**1 лаба**

**Математическое программирование** – это раздел высшей математики, посвященный решению задач, связанных с нахождением экстремумов функций нескольких переменных, при наличии ограничений на переменные.

Методами математического программирования решаются задачи о распределении ресурсов, планировании выпуска продукции, ценообразования, транспортные задачи и т.д.

**Переменными** задачи называются величины x1, x2, x3, …, xn которые полностью характеризуют экономический процесс.

**Сис**т**ема ограничений** включает в себя систему уравнений и неравенств, которым удовлетворяют переменные задачи и которые следуют из ограниченности ресурсов или других экономических или физических условий, например, положительности переменных и т. п.

**Целевой функцией** называют функцию переменных задачи, которая характеризует качество выполнения задачи, и экстремум которой требуется найти.

Допустимое решение, при котором целевая функция достигает оптимального значения, называется оптимальным планом

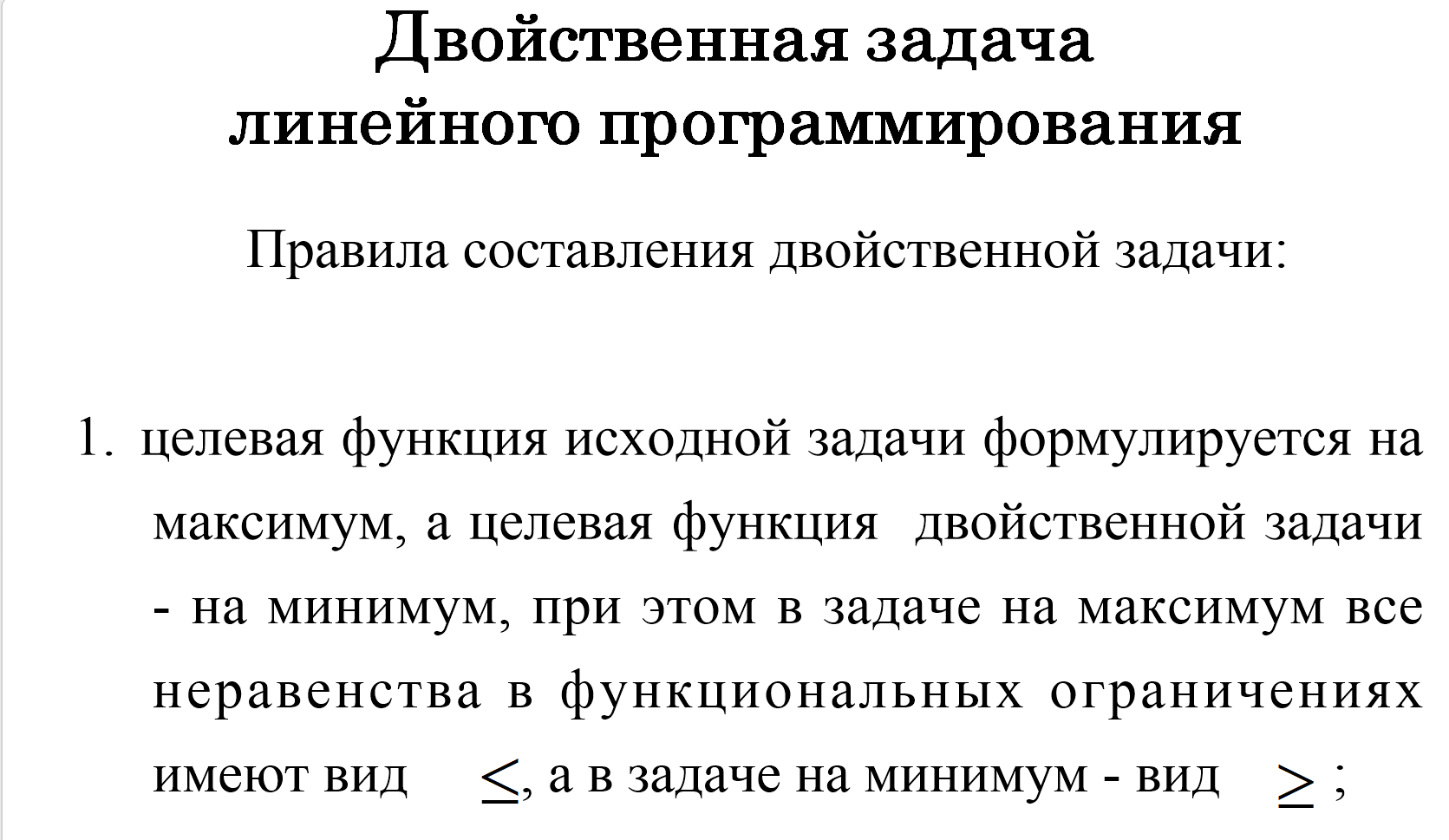
Построение математической модели экономической задачи включает следующие этапы:

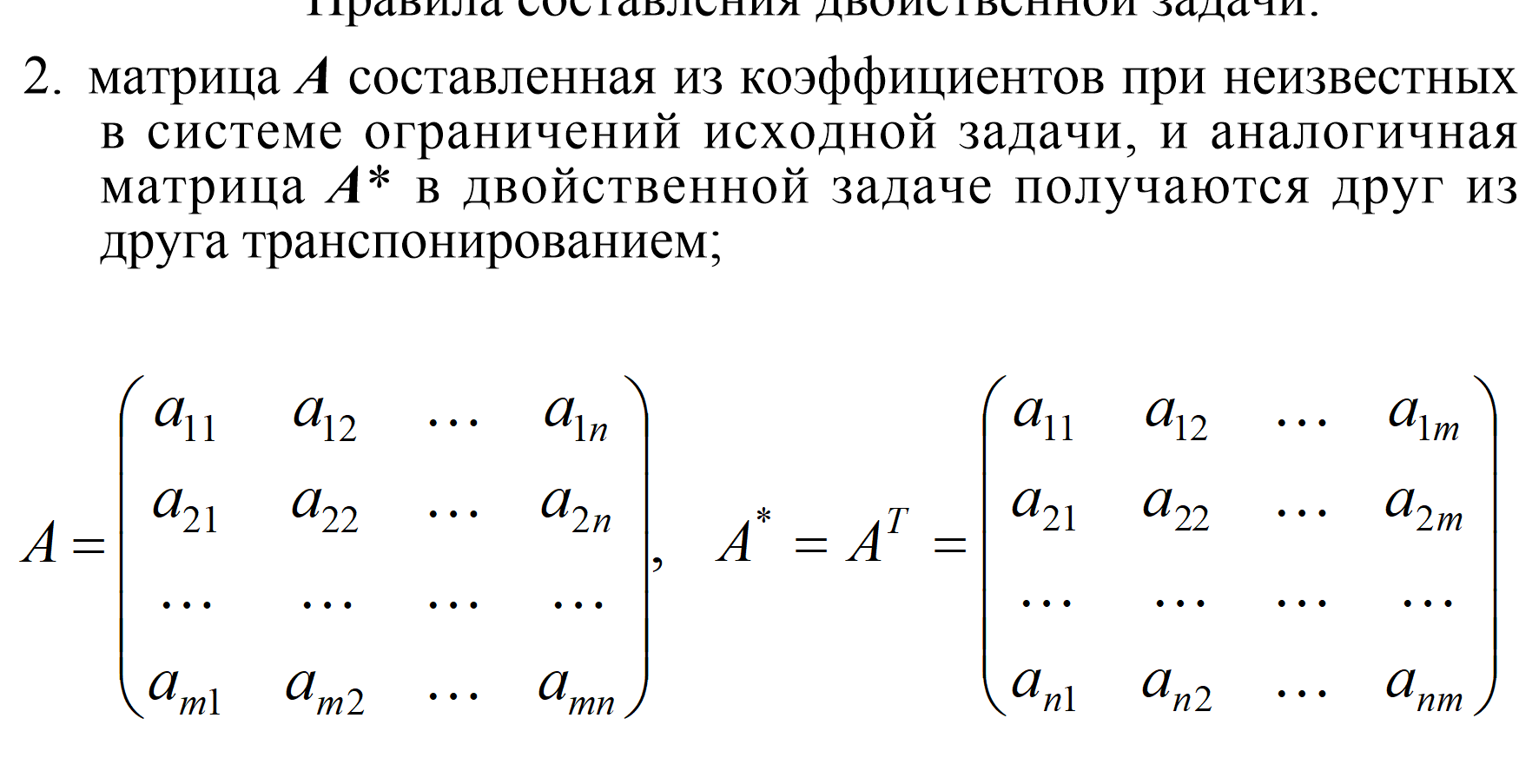
* Выбор переменных задачи
* Составление системных ограничений
* Выбор целевой функции

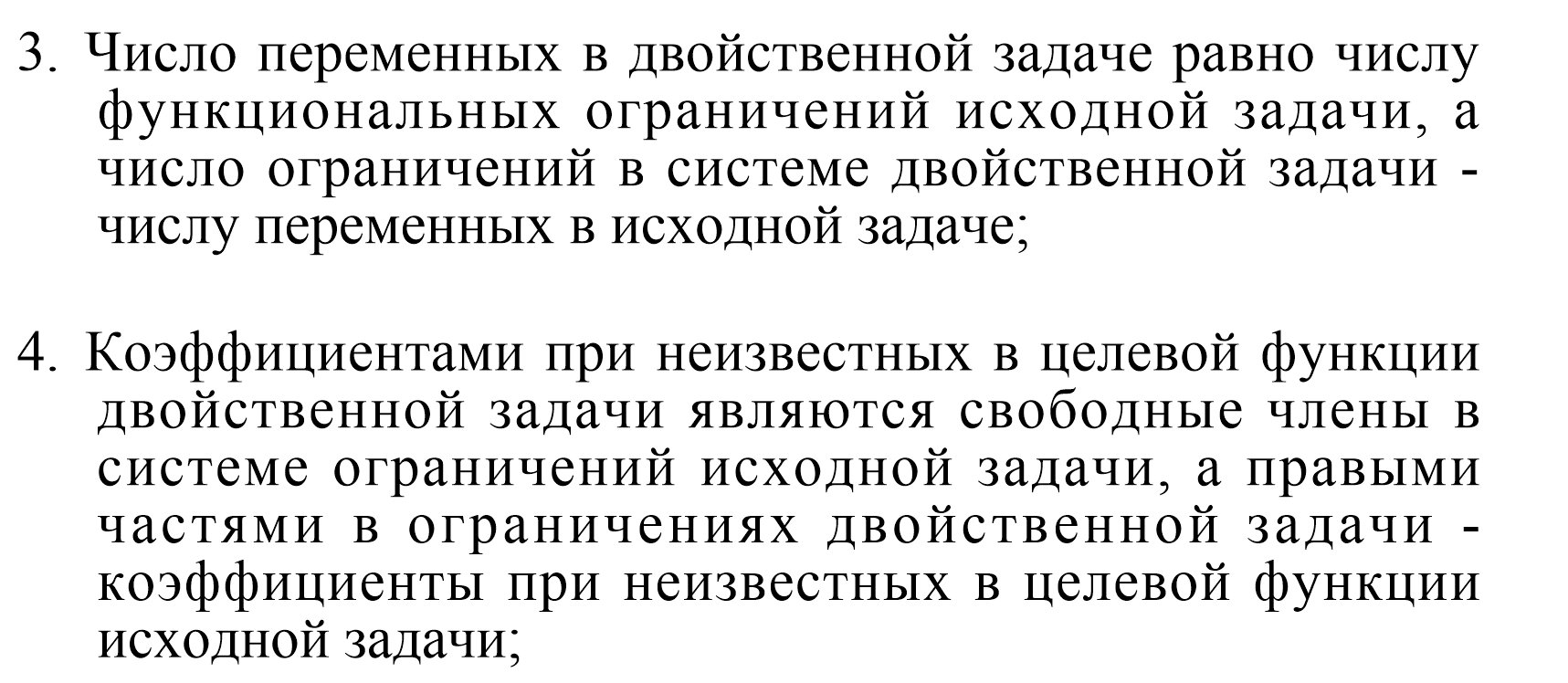


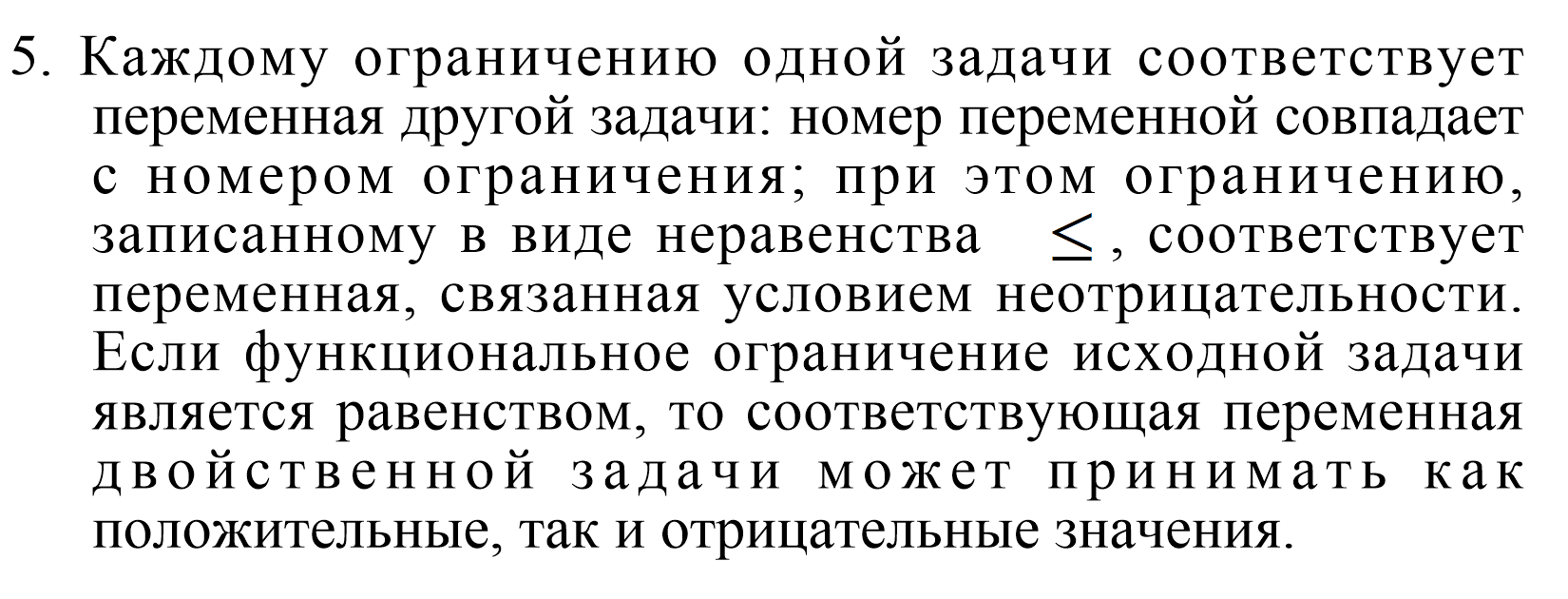
Элементарные преобразования системы (или расширенной матрицы)

* Перестановка любых двух уравнений;
* Умножение обеих частей одного из уравнений на любое отличное от нуля число;
* Прибавление к обеим частям одного и уравнения соответствующих частей другого, умноженных на любое число отличное от нуля;
* Вычеркивание нулевой строки (уравнения с нулевым коэффициентом и своборным членом, равным нулю)



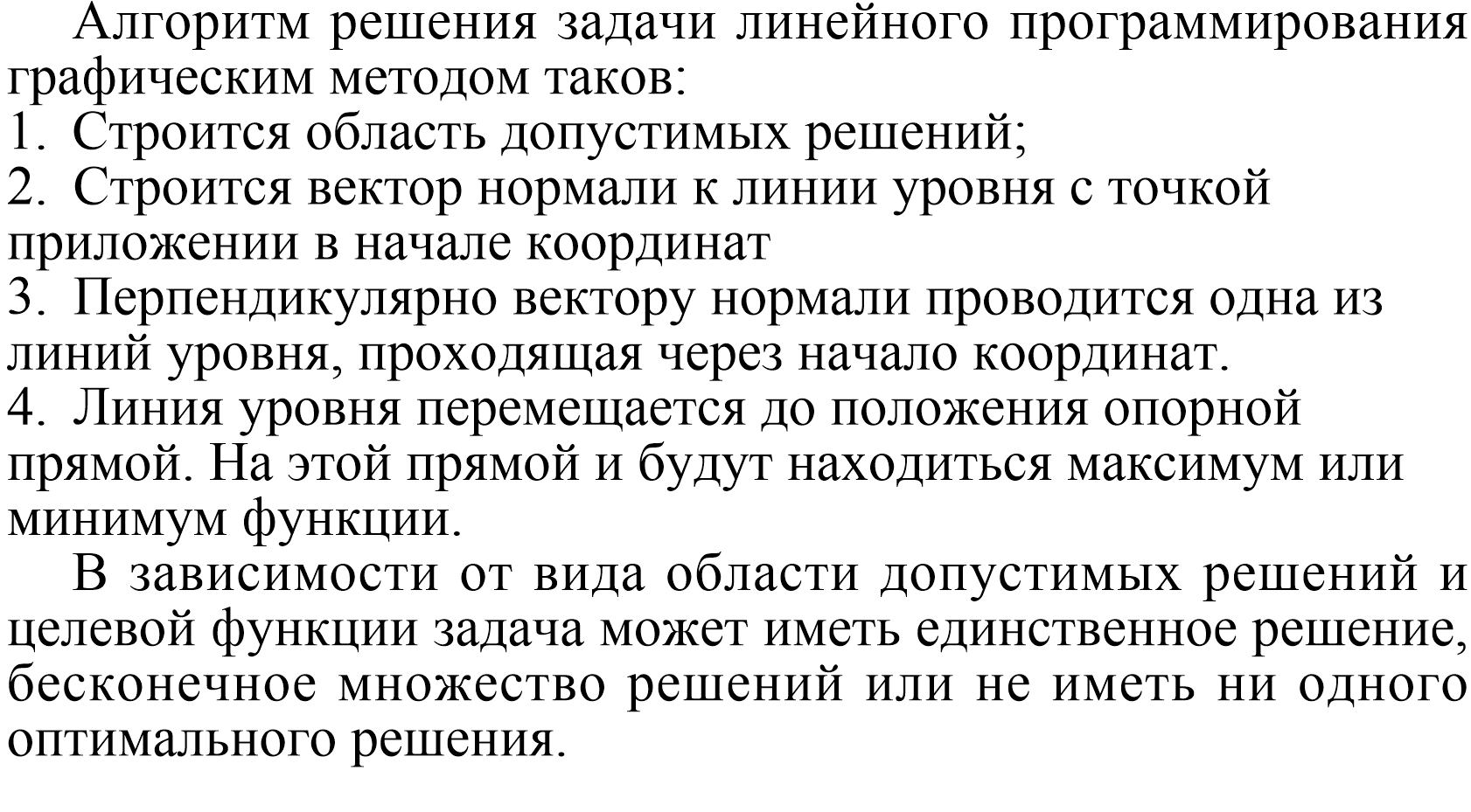






Графический метод решения задачи ЛП состоит из 2 этапов:

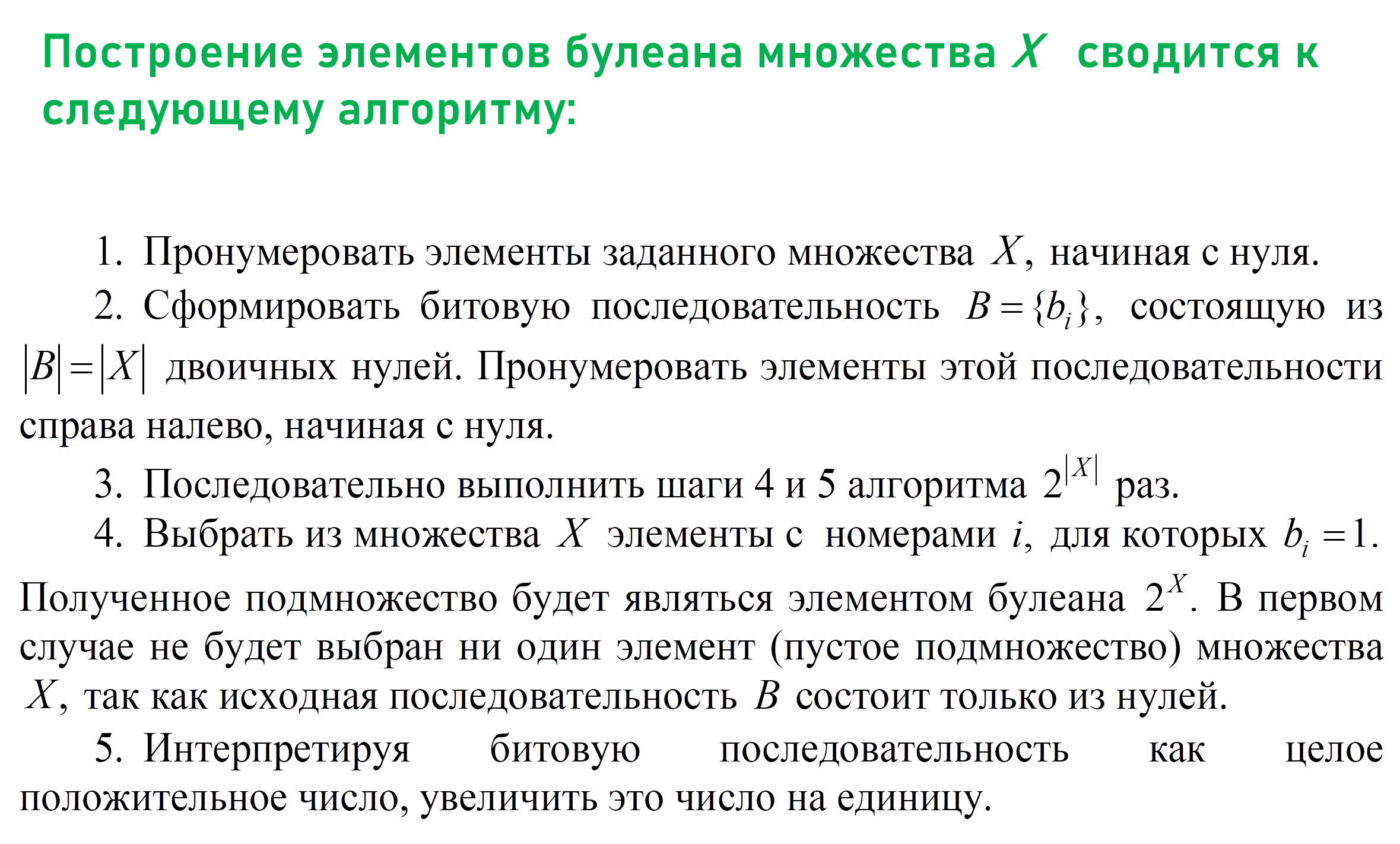
1. построение пространства допустимых решений, удовлетворяющих всем ограничениям модели;
2. нахождение оптимального решения среди всех точек пространства допустимых решений.



**2 лаба**

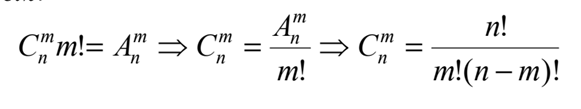
**Разобрать и разработать генератор подмножеств заданного множества.**

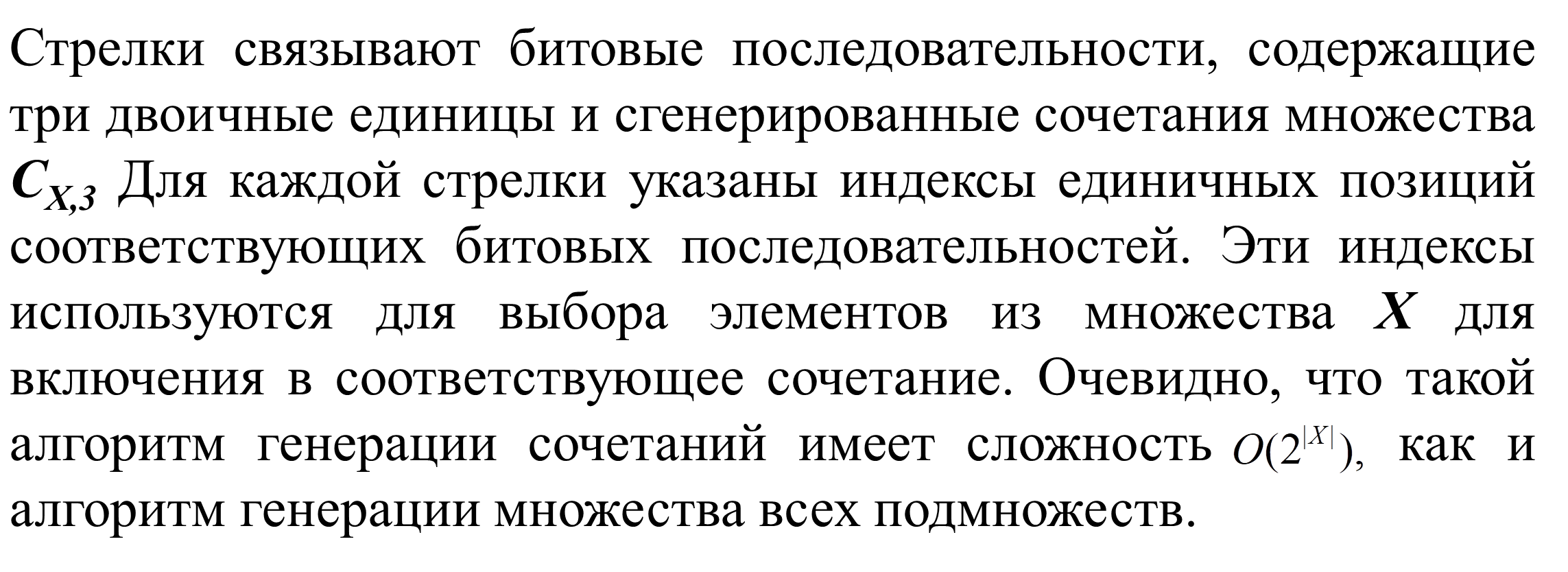
соединения из n элементов, каждое из которых содержит все n элементов, и которые отличаются лишь порядком элементов



**Разобрать и разработать генератор сочетаний.**

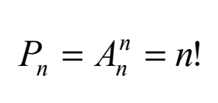
Сочетания – соединения, каждое из которых содержит ровно м различных элементов и которые отличаются хотя бы одним элементом.

****



**Разобрать и разработать генератор перестановок.**

Перестановки - соединения из n элементов, каждое из которых содержит все n элементов, и которые отличаются лишь порядком элементов

****

Построение множества всех перестановок с помощью алгоритма Джонсона – Троттера сводится к следующей процедуре:

1. Построить первую перестановку. Первая перестановка – это последовательность всех элементов множества *X* перечисленных в порядке возрастания. Стрелки всех элементов последовательности направлены влево.
2. Найти наибольший мобильный элемент в текущей перестановке. Если в последовательности нет мобильного элемента, то построены все перестановки элементов множества *X* – алгоритм закончил свою работу.

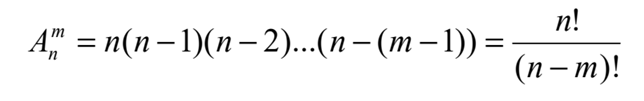
3. Поменять местами наибольший мобильный элемент и элемент, на который указывает стрелка наибольшего мобильного элемента.

4. Найти все элементы, большие, чем наибольший мобильный элемент (если они есть) и изменить их стрелки на противоположное направление.

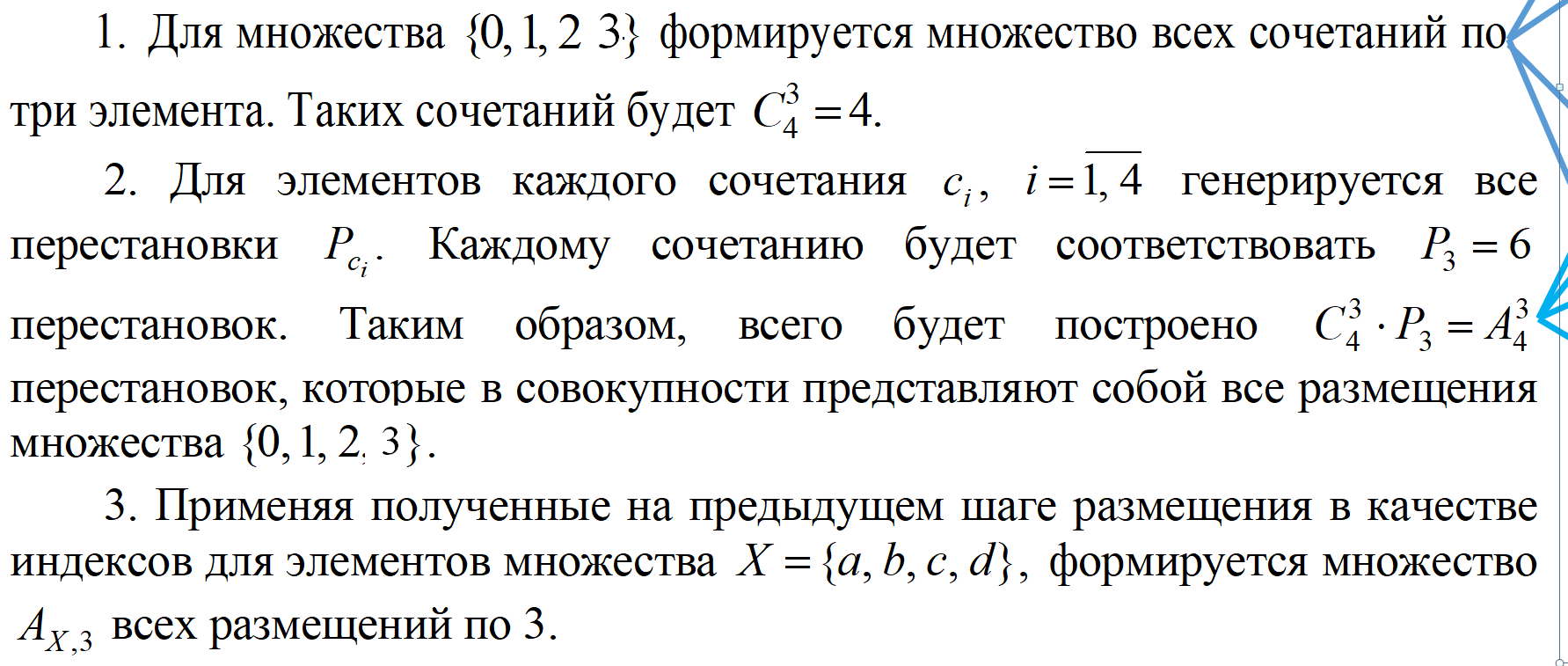
5. Перейти к пункту 2.

**Разобрать и разработать генератор размещений.**

Размещение - соединения из м элементов по н, каждое из которых содержит ровно м различных элементов и которые отличаются либо самими элементами, либо порядком элементов.

****

****

****

**1. Что такое комбинаторный анализ?**

Комбинаторный анализ - Это раздел математики, посвященный решению задач выбора и расположения элементов некоторого, обычно конечного, множества в соответствии с заданными правилами

**6. Треугольник Паскаля**

Треугольник Паскаля является структурой чисел, в которой каждое число является суммой двух чисел, расположенных над ним. Таблица биномиальных коэфицентов

**13. Как реализована структура xcombination**

public:

xcombination(short n, short m); // Конструктор

~xcombination(); // Деструктор

void reset(); // Сброс состояния

short getfirst(); // Получение первого сочетания

short getnext(); // Получение следующего сочетания

short ntx(short i); // Получение элемента сочетания по индексу i

unsigned \_\_int64 count() const; // Вычисление общего числа сочетаний

**14. Для чего нужны функции getfirst и getnext генератора сочетаний**

1. getfirst():
   * getfirst предназначена для получения первого сочетания в порядке увеличения лексикографического порядка.
   * Когда вызывается getfirst, генератор переходит к начальному состоянию, где выбраны первые m элементов множества (например, первые m элементов массива).
   * Возвращает размер текущего сочетания.
2. getnext():
   * getnext используется для получения следующего сочетания после текущего.
   * При каждом вызове getnext, генератор переходит к следующему сочетанию в лексикографическом порядке.
   * Если достигнуто последнее сочетание, дальнейшие вызовы getnext будут возвращать информацию об окончании последовательности.
   * Возвращает размер текущего сочетания (если сочетание существует) или -1, если достигнут конец последовательности.

**15. Для чего нужны функции ntx, count и reset генератора сочетаний**

ntx(short i):Функция ntx используется для получения элемента сочетания по его индексу i.

Например, если текущее сочетание - [2, 4, 6], то вызов ntx(1) вернет второй элемент, то есть 4.

count():Функция count возвращает общее количество сочетаний для данного генератора сочетаний.

Обычно используется для определения размера пространства поиска сочетаний.

reset():Функция reset служит для сброса состояния генератора сочетаний к начальному состоянию.

После вызова reset генератор будет находиться в состоянии, соответствующем первому сочетанию.

**3 лаба**

***Метод ветвей и границ* – это общий алгоритмический метод решения задач комбинаторной оптимизации.**

**Метод ветвей и границ был предложен для решения общей задачи целочисленного линейного программирования.**

**Метод является вариацией полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений.**

* **Задача о коммивояжере**
* **Минимаксные задачи о назначениях**
* **Задачи календарного планирования**
* **Задача о трех станках(вероятностные задачи)**

**Inf 0 6 Inf 0**

**0 Inf 0 44 60**

**0 10 Inf 72 47**

**0 5 4 Inf 9**

**56 41 12 0 Inf**

**4 лаба**

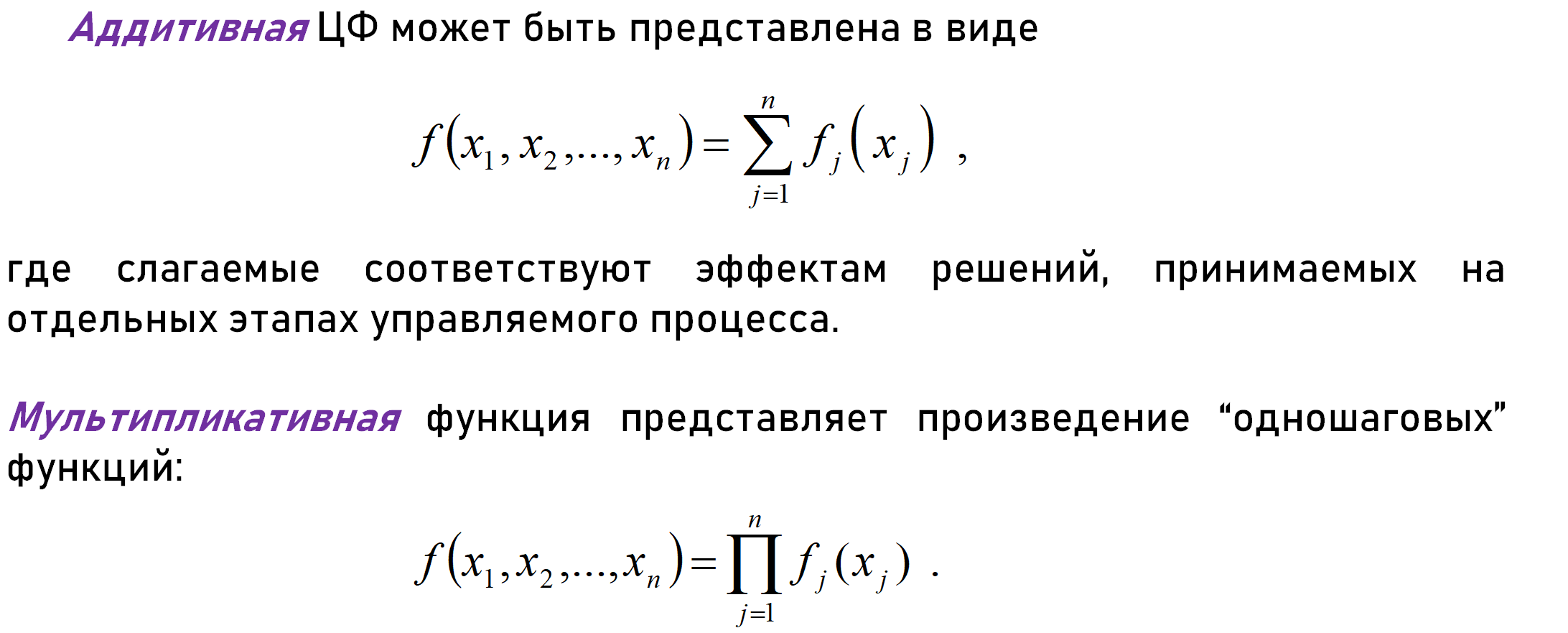
* + - 1. **В каких областях используется динамическое программирование?**

Динамическое программирование используется при оптимальном планировании управляемых процессов и наиболее эффективно в случае многошаговых или многоэтапных процессов принятия решений.

* + - 1. **В чем заключается задача динамического программирования?**

Задача ДП состоит в поиске оптимального управления, переводящего систему из начального состояния в конечное, и обеспечивающего экстремум целевой функции. Классическим примером задачи ДП является планирование промышленного объединения, состоящего из ***k*** предприятий на период ***n*** лет. Выделяемые в начале каждого года средства должны быть распределены между предприятиями таким образом, чтобы суммарный доход за весь период планирования был максимальным.

* + - 1. **Чем аддитивная функция отличается от мультипликативной?**

****

* + - 1. **Каков принцип оптимальности Беллмана?**

Каково бы ни было состояние управляемой системы *S* перед очередным шагом, шаговое управление необходимо выбирать так, чтобы выигрыш на данном шаге плюс оптимальный выигрыш на всех последующих шагах был максимальным.

* + - 1. **Что такое рекурсивный алгоритм?**

Рекурсивный алгоритм– это алгоритм, решающий задачу путем сведения ее к решению одной или нескольких таких же задач, но в сокращенном их варианте

* + - 1. **Что такое рекурсивная функция?**

Рекурсивная функция – это функция, которая вызывает саму себя.

* + - 1. **Что такое системный стек?**

Специальная секция памяти которая резервируется для хранения контекста операционной системой

* + - 1. **Объясните понятие «глубина рекурсии»**

цепочка вызовов функций

* + - 1. **Поясните своими словами схему решения задачи по принципу «разделяй и властвуй»**

Часто рекурсивные функции, применяемые для решения оптимизационных задач, используют более одного рекурсивного вызова, каждый из которых работает приблизительно с половиной входных данных.

* + - 1. **Что такое редакционное расстояние?**

Дистанция Левенштейна определяется между двумя строками и равна минимальному количеству операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.

* + - 1. **Что такое подпоследовательность и как её можно получить из последовательности?**



**6 лаба**

1. Какие представления графов Вы знаете?
   * Матрица смежности: Это представление графа в виде квадратной матрицы, где строки и столбцы представляют вершины графа, а элементы матрицы указывают на наличие (или отсутствие) рёбер между вершинами. В случае ненаправленных графов матрица смежности симметрична относительно главной диагонали.
   * Список смежности: Каждой вершине сопоставляется список вершин, с которыми она соединена рёбрами. Это представление позволяет эффективно обрабатывать соседние вершины.
   * Матрица инцидентности: В этом представлении строки матрицы соответствуют вершинам, а столбцы - рёбрам. Элемент матрицы указывает, инцидентен ли рассматриваемый ребро данной вершине. Это представление чаще используется в направленных графах.
2. В чем заключается поиск в ширину? Где рационально его использовать?

Поиск в ширину (BFS - Breadth-First Search) - это алгоритм поиска в графе, который исследует все узлы графа поочередно, начиная с начальной вершины, а затем переходя к следующему слою узлов по мере удаления от начальной вершины.

Основная идея BFS заключается в использовании очереди для посещения вершин в порядке, соответствующем расстоянию от начальной вершины. Начиная с начальной вершины, BFS сначала посещает всех её соседей, затем соседей соседей и так далее.

Применение BFS:

* Поиск кратчайшего пути: BFS позволяет находить кратчайший путь между двумя вершинами в ненаправленных и взвешенных графах. Поскольку BFS идёт от вершины к вершине в порядке их удаления от начальной, первый найденный путь обычно является кратчайшим.
* Поиск в графах без весов: BFS часто используется для обхода графов без весов, таких как деревья или графы с одинаковой длиной всех рёбер.
* Поиск в ширину также может использоваться для проверки связности графа: Если BFS достигает всех вершин из стартовой вершины, то граф связный.

1. В чем заключается поиск в глубину? В каких ситуациях рационально его использовать?

Поиск в глубину (DFS - Depth-First Search) - это алгоритм поиска в графе, который исследует одну ветвь графа до тех пор, пока не достигнет конечной вершины, а затем возвращает и исследует другие ветви.

Основная идея DFS заключается в использовании стека (или рекурсии) для рекурсивного исследования вершин. Начиная с начальной вершины, DFS переходит к первой соседней вершине, затем к соседней вершине этой вершины и так далее, пока не будет достигнута конечная вершина, после чего возвращается к предыдущей вершине и продолжает исследование.

Применение DFS:

* Топологическая сортировка: DFS может использоваться для топологической сортировки направленных ациклических графов (DAG). Топологическая сортировка позволяет упорядочить вершины графа так, чтобы все рёбра шли от вершин с меньшим порядковым номером к вершинам с большим.
* Поиск в глубину также используется в задачах поиска циклов: Он может быть использован для обнаружения циклов в направленных графах.
* Генерация подмножеств и перестановок: DFS может использоваться для генерации всех подмножеств или перестановок элементов.
* Решение задач на графах: DFS используется в алгоритмах для нахождения компонент связности, проверки связности, нахождения сильно связанных компонент, нахождения мостов и точек сочленения.

1. В чем смысл топологической сортировки? Для чего она применяется?

Топологическая сортировка - это процесс упорядочивания вершин ориентированного ациклического графа (DAG) таким образом, что каждое ребро в графе идёт от вершины с меньшим порядковым номером к вершине с большим порядковым номером. То есть, если существует ребро от вершины A к вершине B, то вершина A должна идти перед вершиной B в топологической сортировке.

Смысл топологической сортировки заключается в том, чтобы упорядочить вершины таким образом, чтобы все зависимости между вершинами графа были удовлетворены. Это особенно важно в контексте задач, где существует порядок выполнения или зависимости между элементами, например:

* В процессе сборки программы некоторые компоненты зависят от других. Топологическая сортировка может определить правильный порядок компиляции или сборки этих компонентов.
* Если задачи имеют зависимости друг от друга, то топологическая сортировка может помочь определить порядок их выполнения.
* В процессе анализа данных или в программировании может быть важно понять, какие элементы зависят от других. Топологическая сортировка позволяет увидеть эти зависимости и определить порядок обработки данных или выполнения операций.

1. Что такое минимальное остовное дерево?

Минимальное остовное дерево (Minimum Spanning Tree, MST) в неориентированном взвешенном графе - это подмножество рёбер графа, образующее дерево и содержащее все вершины графа, причём сумма весов рёбер этого дерева минимальна. Другими словами, минимальное остовное дерево является деревом с минимальной суммой весов его рёбер, которое связывает все вершины графа.

1. В чем заключается стандартный алгоритм построения минимального остовного дерева?

Стандартный алгоритм для построения минимального остовного дерева обычно базируется на двух основных подходах: алгоритме Прима и алгоритме Крускала. Оба алгоритма работают на неориентированных взвешенных графах и стремятся найти минимальное остовное дерево.

1. К какой категории алгоритмов относятся алгоритмы Прима и Крускала?

Алгоритмы Прима и Крускала относятся к классу жадных алгоритмов. Жадные алгоритмы принимают локально оптимальные решения на каждом шаге в надежде, что они приведут к глобально оптимальному решению.

Алгоритм Прима ищет минимальное остовное дерево, выбирая каждый раз ребро минимального веса, которое связывает уже включённые вершины с вершинами, ещё не включенными в остовное дерево.

Алгоритм Крускала, в свою очередь, выбирает рёбра в порядке возрастания их веса и добавляет их к остовному дереву, при условии, что добавление этого ребра не создаст цикл в дереве.

1. Опишите один шаг алгоритма Крускала? Когда алгоритм прекращает свою работу?

Один шаг алгоритма Крускала состоит из выбора ребра с наименьшим весом из списка всех рёбер, которые ещё не включены в минимальное остовное дерево, и проверки, не создаст ли включение этого ребра в дерево цикл.

Выбор ребра с наименьшим весом: Из списка всех рёбер графа, которые ещё не включены в остовное дерево, выбирается ребро с наименьшим весом.

Проверка наличия цикла: Проверяется, создаст ли включение этого ребра в остовное дерево цикл. Это можно сделать, например, с помощью алгоритма поиска в глубину или поиска в ширину, чтобы проверить, существует ли путь между вершинами, соединяемыми ребром.

Добавление ребра в остовное дерево: Если включение ребра не создаёт цикл, то оно добавляется в минимальное остовное дерево.

Продолжение процесса: Процесс продолжается снова с выбора ребра с наименьшим весом из оставшихся рёбер и проверки его наличия цикла.

Алгоритм Крускала завершает свою работу, когда все вершины графа соединены между собой рёбрами, и они образуют единственное связное минимальное остовное дерево. Это происходит, когда количество рёбер в дереве достигает V−1, где V - количество вершин в графе.

**7 лаба**

Основные методы сетевого планирования.

* метод критического пути (Critical Path Method - СРМ)
* метод оценки и обзора программ (Program Evaluation and Review Technique - PERT).

Какой ключевой фактор проекта учитывается при выборе между методом критического пути и методом оценки и обзора программ.

Метод критического пути (СРМ) применяется тогда, когда операции, входящие в состав комплекса работ, имеют известные строго определенные продолжительности (являются детерминированными).

В свою очередь, метод оценки и обзора программ (РЕRТ) применяется при планировании проектов, для которых характерна неопределенность в оценке затрат времени, необходимого для выполнения отдельных операций

Три основных этапа сетевого планирования и управления.

* Структурное планирование
* Календарное планирование
* Оперативное управление.

Что такое сетевая модель?

Сетевой моделью называется модель, отражающая комплекс работ (операций) и событий, связанных с реализацией некоторого проекта в их логической и технологической последовательности и связи.

Три вида событий в сетевом проектировании и управлении.

исходное, завершающее и промежуточное. С исходного события начинается выполнение комплекса операций. Завершающее событие соответствует достижению конечной цели.

Какой сетевой график называется многоцелевым?

Сетевые графики с несколькими завершающими событиями называются многоцелевыми.

Три вида операций в сетевом графике.

В чем разница между событием и операцией? вершины, называемые событиями, соответствуют моментам времени начала или окончания одной или нескольких операций, а дуги – операциям.

Что такое коэффициент дополнительных затрат - это отношение полного объёма выполненной работы к реальному количеству времени, затраченному на выполнение этой работы.

**8 лаба**