>
A1	1	7	3	5	1/3	2,03617	0,255847
A2	1/7	1	1/5	1/3	1/9	0,254047	0,0319212
A3	1/3	5	1	1/3	1/5	0,644394	0,0809687
A4	1/5	3	3	1	1/7	0,76214	0,0957635
A5	3	9	5	7	1	3,93628	0,494597
$\sum V_{K1Y}$						7,958561	

Относительная ценность:

$$V_{K11} = \left(1 * 7 * 3 * 5 * \frac{1}{3}\right)^{\frac{1}{5}} = 2,03617$$

$$V_{K12} = \left(\frac{1}{7} * 1 * \frac{1}{5} * \frac{1}{3} * \frac{1}{9}\right)^{\frac{1}{5}} = 0,254047$$

$$V_{K13} = \left(\frac{1}{3} * 5 * 1 * \frac{1}{3} * \frac{1}{5}\right)^{\frac{1}{5}} = 0,644394$$

$$V_{K14} = \left(\frac{1}{5} * 3 * 3 * 1 * \frac{1}{7}\right)^{\frac{1}{5}} = 0,76214$$

$$V_{K15} = (3 * 9 * 5 * 7 * 1)^{\frac{1}{5}} = 3,93628$$

Нормирующий коэффициент  $\sum V_{K1Y}$ :

$$\begin{aligned} \sum V_{K1Y} &= V_{K11} + V_{K12} + V_{K13} + V_{K14} + V_{K15} = \\ &= 2,03617 + 0,254047 + 0,644394 + 0,76214 + 3,93628 = \\ &= 7,958561 \end{aligned}$$

Важность приоритетов:

$$W_{3K11} = \frac{2,03617}{7,958561} = 0,255847$$

$$W_{3K12} = \frac{0,254047}{7,958561} = 0,0319212$$

$$W_{3K13} = \frac{0,644394}{7,958561} = 0,0809687$$

$$W_{3K14} = \frac{0,76214}{7,958561} = 0,0957635$$

$$W_{3K15} = \frac{3,93628}{7,958561} = 0,494597$$

Вектор приоритетов:

$$W_{3K1Y} = (0,26; 0,03; 0,08; 0,1; 0,5)$$

## КРИТЕР\_2 – Год

Составим обратно симметричную матрицу для КРИТЕР\_2 (Таблица 3.5.2.3).

Таблица 3.5.2.3 - Обратно симметричная матрица для КРИТЕР\_2

К2	A1	A2	A3	A4	A5	V <sub>K2Y</sub>	W <sub>3K2Y</sub>
A1	1	3	5	7	5	3,49971	0,470034
A2	1/3	1	5	7	5	2,25519	0,302887
A3	1/5	1/5	1	5	1	0,72478	0,0973427
A4	1/7	1/7	1/5	1	1/5	0,241197	0,0323943
A5	1/5	1/5	1	5	1	0,72478	0,0973427
$\sum V_{K2Y}$						7,445657	

Относительная ценность:

$$V_{K21} = (1 * 3 * 5 * 7 * 5)^{\frac{1}{5}} = 3,49971$$

$$V_{K22} = \left(\frac{1}{3} * 1 * 5 * 7 * 5\right)^{\frac{1}{5}} = 2,25519$$

$$V_{K23} = \left(\frac{1}{5} * \frac{1}{5} * 1 * 5 * 1\right)^{\frac{1}{5}} = 0,72478$$

$$V_{K24} = \left( \frac{1}{7} * \frac{1}{7} * \frac{1}{5} * 1 * \frac{1}{5} \right)^{\frac{1}{5}} = 0,241197$$

$$V_{K25} = \left( \frac{1}{5} * \frac{1}{5} * 1 * 5 * 1 \right)^{\frac{1}{5}} = 0,72478$$

Нормирующий коэффициент  $\sum V_{K2Y}$ :

$$\begin{aligned} \sum V_{K2Y} &= V_{K21} + V_{K22} + V_{K23} + V_{K24} + V_{K25} = \\ &= 3,49971 + 2,25519 + 0,72478 + 0,241197 + 0,72478 = \\ &= 7,445657 \end{aligned}$$

Важность приоритетов:

$$W_{3K21} = \frac{3,49971}{7,445657} = 0,470034$$

$$W_{3K22} = \frac{2,25519}{7,445657} = 0,302887$$

$$W_{3K23} = \frac{0,72478}{7,445657} = 0,0973427$$

$$W_{3K24} = \frac{0,241197}{7,445657} = 0,0323943$$

$$W_{3K25} = \frac{0,72478}{7,445657} = 0,0973427$$

Вектор приоритетов:

$$W_{3K2Y} = (0,47; 0,303; 0,97; 0,032; 0,97)$$

### **КРИТЕР\_3 – Гарантия**

Составим обратно симметричную матрицу для КРИТЕР\_3 (Таблица 3.5.2.4).

*Таблица 3.5.2.4 - Обратно симметричная матрица для КРИТЕР\_3*

<b>К3</b>	<b>A1</b>	<b>A2</b>	<b>A3</b>	<b>A4</b>	<b>A5</b>	<b>V<sub>K3Y</sub></b>	<b>W<sub>3K3Y</sub></b>
-----------	-----------	-----------	-----------	-----------	-----------	------------------------	-------------------------

Продолжение Таблицы 3.5.2.4

<b>A1</b>	1	3	1/5	3	5	1,55185	0,196336
<b>A2</b>	1/3	1	1/7	1/3	3	0,543946	0,0688186
<b>A3</b>	5	7	1	7	9	4,66318	0,589973
<b>A4</b>	1/3	3	1/7	1	3	0,844121	0,106796
<b>A5</b>	1/5	1/3	1/9	1/5	1	0,30096	0,0380766
<b>V<sub>K35</sub></b>						7,904057	

Относительная ценность:

$$V_{K31} = \left(1 * 3 * \frac{1}{5} * 3 * 5\right)^{\frac{1}{5}} = 1,55185$$

$$V_{K32} = \left(\frac{1}{3} * 1 * \frac{1}{7} * \frac{1}{3} * 3\right)^{\frac{1}{5}} = 0,543946$$

$$V_{K33} = (5 * 7 * 1 * 7 * 9)^{\frac{1}{5}} = 4,66318$$

$$V_{K34} = \left(\frac{1}{3} * 3 * \frac{1}{7} * 1 * 3\right)^{\frac{1}{5}} = 0,844121$$

$$V_{K35} = \left(\frac{1}{5} * \frac{1}{3} * \frac{1}{9} * \frac{1}{5} * 1\right)^{\frac{1}{5}} = 0,30096$$

Нормирующий коэффициент  $\sum V_{K3Y}$ :

$$\begin{aligned} \sum V_{K3Y} &= V_{K31} + V_{K32} + V_{K33} + V_{K34} + V_{K35} = \\ &= 1,55185 + 0,543946 + 4,66318 + 0,844121 + 0,30096 = \\ &= 7,904057 \end{aligned}$$

Важность приоритетов:

$$W_{3K31} = \frac{1,55185}{7,904057} = 0,196336$$

$$W_{3K32} = \frac{0,543946}{7,904057} = 0,0688186$$

$$W_{3K33} = \frac{4,66318}{7,904057} = 0,589973$$

$$W_{3K34} = \frac{0,844121}{7,904057} = 0,106796$$

$$W_{3K35} = \frac{0,30096}{7,904057} = 0,0380766$$

Вектор приоритетов:

$$W_{3K3Y} = (0,197; 0,068; 0,591; 0,106; 0,038)$$

#### КРИТЕР\_4 – Вес

Составим обратно симметричную матрицу для КРИТЕР\_4 (Таблица 3.5.2.5).

Таблица 3.5.2.5 - Обратно симметричная матрица для КРИТЕР\_4

К4	A1	A2	A3	A4	A5	V <sub>K4Y</sub>	W <sub>3K4Y</sub>
A1	1	3	1/5	3	7	1,65987	0,205181
A2	1/3	1	1/7	3	5	0,93492	0,115568
A3	5	7	1	7	9	4,66318	0,576427
A4	1/3	1/3	1/7	1	5	0,602457	0,0744712
A5	1/7	1/5	1/9	1/5	1	0,229374	0,0283535
$\sum V_{K4Y}$						8,089801	

Относительная ценность:

$$V_{K41} = \left(1 * 3 * \frac{1}{5} * 3 * 7\right)^{\frac{1}{5}} = 1,65987$$

$$V_{K42} = \left(\frac{1}{3} * 1 * \frac{1}{7} * 3 * 5\right)^{\frac{1}{5}} = 0,93492$$

$$V_{K43} = (5 * 7 * 1 * 7 * 9)^{\frac{1}{5}} = 4,66318$$

$$V_{K44} = \left(\frac{1}{3} * \frac{1}{3} * \frac{1}{7} * 1 * 5\right)^{\frac{1}{5}} = 0,602457$$

$$V_{K45} = \left(\frac{1}{7} * \frac{1}{5} * \frac{1}{9} * \frac{1}{5} * 1\right)^{\frac{1}{5}} = 0,229374$$

Нормирующий коэффициент  $\sum V_{K4Y}$ :

$$\begin{aligned}\sum V_{K4Y} &= V_{K41} + V_{K42} + V_{K43} + V_{K44} + V_{K45} = \\ &= 1,65987 + 0,93492 + 4,66318 + 0,602457 + 0,229374 = \\ &= 8,089801\end{aligned}$$

Важность приоритетов:

$$\begin{aligned}W_{3K41} &= \frac{1,65987}{8,089801} = 0,205181 \\ W_{3K42} &= \frac{0,93492}{8,089801} = 0,115568 \\ W_{3K43} &= \frac{4,66318}{8,089801} = 0,576427 \\ W_{3K44} &= \frac{0,602457}{8,089801} = 0,0744712 \\ W_{3K45} &= \frac{0,229374}{8,089801} = 0,0283535\end{aligned}$$

Вектор приоритетов:

$$W_{3K4Y} = (0,205; 0,115; 0,577; 0,074; 0,028)$$

### КРИТЕР\_5 – Диагональ

Составим обратно симметричную матрицу для КРИТЕР\_5 (Таблица 3.5.2.6).

Таблица 3.5.2.6 - Обратно симметричная матрица для КРИТЕР\_5

K5	A1	A2	A3	A4	A5	V <sub>K5Y</sub>	W <sub>3K5Y</sub>
A1	1	5	7	3	1/3	2,03617	0,277415
A2	1/5	1	5	1/3	1/5	0,581811	0,0792681
A3	1/7	1/5	1	1/7	1/9	0,214446	0,0292169
A4	1/3	3	7	1	1/3	1,18466	0,161403
A5	3	5	9	3	1	3,3227	0,452697
$\sum V_{K5Y}$						7,339787	

Относительная ценность:

$$V_{K51} = \left(1 * 5 * 7 * 3 * \frac{1}{3}\right)^{\frac{1}{5}} = 2,03617$$

$$V_{K52} = \left(\frac{1}{5} * 1 * 5 * \frac{1}{3} * \frac{1}{5}\right)^{\frac{1}{5}} = 0,581811$$

$$V_{K53} = \left(\frac{1}{7} * \frac{1}{5} * 1 * \frac{1}{7} * \frac{1}{9}\right)^{\frac{1}{5}} = 0,214446$$

$$V_{K54} = \left(\frac{1}{3} * 3 * 7 * 1 * \frac{1}{3}\right)^{\frac{1}{5}} = 1,18466$$

$$V_{K55} = (3 * 5 * 9 * 3 * 1)^{\frac{1}{5}} = 3,3227$$

Нормирующий коэффициент  $\sum V_{K5Y}$ :

$$\begin{aligned}\sum V_{K5Y} &= V_{K51} + V_{K52} + V_{K53} + V_{K54} + V_{K55} = \\ &= 2,03617 + 0,581811 + 0,214446 + 1,18466 + 3,3227 = \\ &= 7,339787\end{aligned}$$

Важность приоритетов:

$$W_{3K51} = \frac{2,03617}{7,339787} = 0,277415$$

$$W_{3K52} = \frac{0,581811}{7,339787} = 0,0792681$$

$$W_{3K53} = \frac{0,214446}{7,339787} = 0,0292169$$

$$W_{3K54} = \frac{1,18466}{7,339787} = 0,161403$$

$$W_{3K55} = \frac{3,3227}{7,339787} = 0,452697$$

Вектор приоритетов:

$$W_{3K5Y} = (0,277; 0,0798; 0,029; 0,161; 0,453)$$

## Согласованность локальных приоритетов

Любая матрица суждений в общем случае не согласована, так как суждения отражают субъективные мнения ЛПР, а сравнение элементов, которые имеют количественные эквиваленты, может быть несогласованным из-за присутствия погрешности проведения при проведении измерений. Совершенной согласованности парных сравнений даже в идеальном случае на практике достичь трудно. Нужен способ оценки степени согласованности при решении конкретной задачи.

Метод анализа иерархий дает возможность провести такую оценку.

Вместе с матрицей парных сравнений мы имеем меру оценки степени отклонения от согласованности. Когда такие отклонения превышают установленные пределы тем, кто проводит решение задачи, необходимо их пересмотреть [2].

В нашей задаче размерность матрицы  $n=5$ , тогда среднее значение индекса случайной согласованности  $СИ = 1,12$ .

Теперь определяем индекс согласованности и отношение согласованности.

Определим индекс согласованности и отношение согласованности для матрицы «цель» (Таблица 3.5.2.7).

Таблица 3.5.2.7 – Матрица «Цель»

Цель	К 1	К 2	К 3	К 4	К 5	W <sub>2i</sub>
К 1	1	3	3	5	7	0,47
К 2	1/3	1	1	3	5	0,2
К 3	1/3	1	1	3	5	0,2
К 4	1/5	1/3	1/3	1	3	0,09
К 5	1/7	1/5	1/5	1/3	1	0,04

$$S_1 = 1 + \frac{1}{3} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} = \frac{211}{105} = 2,01$$

$$S_2 = 3 + 1 + 1 + \frac{1}{3} + \frac{1}{5} = \frac{83}{15} = 5,53$$

$$S_3 = 3 + 1 + 1 + \frac{1}{3} + \frac{1}{5} = \frac{83}{15} = 5,53$$



$$S_4 = 5 + 3 + 3 + 1 + \frac{1}{3} = \frac{37}{3} = 12,3$$

$$S_5 = 7 + 5 + 5 + 3 + 1 = 21$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов, т.е. сумму суждений первого столбца на первую компоненту, сумму суждений второго столбца - на вторую и т.д.

$$P_1 = S_1 * W_{21} = 2,01 * 0,47 = 0,9447$$

$$P_2 = S_2 * W_{22} = 0,2 * 5,53 = 1,106$$

$$P_3 = S_3 * W_{23} = 0,2 * 5,53 = 1,106$$

$$P_4 = S_4 * W_{24} = 0,09 * 12,3 = 1,107$$

$$P_5 = S_5 * W_{25} = 0,04 * 21 = 0,84$$

Сумма чисел  $P_j$  отражает пропорциональность предпочтений, чем ближе эта величина к  $n$  (числу объектов и видов действия в матрице парных сравнений), тем более согласованы суждения.

$$\lambda_{max} = P_1 + P_2 + P_3 + P_4 + P_5 = 0,9447 + 1,106 + 1,106 + 1,107 + 0,84 = 5,1037$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС = \frac{(\lambda_{max} - n)}{(n - 1)} = \frac{(5,1037 - 5)}{(5 - 1)} = 0,0275$$

Отношение индекса согласованности ИС к среднему значению случайного индекса согласованности СИ называется отношением согласованности ОС.

$$ОС = \frac{ИС}{СИ} = \frac{0,0275}{1,12} = 0,02$$

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица «цель» согласована.

### КРИТЕР\_1 – Цена

Определим индекс согласованности и отношение согласованности для матрицы КРИТЕР\_1 (Таблица 3.5.2.8).

Таблица 3.5.2.8 - Матрица КРИТЕР\_1

К1	A1	A2	A3	A4	A5	W <sub>3K1Y</sub>
A1	1	7	3	5	1/3	0,267
A2	1/7	1	1/5	1/3	1/9	0,033
A3	1/3	5	1	1/3	1/5	0,084
A4	1/5	3	3	1	1/7	0,1
A5	3	9	5	7	1	0,516

$$S_{1K1} = 1 + \frac{1}{7} + \frac{1}{3} + \frac{1}{5} + 3 = \frac{491}{105} = 4,67$$

$$S_{2K1} = 7 + 1 + 5 + 3 + 9 = 25$$

$$S_{3K1} = 3 + \frac{1}{5} + 1 + 3 + 5 = \frac{61}{5} = 12,2$$

$$S_{4K1} = 5 + \frac{1}{3} + \frac{1}{3} + 1 + 7 = \frac{41}{3} = 13,6$$

$$S_{5K1} = \frac{1}{3} + \frac{1}{9} + \frac{1}{5} + \frac{1}{7} + 1 = \frac{563}{315} = 1,78$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов.

$$P_{1K1} = S_1 * W_{3K11} = 0,267 * 4,67 = 1,25$$

$$P_{2K1} = S_2 * W_{3K12} = 0,033 * 25 = 8,25$$

$$P_{3K1} = S_3 * W_{3K13} = 0,084 * 12,2 = 1,02$$

$$P_{4K1} = S_4 * W_{3K14} = 0,1 * 13,6 = 1,36$$

$$P_{5K1} = S_5 * W_{3K15} = 0,516 * 1,78 = 0,92$$

Находим пропорциональность предпочтений.

$$\lambda_{maxK1} = P_{1K1} + P_{2K1} + P_{3K1} + P_{4K1} + P_{5K1} = 1,25 + 8,25 + 1,02 + 1,36 + 0,92 = 12,8$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{K1} = \frac{(\lambda_{maxK1} - n)}{(n - 1)} = \frac{(12,8 - 5)}{(5 - 1)} = 0,095$$

Найдем отношением согласованности ОС.

$$ОС_{K1} = \frac{ИС}{СИ} = \frac{0,095}{1,12} = 0,08$$

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица КРИТЕР\_1 согласована.

## КРИТЕР\_2 – Год

Определим индекс согласованности и отношение согласованности для матрицы КРИТЕР\_2 (Таблица 3.5.2.9).

Таблица 3.5.2.9 - Матрица КРИТЕР\_2

<b>К2</b>	<b>А1</b>	<b>А2</b>	<b>А3</b>	<b>А4</b>	<b>А5</b>	<b>W<sub>3K2Y</sub></b>
<b>А1</b>	1	3	5	7	5	0,47
<b>А2</b>	1/3	1	5	7	5	0,303
<b>А3</b>	1/5	1/5	1	5	1	0,097
<b>А4</b>	1/7	1/7	1/5	1	1/5	0,032
<b>А5</b>	1/5	1/5	1	5	1	0,097

$$S_{1K2} = 1 + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \frac{1}{5} = \frac{197}{105} = 1,87$$

$$S_{2K2} = 3 + 1 + \frac{1}{5} + \frac{1}{7} + \frac{1}{5} = \frac{159}{35} = 4,54$$

$$S_{3K2} = 5 + 5 + 1 + \frac{1}{5} + 1 = \frac{61}{5} = 12,2$$

$$S_{4K2} = 7 + 7 + 5 + 1 + 5 = 25$$

$$S_{5K2} = 5 + 5 + 1 + \frac{1}{5} + 1 = \frac{61}{5} = 12,2$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов.

$$P_{1K2} = S_1 * W_{3K21} = 0,47 * 1,87 = 0,88$$

$$P_{2K2} = S_2 * W_{3K22} = 0,303 * 4,54 = 1,38$$

$$P_{3K2} = S_3 * W_{3K23} = 0,097 * 12,2 = 1,18$$

$$P_{4K2} = S_4 * W_{3K24} = 0,032 * 25 = 0,8$$

$$P_{5K2} = S_5 * W_{3K25} = 0,097 * 12,2 = 1,18$$

Находим пропорциональность предпочтений.

$$\begin{aligned} \lambda_{maxK2} &= P_{1K2} + P_{2K2} + P_{3K2} + P_{4K2} + P_{5K2} = 0,88 + 1,38 + 1,18 + 0,8 + 1,18 \\ &= 5,42 \end{aligned}$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{K2} = \frac{(\lambda_{maxK2} - n)}{(n - 1)} = \frac{(5,42 - 5)}{(5 - 1)} = 0,105$$

Найдем отношением согласованности ОС.

$$ОС_{K2} = \frac{ИС}{СИ} = \frac{0,105}{1,12} = 0,09$$

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица КРИТЕР\_2 согласована.

### КРИТЕР\_3 – Гарантия

Определим индекс согласованности и отношение согласованности для матрицы КРИТЕР\_3 (Таблица 3.5.2.10)

Таблица 3.5.2.10 - Матрица КРИТЕР\_3

К3	A1	A2	A3	A4	A5	W <sub>зкзy</sub>
A1	1	3	1/5	3	5	0,197
A2	1/3	1	1/7	1/3	3	0,068
A3	5	7	1	7	9	0,591
A4	1/3	3	1/7	1	3	0,106
A5	1/5	1/3	1/9	1/5	1	0,038

$$S_{1K3} = 1 + \frac{1}{3} + 5 + \frac{1}{3} + \frac{1}{5} = \frac{103}{15} = 6,86$$

$$S_{2K3} = 3 + 1 + 7 + 3 + \frac{1}{3} = \frac{43}{3} = 14,3$$

$$S_{3K3} = \frac{1}{5} + \frac{1}{7} + 1 + \frac{1}{7} + \frac{1}{9} = \frac{503}{315} = 1,6$$

$$S_{4K3} = 3 + \frac{1}{3} + 7 + 1 + \frac{1}{3} = \frac{35}{3} = 11,6$$

$$S_{5K3} = 5 + 3 + 9 + 3 + 1 = 21$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов.

$$P_{1K3} = S_1 * W_{3K31} = 0,197 * 6,86 = 1,35$$

$$P_{2K3} = S_2 * W_{3K32} = 0,068 * 14,3 = 0,97$$

$$P_{3K3} = S_3 * W_{3K33} = 0,591 * 1,6 = 0,95$$

$$P_{4K3} = S_4 * W_{3K34} = 0,106 * 11,6 = 1,23$$

$$P_{5K3} = S_5 * W_{3K35} = 0,038 * 21 = 0,8$$

Находим пропорциональность предпочтений.

$$\lambda_{maxK1} = P_{1K3} + P_{2K3} + P_{3K3} + P_{4K3} + P_{5K3} = 1,35 + 0,97 + 0,95 + 1,23 + 0,8 = 5,3$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{K3} = \frac{(\lambda_{maxK3} - n)}{(n - 1)} = \frac{(5,3 - 5)}{(5 - 1)} = 0,075$$

Найдем отношением согласованности ОС.

$$ОС_{K3} = \frac{ИС}{СИ} = \frac{0,075}{1,12} = 0,07$$

Значение ОС меньше или равное 0.10 считается приемлемым, значит матрица КРИТЕР\_3 согласована.

#### **КРИТЕР\_4 – Вес**

Определим индекс согласованности и отношение согласованности для матрицы КРИТЕР\_4 (Таблица 3.5.2.11).

Таблица 3.5.2.11 - Матрица КРИТЕР\_4

<b>K4</b>	<b>A1</b>	<b>A2</b>	<b>A3</b>	<b>A4</b>	<b>A5</b>	<b>W<sub>зк4у</sub></b>
<b>A1</b>	1	3	1/5	3	7	0,205
<b>A2</b>	1/3	1	1/7	3	5	0,115
<b>A3</b>	5	7	1	7	9	0,577
<b>A4</b>	1/3	1/3	1/7	1	5	0,074
<b>A5</b>	1/7	1/5	1/9	1/5	1	0,028

$$S_{1K4} = 1 + \frac{1}{3} + 5 + \frac{1}{3} + \frac{1}{7} = \frac{143}{21} = 6,8$$

$$S_{2K4} = 3 + 1 + 7 + \frac{1}{3} + \frac{1}{5} = \frac{173}{15} = 11,53$$

$$S_{3K4} = \frac{1}{5} + \frac{1}{7} + 1 + \frac{1}{7} + \frac{1}{9} = \frac{503}{315} = 1,6$$

$$S_{4K4} = 3 + 3 + 7 + 1 + \frac{1}{5} = \frac{71}{5} = 14,2$$

$$S_{5K4} = 7 + 5 + 9 + 5 + 1 = 27$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов.

$$P_{1K4} = S_1 * W_{3K41} = 0,205 * 6,8 = 1,4$$

$$P_{2K4} = S_2 * W_{3K42} = 0,115 * 11,53 = 1,33$$

$$P_{3K4} = S_3 * W_{3K43} = 0,577 * 1,6 = 0,92$$

$$P_{4K4} = S_4 * W_{3K44} = 0,074 * 14,2 = 1,05$$

$$P_{5K4} = S_5 * W_{3K45} = 0,028 * 27 = 0,76$$

Находим пропорциональность предпочтений.

$$\begin{aligned} \lambda_{maxK1} &= P_{1K4} + P_{2K4} + P_{3K4} + P_{4K4} + P_{5K4} = 1,4 + 1,33 + 0,92 + 1,05 + 0,76 \\ &= 5,46 \end{aligned}$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{K4} = \frac{(\lambda_{maxK4} - n)}{(n - 1)} = \frac{(5,46 - 5)}{(5 - 1)} = 0,1125$$

Найдем отношением согласованности ОС.

$$ОС_{K4} = \frac{ИС}{СИ} = \frac{0,1125}{1,12} = 0,1$$

### КРИТЕР\_5 – Диагональ

Определим индекс согласованности и отношение согласованности для матрицы КРИТЕР\_5 (Таблица 3.5.2.12).

Таблица 3.5.2.12 - Матрица КРИТЕР\_5

<b>K5</b>	<b>A1</b>	<b>A2</b>	<b>A3</b>	<b>A4</b>	<b>A5</b>	<b>W<sub>3K5Y</sub></b>
<b>A1</b>	1	5	7	3	1/3	0,277
<b>A2</b>	1/5	1	5	1/3	1/5	0,079
<b>A3</b>	1/7	1/5	1	1/7	1/9	0,029
<b>A4</b>	1/3	3	7	1	1/3	0,161
<b>A5</b>	3	5	9	3	1	0,453

$$S_{1K5} = 1 + \frac{1}{5} + \frac{1}{7} + \frac{1}{3} + 3 = \frac{491}{105} = 4,68$$

$$S_{2K5} = 5 + 1 + \frac{1}{5} + 3 + 5 = \frac{71}{5} = 14,2$$

$$S_{3K5} = 7 + 5 + 1 + 7 + 9 = 29$$

$$S_{4K5} = 3 + \frac{1}{3} + \frac{1}{7} + 1 + 3 = \frac{157}{21} = 7,48$$

$$S_{5K5} = \frac{1}{3} + \frac{1}{5} + \frac{1}{9} + \frac{1}{3} + 1 = \frac{89}{45} = 1,97$$

Затем полученный результат умножается на компоненту нормализованного вектора приоритетов.

$$P_{1K5} = S_1 * W_{3K51} = 0,277 * 4,68 = 1,3$$

$$P_{2K5} = S_2 * W_{3K52} = 0,079 * 14,2 = 1,12$$

$$P_{3K5} = S_3 * W_{3K53} = 0,029 * 29 = 0,84$$

$$P_{4K5} = S_4 * W_{3K54} = 0,161 * 7,48 = 1,20$$

$$P_{5K5} = S_5 * W_{3K55} = 0,453 * 1,97 = 0,9$$

Находим пропорциональность предпочтений.



$$\lambda_{maxK5} = P_{1K5} + P_{2K5} + P_{3K5} + P_{4K5} + P_{5K5} = 1,3 + 1,12 + 0,84 + 1,20 + 0,9 = 5,36$$

Отклонение от согласованности выражается индексом согласованности.

$$ИС_{K5} = \frac{(\lambda_{maxK5} - n)}{(n - 1)} = \frac{(5,36 - 5)}{(5 - 1)} = 0,085$$

Найдем отношением согласованности ОС.

$$ОС_{K5} = \frac{ИС}{СИ} = \frac{0,085}{1,12} = 0,08$$

Значение ОС меньше или равное 0,10 считается приемлемым, значит матрица КРИТЕР\_5 согласована.

### **Синтез альтернатив**

Векторы приоритетов и отношения согласованности определяются для всех матриц суждений, начиная со второго уровня [1].

Для определения приоритетов альтернатив необходимо локальные приоритеты умножить на приоритет соответствующего критерия на высшем уровне и найти суммы по каждому элементу в соответствии с критериями, на которые воздействует этот элемент [2].

$$W_{2i} = (0,47; 0,2; 0,2; 0,09; 0,04);$$

$$W_{3K_1Y} = (0,267; 0,033; 0,084; 0,1; 0,516);$$

$$W_{3K_2Y} = (0,47; 0,303; 0,097; 0,032; 0,097);$$

$$W_{3K_3Y} = (0,197; 0,068; 0,591; 0,106; 0,038);$$

$$W_{3K_4Y} = (0,205; 0,115; 0,577; 0,074; 0,028);$$

$$W_{3K_5Y} = (0,277; 0,079; 0,029; 0,161; 0,453).$$

Приоритеты альтернатив получим следующим образом:

$$\begin{aligned} W_1 &= W_{21} * W_{3K11} + W_{22} * W_{3K21} + W_{23} * W_{3K31} + W_{24} * W_{3K41} + W_{25} * W_{3K51} \\ &= 0,47 * 0,267 + 0,2 * 0,47 + 0,2 * 0,198 + 0,09 * 0,205 + 0,04 \\ &\quad * 0,277 = 0,287 \end{aligned}$$

$$\begin{aligned} W_2 &= W_{21} * W_{3K12} + W_{22} * W_{3K22} + W_{23} * W_{3K32} + W_{24} * W_{3K42} + W_{25} * W_{3K52} \\ &= 0,47 * 0,033 + 0,2 * 0,303 + 0,2 * 0,068 + 0,09 * 0,115 + 0,04 \\ &\quad * 0,029 = 0,104 \end{aligned}$$

$$\begin{aligned} W_3 &= W_{21} * W_{3K13} + W_{22} * W_{3K23} + W_{23} * W_{3K33} + W_{24} * W_{3K43} + W_{25} * W_{3K53} \\ &= 0,47 * 0,084 + 0,2 * 0,097 + 0,2 * 0,591 + 0,09 * 0,577 + 0,04 \\ &\quad * 0,029 = 0,229 \end{aligned}$$

$$\begin{aligned} W_4 &= W_{21} * W_{3K14} + W_{22} * W_{3K24} + W_{23} * W_{3K34} + W_{24} * W_{3K44} + W_{25} * W_{3K54} \\ &= 0,47 * 0,1 + 0,2 * 0,032 + 0,2 * 0,106 + 0,09 * 0,074 + 0,04 \\ &\quad * 0,161 = 0,087 \end{aligned}$$

$$\begin{aligned} W_5 &= W_{21} * W_{3K15} + W_{22} * W_{3K25} + W_{23} * W_{3K35} + W_{24} * W_{3K45} + W_{25} * W_{3K55} \\ &= 0,47 * 0,516 + 0,2 * 0,097 + 0,2 * 0,038 + 0,09 * 0,028 + 0,04 \\ &\quad * 0,453 = 0,291 \end{aligned}$$

### 3.5.3 Третий этап

Таким образом, приоритеты альтернатив равны:

- Альтернатива A1 (LG QLED) -  $W_1$  приоритет равен 0,287;
- Альтернатива A2 (SAMSUNG NEO QLED)-  $W_2$  приоритет равен 0,104;
- Альтернатива A3 (SAMSUNG UHD QLED) -  $W_3$  приоритет равен 0,229.
- Альтернатива A4 (SONY UHD LED) -  $W_4$  приоритет равен 0,087;
- Альтернатива A5 (LG LED) -  $W_5$  приоритет равен 0,291.

Наиболее перспективным с позиции метода анализа иерархий признается выбор телевизора АЛТЕРН\_5 – LG LED.

### 3.6 Программная реализация

Результат программной реализации представлен на Рисунке 3.6.1. Код реализации представлен в Приложении В.

```
C:\Users\Anastasia\source\repos\prmmmmmm3\prmmmmmm3\bin\Debug\prmmmmmm3.exe
Вектор приоритетов:
0,47  0,2  0,2  0,09  0,04

Вектора приоритетов для каждого критерия:
Цена      : 0,267 0,033 0,084 0,1 0,516
Год       : 0,47 0,303 0,097 0,032 0,097
Гарантия  : 0,197 0,068 0,591 0,106 0,038
Вес       : 0,205 0,115 0,577 0,074 0,028
Диагональ : 0,277 0,079 0,029 0,161 0,453

Индекс согласованности для матрицы "цели": 0,0275
Отношение согласованности: 0,02

Индексы согласованности для матриц критериев:
Цена: 0,095
Отношение согласованности для Цена: 0,08
Год: 0,105
Отношение согласованности для Год: 0,09
Гарантия: 0,075
Отношение согласованности для Гарантия: 0,07
Вес: 0,1125
Отношение согласованности для Вес: 0,1
Диагональ: 0,085
Отношение согласованности для Диагональ: 0,08

Синтез альтернатив.
Приоритеты альтернатив:
LG QLED: 0,287
SAMSUNG NEO QLED: 0,104
SAMSUNG UHD QLED: 0,229
SONY UHD LED: 0,087
LG LED: 0,291
Наилучший вариант: LG LED
```

Рисунок 3.6.1 – Результат работы программы

## 4 ОПИСАНИЕ АЛГОРИТМА ГРАФИЧЕСКОГО МЕТОДА

Сущность линейного программирования состоит в нахождении точек наибольшего или наименьшего значения некоторой функции при определенном наборе ограничений, налагаемых на аргументы и образующих систему ограничений, которая имеет, как правило, бесконечное множество решений [3].

Математическая модель любой задачи линейного программирования включает в себя:

- максимум или минимум целевой функции (критерий оптимальности);
- систему ограничений в форме линейных уравнений и неравенств;
- требование неотрицательности переменных.

В других ситуациях могут возникать задачи с большим количеством переменных, в систему ограничений которых, кроме неравенств, могут входить и равенства [3].

### 4.1 Постановка задачи

Целевая функция  $\overline{f(x)} = 2x_1 + x_2 \rightarrow \max/\min$

$$\text{Ограничения: } \begin{cases} x_1 + x_2 \leq 3 \\ -x_1 + x_2 \leq 2 \\ x_1 - x_2 \leq 1 \\ x_1, x_2 \geq 0 \end{cases}$$

### 4.2 Выполнение работы

#### 4.2.1 Первый этап

Построение ОДР ЗЛП. График реализации представлен на Рисунке 4.2.1.1.

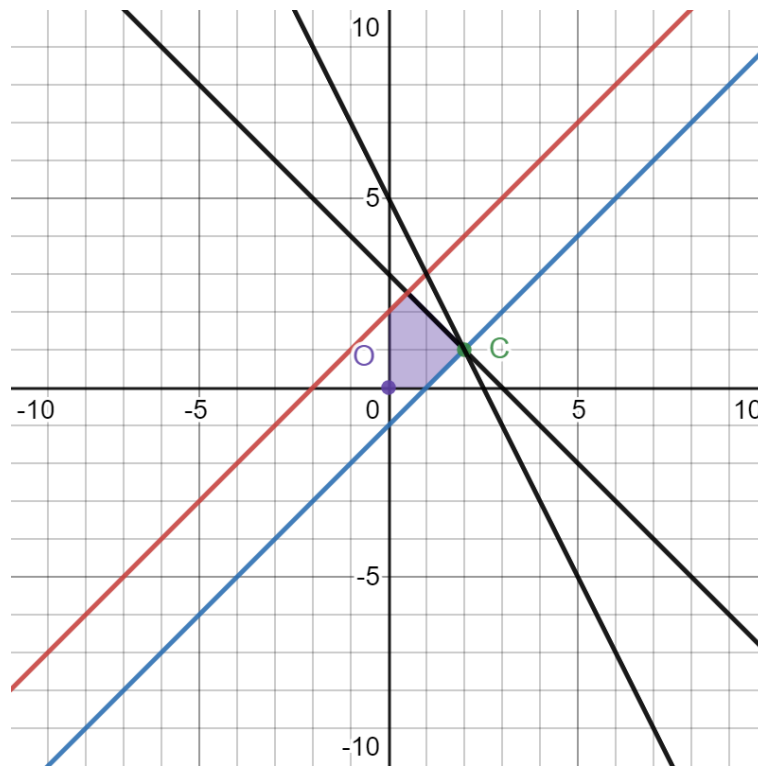


Рисунок 4.2.1.1 - Построение области допустимых решений задачи

## 4.2.2 Второй этап

Вектор, координаты которого являются частными производными функции  $\overline{f(x)}$  – называется градиентом функции  $\overline{\text{grad } f(x)}$ . Градиент перпендикулярен линиям уровня и показывает направление наибольшего возрастания функции  $\overline{f(x)}$ .

$\overline{\text{grad } f(x)} = \{c_1, c_2\} \rightarrow \max$ , где

$$c_1 = \frac{df(x)}{dx_1}, c_2 = \frac{df(x)}{dx_2}$$

$$\overline{\text{grad } f(x)} = \{2, 1\} \rightarrow \max$$

Соответственно, антиградиент перпендикулярен линиям уровня и показывает направление наименьшего убывания функции  $\overline{f(x)}$ .

$-\overline{\text{grad } f(x)} = \{c_1, c_2\} \rightarrow \min$ , где

$$c_1 = -\frac{df(x)}{dx_1}, c_2 = -\frac{df(x)}{dx_2}$$

$$-\overline{\text{grad } f(x)} = \{-2, -1\} \rightarrow \min$$

Оптимальное решение для задачи о максимальном доходе будут координаты точки  $C(x_1^*, x_2^*)$ . Для их нахождения необходимо решить систему линейных уравнений, соответствующих прямым, пересекающихся в точке оптимума С.

$$\begin{cases} x_1 + x_2 = 3 \\ x_1 - x_2 = 1 \end{cases}$$

Решив систему линейных уравнений, получим, что точка максимума соответствует  $(x_1^*, x_2^*) = (2, 1)$ .

Подставив координаты  $x_1^*$  и  $x_2^*$  в целевую функцию получим

$$f(x) = 4 + 1 = 5$$

Координаты точки минимума соответствуют  $(x_1^*, x_2^*) = (0, 0)$ .

Сделаем проверку, подставим в наши ограничения, точки максимума и минимума.

$$\text{Ограничения: } \begin{cases} x_1 + x_2 \leq 3 \\ -x_1 + x_2 \leq 2 \\ x_1 - x_2 \leq 1 \\ x_1, x_2 \geq 0 \end{cases}$$

Проверка максимума:

$$\begin{cases} 3 \leq 3 \\ -1 \leq 2 \\ 1 \leq 1 \\ 2 \geq 0 \\ 1 \geq 0 \end{cases}$$

Проверка минимума:

$$\begin{cases} 0 \leq 3 \\ 0 \leq 2 \\ 0 \leq 1 \\ 0 \geq 0 \\ 0 \geq 0 \end{cases}$$

## 5 ОПИСАНИЕ АЛГОРИТМА СИМПЛЕКС МЕТОДА

Регулярным симплексом в  $n$ -мерном пространстве называется правильный многогранник с  $n+1$  вершиной. При  $n = 2$  симплексом является правильный треугольник, при  $n = 3$  – тетраэдр и т.д.

В симплексе решение задачи начинается с рассмотрений одной из вершин многогранника условий. Если исследуемая вершина не соответствует максимуму (минимуму), то переходят к соседней, увеличивая значение функции цели при решении задачи на максимум и уменьшая при решении задачи на минимум. Таким образом, переход от одной вершины к другой улучшает значение функции цели. Так как число вершин многогранника ограничено, то за конечное число шагов гарантируется нахождение оптимального значения или установление того факта, что задача неразрешима [3].

Таким образом, симплексный метод - это метод целенаправленного перебора опорных решений ЗЛП.

### 5.1 Постановка задачи

Решить прямую ЗЛП с помощью симплексного метода и обратную с помощью теорем двойственности. Определить интервалы устойчивости.

Для производства двух видов изделия используется 3 вида ресурсов. Объем ресурсов ограничен. В Таблице 5.1 даны объемы ресурсов, нормы расходов каждого из ресурсов на одно изделие каждого вида и прибыль, получаемая от реализации одного изделия каждого вида.

Таблица 5.1 - Исходные данные задачи.

Виды ресурсов	Объем ресурсов	Нормы расхода	
		I	II
Сталь, т	500	10	70
Цветные металлы, кг	510	20	50
Станки, станко-час	3100	200	100
Прибыль, ден. ед.		5	5

Определить план выпуска продукции, при котором будет достигнута максимальная прибыль.

## 5.2 Расчетная часть

Целевая функция:

$$f(\bar{x}) = 5x_1 + 5x_2 \rightarrow \max;$$

Ограничения:

$$\begin{cases} 10x_1 + 70x_2 \leq 500 \\ 20x_1 + 50x_2 \leq 510 \\ 200x_1 + 100x_2 \leq 3100 \\ x_1, x_2 \geq 0 \end{cases}$$

Приведем задачу к канонической форме. Для этого в левые части ограничений вводим дополнительные переменные:

$$\begin{cases} x_3 \geq 0 \\ x_4 \geq 0 \\ x_5 \geq 0. \end{cases}$$

Эти переменные выбираются так, чтобы они обращали неравенства в равенства.

$$\begin{cases} 10x_1 + 70x_2 + x_3 = 500 \\ 20x_1 + 50x_2 + x_4 = 510 \\ 200x_1 + 100x_2 + x_5 = 3100 \\ x_j \geq 0, j = \overline{1,5} \end{cases}$$

$$f(x) = 5x_1 + 5x_2 + 0x_3 + 0x_4 + 0x_5$$

$j = \overline{1,5}$  – остатки неиспользованных ресурсов

Построим начальную симплекс-таблицу. Запишем нашу систему в векторной форме:

$$A_1x_1 + A_2x_2 + A_3x_3 + A_4x_4 + A_5x_5 = A_0$$

$$A_1 = \begin{pmatrix} 10 \\ 20 \\ 200 \end{pmatrix}, A_2 = \begin{pmatrix} 70 \\ 50 \\ 100 \end{pmatrix}, A_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, A_4 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, A_5 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, A_0 = \begin{pmatrix} 500 \\ 510 \\ 3100 \end{pmatrix}.$$

Векторы  $A_3, A_4, A_5$  – образуют базис.

$x_3, x_4, x_5$  – базисные переменные.

$x_1, x_2$  – небазисные переменные.

Приравняем  $x_1, x_2$  к 0, и получим первоначальный опорный план.



$$x^{(0)} = (x_1, x_2, x_3, x_4, x_5) = (0, 0, 500, 510, 3100),$$

$$f(x^{(0)}) = 0$$

Для проверки плана  $x^{(0)}$  на оптимальность построим первую симплекс – таблицу (Таблица 5.2.1). Введем в рассмотрение вектор коэффициентов целевой функции при базисных переменных.

$$\overline{C}_B = (c_3, c_4, c_5)^T = (0, 0, 0)^T$$

Для заполнения f-строки найдем относительные оценки  $\Delta_1$ ,  $\Delta_2$ , и значение целевой функции  $Q$ .

$$\Delta_1 = (\overline{C}_B * \overline{A}_1) - C_1 = 0 * 10 + 0 * 20 + 0 * 200 - 5 = -5$$

$$\Delta_2 = (\overline{C}_B * \overline{A}_2) - C_2 = 0 * 70 + 0 * 50 + 0 * 100 - 5 = -5$$

$$Q = (\overline{C}_B * \overline{A}_0) = 0 * 500 + 0 * 510 + 0 * 3100 = 0$$

Таблица 5.2.1 - Симплекс-таблица задачи о максимальном доходе

$\overline{C}_B$	Basis	$\overline{A}_0$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
0	$x_3$	500	10	70	1	0	0
0	$x_4$	510	20	50	0	1	0
0	$x_5$	3100	200	100	0	0	1
<b>F</b>		0	-5	-5	0	0	0
		<b>Q</b>	$\Delta_1$	$\Delta_2$			

### Итерация №0

Итерация №0 представлена в Таблице 1.2.2.

Таблица 1.2.2 – Итерация №0

$\overline{C}_B$	Basis	$\overline{A}_0$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
0	$x_3$	345	0	65	1	0	-0.05
0	$x_4$	200	0	40	0	1	-0.1
0	$x_1$	15.5	1	0.5	0	0	0
<b>F</b>	77.5	0	-2.5	0	0	0.02	<b>F</b>
		<b>Q</b>	$\Delta_1$	$\Delta_2$			

При  $i = 1, 2, 3$

$$i_1 = \frac{500}{10} = 50$$

$$i_2 = \frac{510}{20} = 25.5$$

$$i_3 = \frac{3100}{200} = 15.5$$

$$i_3 \rightarrow \min$$

Запишем опорный план:

$$x^{(1)} = (15.5, 0, 345, 200, 0),$$

$$f(x^{(1)}) = 5 * 15.5 + 5 * 0 + 0 * 345 + 0 * 200 + 0 = 77.5$$

### Итерация №1

Итерация №1 представлена в Таблице 1.2.3.

Таблица 1.2.3 – Итерация №1

$\overline{C_B}$	Basis	$\overline{A_0}$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
0	$x_3$	20	0	0	1	-1.62	0.11
0	$x_2$	5	0	1	0	0.02	0
0	$x_1$	13	1	0	0	-0.01	0.01
<b>F</b>		90	0	0	0	0.06	0.02
		<b>Q</b>	$\Delta_1$	$\Delta_2$			

При  $i = 1, 2, 3$

$$i_1 = \frac{345}{65} = 5.3$$

$$i_2 = \frac{200}{40} = 5$$

$$i_3 = \frac{15.5}{0.5} = 31$$

$$i_2 \rightarrow \min$$

Запишем опорный план:

$$x^{(2)} = (13, 5, 20, 0, 0),$$

$$f(x^{(2)}) = 5 * 13 + 5 * 5 + 0 * 20 + 0 + 0 = 90$$

Текущий опорный план является оптимальным:

$$x_0^* = (13, 5, 20, 0, 0)$$

Решение исходной задачи:

$$x^* = (13, 5)$$

$$x_1 = 13,$$

$$x_2 = 5$$

$$f(x) = 5 * 13 + 5 * 5 = 90$$

### 5.3 Программная реализация

Результат программной реализации представлен на Рисунке 5.3.1 – Рисунке 5.3.4. Код реализации представлен в Приложении Д.

The screenshot shows a window titled "Simplex Method". It has two input fields: "N. of constraints" with the value 3 and "N. of variables" with the value 2. There are "Default" and "OK" buttons. Below these are two tables. The first table has columns x1, x2, Sign, and a value. The second table has columns x1, x2, and C. At the bottom, there is a "GO" button and a "CLEAR" button.

x1	x2	Sign	
10	70	<=	500
20	50	<=	510
200	100	<=	3100

x1	x2	C
5	5	0

Buttons: Default, OK, GO, CLEAR. A dropdown menu shows "Max".

Рисунок 5.3.1 – Ввод исходных данных

i	Basis	C	B	A1	A2	A3	A4	A5
1	A3	0	500	10	70	1	0	0
2	A4	0	510	20	50	0	1	0
3	A5	0	3100	200	100	0	0	1
m+1	F	$\Delta_j$	0	-5	-5	0	0	0
m+2	M	$\Delta_j$		0	0	0	0	0

Рисунок 5.3.2 – Базисная таблица

i	Basis	C	B	A1	A2	A3	A4	A5
1	A3	0	345	0	65	1	0	-0,05
2	A4	0	200	0	40	0	1	-0,1
3	A1	5	15,5	1	0,5	0	0	0
m+1	F	$\Delta_j$	77,5	0	-2,5	0	0	0,02

Рисунок 5.3.3 – Итерация №0

i	Basis	C	B	A1	A2	A3	A4	A5
1	A3	0	20	0	0	1	-1,62	0,11
2	A2	5	5	0	1	0	0,02	0
3	A1	5	13	1	0	0	-0,01	0,01
m+1	F	$\Delta_j$	90	0	0	0	0,06	0,02

**Рисунок 5.3.4 – Итерация №1**

## 6 ОПИСАНИЕ АЛГОРИТМА ДВОЙСТВЕННОЙ ЗАДАЧИ

С каждой ЗЛП тесно связана другая линейная задача, называемая двойственной. Обе задачи можно считать двойственными одну по отношению к другой, считать равносильными. Первоначальная задача называется исходной или прямой, другая будет обратной. Переменные, используемые в двойственной задаче, называются двойственными. На них не накладывается ограничений по знаку. Рассматриваются двойственные критерии оптимальности. Связь между оптимальными решениями двойственных задач устанавливается теоремой двойственности [3].

### 6.1 Постановка задачи

Дана прямая задача на максимум. Построить к ней двойственную задачу.

$$f(\bar{x}) = 5x_1 + 5x_2 \rightarrow \max;$$
$$\begin{cases} 10x_1 + 70x_2 \leq 500 \\ 20x_1 + 50x_2 \leq 510 \\ 200x_1 + 100x_2 \leq 3100 \\ x_1, x_2 \geq 0 \end{cases}$$

### 6.2 Расчетная часть

#### 6.2.1 Первая теорема двойственности

Если одна из пары двойственных задач имеет оптимальный план, то и другая имеет оптимальный план, причем экстремальные значения целевых функций равны:

$$\max f(\bar{x}) = \min g(\bar{y})$$

Если задача определения оптимального плана, максимизирующего выпуск продукции, разрешима, то разрешима и задача определения оценок ресурсов.

Стоимость выпущенной продукции, полученной при реализации оптимального плана, совпадает с суммарной оценкой ресурсов. Совпадение значений целевых функций для соответствующих планов пары двойственных задач достаточно, чтобы эти планы были оптимальными.

Двойственные оценки обладают тем свойством, что они гарантируют рентабельность оптимального плана, т.е. равенство общей оценки продукции и ресурсов, и обуславливают убыточность всякого другого плана, отличного от оптимального [3].

Составим математическую модель двойственной задачи. В качестве переменных двойственной задачи возьмем  $y_1, y_2, y_3$  представляющие собой условные оценки запасов сырья. Данная задача является симметричной, тогда двойственная задача в матричном виде будет выглядеть следующим образом:

$$\begin{aligned} g(\bar{y}) = (\bar{b}, \bar{y}) &\rightarrow \min \\ A^T \bar{y} &\geq \bar{c}, \\ \bar{y} &\geq 0, \end{aligned}$$

где  $\bar{y} = (y_1, y_2, y_3)$  - вектор двойственных переменных.

$$A^T = \begin{pmatrix} 10 & 20 & 200 \\ 70 & 50 & 100 \end{pmatrix}$$

$A^T$  - транспонированная матрица коэффициентов системы ограничений.

Отсюда можно сформулировать двойственную задачу.

Целевая функция:

$$g(\bar{y}) = (\bar{b}, \bar{y}) = 500y_1 + 510y_2 + 3100y_3 \rightarrow \min;$$

Ограничения:

$$\begin{cases} 10y_1 + 20y_2 + 200y_3 \geq 5 \\ 70y_1 + 50y_2 + 100y_3 \geq 5 \\ y_1, y_2, y_3 \geq 0 \end{cases}$$

Оптимальное решение двойственной задачи может быть получено из оптимального решения прямой задачи.

Так как прямая задача имеет решение, то на основании первой теоремы о двойственности задача также разрешима. Ее решение может быть найдено из формулы:

$$\bar{g}^* = \bar{C}_B * D^{-1},$$

где D – матрица, составленная из компонентов векторов, входящих в последний базис, при котором получен оптимальный план исходной задачи.

В нашем примере в последней симплекс-таблице базисными переменными являются  $x_2, x_1, x_3$ . Соответствующие этим переменным векторы  $\bar{A}_2, \bar{A}_1, \bar{A}_3$  используются для формирования столбцов матрицы D.

$$A_2 = \begin{pmatrix} 70 \\ 50 \\ 100 \end{pmatrix}, A_1 = \begin{pmatrix} 10 \\ 20 \\ 200 \end{pmatrix}, A_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$D = (\bar{A}_2, \bar{A}_1, \bar{A}_3) = \begin{pmatrix} 70 & 10 & 1 \\ 50 & 20 & 0 \\ 100 & 200 & 0 \end{pmatrix}$$

$$D^{-1} = (y_2^*, y_1^*, y_3^*) = \begin{pmatrix} 0 & 0.02 & -0.002 \\ 0 & -0.01 & 0.006 \\ 1 & -1.62 & 0.11 \end{pmatrix}$$

Так как  $\bar{C}_B = (5, 5, 0)$ , то

$$\begin{aligned} \bar{y}^* = (y_1^*, y_2^*, y_3^*) &= \bar{C}_B * D^{-1} = (5, 5, 0) * \begin{pmatrix} 0 & 0.02 & -0.002 \\ 0 & -0.01 & 0.006 \\ 1 & -1.62 & 0.11 \end{pmatrix} = \\ &= (0; 0.06; 0.02) \end{aligned}$$

Минимальное значение целевой функции двойственной задачи:

$$g_{min} = g(\bar{y}^*) = (\bar{b}, \bar{y}^*) = 500 * 0 + 510 * 0.06 + 3100 * 0.02 = 90$$

Оно совпадает с максимальным значением  $f_{max} = 90$  прямой задачи, что является результатом взаимодвойственности.

Таким образом условие выполняется:

$$\max f(\bar{x}) = \min g(\bar{y}) = 90$$

Запасы составляют:

$$y_1 = 0,$$

$$y_2 = 0.06,$$

$$y_3 = 0.02$$

### 6.2.2 Вторая теорема двойственности

Вторую теорему двойственности иногда называют теоремой о дополняющей нежесткости. Используя ее также можно прийти к решению двойственной задачи.

Для того, чтобы планы  $\bar{x}^* = (x_1^*, x_2^*, \dots, x_n^*)$  и  $\bar{y}^* = (y_1^*, y_2^*, \dots, y_m^*)$  ЗЛП двойственной пары были оптимальными, необходимо и достаточно, чтобы эти планы удовлетворяли условиям дополняющей нежесткости.

$$\begin{cases} x_j^* \left( \sum_{i=1}^m a_{ij} y_i^* - c_j \right) = 0, j = \overline{1, n} \\ y_i^* \left( \sum_{j=1}^n a_{ij} x_j^* - b_i \right) = 0, i = \overline{1, m} \end{cases}$$

Мы имеем оптимальное решение прямой задачи:

$$x_1 = 13;$$

$$x_2 = 5;$$

$$f_{\max} = 90.$$

Рассмотрим выполнение неравенств прямой задачи при подстановке  $x_1$  и  $x_2$  в систему ограничений (Таблица 6.2.2).

Таблица 6.2.2 - Выполнение неравенств прямой задачи

Ограничение	Расчет	Вывод
$10x_1 + 70x_2 \leq 500$	$10 * 13 + 70 * 5 < 500$	Первое ограничение прямой задачи выполняется как строгое неравенство. Это означает, что ресурс 1-го вида израсходован не полностью. Следовательно, этот ресурс не является дефицитным и его оценка в оптимальном плане равна нулю ( $y_1 = 0$ ).



Продолжение Таблицы 6.2.2

$20x_1 + 50x_2 \leq 510$	$20 * 13 + 50 * 5 = 510$	Второе ограничение прямой задачи выполняется как равенство. Это означает, что ресурс 2-го вида полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля ( $y_2 \neq 0$ ).
$200x_1 + 100x_2 \leq 3100$	$200 * 13 + 100 * 5 = 3100$	Третье ограничение прямой задачи выполняется как равенство. Это означает, что ресурс 3-го вида полностью используется в оптимальном плане, является дефицитным и его оценка согласно второй теоремы двойственности отлична от нуля ( $y_3 \neq 0$ ).
$x_1 \geq 0$	$13 > 0$	Первое ограничение в двойственной задаче будет равенством $20y_2 + 200y_3 = 5$
$x_2 \geq 0$	$5 > 0$	Второе ограничение в двойственной задаче будет равенством $50y_2 + 100y_3 = 5$

Согласно таблице, имеем следующую систему уравнений:

$$\begin{cases} 20y_2 + 200y_3 = 5 \\ 50y_2 + 100y_3 = 5 \end{cases}$$

Отсюда получаем:

$$y_2 = 0.06;$$

$$y_3 = 0.02.$$

Решение, найденное из первой теоремы двойственности равнозначно решению из второй теоремы.

$$\begin{aligned} g(\bar{y}) &= (\bar{b}, \bar{y}) = 500y_1 + 510y_2 + 3100y_3 = \\ &= 500 * 0 + 510 * 0.06 + 3100 * 0.02 = 90 \\ \min g(\bar{y}) &= 90 \end{aligned}$$

Таким образом, вторая теорема дает нахождение оптимального решения двойственной задачи, пользуясь условием обращения в равенство сопряженных неравенств в системах ограничения.

### 6.2.3 Третья теорема двойственности

Из теоремы об оценках вытекает, что при малом изменении правой части  $i$ -го ограничения в системе ограничений ЗЛП максимальное значение целевой функции изменяется на величину.

$$\Delta G_{max}^i \approx y_i^* \times \Delta b_i$$

Оценим чувствительность решения задачи о максимальном доходе к изменению запасов сырья и спроса на продукцию.

Оптимальное решение двойственной задачи равно:

$$\overline{y}^* = (y_1^*, y_2^*, y_3^*) = (0; 0.06; 0.02)$$

Мы предположим, что спрос увеличился на единицу.

$$\Delta G_{max_1} = y_1 \times \Delta b_1 = 0 \times 1 = 0$$

Для двойственных оценок  $y_2$  и  $y_3$  имеем:

$$\Delta G_{max_2} = y_2 \times \Delta b_2 = 0.06 \times 1 = 0.06$$

$$\Delta G_{max_3} = y_3 \times \Delta b_3 = 0.02 \times 1 = 0.02$$

Это означает, что если увеличить запасы ингредиентов А и В на 1, то данное решение приведет к увеличению значения целевой функции, следовательно, возрастет доход.

$$G_{max_2} + \Delta G_{max_2} = 90 + 0.06 = 90.06$$

$$G_{max_3} + \Delta G_{max_3} = 90 + 0.02 = 90.02$$

Таким образом, снова подтверждается, что запасы ингредиентов А и В полностью используются в оптимальном плане, являются дефицитными и сдерживают рост целевой функции.

Из теоремы также вытекает, что если изменится объем каждого ресурса на величину  $\Delta b_i (i = \overline{1, m})$ , то эти изменения приведут к суммарному изменению прибыли  $\Delta G_{max}$ , которое может быть вычислено по формуле:

$$\Delta G_{max} \approx \sum_{i=1}^m y_i^* \times \Delta b_i$$

Эта формула имеет место лишь тогда, когда при изменении величин  $b_i$  значения переменных  $y_i^*$  в оптимальном плане соответствующей двойственной задачи остаются неизменными, поэтому представляет интерес определить такие интервалы изменения каждого из свободных членов  $b_i$ , в которых оптимальный план двойственной задачи не меняется. Такие интервалы называют интервалами устойчивости двойственных оценок.

Нижнюю и верхнюю границы интервала  $(b_i - \Delta b_i^H, b_i + \Delta b_i^B)$  устойчивости двойственных оценок определяют по формулам:

$$\Delta b_i^H = \begin{cases} -\infty, \text{ если } \forall d_{ji} \leq 0, \\ \min_j \left\{ \frac{x_j^*}{d_{ji}} \right\}, \text{ для } d_{ji} > 0 \end{cases}$$

$$\Delta b_i^B = \begin{cases} \left| \max_j \left\{ \frac{x_j^*}{d_{ji}} \right\} \right|, \text{ если } d_{ji} < 0, \\ +\infty, \text{ если } \forall d_{ji} \geq 0 \end{cases}$$

Для удобства проведения расчета выпишем необходимые элементы из прямой задачи о максимальном доходе:

1. Обратная матрица базиса оптимального плана:

$$D^{-1} = (y_2^*, y_1^*, y_3^*) = \begin{pmatrix} 0 & 0.02 & -0.002 \\ 0 & -0.01 & 0.006 \\ 1 & -1.62 & 0.11 \end{pmatrix}.$$

2. Индексы базисных переменных оптимального плана:

$$\overline{A_0^*} = (x_2^*, x_1^*, x_3^*) = \begin{pmatrix} 5 \\ 13 \\ 20 \end{pmatrix}.$$

3. Свободные члены неравенств (ограничений) прямой задачи:

$$\overline{A_0} = (b_1, b_2, b_3) = \begin{pmatrix} 500 \\ 510 \\ 3100 \end{pmatrix}.$$

### Ресурс 1

Находим нижнюю границу:

$$\Delta b_1^H = \min \left\{ \frac{5}{0.02} \right\} = 250$$

Находим верхнюю границу:

$$\Delta b_1^B = \begin{cases} \left| \max \left\{ -\frac{13}{0.01} \right\} \right| = |-1300| = 1300 \\ \left| \max \left\{ -\frac{20}{1.62} \right\} \right| = |-12| = 12 \end{cases}$$

Получаем  $\Delta b_1 \in (-250; 1300)$ .

Первый ресурс может изменяться в интервале:

$$(b_1 - \Delta b_1^H, b_1 + \Delta b_1^B) = (500 - 250; 500 + 1300) = (250; 1800).$$

### Ресурс 2

Находим нижнюю границу:

$$\Delta b_2^H = \min \left\{ \frac{20}{1} \right\} = 20$$

Находим верхнюю границу:

$$\Delta b_2^B = +\infty,$$

так как среди элементов первого столбца нет отрицательных значений.

Получаем  $\Delta b_2 \in (-20; +\infty)$ .

Второй ресурс может изменяться в интервале:

$$(b_2 - \Delta b_2^H, b_2 + \Delta b_2^B) = (510 - 20; +\infty) = (490; +\infty).$$

### Ресурс 3

Находим нижнюю границу:

$$\Delta b_3^H = \begin{cases} \min \left\{ \frac{13}{0.006} \right\} = 2167 \\ \min \left\{ \frac{20}{0.11} \right\} = 181 \end{cases}$$

Находим верхнюю границу:

$$\Delta b_3^B = \left| \max \left\{ -\frac{5}{0.002} \right\} \right| = |-2500| = 2500$$

Получаем  $\Delta b_3 \in (-2167; 2500)$ .

Третий ресурс может изменяться в интервале:

$$(b_3 - \Delta b_3^H, b_3 + \Delta b_3^B) = (3100 - 2167; 3100 + 2500) = (933; 5600).$$

Далее оценим влияние изменения объема ресурсов на величину максимальной стоимости продукции. Как известно, это дефицитные ресурсы  $y_1^* = 0.06$  и  $y_2^* = 0.02$ .

$$\Delta G_{\max_1} = y_1 \times \Delta b_1^B = 1300 \times 0.06 = 78$$

$$\Delta G_{\max_2} = y_2 \times \Delta b_2^B = 2167 \times 0.02 = 43$$

Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции  $G_{\max}$  на величину:

$$\Delta G_{\max} = \Delta G_{\max_1} + \Delta G_{\max_2} = 78 + 43 = 121$$

Следовательно, оптимальное значение целевой функции при максимальном изменении ресурсов:

$$G_{\max} \approx 90 + 121 = 211.$$

Таким образом, двойственные оценки позволяют судить о чувствительности решения к изменениям.

Из теоремы об оценках вытекает, что при малом изменении правой части  $i$ -го ограничения в системе ограничений ЗЛП максимальное значение целевой функции изменяется на величину.

$$\Delta G_{\max}^i \approx y_i^* \times \Delta b_i$$

Оценим чувствительность решения задачи о максимальном доходе к изменению запасов сырья и спроса на продукцию.

Оптимальное решение двойственной задачи равно:

$$\overline{y}^* = (y_1^*, y_2^*, y_3^*) = (0; 0.06; 0.02)$$

Мы предположим, что спрос увеличился на единицу.

$$\Delta G_{\max_1} = y_1 \times \Delta b_1 = 0 \times 1 = 0$$

Для двойственных оценок  $y_2$  и  $y_3$  имеем:

$$\Delta G_{max_2} = y_2 \times \Delta b_2 = 0.06 \times 1 = 0.06$$

$$\Delta G_{max_3} = y_3 \times \Delta b_3 = 0.02 \times 1 = 0.02$$

Это означает, что если увеличить запасы ингредиентов А и В на 1, то данное решение приведет к увеличению значения целевой функции, следовательно, возрастет доход.

$$G_{max_2} + \Delta G_{max_2} = 90 + 0.06 = 90.06$$

$$G_{max_3} + \Delta G_{max_3} = 90 + 0.02 = 90.02$$

Таким образом, снова подтверждается, что запасы ингредиентов А и В полностью используются в оптимальном плане, являются дефицитными и сдерживают рост целевой функции.

Из теоремы также вытекает, что если изменится объем каждого ресурса на величину  $\Delta b_i (i = \overline{1, m})$ , то эти изменения приведут к суммарному изменению прибыли  $\Delta G_{max}$ , которое может быть вычислено по формуле:

$$\Delta G_{max} \approx \sum_{i=1}^m y_i^* \times \Delta b_i$$

Эта формула имеет место лишь тогда, когда при изменении величин  $b_i$  значения переменных  $y_i^*$  в оптимальном плане соответствующей двойственной задачи остаются неизменными, поэтому представляет интерес определить такие интервалы изменения каждого из свободных членов  $b_i$ , в которых оптимальный план двойственной задачи не меняется. Такие интервалы называют интервалами устойчивости двойственных оценок.

Нижнюю и верхнюю границы интервала  $(b_i - \Delta b_i^H, b_i + \Delta b_i^B)$  устойчивости двойственных оценок определяют по формулам:

$$\Delta b_i^H = \begin{cases} -\infty, \text{ если } \forall d_{ji} \leq 0, \\ \min_j \left\{ \frac{x_j^*}{d_{ji}} \right\}, \text{ для } d_{ji} > 0 \end{cases}$$

$$\Delta b_i^B = \begin{cases} \left| \max_j \left\{ \frac{x_j^*}{d_{ji}} \right\} \right|, \text{ если } d_{ji} < 0, \\ +\infty, \text{ если } \forall d_{ji} \geq 0 \end{cases}$$

Для удобства проведения расчета выпишем необходимые элементы из прямой задачи о максимальном доходе:

4. Обратная матрица базиса оптимального плана:

$$D^{-1} = (y_2^*, y_1^*, y_3^*) = \begin{pmatrix} 0 & 0.02 & -0.002 \\ 0 & -0.01 & 0.006 \\ 1 & -1.62 & 0.11 \end{pmatrix}.$$

5. Индексы базисных переменных оптимального плана:

$$\overline{A}_0^* = (x_2^*, x_1^*, x_3^*) = \begin{pmatrix} 5 \\ 13 \\ 20 \end{pmatrix}.$$

6. Свободные члены неравенств (ограничений) прямой задачи:

$$\overline{A}_0 = (b_1, b_2, b_3) = \begin{pmatrix} 500 \\ 510 \\ 3100 \end{pmatrix}.$$

### Ресурс 1

Находим нижнюю границу:

$$\Delta b_1^H = \min \left\{ \frac{5}{0.02} \right\} = 250$$

Находим верхнюю границу:

$$\Delta b_1^B = \begin{cases} \left| \max \left\{ -\frac{13}{0.01} \right\} \right| = |-1300| = 1300 \\ \left| \max \left\{ -\frac{20}{1.62} \right\} \right| = |-12| = 12 \end{cases}$$

Получаем  $\Delta b_1 \in (-250; 1300)$ .

Первый ресурс может изменяться в интервале:

$$(b_1 - \Delta b_1^H, b_1 + \Delta b_1^B) = (500 - 250; 500 + 1300) = (250; 1800).$$

### Ресурс 2

Находим нижнюю границу:

$$\Delta b_2^H = \min \left\{ \frac{20}{1} \right\} = 20$$

Находим верхнюю границу:

$$\Delta b_2^B = +\infty,$$

так как среди элементов первого столбца нет отрицательных значений.

Получаем  $\Delta b_2 \in (-20; +\infty)$ .

Второй ресурс может изменяться в интервале:

$$(b_2 - \Delta b_2^H, b_2 + \Delta b_2^B) = (510 - 20; +\infty) = (490; +\infty).$$

### Ресурс 3

Находим нижнюю границу:

$$\Delta b_3^H = \begin{cases} \min \left\{ \frac{13}{0.006} \right\} = 2167 \\ \min \left\{ \frac{20}{0.11} \right\} = 181 \end{cases}$$

Находим верхнюю границу:

$$\Delta b_3^B = \left| \max \left\{ -\frac{5}{0.002} \right\} \right| = |-2500| = 2500$$

Получаем  $\Delta b_3 \in (-2167; 2500)$ .

Третий ресурс может изменяться в интервале:

$$(b_3 - \Delta b_3^H, b_3 + \Delta b_3^B) = (3100 - 2167; 3100 + 2500) = (933; 5600).$$

Далее оценим влияние изменения объема ресурсов на величину максимальной стоимости продукции. Как известно, это дефицитные ресурсы  $y_1^* = 0.06$  и  $y_2^* = 0.02$ .

$$\Delta G_{max_1} = y_1 \times \Delta b_1^B = 1300 \times 0.06 = 78$$

$$\Delta G_{max_2} = y_2 \times \Delta b_2^B = 2167 \times 0.02 = 43$$

Совместное влияние изменений этих ресурсов приводит к изменению максимальной стоимости продукции  $G_{max}$  на величину:

$$\Delta G_{max} = \Delta G_{max_1} + \Delta G_{max_2} = 78 + 43 = 121$$

Следовательно, оптимальное значение целевой функции при максимальном изменении ресурсов:

$$G_{max} \approx 90 + 121 = 211.$$

Таким образом, двойственные оценки позволяют судить о чувствительности решения к изменениям.



## 6.3 Программная реализация

Результат программной реализации представлен на Рисунке 6.3.1 – Рисунке 6.3.5. Реализация программы представлена в Приложении Г.

The screenshot shows a software window titled "Simplex Method". It contains two input tables. The first table is for constraints, with columns for variables (x1, x2, x3), a sign, and a right-hand side value. The second table is for the objective function, with columns for variables (x1, x2, x3) and a coefficient (C). Below the tables are buttons for "Default", "OK", "GO", and "CLEAR".

x1	x2	x3	Sign	
10	20	200	>=	5
70	50	100	>=	5

x1	x2	x3	C
500	510	3100	0

Рисунок 6.3.1 – Ввод исходных данных

i	Basis	C	B	A1	A2	A3	A4	A5	A6	A7
1	A6	-M	5	10	20	200	-1	0	1	0
2	A7	-M	5	70	50	100	0	-1	0	1
m+1	F	$\Delta_j$	0	500	510	3100	0	0	0	0
m+2	M	$\Delta_j$		-80	-70	-300	1	1	0	0

Рисунок 6.3.2 – Базисная таблица

i	Basis	C	B	A1	A2	A3	A4	A5	A6	A7
1	A3	-31...	0,02	0,05	0,1	1	0	0	0	0
2	A7	-M	2,5	65	40	0	0,5	-1	-0,5	1
m+1	F	$\Delta_j$	-77,5	345	200	0	15,5	0	-15,5	0
m+2	M	$\Delta_j$		-65	-40	0	-0,5	1	1,5	0

Рисунок 6.3.3 – Итерация №0

i	Basis	C	B	A1	A2	A3	A4	A5	A6	A7
1	A3	-31...	0,02	0	0,07	1	-0,01	0	0,01	0
2	A1	-500	0,04	1	0,62	0	0,01	-0,02	-0,01	0,02
m+1	F	$\Delta_j$	-90,...	0	-12,...	0	12,85	5,31	-12,...	-5,31
m+2	M	$\Delta_j$		0	0	0	0	0	1	1

Рисунок 6.3.4 – Итерация №1

i	Basis	C	B	A1	A2	A3	A4	A5	A6	A7
1	A3	-31...	0,02	-0,11	0	1	-0,01	0	0,01	0
2	A2	-510	0,06	1,62	1	0	0,01	-0,02	-0,01	0,02
m+1	F	$\Delta_j$	-90	20	0	0	13	5	-13	-5

The optimal solution is found:  $F_{\min} = -90$

**Рисунок 6.3.5 – Результат двойственного метода**

## 7 ОПИСАНИЕ АЛГОРИТМА ТРАНСПОРТНОЙ ЗАДАЧИ

Сущность транспортной задачи заключается в нахождении наиболее рационального плана перевозки однородного продукта из пунктов нахождения в пункты потребления. В общем виде транспортная задача формулируется следующим образом. У нас есть пункты отправления продукта, на которых сосредоточено соответственно количество продукта. И есть пункты потребления, потребность которых в этом продукте. Нам требуется найти оптимальное распределение поставок [3].

### 7.1 Постановка задачи

Требуется составить план перевозок, при котором общая стоимость доставки продукции будет наименьшей. Условие задачи представлено в Таблица 7.1.

Таблица 7.1 – Условие задачи

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	7	4	40
$A_2$	7	4	9	7	25
$A_3$	6	14	8	10	35
Потребности	15	40	30	15	0

### 7.2 Расчетная часть

Число пунктов отправления  $m = 3$ , а число пунктов назначения  $n = 4$ . Следовательно, опорный план задачи определяется числами, стоящими в  $m + n - 1 = 3 + 4 - 1 = 6$  заполненных клетках таблицы.

Запасы поставщиков:

$$\sum A_i = 40 + 25 + 35 = 100 \text{ единиц.}$$

Потребность потребителей

$$\sum B_i = 15 + 40 + 30 + 15 = 100 \text{ единиц.}$$

$$\sum A_i = \sum B_i$$

Мы имеем закрытую модель, или модель, удовлетворяющую условию баланса. В этой модели суммарный объем груза у поставщиков равен суммарному спросу потребителей.

Процедура нахождения оптимального плана транспортной задачи имеет два этапа [1]. На первом находят опорный план транспортной задачи. Далее последовательно улучшают найденный опорный план до получения оптимального плана. Чтобы найти опорный план, мы будем использовать метод северо-западного угла, и метод минимальной стоимости.

### 7.2.1 Метод северо-западного угла

В этом методе, прежде всего, заполняются клетки первой горизонтальной строки, начиная с левой верхней клетки («северо-западного угла таблицы»), пока не будут исчерпаны запасы  $a_1$ . Затем заполняют клетки второй строки, начиная с той, которая имеет номер, аналогичный номеру последней заполненной клетки первой горизонтальной строки до полного исчерпывания запасов  $a_2$  и т.д.

План начинает заполняться с верхнего левого угла. Искомый элемент равен  $c_{11} = 10$ . Для этого элемента запасы равны 40, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.1.1).

$$x_{11} = \min(40, 15) = 15$$

Таблица 7.2.1.1 – Первый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	7	4	25
$A_2$	x	4	9	7	25
$A_3$	x	14	8	10	35
Потребности	0	40	30	15	100

Искомый элемент равен  $c_{12} = 5$ . Для этого элемента запасы равны 25, а потребности 40. Поскольку минимальным является 25, то вычитаем его (Таблица 7.2.1.2).

$$x_{12} = \min(25, 40) = 25$$

Таблица 7.2.1.2 – Второй этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	x	x	0
$A_2$	x	4	9	7	25
$A_3$	x	14	8	10	35
Потребности	0	15	30	15	100

Искомый элемент равен  $c_{22} = 4$ . Для этого элемента запасы равны 25, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.1.3).

$$x_{22} = \min(25, 15) = 15$$

Таблица 7.2.1.3 – Третий этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	x	x	0
$A_2$	x	4	9	7	10
$A_3$	x	x	8	10	35
Потребности	0	0	30	15	100

Искомый элемент равен  $c_{23} = 9$ . Для этого элемента запасы равны 10, а потребности 30. Поскольку минимальным является 10, то вычитаем его (Таблица 7.2.1.4).

$$x_{23} = \min(10, 30) = 10$$

Таблица 7.2.1.4 – Четвертый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	x	x	0
$A_2$	x	4	9	x	0

Продолжение Таблицы 7.2.1.4

$A_3$	x	x	8	10	35
<b>Потребности</b>	0	0	20	15	100

Искомый элемент равен  $c_{33} = 8$ . Для этого элемента запасы равны 35, а потребности 20. Поскольку минимальным является 20, то вычитаем его (Таблица 7.2.1.5).

$$x_{33} = \min(35, 20) = 20$$

Таблица 7.2.1.5 – Пятый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	x	x	0
$A_2$	x	4	9	x	0
$A_3$	x	x	8	10	15
<b>Потребности</b>	0	0	0	15	100

Искомый элемент равен  $c_{34} = 10$ . Для этого элемента запасы равны 15, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.1.6).

$$x_{34} = \min(15, 15) = 15$$

Таблица 7.2.1.6 – Шестой этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	x	x	0
$A_2$	x	4	9	x	0
$A_3$	x	x	8	10	0
<b>Потребности</b>	0	0	0	0	0

Результат всех преобразований представлен в Таблице 7.2.1.7.

Таблица 7.2.1.7 – Оптимальный опорный план

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [15]	5 [25]	7	4	40

Продолжение Таблицы 7.2.1.7

$A_2$	7	4 [15]	9 [10]	7	25
$A_3$	6	14	8 [20]	10 [15]	35
<b>Потребности</b>	15	40	30	15	100

В результате был получен опорный план, который является допустимым, так как все грузы вывезены, потребность потребителей удовлетворена, план соответствует системе ограничений транспортной задачи.

Подсчитаем число занятых клеток, их 6, а должно быть:

$$m + n - 1 = 6.$$

Опорный план является невырожденным. Общая стоимость перевозок груза составляет:

$$f_0 = 10 * 15 + 5 * 25 + 4 * 15 + 9 * 10 + 8 * 20 + 10 * 15 = 735$$

## 7.2.2 Метод минимальной стоимости

Суть метода заключается в том, что из всей таблицы стоимости  $c_{ij}$  выбирают наименьшую, и в клетку которая ей соответствует, помещают меньшие числа из  $a_i$  и  $b_j$ . Затем из рассмотрения мысленно исключают либо строку, соответствующую поставщику, запасы которого полностью израсходованы, либо столбец, соответствующий потребителю, потребности которого полностью удовлетворены, либо строку и столбец, если израсходованы запасы поставщика и удовлетворены потребности потребителя. Из оставшейся части таблицы стоимостей снова выбирают наименьшую стоимость, и процесс распределения запасов продолжают, пока все запасы не будут распределены, а потребности удовлетворены [2].

Искомый элемент равен  $c_{14} = 4$ . Для этого элемента запасы равны 40, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.2.1).

$$x_{14} = \min(40, 15) = 15$$

Таблица 7.2.2.1 – Первый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	7	4	25
$A_2$	7	4	9	x	25
$A_3$	6	14	8	x	35
Потребности	15	40	30	0	0

Искомый элемент равен  $c_{22} = 4$ . Для этого элемента запасы равны 25, а потребности 40. Поскольку минимальным является 25, то вычитаем его (Таблица 7.2.2.2).

Таблица 7.2.2.2 – Второй этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	7	4	25
$A_2$	x	4	x	x	0
$A_3$	6	14	8	x	35
Потребности	15	15	30	0	0

Искомый элемент равен  $c_{12} = 5$ . Для этого элемента запасы равны 25, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.2.3).

$$x_{12} = \min(25, 15) = 15$$

Таблица 7.2.2.3 – Третий этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5	7	4	10
$A_2$	x	4	x	x	0
$A_3$	6	x	8	x	35
Потребности	0	0	30	0	0

Искомый элемент равен  $c_{31} = 6$ . Для этого элемента запасы равны 35, а потребности 15. Поскольку минимальным является 15, то вычитаем его (Таблица 7.2.2.4).

$$x_{31} = \min(35, 15) = 15$$



Таблица 7.2.2.4 – Четвертый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	x	5	7	4	10
$A_2$	x	4	x	x	0
$A_3$	6	x	8	x	20
Потребности	0	0	30	0	0

Искомый элемент равен  $c_{13} = 7$ . Для этого элемента запасы равны 10, а потребности 30. Поскольку минимальным является 10, то вычитаем его (Таблица 7.2.2.5).

$$x_{13} = \min(10, 30) = 10$$

Таблица 7.2.2.5 – Пятый этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	x	5	7	4	0
$A_2$	x	4	x	x	0
$A_3$	6	x	8	x	20
Потребности	0	0	20	0	0

Искомый элемент равен  $c_{33} = 8$ . Для этого элемента запасы равны 20, а потребности 20. Поскольку минимальным является 20, то вычитаем его (Таблица 7.2.2.6).

$$x_{33} = \min(20, 20) = 20$$

Таблица 7.2.2.6 – Шестой этап

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	x	5	7	4	0
$A_2$	x	4	x	x	0
$A_3$	6	x	8	x	0
Потребности	0	0	0	0	0

Результат всех преобразований представлен в Таблице 7.2.2.7.

Таблица 7.2.2.7 – Оптимальный опорный план

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5 [15]	7 [10]	4 [15]	40
$A_2$	7	4 [25]	9	7	25
$A_3$	6 [15]	14	8 [20]	10	35
Потребности	15	40	30	15	100

В результате был получен опорный план, который является допустимым, так как все грузы вывезены, потребность потребителей удовлетворена, план соответствует системе ограничений транспортной задачи.

Подсчитаем число занятых клеток, их 6, а должно быть:

$$m + n - 1 = 6.$$

Опорный план является невырожденным. Общая стоимость перевозок груза составляет:

$$f_0 = 5 * 15 + 7 * 10 + 4 * 15 + 4 * 25 + 6 * 15 + 8 * 20 = 555$$

### 7.2.3 Метод потенциалов

Метод потенциалов является модификацией симплекс-метода решения задачи линейного программирования применительно к транспортной задаче. Он позволяет, отправляясь от некоторого опорного решения, получить оптимальное решение за конечное число итераций [3].

#### 1. Определение исходного плана перевозок.

Для составления исходного плана перевозок используем метод северо-западного угла (Таблица 7.2.3.1). Общее количество базисных клеток равно 6.

Таблица 7.2.3.1 – Опорный план по методу северо-западного угла

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	

Продолжение Таблицы 7.2.3.1

$A_1$	10 [15]	5 [25]	7	4	40
$A_2$	7	4 [15]	9 [10]	7	25
$A_3$	6	14	8 [20]	10 [15]	35
<b>Потребности</b>	15	40	30	15	100

Стоимость перевозок по этому плану равна:

$$f_0 = 10 * 15 + 5 * 25 + 4 * 15 + 9 * 10 + 8 * 20 + 10 * 15 = 735.$$

## 2. Исследование базисного решения на оптимальность.

Проверим оптимальность опорного плана. Найдем предварительные потенциалы:

$U_i$  — потенциал поставщика;

$V_j$  — потенциал потребителя.

Последовательно найдем значения потенциалов. Значение одного потенциала необходимо задать. Пусть  $U_1 = 0$ .

$$A_1B_1: V_1 + U_1 = 10 \quad V_1 = 10$$

$$A_1B_2: V_2 + U_1 = 5 \quad V_2 = 5$$

$$A_2B_2: V_2 + U_2 = 4 \quad U_2 = -1$$

$$A_2B_3: V_3 + U_2 = 9 \quad V_3 = 10$$

$$A_3B_3: V_3 + U_3 = 8 \quad U_3 = -2$$

$$A_3B_4: V_4 + U_3 = 10 \quad V_4 = 12$$

Полученные числа заключаем в рамки и записываем их в соответствующие клетки, Таблица 7.2.3.2.

Таблица 7.2.3.2 – Результат исследования на оптимальность

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	

Продолжение Таблицы 7.2.3.2

$A_1$	10 [15]	5 [25]	7	4	40	$U_1 = 0$
$A_2$	7	4 [15]	9 [10]	7	25	$U_2 = -1$
$A_3$	6	14	8 [20]	10 [15]	35	$U_3 = -2$
<b>Потребности</b>	$V_1 = 10$	$V_2 = 5$	$V_3 = 10$	$V_4 = 12$		

### 3. Определение нового базисного решения

Данный опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых  $U_i + V_j > C_{ij}$ .

Найдем оценки незадействованных маршрутов ( $C_{ij}$  – стоимость доставки):

$$A_1B_3: \Delta_{13} = C_{13} - (U_1 + V_3) = 3$$

$$A_1B_4: \Delta_{14} = C_{14} - (U_1 + V_4) = 8$$

$$A_2B_1: \Delta_{21} = C_{21} - (U_2 + V_1) = 2$$

$$A_2B_4: \Delta_{24} = C_{24} - (U_2 + V_4) = 4$$

$$A_3B_1: \Delta_{31} = C_{31} - (U_3 + V_1) = 2$$

$$A_3B_2: \Delta_{32} = C_{32} - (U_3 + V_2) = -11$$

Наибольшая оценка свободной клетки 8 находится на пересечении строки  $A_1$  и столбца  $B_4$ . Для данной свободной клетки строим цикл пересчета (Таблица 7.2.3.3).

Таблица 7.2.3.3 – Цикл перерасчета

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [15]	5 [25][-]	7	4 [+]	40
$A_2$	7	4 [15][+]	9 [10][-]	7	25
$A_3$	6	14	8 [20][+]	10 [15][-]	35

Продолжение Таблицы 7.2.3.3

<b>Потребности</b>	15	40	30	15	100
--------------------	----	----	----	----	-----

Наименьшее из чисел в минусовых клетках равно 10. Клетка, в которой находится это число становится свободной (Таблица 1.2.3.4).

Таблица 1.2.3.4 – Новое базисное решение

<b>Пункты отправления</b>	<b>Пункты назначения</b>				<b>Запасы</b>
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [15]	5 [15]	7	4 [10]	40
$A_2$	7	4 [25]	9	7	25
$A_3$	6	14	8 [30]	10 [5]	35
<b>Потребности</b>	15	40	30	15	100

Общая стоимость перевозок груза по этому плану составляет:

$$f_0 = 10 * 15 + 5 * 15 + 4 * 10 + 4 * 25 + 8 * 30 + 10 * 5 = 655$$

#### 4. Исследование базисного решения на оптимальность.

Проверим полученный опорный план на оптимальность. Последовательно найдем значения потенциалов. Значение одного потенциала необходимо задать.

Пусть  $U_1 = 0$ .

$$\begin{array}{ll}
 A_1B_1: V_1 + U_1 = 10 & V_1 = 10 \\
 A_1B_2: V_2 + U_1 = 5 & V_2 = 5 \\
 A_1B_4: V_4 + U_1 = 4 & V_4 = 4 \\
 A_2B_2: V_2 + U_2 = 4 & U_2 = -1 \\
 A_3B_3: V_3 + U_3 = 8 & U_3 = 6 \\
 A_3B_4: V_4 + U_3 = 10 & V_3 = 2
 \end{array}$$

Полученные числа заключаем в рамки и записываем их в соответствующие клетки (Таблица 7.2.3.5).

Таблица 7.2.3.5 – Результат исследования на оптимальность

Пункты отправления	Пункты назначения				Запасы	
	$B_1$	$B_2$	$B_3$	$B_4$		
$A_1$	10 [15]	5 [15]	7	4 [10]	40	$U_1 = 0$
$A_2$	7	4 [25]	9	7	25	$U_2 = -1$
$A_3$	6	14	8 [30]	10 [5]	35	$U_3 = 6$
Потребности	$V_1 = 10$	$V_2 = 5$	$V_3 = 2$	$V_4 = 4$		

## 5. Определение нового базисного решения.

Данный опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых  $U_i + V_j > C_{ij}$ .

Найдем оценки незадействованных маршрутов ( $C_{ij}$  – стоимость доставки):

$$A_1B_3: \Delta_{13} = C_{13} - (U_1 + V_3) = 5$$

$$A_2B_1: \Delta_{21} = C_{21} - (U_2 + V_1) = -2$$

$$A_2B_3: \Delta_{23} = C_{23} - (U_2 + V_3) = 8$$

$$A_2B_4: \Delta_{24} = C_{24} - (U_2 + V_4) = 4$$

$$A_3B_1: \Delta_{31} = C_{31} - (U_3 + V_1) = -10$$

$$A_3B_2: \Delta_{32} = C_{32} - (U_3 + V_2) = 3$$

Наибольшая оценка свободной клетки 6 находится на пересечении строки  $A_3$  и столбца  $B_1$ . Для данной свободной клетки строим цикл пересчета (Таблица 7.2.3.6).

Таблица 7.2.3.6 – Цикл перерасчета

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [15][-]	5 [15]	7	4 [10][+]	40
$A_2$	7	4 [25]	9	7	25

Продолжение Таблицы 7.2.3.6

$A_3$	6 [+]	14	8 [30]	10 [5][-]	35
<b>Потребности</b>	15	40	30	15	100

Наименьшее из чисел в минусовых клетках равно 5. Клетка, в которой находится это число становится свободной (Таблица 7.2.3.7).

Таблица 7.2.3.7 – Новое базисное решение

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [10]	5 [15]	7	4 [15]	40
$A_2$	7	4 [25]	9	7	25
$A_3$	6 [5]	14	8 [30]	10	35
<b>Потребности</b>	15	40	30	15	100

Общая стоимость перевозок груза по этому плану составляет:

$$f_0 = 10 * 10 + 5 * 15 + 4 * 15 + 4 * 25 + 6 * 5 + 8 * 30 = 605$$

#### 6. Исследование базисного решения на оптимальность.

Проверим полученный опорный план на оптимальность. Последовательно найдем значения потенциалов. Значение одного потенциала необходимо задать.

Пусть  $U_1 = 0$ .

$$A_1B_1: V_1 + U_1 = 10$$

$$V_1 = 10$$

$$A_1B_2: V_2 + U_1 = 5$$

$$V_2 = 5$$

$$A_1B_4: V_4 + U_1 = 4$$

$$V_4 = 4$$

$$A_2B_2: V_2 + U_2 = 4$$

$$U_2 = -1$$

$$A_3B_1: V_1 + U_3 = 6$$

$$U_3 = -4$$

$$A_3B_3: V_3 + U_3 = 8$$

$$V_3 = 12$$

Полученные числа заключаем в рамки и записываем их в соответствующие клетки (Таблица 7.2.3.8).

Таблица 7.2.3.8 – Результат исследования на оптимальность

Пункты отправления	Пункты назначения				Запасы	
	$B_1$	$B_2$	$B_3$	$B_4$		
$A_1$	10 [10]	5 [15]	7	4 [15]	40	$U_1 = 0$
$A_2$	7	4 [25]	9	7	25	$U_2 = -1$
$A_3$	6 [5]	14	8 [30]	10	35	$U_3 = -4$
Потребности	$V_1 = 10$	$V_2 = 5$	$V_3 = 12$	$V_4 = 4$		

#### 7. Определение нового базисного решения.

Данный опорный план не является оптимальным, так как существуют оценки свободных клеток, для которых  $U_i + V_j > C_{ij}$ .

Найдем оценки незадействованных маршрутов ( $C_{ij}$  – стоимость доставки):

$$A_1B_3: \Delta_{13} = C_{13} - (U_1 + V_3) = -5$$

$$A_2B_1: \Delta_{21} = C_{21} - (U_2 + V_1) = -2$$

$$A_2B_3: \Delta_{23} = C_{23} - (U_2 + V_3) = -2$$

$$A_2B_4: \Delta_{24} = C_{24} - (U_2 + V_4) = 4$$

$$A_3B_2: \Delta_{32} = C_{31} - (U_3 + V_1) = 13$$

$$A_3B_4: \Delta_{34} = C_{32} - (U_3 + V_2) = 10$$

Наибольшая оценка свободной клетки 7 находится на пересечении строки  $A_1$  и столбца  $B_3$ . Для данной свободной клетки строим цикл пересчета (Таблица 7.2.3.9).

Таблица 7.2.3.9 – Цикл перерасчета

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10 [10] [-]	5 [15]	7 [+]	4 [15]	40



Продолжение Таблицы 7.2.3.9

$A_2$	7	4 [25]	9	7	25
$A_3$	6 [5][+]	14	8 [30][-]	10	35
<b>Потребности</b>	15	40	30	15	100

Наименьшее из чисел в минусовых клетках равно 10. Клетка, в которой находится это число становится свободной (Таблица 7.2.3.10).

Таблица 7.2.3.10 – Новое базисное решение

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5 [15]	7 [10]	4 [15]	40
$A_2$	7	4 [25]	9	7	25
$A_3$	6 [15]	14	8 [20]	10	35
<b>Потребности</b>	15	40	30	15	100

Общая стоимость перевозок груза по этому плану составляет:

$$f_0 = 5 * 15 + 7 * 10 + 4 * 15 + 4 * 25 + 6 * 15 + 8 * 20 = 555$$

## 8. Исследование базисного решения на оптимальность.

Проверим полученный опорный план на оптимальность. Последовательно найдем значения потенциалов. Значение одного потенциала необходимо задать.

Пусть  $U_1 = 0$ .

$$A_1B_2: V_2 + U_1 = 5$$

$$V_2 = 5$$

$$A_1B_3: V_3 + U_1 = 7$$

$$V_3 = 7$$

$$A_1B_4: V_4 + U_1 = 4$$

$$V_4 = 4$$

$$A_2B_2: V_2 + U_2 = 4$$

$$U_2 = -1$$

$$A_3B_3: V_3 + U_3 = 6$$

$$U_3 = 1$$

$$A_3B_1: V_1 + U_3 = 8$$

$$V_1 = 5$$

Полученные числа заключаем в рамки и записываем их в соответствующие клетки (Таблица 7.2.3.11).

Таблица 7.2.3.11 – Результат исследования на оптимальность

Пункты отправления	Пункты назначения				Запасы	
	$B_1$	$B_2$	$B_3$	$B_4$		
$A_1$	10	5 [15]	7 [10]	4 [15]	40	$U_1 = 0$
$A_2$	7	4 [25]	9	7	25	$U_2 = -1$
$A_3$	6 [15]	14	8 [20]	10	35	$U_3 = 1$
Потребности	$V_1 = 5$	$V_2 = 5$	$V_3 = 7$	$V_4 = 4$		

Найдем оценки незадействованных маршрутов ( $C_{ij}$  – стоимость доставки):

$$A_1B_1: \Delta_{11} = C_{11} - (U_1 + V_1) = 5$$

$$A_2B_1: \Delta_{21} = C_{21} - (U_2 + V_1) = 3$$

$$A_2B_3: \Delta_{23} = C_{23} - (U_2 + V_3) = 3$$

$$A_2B_4: \Delta_{24} = C_{24} - (U_2 + V_4) = 4$$

$$A_3B_2: \Delta_{32} = C_{31} - (U_3 + V_1) = 8$$

$$A_3B_4: \Delta_{34} = C_{32} - (U_3 + V_2) = 5$$

Среди оценок нет отрицательных чисел. Следовательно, данный опорный план является оптимальным (Таблица 7.2.3.12).

Таблица 7.2.3.12 – Оптимальный опорный план

Пункты отправления	Пункты назначения				Запасы
	$B_1$	$B_2$	$B_3$	$B_4$	
$A_1$	10	5 [15]	7 [10]	4 [15]	40
$A_2$	7	4 [25]	9	7	25
$A_3$	6 [15]	14	8 [20]	10	35

Продолжение Таблицы 7.2.3.12

<b>Потребности</b>	15	40	30	15	100
--------------------	----	----	----	----	-----

Общая стоимость перевозок груза по этому плану составляет:

$$f_0 = 5 * 15 + 7 * 10 + 4 * 15 + 4 * 25 + 6 * 15 + 8 * 20 = 555.$$

## 7.3 Программная реализация

Реализация интерфейса программы представлена на Рисунке 7.3.1.

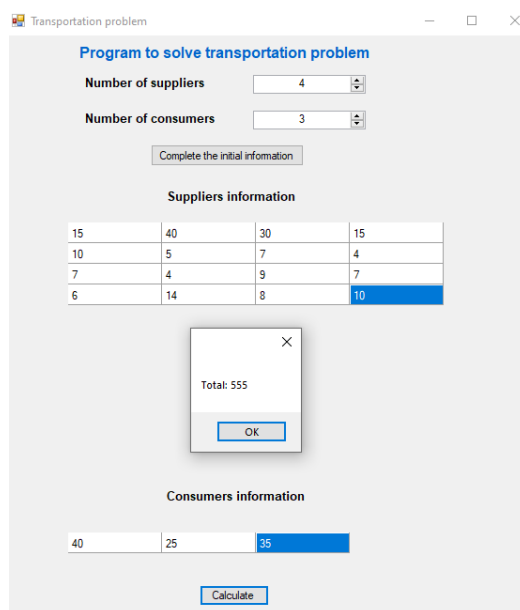


Рисунок 7.3.1 – Интерфейс программы

Реализация работы программы представлена на Рисунке 7.3.2.

	A	B	C	D
1	1 iteration	Consumer need 40	Consumer need 25	Consumer need 35
2	Suppliers stock 15	10	7	6 [15]
3	Suppliers stock 40	5 [15]	4 [25]	14
4	Suppliers stock 30	7 [10]	9	8 [20]
5	Suppliers stock 15	4 [15]	7	10

Рисунок 7.3.2 – Решение задачи

Код реализации представлен в Приложении Д.

## ЗАКЛЮЧЕНИЕ

Любая сфера человеческой деятельности связана с принятием решений. В зависимости от предметной дисциплины, посвященные этой проблематике, называются исследованием операций, экономико-математические методы, математические методы в управлении.

В процессе выполнения курсовой работы я научилась, детально анализировать данные и осознано выбирать наилучший вариант решения задачи. Сама работа заключалась в анализе проблемы, ее идентификации и определении задач, которые помогут в выборе. В моей работе стояла проблема в выборе лучшего варианта телевизора. Для этого была собрана информация, определены альтернативы и их оценки. Был создан план действий, его реализация, и оценка результата.

Проходить все эти этапы, в зависимости от специфики ситуации, можно параллельно, одновременно или с возвратом к пройденным этапам. Прохождение всех этапов должно быть рационально обоснованно. Теория принятия решений говорит и о том, что нужно уметь статистически прогнозировать развитие событий. И для этого необходимо располагать данными и их статистикой.

В результате проделанной работы, были изучены различные методы поиска оптимального варианта. Каждый из них имеет свою особенность, они применяются в разных областях, но при этом их объединяет одно целое, сам механизм анализа и обработки данных, принятие человеком оптимального, наиболее подходящего для него варианта.

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Многокритериальная оптимизация [Электронный ресурс]: метод. указания по вып. курсовой работы / Л. С. Болотова, А. Б. Сорокин. — М.: МИРЭА, 2015. — Электрон. опт. диск (ISO).
2. Методы оптимизации: гибридные генетические алгоритмы [Электронный ресурс]: учебно-метод. пособие / А. Б. Сорокин. — М.: МИРЭА, 2016. — Электрон. опт. диск (ISO).
3. Линейное программирование: практикум [Электронный ресурс]: учебно-метод. пособие / А. Б. Сорокин, Е. В. Бражникова, О. В. Платонова. — М.: МИРЭА, 2017. — Электрон. опт. диск (ISO).

## **ПРИЛОЖЕНИЯ**

Приложение А – Реализация метода Парето

Приложение Б – Реализация метода ЭЛЕКТРА II

Приложение В – Реализация метода МАИ

Приложение Г – Реализация симплекс метода и двойственной задачи

Приложение Д – Реализация транспортной задачи

## Приложение А

### Листинг А.1 – Реализация метода Парето

```
static void ShowList(this List<TV> tv)
{
    Console.WriteLine("№      price      per      year      garant
weight      diag");

    foreach (TV t in tv)
    {
        Console.WriteLine("{0, -5}{1, -10}{2, -10}{3, -10}{4, -
10}{5, -10}{6, -10}", t.Number, t.Price, t.Per, t.Year, t.Garant, t.Weight,
t.Diag);
    }

    Console.WriteLine();
}

static bool IsConsist(this List<TV> tv, int number)
{
    foreach (TV t in tv)
        if (t.Number == number)
            return true;
    return false;
}

static List<TV> PairComparison(this List<TV> tv)
{
    string[,] array = new string[tv.Count, tv.Count];
    List<int> result = new List<int>();
    List<TV> ret = new List<TV>();

    for (int i = 0; i < tv.Count; i++)
        for (int j = 0; j < tv.Count; j++)
            array[i, j] = "x";

    for (int i = 1; i < tv.Count; i++)
        for (int j = 0; j < i; j++)
            if (tv[i].CompareTo(tv[j]))
                array[i, j] = (i + 1).ToString();
            else if (tv[j].CompareTo(tv[i]))
                array[i, j] = (j + 1).ToString();
            else
                array[i, j] = "H";

    if (canShowPareto)
        Console.WriteLine("    1    2    3    4    5    6    7    8    9
10");

    for (int i = 0; i < tv.Count; i++)
        for (int j = 0; j < tv.Count; j++)
        {
            if (canShowPareto)
            {
                if (j == 0) Console.Write((i + 1).ToString() + "
");
                Console.Write(j == 8 ? array[i, j] + "\n" :
array[i, j] + "    ");
            }

            if (Char.IsDigit(array[i, j], 0) &&
!result.Contains(int.Parse(array[i, j]) - 1))
```

*Продолжение Листинга А.1*

```
        result.Add(int.Parse(array[i, j]) - 1);
    }

    Console.WriteLine();

    result.Sort();

    foreach (int r in result)
        ret.Add(tv[r]);

    return ret;
}
```

*Листинг А.2 – Реализация метода верхних/нижних границ критериев*

```
static List<TV> CriteriaBorders(this List<TV> tv,
int excludedCriteria = 0)//критерии, и их характеристики
{
    int minPrice = 200;
    int maxPer = 4;
    int minYear = 2021;
    int maxGarant = 12;
    int minWeight = 40;
    int maxDiag = 191;

    string choice = "";

    if (excludedCriteria != 1)
    {
        Console.Write("Задайте значения границ критериев (или
введите def, чтобы пропустить критерий):\n\nМинимальная цена: ");
        choice = Console.ReadLine();
        try
        {
            minPrice = choice == "def" ? 1000000000 :
int.Parse(choice);
        }
        catch (Exception ex)
        {
            Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
            minPrice = 1000000000;
        }
    }
    if (excludedCriteria != 2)
    {
        Console.Write("Максимальное разрешение: ");
        choice = Console.ReadLine();
        try
        {
            maxPer = choice == "def" ? 0 :
int.Parse(choice.Replace(".", ","));
        }
        catch (Exception ex)
        {
            Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
            maxPer = 0;
        }
    }
    if (excludedCriteria != 3)
```



*Продолжение Листинга А.2*

```
{
    Console.Write("Минимальный год: ");
    choice = Console.ReadLine();
    try
    {
        minYear = choice == "def" ? 0 : int.Parse(choice);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
        minYear = 0;
    }
}
if (excludedCriteria != 4)
{
    Console.Write("Максимальная гарантия: ");
    choice = Console.ReadLine();
    try
    {
        maxGarant = choice == "def" ? 0 : int.Parse(choice);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
        maxGarant = 0;
    }
}
if (excludedCriteria != 5)
{
    Console.Write("Минимальная вес: ");
    choice = Console.ReadLine();
    try
    {
        minWeight = choice == "def" ? 1000000000 :
int.Parse(choice);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
        minWeight = 1000000000;
    }
}
if (excludedCriteria != 6)
{
    Console.Write("Максимальная диагональ: ");
    choice = Console.ReadLine();
    try
    {
        maxDiag = choice == "def" ? 1000000000 :
int.Parse(choice);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
        maxDiag = 1000000000;
    }
}
```

### *Продолжение Листинга А.2*

```
        Console.WriteLine();

        var groupedList = tv.Where(p => p.Price <= minPrice).
                               Where(r => r.Per >= maxPer).
                               Where(y => y.Year <= minYear).
                               Where(g => g.Garant >=
maxGarant).
                               Where(w => w.Weight <=
minWeight).
                               Where(d => d.Diag >= maxDiag);

        List<TV> typedGroupedList = new List<TV>();

        foreach (var gl in groupedList)
        {
            typedGroupedList.Add((TV)gl);
        }

        return typedGroupedList;
    }
}
```

### *Листинг А.3 – Реализация метода субоптимизации*

```
        static int GetSuboptimizationCriteria()//субоптимизация
        {
            Console.Write("Укажите главный критерий (1 - цена, 2 -
разрешение, " +
                           "3 - год, 4 - гарантия, 5 - вес, 6 -
диагональ):\n>> ");

            int choice = 0;

            while (choice != 1 && choice != 2 && choice != 3 && choice !=
4 && choice != 5 && choice != 6)
            {
                try
                {
                    choice = Convert.ToInt32(Console.ReadLine());
                }
                catch (Exception ex)
                {
                    Console.WriteLine("Неправильный формат входных
данных.\n" + ex.Message);
                }
            }

            Console.WriteLine();

            return choice;
        }

        static List<TV> SelectByCriteria(this List<TV> tv, int
suboptimizationCriteria)
        {
            List<TV> result = new List<TV>();

            int index = 0;

            for (int i = 1; i < tv.Count; i++)
            {
                switch (suboptimizationCriteria)
                {

```

```

        case 1:
            if (tv[index].Price > tv[i].Price)
                index = i;
            break;
        case 2:
            if (tv[index].Per < tv[i].Per)
                index = i;
            break;
        case 3:
            if (tv[index].Year > tv[i].Year)
                index = i;
            break;
        case 4:
            if (tv[index].Garant < tv[i].Garant)
                index = i;
            break;
        case 5:
            if (tv[index].Weight > tv[i].Weight)
                index = i;
            break;
        case 6:
            if (tv[index].Diag < tv[i].Diag)
                index = i;
            break;
        default:
            if (tv[index].Price > tv[i].Price)
                index = i;
            break;
    }
}

for (int i = 0; i < tv.Count; i++)
{
    switch (suboptimizationCriteria)
    {
        case 1:
            if (tv[i].Price == tv[index].Price)
                result.Add(tv[i]);
            break;
        case 2:
            if (tv[i].Per == tv[index].Per)
                result.Add(tv[i]);
            break;
        case 3:
            if (tv[i].Year == tv[index].Year)
                result.Add(tv[i]);
            break;
        case 4:
            if (tv[i].Garant == tv[index].Garant)
                result.Add(tv[i]);
            break;
        case 5:
            if (tv[i].Weight == tv[index].Weight)
                result.Add(tv[i]);
            break;
        case 6:
            if (tv[i].Diag == tv[index].Diag)
                result.Add(tv[i]);
            break;
        default:
            if (tv[i].Price == tv[index].Price)

```

### *Продолжение Листинга А.3*

```
                result.Add(tv[i]);
                break;
            }
        }
        return result;
    }
}
```

### *Листинг А.4 – Реализация метода лексической оптимизации*

```
static List<TV> Lexicography(this List<TV> tv) //лексикографический
{
    List<TV> result = tv;

    List<int> criterias = new List<int>() { 1, 2, 3, 4, 5, 6 };

    int criteria;

    while (criterias != null)
    {
        criteria = GetSuboptimizationCriteria();

        if (!criterias.Contains(criteria))
        {
            Console.WriteLine("Критерий уже был использован.\n");
            continue;
        }
        else
        {
            criterias.Remove(criteria);
        }

        result = result.SelectByCriteria(criteria);

        if (result.Count <= 1)
            return result;
        else
            result.ShowList();

        Console.WriteLine("Нажмите ESC, чтобы закончить
оптимизацию, или ENTER, чтобы продолжить.\n");
        while (true)
        {
            ConsoleKey key = Console.ReadKey().Key;
            if (key == ConsoleKey.Escape)
                return result;
            else if (key == ConsoleKey.Enter)
                break;
        }
    }

    return result;
}
}
```

## Приложение Б

### Листинг Б.1 - Реализация первого этапа

```
static double[,] GetPreferenceMatrix(this List<TV> tv, List<Criteria>
criterias)//часть с расчетами
{
    TV t1, t2;
    double P = 0, N = 0;

    double[,] pMatrix = new double[tv.Count, tv.Count];

    for (int i = 0; i < tv.Count; i++)
    {
        t1 = tv[i];

        for (int j = i + 1; j < tv.Count; j++)
        {
            t2 = tv[j];

            for (int k = 0; k < criterias.Count; k++)
            {
                if (criterias[k].Vector)
                {
                    if (t1.Criterias[k] > t2.Criterias[k])
                        P += criterias[k].Weight;
                    else if (t1.Criterias[k] < t2.Criterias[k])
                        N += criterias[k].Weight;
                }
                else
                {
                    if (t1.Criterias[k] < t2.Criterias[k])
                        P += criterias[k].Weight;
                    else if (t1.Criterias[k] > t2.Criterias[k])
                        N += criterias[k].Weight;
                }
            }

            if (N != 0 && P / N > 1)
            {
                pMatrix[i, j] = Math.Round(P / N, 2);
                pMatrix[j, i] = 0;
            }
            else if (P != 0 && N / P > 1)
            {
                pMatrix[i, j] = 0;
                pMatrix[j, i] = Math.Round(N / P, 2);
            }
            else
            {
                pMatrix[i, j] = 0;
                pMatrix[j, i] = 0;
            }

            P = 0;
            N = 0;
        }
    }

    return pMatrix;
}
```

### *Листинг Б.2 - Реализация второго этапа*

```
static void ShowPreferenceMatrix(double[,] pMatrix)
{
    if (Math.Sqrt(pMatrix.Length) > 20)
        return;
    for (int i = 0; i < Math.Sqrt(pMatrix.Length); i++)
        if (i == 0)
            Console.Write("      {0, -8}", i + 1);
        else
            Console.Write("{0, -8}", i + 1);

    Console.WriteLine();

    for (int i = 0; i < Math.Sqrt(pMatrix.Length); i++)
    {
        Console.Write("{0, -4} ", i + 1);

        for (int j = 0; j < Math.Sqrt(pMatrix.Length); j++)
            Console.Write("{0, -8}", pMatrix[i, j]);

        Console.WriteLine();
    }
}
```

### *Листинг Б.3 – Реализация построения графа на 1 этапе*

```
static void ShowLevels(double[,] pMatrix)
{
    bool isChanged = false;

    int[] levels = new
int[Convert.ToInt32(Math.Sqrt(pMatrix.Length))];

    for (int i = 0; i < levels.Length; i++)
        levels[i] = 0;

    List<int> levelPeaks = null;

    levelPeaks = GetLevelVertices(pMatrix, 0);

    foreach (int lp in levelPeaks)
    {
        for (int j = 0; j < levels.Length; j++)
            if (pMatrix[lp, j] != 0 && levels[lp] <= levels[j])
                levels[j] = levels[lp] + 1;
    }

    for (int i = 1; i < levels.Length; i++)
    {
        levelPeaks.Clear();
        Console.Write("Уровень: " + i + "; ");
        for (int j = 0; j < levels.Length; j++)
            if (levels[j] == i)
            {
                levelPeaks.Add(j);
                Console.Write((j + 1) + " ");
            }
        Console.WriteLine();

        if (levelPeaks.Count == 0)
            break;
    }
}
```

### *Продолжение Листинга Б.3*

```
        foreach (int lp in levelPeaks)
        {
            for (int j = 0; j < levels.Length; j++)
                if (pMatrix[lp, j] != 0 && levels[lp] >= levels[j])
                    levels[j] = levels[lp] + 1;

            Console.Write("[ " + (lp + 1) + " ] ");
            for (int k = 0; k < levels.Length; k++)
                Console.Write(levels[k] + " ");
            Console.WriteLine();
        }
    }

    Console.WriteLine("\n\nУровни графа: ");
    for (int i = 0; i <= levels.Max(); i++)
    {
        Console.Write("Уровень " + (i + 1) + ": ");
        for (int j = 0; j < levels.Length; j++)
        {
            if (levels[j] == i)
                Console.Write((j + 1) + " ");
        }
        Console.WriteLine();
    }
}
```

### *Листинг Б.4 - Реализация построения графа на 2 этапе*

```
static List<int> GetLevelVertices(double[,] pMatrix, int level)
{
    List<int> zeros = new List<int>();
    int count = 0;

    for (int i = 0; i < Math.Sqrt(pMatrix.Length); i++)
    {
        count = 0;
        for (int j = 0; j < Math.Sqrt(pMatrix.Length); j++)
        {
            if (pMatrix[j, i] != 0) count++;
            if (count > level) break;
            if (j == Convert.ToInt32(Math.Sqrt(pMatrix.Length)) - 1 &&
count == level)
                zeros.Add(i);
        }
    }

    return zeros;
}
```

## Приложение В

### Листинг В.1 – Реализация ввода критериев и альтернатив

```
static void GetDataFromFile(string path, out int CriteriasNumber, out string[]
CriteriasNames, out int AlternativesNumber,
                                out string[] AlternativesNames, out
double[,] AimTable, out List<double[,]> CriteriasTables)
{
    CriteriasNumber = -1;
    CriteriasNames = null;
    AlternativesNumber = -1;
    AlternativesNames = null;
    AimTable = null;
    CriteriasTables = null;

    try
    {
        StreamReader reader = new StreamReader(@path);

        CriteriasNumber = Convert.ToInt32(reader.ReadLine().Split('
')[2]); // Количество критериев:

        reader.ReadLine(); // Названия критериев:
        CriteriasNames = new string[CriteriasNumber];
        for (int i = 0; i < CriteriasNumber; i++)
        {
            CriteriasNames[i] = reader.ReadLine();
        }

        AlternativesNumber = Convert.ToInt32(reader.ReadLine().Split('
')[2]); // Количество альтернатив:

        reader.ReadLine(); // Названия альтернатив:
        AlternativesNames = new string[AlternativesNumber];
        for (int i = 0; i < AlternativesNumber; i++)
        {
            AlternativesNames[i] = reader.ReadLine();
        }

        reader.ReadLine(); // Таблица "цели":
        AimTable = new double[CriteriasNumber, CriteriasNumber];
        for (int i = 0; i < CriteriasNumber; i++)
        {
            string[] line = reader.ReadLine().Replace("\t\t", "
").Split(' ');

            for (int j = 0; j < CriteriasNumber; j++)
            {
                if (line[j].Contains("/"))
                {
                    AimTable[i, j] =
Math.Round(Convert.ToDouble(line[j].Split('/')[0]) /
Convert.ToDouble(line[j].Split('/')[1]), 2);
                }
                else
                {
                    AimTable[i, j] = Convert.ToDouble(line[j]);
                }
            }
        }

        CriteriasTables = new List<double[,]>();
    }
}
```



### Продолжение Листинга В.1

```
        for (int i = 0; i < CriteriasNumber; i++)
        {
            reader.ReadLine(); // Таблица i критерия:
            CriteriasTables.Add(new double[AlternativesNumber,
AlternativesNumber]);
            for (int j = 0; j < AlternativesNumber; j++)
            {
                string[] line = reader.ReadLine().Replace("\t\t", "
").Split(' ');

                for (int k = 0; k < AlternativesNumber; k++)
                {
                    if (line[k].Contains("/"))
                    {
                        CriteriasTables[i][j, k] =
Math.Round(Convert.ToDouble(line[k].Split('/')[0]) /
Convert.ToDouble(line[k].Split('/')[1]), 2);
                    }
                    else
                    {
                        CriteriasTables[i][j, k] =
Convert.ToDouble(line[k]);
                    }
                }
            }
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Ошибка чтения входных данных: \n" +
ex.Message);
        CriteriasNumber = -1;
        Console.ReadLine();
    }
}
```

### Листинг В.2 – Реализация вывода вектора приоритетов цели

```
static double[] GetAimPriorityVector(double[,] AimTable, int
CriteriasNumber)
{
    double[] PriorityVector = new double[CriteriasNumber];
    double geometryAvg = 0.0;
    double[] V = new double[CriteriasNumber];

    for (int i = 0; i < CriteriasNumber; i++)
    {
        double Vi = 1;

        for (int j = 0; j < CriteriasNumber; j++)
        {
            Vi *= AimTable[i, j];
        }

        V[i] = Math.Round(Math.Pow(Vi, (1.0 /
(double)CriteriasNumber)), 2);

        geometryAvg += V[i];
    }

    for (int i = 0; i < CriteriasNumber; i++)
    {

```

### *Продолжение Листинга В.2*

```
        PriorityVector[i] = Math.Round(V[i] / geometryAvg, 2);
    }

    return PriorityVector;
}
```

### *Листинг В.3 – Реализация вывода приоритетов для всех критериев*

```
static List<double[]> GetCriteriaPriorityVectors(List<double[,]>
CriteriaTables, int AlternativesNumber, int CriteriaNumber)
{
    List<double[]> CriteriaPriorityVectors = new List<double[]>();
    double geometryAvg = 0.0;
    double[] V = new double[AlternativesNumber];

    for (int i = 0; i < CriteriaNumber; i++)
    {
        geometryAvg = 0.0;

        for (int j = 0; j < AlternativesNumber; j++)
        {
            double Vi = 1;

            for (int k = 0; k < AlternativesNumber; k++)
            {
                Vi *= CriteriaTables[i][j, k];
            }

            V[j] = Math.Round(Math.Pow(Vi, (1.0 /
(double)AlternativesNumber)), 3);

            geometryAvg += V[j];
        }

        CriteriaPriorityVectors.Add(new double[AlternativesNumber]);

        for (int j = 0; j < AlternativesNumber; j++)
        {
            CriteriaPriorityVectors[i][j] = Math.Round(V[j] /
geometryAvg, 3);
        }
    }

    return CriteriaPriorityVectors;
}
```

### *Листинг В.4 – Реализация вывода ИС и ОС для цели*

```
static double GetAimConsistencyIndex(double[,] AimTable, int CriteriaNumber,
double[] AimPriorityVector)
{
    double AimConsistencyIndex = 0.0;
    double[] P = new double[CriteriaNumber];
    double PSum = 0.0;

    for (int i = 0; i < CriteriaNumber; i++)
    {
        double S = 0.0;

        for (int j = 0; j < CriteriaNumber; j++)
        {
            S += AimTable[j, i];
        }
    }
}
```

#### *Продолжение Листинга В.4*

```
        P[i] = Math.Round(S * AimPriorityVector[i], 2);
    }

    foreach (double p in P)
        PSum += p;

    AimConsistencyIndex = Math.Round((PSum - CriteriasNumber) /
(CriteriasNumber - 1), 4);

    return AimConsistencyIndex;
}
```

#### *Листинг В.5 - Реализация вывода ИС и ОС для всех критериев*

```
static List<double> GetCriteriasConsistencyIndexes(List<double[,]>
CriteriasTables, int CriteriasNumber, int AlternativesNumber,
                                                    List<double[]>
CriteriasPriorityVectors)
{
    List<double> CriteriasConsistencyIndexes = new List<double>();
    double[] P = new double[AlternativesNumber];
    double PSum = 0.0;

    for (int i = 0; i < CriteriasNumber; i++)
    {
        PSum = 0.0;
        for (int j = 0; j < AlternativesNumber; j++)
        {
            double S = 0.0;

            for (int k = 0; k < AlternativesNumber; k++)
            {
                S += CriteriasTables[i][k, j];
            }

            P[j] = Math.Round(S * CriteriasPriorityVectors[i][j], 2);
        }

        foreach (double p in P)
            PSum += p;

        CriteriasConsistencyIndexes.Add(Math.Round((PSum -
AlternativesNumber) / (AlternativesNumber - 1), 4));
    }

    return CriteriasConsistencyIndexes;
}
```

#### *Листинг В.6 – Реализация вывода всех альтернатив*

```
static double[] GetAlternativesPriorities(double[] AimPriorityVector,
List<double[]> CriteriasPriorityVectors,
int CriteriasNumber,
                                                    int AlternativesNumber)
{
    double[] AlternativesPriorities = new double[AlternativesNumber];

    for (int i = 0; i < CriteriasNumber; i++)
        AlternativesPriorities[i] = 0.0;

    for (int i = 0; i < AlternativesNumber; i++)
```

*Продолжение Листинга В.6*

```
        {
            for (int k = 0; k < CriteriasNumber; k++)
            {
                AlternativesPriorities[i] +=
Math.Round(AimPriorityVector[k] * CriteriasPriorityVectors[k][i], 3);
            }

            return AlternativesPriorities;
        }
    }
```

*Листинг В.7 – Реализация вывода лучшей альтернативы*

```
static int GetBestAlternativeIndex(double[] AlternativesPriorities)
{
    double max = AlternativesPriorities.Max();

    for (int i = 0; i < AlternativesPriorities.Length; i++)
        if (AlternativesPriorities[i] == max)
            return i;

    return 0;
}
```

## Приложение Г

*Листинг Г.1 – Отвечает за сборку таблицы и ввод данных*

```
namespace SimplexMethod
{
    public partial class MainScreen : System.Windows.Forms.Form
    {
        public MainScreen()
        {
            InitializeComponent();

            int constraintsCount = 0;
            int variablesCount = 0;

            private void okBtn_Click(object sender, EventArgs e)
            {
                try
                {
                    constraintsCount = Convert.ToInt32(nOfConstraintsTextBox.Text);
                    variablesCount = Convert.ToInt32(nOfVariablesTextBox.Text);
                    fillConstraintsGrid();
                    fillFunctionGrid();
                }
                catch (Exception err)
                {
                    MessageBox.Show(err.Message);
                }
            }

            void fillConstraintsGrid()
            {
                constraintsGridView.Rows.Clear();
                constraintsGridView.ColumnCount = variablesCount + 2;
                constraintsGridView.RowHeadersVisible = false;
                for (int i = 0; i < variablesCount + 2; i++)
                {
                    constraintsGridView.Columns[i].Width = 50;
                    constraintsGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
                    if (i < variablesCount)
                    {
                        constraintsGridView.Columns[i].Name = "x" + (i +
1).ToString();
                    } else if (i == variablesCount)
                    {
                        constraintsGridView.Columns[i].Name = "Sign";
                    }
                }

                for (int i = 0; i < constraintsCount; i++)
                {
                    string[] row = new string[variablesCount + 2];
                    constraintsGridView.Rows.Add(row);
                    constraintsGridView.Rows[i].Height = 20;
                }
            }

            void fillFunctionGrid()
            {

```

```

        functionGridView.Rows.Clear();
        functionGridView.ColumnCount = variablesCount + 1;
        functionGridView.RowHeadersVisible = false;
        for (int i = 0; i < variablesCount + 1; i++)
        {
            functionGridView.Columns[i].Width = 50;
            functionGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
            if (i < variablesCount)
            {
                functionGridView.Columns[i].Name = "x" + (i +
1).ToString();
            }
            else
            {
                functionGridView.Columns[i].Name = "C";
            }
        }
        string[] row = new string[variablesCount + 2];
        constraintsGridView.Rows.Add(row);
        constraintsGridView.Rows[0].Height = 20;
    }

    void Proceed()
    {
        Constraint[] constraints = new Constraint[constraintsCount];
        for (int i = 0; i < constraintsCount; i++)
        {
            double[] variables = new double[variablesCount];
            double b =
Convert.ToDouble(constraintsGridView.Rows[i].Cells[variablesCount + 1].Value);
            string sign =
Convert.ToString(constraintsGridView.Rows[i].Cells[variablesCount].Value);
            for (int j = 0; j < variablesCount; j++)
            {
                variables[j] =
Convert.ToDouble(constraintsGridView.Rows[i].Cells[j].Value);
            }
            constraints[i] = new Constraint(variables, b, sign);
        }
        double[] functionVariables = new double[variablesCount];
        for (int i = 0; i < variablesCount; i++)
        {
            functionVariables[i] =
Convert.ToDouble(functionGridView.Rows[0].Cells[i].Value);
        }
        double c =
Convert.ToDouble(functionGridView.Rows[0].Cells[variablesCount].Value);

        bool isExtrMax = extrComboBox.SelectedIndex == 0;

        Function function = new Function(functionVariables, c, isExtrMax);

        Simplex simplex = new Simplex(function, constraints);

        Tuple<List<SimplexSnap>, SimplexResult> result =
simplex.GetResult();
    }

```

```

        switch (result.Item2)
        {
            case SimplexResult.Found:
                string extrStr = isExtrMax ? "max" : "min";
                resultsLbl.Text = "The optimal solution is found: F" +
extrStr + $" = {result.Item1.Last().fValue}";
                break;
            case SimplexResult.Unbounded:
                resultsLbl.Text = "The domain of admissible solutions is
unbounded";
                break;
            case SimplexResult.NotYetFound:
                resultsLbl.Text = "Algorithm has made 100 cycles and
hasn't found any optimal solution.";
                break;
        }

        ShowResultsGrid(result.Item1);
    }

    void ShowResultsGrid(List<SimplexSnap> snaps)
    {
        resultsGridView.Rows.Clear();
        resultsGridView.ColumnCount = snaps.First().matrix.Length + 4;
        resultsGridView.RowHeadersVisible = false;
        resultsGridView.ColumnHeaderVisible = false;

        for (int i = 0; i < snaps.First().matrix.Length + 4; i++)
        {
            resultsGridView.Columns[i].Width = 50;
            resultsGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        }

        foreach (SimplexSnap snap in snaps)
        {
            string[] firstRow = new string[snaps.First().matrix.Length +
4];

            firstRow[0] = "i";
            firstRow[1] = "Basis";
            firstRow[2] = "C";
            firstRow[3] = "B";
            for (int i = 4; i < snaps.First().matrix.Length + 4; i++)
            {
                firstRow[i] = $"A{i - 3}";
            }

            resultsGridView.Rows.Add(firstRow);

            for (int i = 0; i < snaps.First().C.Length; i++)
            {
                string[] row = new string[snaps.First().matrix.Length +
4];

                for (int j = 0; j < snaps.First().matrix.Length + 4; j++)
                {
                    if (j == 0)
                    {
                        row[j] = (i + 1).ToString();
                    }
                }
            }
        }
    }

```

```

        else if (j == 1)
        {
            row[j] = $"A{snap.C[i] + 1}";
        }
        else if (j == 2)
        {
            row[j] = snap.m[snap.C[i]] ? "-M" :
4];
            $" {snap.fVars[snap.C[i]]}";
        }
        else if (j == 3)
        {
            row[j] = Round(snap.b[i]).ToString();
        }
        else
        {
            row[j] = Round(snap.matrix[j - 4][i]).ToString();
        }
    }
    resultsGridView.Rows.Add(row);
}
string[] fRow = new string[snap.First().matrix.Length + 4];
fRow[0] = "m+1";
fRow[1] = "F";
fRow[2] = "Δj";
fRow[3] = Round(snap.fValue).ToString();
for (int i = 4; i < snap.First().matrix.Length + 4; i++)
{
    fRow[i] = Round(snap.F[i - 4]).ToString();
}
resultsGridView.Rows.Add(fRow);

if (!snap.isMDone)
{
    string[] mRow = new string[snap.First().matrix.Length +
4];

    mRow[0] = "m+2";
    mRow[1] = "M";
    mRow[2] = "Δj";
    mRow[3] = "";
    for (int i = 4; i < snap.First().matrix.Length + 4; i++)
    {
        mRow[i] = Round(snap.M[i - 4]).ToString();
    }
    resultsGridView.Rows.Add(mRow);
}
string[] emptyRow = new string[snap.First().matrix.Length +
4];
resultsGridView.Rows.Add(emptyRow);
}

}

double Round(double a)
{
    return Math.Round(a, 2);
}

void fillDefaultsConstraints(double[][] consMatrx, string[] signs,
double[] freeVars)
{
    constraintsCount = signs.Length;

```



```

        nofConstraintsTextBox.Text = constraintsCount.ToString();
        variablesCount = consMatrx.First().Length;
        nofVariablesTextBox.Text = variablesCount.ToString();
        fillConstraintsGrid();
        for (int i = 0; i < constraintsCount; i++)
        {
            for (int j = 0; j < variablesCount + 2; j++)
            {
                if (j < variablesCount)
                {
                    constraintsGridView.Rows[i].Cells[j].Value =
consMatrx[i][j];
                }
                else if (j < variablesCount + 1)
                {
                    constraintsGridView.Rows[i].Cells[j].Value = signs[i];
                }
                else if (j < variablesCount + 2)
                {
                    constraintsGridView.Rows[i].Cells[j].Value =
freeVars[i];
                }
            }
        }
    }
    void fillDefaultsFunction(double[] funcVars, double c, bool isExtrMax)
    {
        fillFunctionGrid();
        for (int i = 0; i < variablesCount + 1; i++)
        {
            if (i < variablesCount)
            {
                functionGridView.Rows[0].Cells[i].Value = funcVars[i];
            }
            else
            {
                functionGridView.Rows[0].Cells[i].Value = c;
            }
        }

        extrComboBox.SelectedIndex = isExtrMax ? 0 : 1;
    }
    private void goBtn_Click(object sender, EventArgs e)
    {
        Proceed();
    }

    private void defaultBtn_Click(object sender, EventArgs e)
    {
        Problem p = ProblemsService.shared.GetNext();
        fillDefaultsConstraints(p.consMatrx, p.signs, p.freeVars);
        fillDefaultsFunction(p.funcVars, p.c, p.isExtrMax);
        Proceed();
    }

    private void clearBtn_Click(object sender, EventArgs e)
    {

```

### *Продолжение Листинга Г.1*

```
        nOfConstraintsTextBox.Clear();
        nOfVariablesTextBox.Clear();
        resultsGridView.Columns.Clear();
        functionGridView.Columns.Clear();
        constraintsGridView.Columns.Clear();
        extrComboBox.SelectedIndex = -1;
        resultsLbl.ResetText();
    }
}
```

### *Листинг Г.2 – Отвечает за окно интерфейса*

```
namespace SimplexMethod
{
    partial class MainScreen
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true
        if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }
        #region Windows Form Designer generated code
        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new
            System.ComponentModel.ComponentResourceManager(typeof(MainScreen));
            this.nOfConstraintsTextBox = new System.Windows.Forms.TextBox();
            this.nOfVariablesTextBox = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.okBtn = new System.Windows.Forms.Button();
            this.constraintsGridView = new System.Windows.Forms.DataGridView();
            this.functionGridView = new System.Windows.Forms.DataGridView();
            this.extrComboBox = new System.Windows.Forms.ComboBox();
            this.label2 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.resultsGridView = new System.Windows.Forms.DataGridView();
            this.goBtn = new System.Windows.Forms.Button();
            this.clearBtn = new System.Windows.Forms.Button();
            this.defaultBtn = new System.Windows.Forms.Button();
            this.resultsLbl = new System.Windows.Forms.Label();
        }
    }
}
```

## Продолжение Листинга Г.2

```
((System.ComponentModel.ISupportInitialize)(this.constraintsGridView)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.functionGridView)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.resultsGridView)).BeginInit();

        this.SuspendLayout();
        //
        // nOfConstraintsTextBox
        //
        this.nOfConstraintsTextBox.Location = new System.Drawing.Point(149,
23);

        this.nOfConstraintsTextBox.Name = "nOfConstraintsTextBox";
        this.nOfConstraintsTextBox.Size = new System.Drawing.Size(34, 29);
        this.nOfConstraintsTextBox.TabIndex = 0;
        //
        // nOfVariablesTextBox
        //
        this.nOfVariablesTextBox.Location = new System.Drawing.Point(309,
23);

        this.nOfVariablesTextBox.Name = "nOfVariablesTextBox";
        this.nOfVariablesTextBox.Size = new System.Drawing.Size(34, 29);
        this.nOfVariablesTextBox.TabIndex = 1;
        //
        // label1
        //
        this.label1.AutoSize = true;
        this.label1.Location = new System.Drawing.Point(20, 26);
        this.label1.Name = "label1";
        this.label1.Size = new System.Drawing.Size(123, 21);
        this.label1.TabIndex = 2;
        this.label1.Text = "N. of constraints";
        //
        // okBtn
        //
        this.okBtn.Location = new System.Drawing.Point(732, 23);
        this.okBtn.Name = "okBtn";
        this.okBtn.Size = new System.Drawing.Size(69, 36);
        this.okBtn.TabIndex = 4;
        this.okBtn.Text = "OK";
        this.okBtn.UseVisualStyleBackColor = true;
        this.okBtn.Click += new System.EventHandler(this.okBtn_Click);
        //
        // constraintsGridView
        //
        this.constraintsGridView.BackgroundColor =
System.Drawing.Color.White;
        this.constraintsGridView.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
        this.constraintsGridView.Location = new System.Drawing.Point(24,
67);

        this.constraintsGridView.Name = "constraintsGridView";
        this.constraintsGridView.Size = new System.Drawing.Size(777, 123);
        this.constraintsGridView.TabIndex = 5;
        //
        // functionGridView
        //
```

```

        this.functionGridView.BackgroundColor =
System.Drawing.Color.White;
        this.functionGridView.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
        this.functionGridView.Location = new System.Drawing.Point(24, 213);
        this.functionGridView.Name = "functionGridView";
        this.functionGridView.Size = new System.Drawing.Size(686, 62);
        this.functionGridView.TabIndex = 6;
        //
        // extrComboBox
        //
        this.extrComboBox.AllowDrop = true;
        this.extrComboBox.FormattingEnabled = true;
        this.extrComboBox.Items.AddRange(new object[] {
"Max",
"Min"});
        this.extrComboBox.Location = new System.Drawing.Point(732, 246);
        this.extrComboBox.Name = "extrComboBox";
        this.extrComboBox.Size = new System.Drawing.Size(69, 29);
        this.extrComboBox.TabIndex = 7;
        //
        // label2
        //
        this.label2.AutoSize = true;
        this.label2.Location = new System.Drawing.Point(194, 26);
        this.label2.Name = "label2";
        this.label2.Size = new System.Drawing.Size(109, 21);
        this.label2.TabIndex = 3;
        this.label2.Text = "N. of variables";
        //
        // label3
        //
        this.label3.AutoSize = true;
        this.label3.Location = new System.Drawing.Point(728, 213);
        this.label3.Name = "label3";
        this.label3.Size = new System.Drawing.Size(36, 21);
        this.label3.TabIndex = 8;
        this.label3.Text = "Extr";
        //
        // resultsGridView
        //
        this.resultsGridView.Anchor =
((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top
| System.Windows.Forms.AnchorStyles.Bottom)
| System.Windows.Forms.AnchorStyles.Left)
| System.Windows.Forms.AnchorStyles.Right)));
        this.resultsGridView.BackgroundColor = System.Drawing.Color.White;
        this.resultsGridView.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
        this.resultsGridView.Location = new System.Drawing.Point(24, 354);
        this.resultsGridView.Name = "resultsGridView";
        this.resultsGridView.Size = new System.Drawing.Size(777, 287);
        this.resultsGridView.TabIndex = 9;
        //
        // goBtn
        //
        this.goBtn.Location = new System.Drawing.Point(24, 291);
        this.goBtn.Name = "goBtn";
        this.goBtn.Size = new System.Drawing.Size(686, 34);
        this.goBtn.TabIndex = 10;
        this.goBtn.Text = "GO";
        this.goBtn.UseVisualStyleBackColor = true;

```

```

        this.goBtn.Click += new System.EventHandler(this.goBtn_Click);
        //
        // clearBtn
        //
        this.clearBtn.Location = new System.Drawing.Point(732, 291);
        this.clearBtn.Name = "clearBtn";
        this.clearBtn.Size = new System.Drawing.Size(69, 34);
        this.clearBtn.TabIndex = 11;
        this.clearBtn.Text = "CLEAR";
        this.clearBtn.UseVisualStyleBackColor = true;
        this.clearBtn.Click += new
System.EventHandler(this.clearBtn_Click);
        //
        // defaultBtn
        //
        this.defaultBtn.Location = new System.Drawing.Point(592, 23);
        this.defaultBtn.Name = "defaultBtn";
        this.defaultBtn.Size = new System.Drawing.Size(118, 36);
        this.defaultBtn.TabIndex = 12;
        this.defaultBtn.Text = "Default";
        this.defaultBtn.UseVisualStyleBackColor = true;
        this.defaultBtn.Click += new
System.EventHandler(this.defaultBtn_Click);
        //
        // resultsLbl
        //
        this.resultsLbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom
| System.Windows.Forms.AnchorStyles.Left)));
        this.resultsLbl.AutoSize = true;
        this.resultsLbl.Location = new System.Drawing.Point(20, 657);
        this.resultsLbl.Name = "resultsLbl";
        this.resultsLbl.Size = new System.Drawing.Size(0, 21);
        this.resultsLbl.TabIndex = 13;
        //
        // mainScreen
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(9F, 21F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(824, 697);
        this.Controls.Add(this.resultsLbl);
        this.Controls.Add(this.defaultBtn);
        this.Controls.Add(this.clearBtn);
        this.Controls.Add(this.goBtn);
        this.Controls.Add(this.resultsGridView);
        this.Controls.Add(this.label3);
        this.Controls.Add(this.extrComboBox);
        this.Controls.Add(this.functionGridView);
        this.Controls.Add(this.constraintsGridView);
        this.Controls.Add(this.okBtn);
        this.Controls.Add(this.label2);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.nOfVariablesTextBox);
        this.Controls.Add(this.nOfConstraintsTextBox);
        this.Font = new System.Drawing.Font("Segoe UI", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte) 0));
        this.Icon =
((System.Drawing.Icon)(resources.GetObject("$this.Icon")));

```

### Продолжение Листинга Г.2

```
        this.Margin = new System.Windows.Forms.Padding(5);
        this.Name = "MainScreen";
        this.Text = "Simplex Method";

        ((System.ComponentModel.ISupportInitialize)(this.constraintsGridView)).EndInit();
    };

    ((System.ComponentModel.ISupportInitialize)(this.functionGridView)).EndInit();

    ((System.ComponentModel.ISupportInitialize)(this.resultsGridView)).EndInit();
    this.ResumeLayout(false);
    this.PerformLayout();

    }

    #endregion

    private System.Windows.Forms.TextBox nOfConstraintsTextBox;
    private System.Windows.Forms.TextBox nOfVariablesTextBox;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.Button okBtn;
    private System.Windows.Forms.DataGridView constraintsGridView;
    private System.Windows.Forms.DataGridView functionGridView;
    private System.Windows.Forms.ComboBox extrComboBox;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.DataGridView resultsGridView;
    private System.Windows.Forms.Button goBtn;
    private System.Windows.Forms.Button clearBtn;
    private System.Windows.Forms.Button defaultBtn;
    private System.Windows.Forms.Label resultsLbl;
}
}
```

### Листинг Г.3 – Расчетная часть

```
namespace SimplexMethod
{
    public partial class MainScreen : System.Windows.Forms.Form
    {
        public MainScreen()
        {
            InitializeComponent();

            int constraintsCount = 0;
            int variablesCount = 0;

            private void okBtn_Click(object sender, EventArgs e)
            {
                try
                {
                    constraintsCount = Convert.ToInt32(nOfConstraintsTextBox.Text);
                    variablesCount = Convert.ToInt32(nOfVariablesTextBox.Text);
                    fillConstraintsGrid();
                    fillFunctionGrid();
                }
                catch (Exception err)
                {
                    MessageBox.Show(err.Message);
                }
            }
        }
    }
}
```

```

    }
}

void fillConstraintsGrid()
{
    constraintsGridView.Rows.Clear();
    constraintsGridView.ColumnCount = variablesCount + 2;
    constraintsGridView.RowHeadersVisible = false;
    for (int i = 0; i < variablesCount + 2; i++)
    {
        constraintsGridView.Columns[i].Width = 50;

        constraintsGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        if (i < variablesCount)
        {
            constraintsGridView.Columns[i].Name = "x" + (i +
1).ToString();
        } else if (i == variablesCount)
        {
            constraintsGridView.Columns[i].Name = "Sign";
        }
    }

    for (int i = 0; i < constraintsCount; i++)
    {
        string[] row = new string[variablesCount + 2];
        constraintsGridView.Rows.Add(row);
        constraintsGridView.Rows[i].Height = 20;
    }
}

void fillFunctionGrid()
{
    functionGridView.Rows.Clear();
    functionGridView.ColumnCount = variablesCount + 1;
    functionGridView.RowHeadersVisible = false;
    for (int i = 0; i < variablesCount + 1; i++)
    {
        functionGridView.Columns[i].Width = 50;
        functionGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        if (i < variablesCount)
        {
            functionGridView.Columns[i].Name = "x" + (i +
1).ToString();
        }
        else
        {
            functionGridView.Columns[i].Name = "C";
        }
    }
    string[] row = new string[variablesCount + 2];
    constraintsGridView.Rows.Add(row);
    constraintsGridView.Rows[0].Height = 20;
}

void Proceed()
{

```

```

        Constraint[] constraints = new Constraint[constraintsCount];
        for (int i = 0; i < constraintsCount; i++)
        {
            double[] variables = new double[variablesCount];
            double b =
            Convert.ToDouble(constraintsGridView.Rows[i].Cells[variablesCount + 1].Value);
            string sign =
            Convert.ToString(constraintsGridView.Rows[i].Cells[variablesCount].Value);
            for (int j = 0; j < variablesCount; j++)
            {
                variables[j] =
                Convert.ToDouble(constraintsGridView.Rows[i].Cells[j].Value);
            }
            constraints[i] = new Constraint(variables, b, sign);
        }
        double[] functionVariables = new double[variablesCount];
        for (int i = 0; i < variablesCount; i++)
        {
            functionVariables[i] =
            Convert.ToDouble(functionGridView.Rows[0].Cells[i].Value);
        }
        double c =
        Convert.ToDouble(functionGridView.Rows[0].Cells[variablesCount].Value);

        bool isExtrMax = extrComboBox.SelectedIndex == 0;

        Function function = new Function(functionVariables, c, isExtrMax);

        Simplex simplex = new Simplex(function, constraints);

        Tuple<List<SimplexSnap>, SimplexResult> result =
        simplex.GetResult();
        switch (result.Item2)
        {
            case SimplexResult.Found:
                string extrStr = isExtrMax ? "max" : "min";
                resultsLbl.Text = "The optimal solution is found: F" +
                extrStr + $" = {result.Item1.Last().fValue}";
                break;
            case SimplexResult.Unbounded:
                resultsLbl.Text = "The domain of admissible solutions is
                unbounded";
                break;
            case SimplexResult.NotYetFound:
                resultsLbl.Text = "Algorithm has made 100 cycles and hasn't
                found any optimal solution.";
                break;
        }

        ShowResultsGrid(result.Item1);
    }

    void ShowResultsGrid(List<SimplexSnap> snaps)
    {
        resultsGridView.Rows.Clear();
        resultsGridView.ColumnCount = snaps.First().matrix.Length + 4;
        resultsGridView.RowHeadersVisible = false;
        resultsGridView.ColumnHeadersVisible = false;
    }

```



```

        for (int i = 0; i < snaps.First().matrix.Length + 4; i++)
        {
            resultsGridView.Columns[i].Width = 50;
            resultsGridView.Columns[i].DefaultCellStyle.Alignment =
DataGridViewContentAlignment.MiddleCenter;
        }

        foreach (SimplexSnap snap in snaps)
        {
            string[] firstRow = new string[snaps.First().matrix.Length +
4];

            firstRow[0] = "i";
            firstRow[1] = "Basis";
            firstRow[2] = "C";
            firstRow[3] = "B";

            for (int i = 4; i < snaps.First().matrix.Length + 4; i++)
            {
                firstRow[i] = $"A{i - 3}";
            }

            resultsGridView.Rows.Add(firstRow);

            for (int i = 0; i < snaps.First().C.Length; i++)
            {
                string[] row = new string[snaps.First().matrix.Length + 4];
                for (int j = 0; j < snaps.First().matrix.Length + 4; j++)
                {
                    if (j == 0)
                    {
                        row[j] = (i + 1).ToString();
                    }
                    else if (j == 1)
                    {
                        row[j] = $"A{snap.C[i] + 1}";
                    }
                    else if (j == 2)
                    {
                        row[j] = snap.m[snap.C[i]] ? "-M" :
"${snap.fVars[snap.C[i]]}";
                    }
                    else if (j == 3)
                    {
                        row[j] = Round(snap.b[i]).ToString();
                    }
                    else
                    {
                        row[j] = Round(snap.matrix[j - 4][i]).ToString();
                    }
                }
                resultsGridView.Rows.Add(row);
            }
            string[] fRow = new string[snaps.First().matrix.Length + 4];
fRow[0] = "m+1";
            fRow[1] = "F";
            fRow[2] = "Δj";
            fRow[3] = Round(snap.fValue).ToString();

            for (int i = 4; i < snaps.First().matrix.Length + 4; i++)
            {

```

```

        fRow[i] = Round(snap.F[i - 4]).ToString();
    }
    resultsGridView.Rows.Add(fRow);

    if (!snap.isMDone)
    {
        string[] mRow = new string[snap.First().matrix.Length +
4];

        mRow[0] = "m+2";
        mRow[1] = "M";
        mRow[2] = "Δj";
        mRow[3] = "";
        for (int i = 4; i < snap.First().matrix.Length + 4; i++)
        {
            mRow[i] = Round(snap.M[i - 4]).ToString();
        }
        resultsGridView.Rows.Add(mRow);
    }
    string[] emptyRow = new string[snap.First().matrix.Length +
4];
    resultsGridView.Rows.Add(emptyRow);
}

double Round(double a)
{
    return Math.Round(a, 2);
}

void fillDefaultsConstraints(double[][] consMatrx, string[] signs,
double[] freeVars)
{
    constraintsCount = signs.Length;
    nofConstraintsTextBox.Text = constraintsCount.ToString();
    variablesCount = consMatrx.First().Length;
    nofVariablesTextBox.Text = variablesCount.ToString();
    fillConstraintsGrid();

    for (int i = 0; i < constraintsCount; i++)
    {
        for (int j = 0; j < variablesCount + 2; j++)
        {
            if (j < variablesCount)
            {
                constraintsGridView.Rows[i].Cells[j].Value =
consMatrx[i][j];
            }
            else if (j < variablesCount + 1)
            {
                constraintsGridView.Rows[i].Cells[j].Value = signs[i];
            }
            else if (j < variablesCount + 2)
            {
                constraintsGridView.Rows[i].Cells[j].Value =
freeVars[i];
            }
        }
    }
}

```

*Продолжение Листинга Г.3*

```
void fillDefaultsFunction(double[] funcVars, double c, bool isExtrMax)
{
    fillFunctionGrid();
    for (int i = 0; i < variablesCount + 1; i++)
    {
        if (i < variablesCount)
        {
            functionGridView.Rows[0].Cells[i].Value = funcVars[i];
        }
        else
        {
            functionGridView.Rows[0].Cells[i].Value = c;
        }
    }

    extrComboBox.SelectedIndex = isExtrMax ? 0 : 1;
}

private void goBtn_Click(object sender, EventArgs e)
{
    Proceed();
}

private void defaultBtn_Click(object sender, EventArgs e)
{
    Problem p = ProblemsService.shared.GetNext();
    fillDefaultsConstraints(p.consMatrx, p.signs, p.freeVars);
    fillDefaultsFunction(p.funcVars, p.c, p.isExtrMax);
    Proceed();
}

private void clearBtn_Click(object sender, EventArgs e)
{
    nOfConstraintsTextBox.Clear();
    nOfVariablesTextBox.Clear();
    resultsGridView.Columns.Clear();
    functionGridView.Columns.Clear();
    constraintsGridView.Columns.Clear();
    extrComboBox.SelectedIndex = -1;
    resultsLbl.ResetText();
}
}
```

## Приложение Д

*Листинг Д.1 – Реализация метода минимальной стоимости*

```
using System;
using transport_problem.Table;

namespace transport_problem.SolutionMethods
{
    public class FirstlySolutionMethod
    {
        protected Table.Table _table;

        protected FirstlySolutionMethod(Supplier[] suppliers,
Consumer[] consumers)
        {
            _table = new Table.Table(suppliers, consumers);
        }

        protected void AddTransportation(Cell cell)
        {
            int rate = cell.GetRate();
            int supplierStock = cell.GetRow().GetStock();
            int consumerNeeds = cell.GetColumn().GetRequirement();

            Transportation transportation;

            if (consumerNeeds > supplierStock)
            {
                transportation = new Transportation(supplierStock,
rate);
                cell.AddTransportation(transportation);

                _table.RemoveRow(cell.GetRow());
                cell.GetColumn().SetRequirement(consumerNeeds -
supplierStock);
                cell.GetRow().SetStock(0);
            }
            Else
            {
                transportation = new Transportation(consumerNeeds,
rate);
                cell.AddTransportation(transportation);

                _table.RemoveColumn(cell.GetColumn());
                cell.GetColumn().SetRequirement(0);
                cell.GetRow().SetStock(supplierStock - consumerNeeds);
            }
        }
    }
}
```

*Листинг Д.2 – Реализация метода северо-западного угла*

```
using transport_problem.Table;

namespace transport_problem.SolutionMethods
```

*Продолжение Листинга Д.2*

```
        }
    }
    return _table;
}
}
{
    public class NorthwestCornerMethod : FirstlySolutionMethod
    {
        public NorthwestCornerMethod(Supplier[] suppliers, Consumer[]
consumers) : base(suppliers, consumers)
        {

        }

        public Table.Table GetSolution()
        {

            foreach (TableRow row in _table.GetRows())
            {
                foreach (Cell cell in row.GetCells())
                {
                    if (cell.IsActive())
                    {
                        AddTransportation(cell);
                    }
                }
            }
        }
    }
}
```

*Листинг Д.3 – Реализация метода потенциалов*

```
using System;
using System.Collections;
using System.Linq;
using System.Windows.Forms;
using transport_problem.Table;

namespace transport_problem.SolutionMethods
{
    public class PotentialsMethod
    {
        private Table.Table _table;

        public PotentialsMethod(Table.Table table)
        {
            _table = table;
        }

        public bool IsOptimal()
        {
            if (CheckDegeneracy())
            {
                AddRandomTransportation();
            }

            ResetPotentials();
            CalculatePotentials();
            CalculateDistrubution();
        }
    }
}
```

```

        return _table.GetRows().SelectMany(row => row.GetCells()).All(cell
=> cell.GetDistributionIndex() >= 0);
    }
    public void CalculatePotentials()
    {
        _table.GetRow(0).SetPotential(0);

        while (!AllPotentialsCalculated())
        {
            foreach (Cell cell in _table.GetRows().SelectMany(row =>
row.GetCells().Where(cell => cell.haveTransportation()))
            {

                TableRow row = cell.GetRow();
                TableColumn column = cell.GetColumn();

                if (!row.HavePotential() && !column.HavePotential())
                {
                    continue;
                }

                if (!row.HavePotential())
                {
                    row.SetPotential(cell.GetRate() -
Convert.ToInt32(column.GetPotential()));
                }

                else if (!column.HavePotential())
                {
                    column.SetPotential(cell.GetRate() -
Convert.ToInt32(row.GetPotential()));
                }
            }
        }

        private void CalculateDistrubution()
        {
            foreach (TableRow row in _table.GetRows())
            {
                foreach (Cell cell in row.GetCells())
                {
                    cell.SetDistributionIndex(cell.GetRate() -
Convert.ToInt32(row.GetPotential()) -
Convert.ToInt32(cell.GetColumn().GetPotential()));
                }
            }

            private void ResetPotentials()
            {
                foreach (TableRow row in _table.GetRows())
                {
                    row.RemovePotential();
                }

                foreach (TableColumn column in _table.GetColumns())
                {
                    column.RemovePotential();
                }
            }
        }
    }

```

```

    }

    private bool AllPotentialsCalculated()
    {
        return _table.GetRows().All(row => row.HavePotential()) &&
        _table.GetColumns().All(column => column.HavePotential());
    }

    private bool CheckDegeneracy()
    {
        return _table.GetTransportationsCnt() < (_table.GetColumnsCnt() +
        _table.GetRowsCnt() - 1);
    }

    private void AddRandomTransportation()
    {
        Cell cell = GetRandomFreeCell();

        cell.AddTransportation(new Transportation(0, cell.GetRate()));
    }

    public void Otimize()
    {
        Cell top = GetMinDistributionIndexCell();
        DoRedistribution(BuildRedistributionCycle(top));
    }

    private void DoRedistribution(Cell[] cycle)
    {
        ArrayList calculated = new ArrayList();
        int min = GetMinTransportationCargo(cycle);

        var arr = cycle.ToArray();

        for (int i = 0; i < cycle.Length; i++)
        {
            Cell cell = (Cell) arr[i];

            if (calculated.Contains(cell))
                continue;

            Transportation transportation = cell.GetTransportation();

            if (i == 0)
            {
                cell.AddTransportation(new Transportation(min,
cell.GetRate()));
                calculated.Add(cell);
                continue;
            }

            cell.AddTransportation(i%2 == 0
                ? new Transportation(transportation.GetCargo() + min,
                    cell.GetRate())
                : new Transportation(transportation.GetCargo() - min,
                    cell.GetRate()));

            calculated.Add(cell);
        }
    }

```

```

        if (cell.GetTransportation().GetCargo() == 0)
            cell.RemoveTransportation();
    }

    private int GetMinTransportationCargo(Cell[] cycle)
    {
        int min = cycle[1].GetTransportation().GetCargo();

        for (int i = 1; i < cycle.Length; i++)
        {
            Cell cell = cycle[i];

            if (i%2 != 0 && cell.GetTransportation().GetCargo() < min)
                min = cell.GetTransportation().GetCargo();

        }

        return min;
    }

    private Cell[] BuildRedistributionCycle(Cell top)
    {
        ArrayList cycles = new ArrayList();
        ArrayList firstCycle = new ArrayList {top};

        cycles.Add(firstCycle);

        do
        {
            foreach (ArrayList cycle in cycles.ToArray())
            {
                if (CheckCycleFinal(cycle))
                {
                    return
cycle.Cast<Cell>().Reverse().Skip(1).Reverse().ToArray();
                }

                MakeCycleBranches(cycles, cycle);
            }

        } while (true);
    }

    private void MakeCycleBranches(ArrayList cycles, ArrayList cycle)
    {
        Cell first = cycle.Cast<Cell>().First();

        Cell last = cycle.Cast<Cell>().Last();
        Cell secondLast =
cycle.Cast<Cell>().Reverse().Skip(1).FirstOrDefault();

        ArrayList originalCycle = (ArrayList) cycle.Clone();

        TableRow row = last.GetRow();
        TableColumn column = last.GetColumn();

        bool sameCycle = true;
    }

```



```

        foreach (Cell cell in row.GetCells())
        {
            if(cell == last)
                continue;

            if (secondLast != null && secondLast.GetRow() == row &&
cell.GetColumnIndex() > last.GetColumnIndex() - secondLast.GetColumnIndex())
                continue;

            if(!cell.haveTransportation() && cell != first)
                continue;

            if (sameCycle)
            {
                cycle.Add(cell);
                sameCycle = false;
            }
            else
            {
                ArrayList newBranch = (ArrayList)originalCycle.Clone();
                newBranch.Add(cell);

                cycles.Add(newBranch);
            }
        }

        foreach (Cell cell in column.GetCells())
        {
            if (cell == last)
                continue;

            if (!cell.haveTransportation())
                continue;

            if (secondLast != null && secondLast.GetColumn() == column &&
cell.GetRowIndex() > last.GetColumnIndex() - secondLast.GetRowIndex())
                continue;

            ArrayList newBranch = (ArrayList)originalCycle.Clone();
            newBranch.Add(cell);

            cycles.Add(newBranch);
        }
    }

    private bool CheckCycleFinal(ArrayList cycle)
    {
        if (cycle.Count < 4)
            return false;

        Cell first = cycle.Cast<Cell>().First();
        Cell last = cycle.Cast<Cell>().Last();

        return first == last;
    }

    private Cell GetRandomFreeCell()
    {
        Random rand = new Random();
        Cell cell;
        do

```

### *Продолжение Листинга Д.3*

```
        {
            int row = rand.Next(_table.GetRowsCnt());
            int col = rand.Next(_table.GetColumnsCnt());
            cell = _table.GetRow(row).GetCell(col);
        }

        while (cell.haveTransportation()) ;

        return cell;
    }

    public Cell GetMinDistributionIndexCell()
    {
        Cell min = GetRandomFreeCell();

        foreach (TableRow row in _table.GetRows())
        {
            foreach (Cell cell in row.GetCells())
            {
                if (cell.GetDistributionIndex() <
min.GetDistributionIndex() && !cell.haveTransportation())
                {
                    min = cell;
                }
            }
        }

        return min;
    }
}
```

### *Листинг Д.4 – Реализация интерфейса*

```
using System;
using System.Collections;
using System.Linq;
using System.Windows.Forms;

using transport_problem.Table;
using transport_problem.SolutionMethods;

namespace transport_problem
{
    public partial class App : Form
    {
        private ArrayList _suppliers;
        private ArrayList _consumers;

        public App()
        {
            InitializeComponent();

            ConfigureGui();
        }

        private void CalculateButton_Click(object sender, EventArgs e)
        {
            InitData();
        }
    }
}
```

*Продолжение Листинга Д.4*

```
        Balancer balancer = new Balancer(_suppliers, _consumers);

        if(!balancer.CheckBalance())
            balancer.Balance();

        //var firstlyMethod = new
NorthwestCornerMethod(_suppliers.Cast<Supplier>().ToArray(),
_consumers.Cast<Consumer>().ToArray());
        var firstlyMethod = new
VogelsMethod(_suppliers.Cast<Supplier>().ToArray(),
_consumers.Cast<Consumer>().ToArray());

        Table.Table solution = firstlyMethod.GetSolution();

        var optimizeMethod = new PotentialsMethod(solution);

        Logger logger = new Logger(solution);

        while (!optimizeMethod.IsOptimal())
        {
            logger.Log();
            optimizeMethod.Otimize();
        }

        logger.Log();
        logger.ShowLog();

        MessageBox.Show("Total: " +
solution.GetTotalTransportationsPrice());
    }

    private void InitData()
    {
        _suppliers = new ArrayList();
        _consumers = new ArrayList();

        int suppliersCount = this.dataGridView1.Columns.Count;
        int consumersCount = this.dataGridView2.Columns.Count;

        for (int i = 0; i < suppliersCount; i++)
        {
            int[] rates = new int[consumersCount];

            int stock =
Convert.ToInt32(this.dataGridView1.Rows[0].Cells[i].Value);

            for (int j = 0; j < consumersCount; j++)
            {
                rates[j] = Convert.ToInt32(this.dataGridView1.Rows[j +
1].Cells[i].Value);
            }

            _suppliers.Add(new Supplier(rates, stock));
        }

        for (int i = 0; i < consumersCount; i++)
        {
            int need =
Convert.ToInt32(this.dataGridView2.Rows[0].Cells[i].Value);
```

```
        _consumers.Add(new Consumer(need));
    }

    private void ConfigureGui()
    {
        AutoSize = true;
        AutoSizeMode = AutoSizeMode.GrowOnly;
    }

    private void initButton_Click(object sender, EventArgs e)
    {
        this.dataGridView1.ColumnCount =
Convert.ToInt32(this.numericUpDown1.Value);
        this.dataGridView1.RowCount =
Convert.ToInt32(this.numericUpDown2.Value) + 1;
        this.dataGridView1.AutoSize = true;
        this.dataGridView1.Visible = true;

        this.dataGridView2.ColumnCount =
Convert.ToInt32(this.numericUpDown2.Value);
        this.dataGridView2.RowCount = 1;
        this.dataGridView2.AutoSize = true;
        this.dataGridView2.Visible = true;

        this.label4.Visible = true;
        this.label5.Visible = true;
        this.CalculateButton.Visible = true;
    }
}
```