В. Липский

Комбинаторика для программистов.

Москва
«Книга по Требованию»

УДК 51
ББК 22.1
В11

В. Липский
В11 Комбинаторика для программистов. / В. Липский – М.: Книга по Требованию, 2023. – 200 с.

ISBN 978-5-458-26099-2

В книге изложение начинается с теоретического обзора и заканчивается описанием алгоритмов, практически готовых к автоматическому исполнению на ЭВМ. Конечно, новизна такого подхода относительна и традиция завершения математического трактата правилами вычислений восходит, по меньшей мере, к ал-Хорезми. Но все же оформление операционного багажа математики в виде машинных программ существенно отличает новые книги от классических инженерных руководств по прикладной математике.

ISBN 978-5-458-26099-2

Глава 1

Введение в комбинаторику

1.1 Основные понятия

В этом разделе приводятся основные определения и обозначения, относящиеся к используемым логическим и теоретико-множественным понятиям, а также представленные в приводимых ниже алгоритмах.

Начнем с логических и теоретико-множественных понятий (читателя, заинтересованного в более глубоком знакомстве с этими понятиями, мы отсылаем к работам [49] и [57]). Мы будем употреблять логические связки $\vee$ (или), $\wedge$ (и), $\neg$ (не), $\Rightarrow$ (если ..., то), $\Leftrightarrow$ (тогда и только тогда, когда). Тот факт, что x есть элемент множества X, будем записывать в виде $x \in X$, его отрицание — в виде $x \notin X$. Множество тех элементов множества X, которые удовлетворяют условию Φ, будем обозначать через $\{x \in X : \Phi\}$ (или $\{x : \Phi\}$, если известно, о каком множестве X идет речь), запись же $\{a_1, \ldots, a_n\}$ будет обозначать множество, элементы которого суть $a_1, \ldots, a_n$ (в частности, единственным элементом множества $\{a\}$ является a). Теоретико-множественные операции объединения, пересечения и разности обозначаются соответственно $\cap$, $\cup$ и $\setminus$, пустое множество обозначается $\varnothing$. Тот факт, что множество A содержится в множестве B (т.е. A есть подмножество множества B), будет записываться в виде $A \subseteq B$ или $B \supseteq A$ (всегда имеют место включения $\varnothing \subseteq A$, $A \subseteq A$); символ «$\subset$» зарезервирован для случая, когда исключается равенство $A = B$ (при этом будем говорить, что A есть *собственное подмножество* множества B). Множество всех подмножеств множества X будем обозначать через $\wp(X)$, мощность множества X (т.е. число его элементов) — через $|X|$.

Последовательность длины n, члены которой суть $a_1, \ldots, a_n$, будем обозначать через $\langle a_1, \ldots, a_n \rangle$, либо просто через $a_1, \ldots, a_n$ или $a_1 \ldots a_n$. Последователь-

ность $\langle a, b\rangle$ длины два будем называть *упорядоченной парой*. *Декартово произведение* $A \times B$ множеств A и B определяется как множество всевозможных пар $\langle a, b\rangle$, где $a \in A$, $b \in B$. Под *бинарным отношением* (с *левой областью* A и *правой областью* B) подразумевается произвольное подмножество $R \subseteq A \times B$. Если $A = B$, то будем говорить о *бинарном отношении* на множестве A. Вместо $\langle a, b\rangle \in R$ часто пишут aRb.

По поводу отношения R на множестве X говорят, что оно:

- (а) рефлексивно, если xRx для каждого $x \in X$,
- (б) транзитивно, если $(xRy \wedge yRz) \Rightarrow xRz$ для произвольных $x, y, z \in X$,
- (в) симметрично, если $xRy \Rightarrow yRx$ для произвольных $x, y \in X$,
- (г) антисимметрично, если $(xRy \wedge yRx) \Rightarrow x = y$ для произвольных $x, y \in X$.

Произвольное бинарное отношение, обладающее свойствами рефлексивности, транзитивности и симметричности, называется *отношением эквивалентности*, а обладающее свойствами рефлексивности, транзитивности и антисимметричности, — *отношением частичной упорядоченности*. Отношение частичной упорядоченности обычно обозначается через «$\leq$», а пара $\langle X, \leq\rangle$ называется *частично упорядоченным множеством*. Будем применять также очевидные обозначения, такие как $x \geq y$ для $y \leq x$, $x < y$ для $x \leq y \wedge x \neq y$ и т.д. Примером частично упорядоченного множества может служить множество целых чисел с отношением делимости, множество целых (или вещественных) чисел с обычным отношением меньше или равно «$\leq$», а также множество $\wp(X)$ с отношением включения $\subseteq$.

Если функция (отображение) f сопоставляет каждому элементу $x \in X$ элемент $f(x) \in Y$, то будем писать $f : X \to Y$ (такая функция может трактоваться как отношение $R \subseteq X \times Y$ с тем свойством, что для каждого $x \in X$ существует в R точно одна пара вида $\langle x, y\rangle$, $y \in Y$, для наших же целей достаточно, однако, интуитивного понятия функции). Для произвольных $A \in X$, $B \in Y$ определим

$$f(A) = \{y \in Y : \text{существует такое } x \in A, \text{что } y = f(x)\}$$

$$f^{-1}(B) = \{x \in X : f(x) \in B\}$$

(вместо $f^{-1}(\{b\})$ будем просто писать $f^{-1}(b)$.

Если $f(X) = Y$, то будем говорить о функции из X на Y. Функция $f : X \to Y$ называется *обратимой* (*взаимно однозначной*), если для произвольных $a, b \in X$

$$a \neq b \Rightarrow f(a) \neq f(b).$$

Мы часто будем использовать понятие графа (см. [9],[31]). Под *неориентированным графом* (или короче *графом*) будем донимать такую произвольную пару $G = \langle V, E\rangle$, что

$$E \subseteq \{\{u, v\} : u, v \in V \wedge u \neq v\}.$$

Ориентированным графом[1] будем называть такую произвольную пару $G = \langle V, E\rangle$, что $E \in V \times V$ и в обоих случаях множества V и E будем называть соответственно множеством *вершин* и множеством *ребер*[2] графа G.

Граф обычно изображается на плоскости в виде множества точек, соответствующих вершинам, и соединяющих их линий, соответствующих ребрам.[3] Линия, изображающая ребро $\{u, v\}$, или $\langle u, v\rangle$[4], соединяет точки, изображающие вершины u,v причем во втором случае стрелка обозначает направление от u к v (рис. 1.1).

В контексте определенного графа $G = \langle V, E\rangle$ будем часто использовать обозначения $u - v$, $u \to v$ вместо $\{u, v\} \in E$ и $\langle u, v\rangle \in E$ соответственно. Если ребро e имеет вид $\{u, v\}$ или $\langle u, v\rangle$, то будем говорить, что ребро e *инцидентно* вершинам u и v, в то время как вершины u и v *смежны* между собой. *Степень вершины* определим как число ребер, инцидентных ей.[5] Вершину нулевой степени будем называть *изолированной* (например, вершина v_5 на рис.1.1, а). *Путем* в графе $G = \langle V, E\rangle$ назовем последовательность вершин $v_0, v_1, \ldots, v_n$, такую, что $k \geq 0$ и $v_i - v_{i+1}$ (или $v_i \to v_{i+1}$, если граф G — ориентированный), $i = 0, \ldots, k-1$.[6] Вершины v_0 и v_k будем называть соответственно *началом* и *концом* пути, а число k — *длиной* пути. Путь, начало и конец которого совпадают, будем называть *циклом*[7]. Если все вершины пути $v_1, \ldots, v_k$ различны, то будем говорить об *элементарном* пути. Соответственно цикл $v_1, \ldots, v_k$ $(v_1 = v_k)$ будем называть *элементарным*, если вершины $v_1, \ldots, v_k$ различны. *Подграфом* графа $G = \langle V, E\rangle$ будем называть такой произвольный граф $G' = \langle V', E'\rangle$, что

[1] или короче *орграфом* — *Прим. перев.*

[2] Элементы множества E для орграфа называются *дугами* — *Прим. перев.*

[3] Дуга в орграфе изображается линией со стрелкой, указывающей ориентацию дуги, т.е. направление от ее начала к концу. — *Прим. перев.*

[4] Угловые скобки используются для обозначения дуг орграфа. — *Прим. перев.*

[5] Для вершин орграфа определяются *полустепени захода* (число заходящих в вершину дуг) и *исхода* (число выходящих дуг). Степень вершины определяется как сумма полустепеней захода и исхода. — *Прим. перев.*

[6] Термин «путь» в теории графов используется только в отношении орграфов, для графов используются термины «цепь» или «маршрут». — *Прим. перев.*

[7] Введенный так термин «цикл» в теории графов используется только в отношении графов, для орграфов используется термин «контур». — *Прим. перев.*

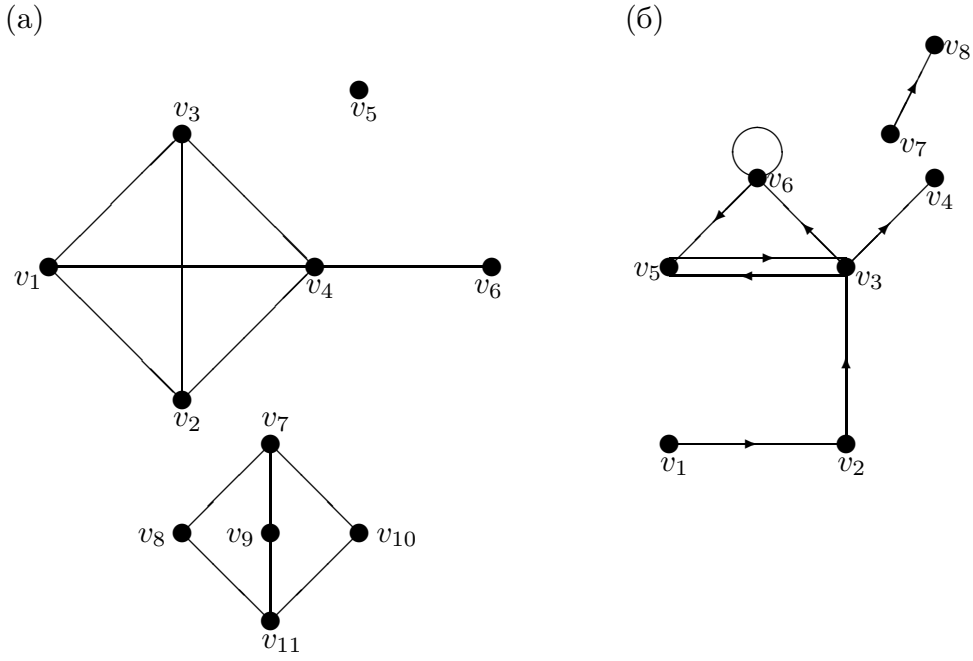

Рис. 1.1: а) неориентированный граф б) неориентированный граф

$V' \subseteq V$ и $E' \subseteq E$.[8]

Пусть $G = \langle V, E\rangle$ — произвольный неориентированный граф, и пусть $v \in V$. Пусть A — множество тех вершин $u \in V$, к которым существует путь из v. Множество A вместе с ребрами графа G, инцидентными вершинам из A, определяет некоторый подграф, называемый компонентой связности графа G. Очевидно, что множества вершин компонент связности произвольного графа попарно не пересекаются. Например, для графа на рис.1.1, а это суть множества $V_1 = \{v_1, v_2, v_3, v_4, v_6\}$, $V_2 = \{v_5\}$ и $V_3 = \{v_7, v_8, v_9, v_{10}, v_{11}, v_{12}\}$.

Будем говорить, что графы $G = \langle V, E\rangle$, $G' = \langle V', E'\rangle$ *изоморфны*, если существует такое взаимно однозначное отображение f из V на V', что для произвольных $u, v \in V$ имеем $\{u, v\} \in E \Leftrightarrow \{f(u), f(v)\} \in E'$ ($\langle u, v\rangle \in E \Leftrightarrow \langle f(u), f(v)\rangle \in E'$ в случае ориентированных графов). Обычно изоморфные графы не различаются между собой.

Для произвольного вещественного числа x мы будем употреблять обозна-

[8] В отечественной литературе по теории графов граф G' называется чаще *частью* графа, или *частичным графом*, под подграфом же понимается частичный граф, удовлетворяющий дополнительному условию $\forall x, y\,(x, y \in V' \wedge \{x, y\} \in E \Rightarrow \{x, y\} \in E')$). — *Прим. перев.*

чения $\lfloor x \rfloor$ и $\lceil x \rceil$ соответственно для наибольшего целого числа, не превосходящего x, и для наименьшего целого числа, не меньшего x, например $\lfloor 3.5 \rfloor = 3$, $\lceil 3.5 \rceil = 4$, $\lfloor -3.5 \rfloor = -4$, $\lceil -3.5 \rceil = -3$.

Перейдем теперь к понятиям, связанным с алгоритмами. Алгоритмы будем обычно записывать на языке программирования, являющимся неформальной версией языка Паскаль [36,75]. Если реализация какого-либо фрагмента программы очевидна, но трудоемка и затемняет идею алгоритма, то такой фрагмент будем иногда заменять описанием на естественном языке. Мы будем также применять неформальные конструкции, такие как, например, циклы (**for** $x \in X$ **do** P (выполнять команду P для всех элементов x множества X в произвольной последовательности)), СТЕК $\Leftarrow x$ (поместить значение переменной x в стек), $x \Rightarrow$ СТЕК (считать элемент x из вершины стека и принять его за значение переменной x), ОЧЕРЕДЬ $\Leftarrow x$ (включить x в очередь в качестве последнего элемента), $x \Rightarrow$ ОЧЕРЕДЬ (взять первый элемент из очереди и принять его в качестве значения переменной x) и т.д. Мы будем обычно опускать описания типов и переменных (иногда для избежания недоразумений будем помещать соответствующие пояснения в комментарий). Переменная, появляющаяся в процедуре, рассматривается как локальная для данной процедуры, исключая тот случай, когда в комментарии сказано что-либо иное. Строки программы нумеруются так, чтобы можно было указать на «цикл 17», «блок 9» и т.д.

Основным параметром алгоритма, который будет нас интересовать, является его *вычислительная сложность* (или просто *сложность*), т.е. число шагов, выполняемых алгоритмом в худшем случае как функция размерности задачи, представленной входными данными. Например, если алгоритм принимает как данные произвольный граф $G = \langle V, E \rangle$, то под размерностью аадачи можно понимать $|V|$. Сложность алгоритма определяется тогда как функция f, такая что $f(n)$ равно наибольшему числу шагов алгоритма для произвольного графа с n вершинами. Можно также считать размерностью задачи пару $\langle |V|, |E| \rangle$ — тогда сложностью является функция двух переменных и $f(n, m)$ равно наибольшему числу шагов, выполняемых алгоритмом для произвольного графа с n вершинами и m ребрами. Остается еще объяснить точнее, что мы понимаем под «шагом» алгоритма. Допустим, что наши программы транслируются на машинный язык типичной ЭВМ, имеющей в наборе своих команд команды переноса слова из памяти в буфер и наоборот, арифметические операции сложения, вычитания, умножения и деления, условные переходы, операции ввода-вывода, а также косвенной адресации, выполненной аппаратно (т.е. определение аргумента операции через адрес ячейки памяти, содержащей адрес этого аргумента). Выполнение любой из указанных выше команд мы и будем считать шагом алгоритма. Очевидно, что при таком определении шага сложность алгоритма зависит от конкретного вида машинных команд. Однако нас никогда не будет интересовать

точная сложность алгоритма, а только асимптотическая сложность, т.е. асимптотическая скорость увеличения числа шагов алгоритма, когда размерность задачи неограниченно растет (чтобы можно было говорить о такой скорости роста, предполагаем, что объем памяти нашего компьютера неограниченный, а также, что каждая ячейка памяти может содержать произвольно большое целое число). Ясно, что при двух произвольных «разумных» способах трансляции соответствующие сложности различаются не более чем на мультипликативную постоянную, а их скорость роста одинакова. Читателя, желающего уточнить приведенные выше очень неформальные рассуждения, отсылаем к работам [1] и [2].

При сравнении скорости роста двух функций $f(n)$ и $g(n)$ (с неотрицательными значениями) очень удобны следующие обозначения:

- $f(n) = O(g(n)) \Leftrightarrow$ существуют константы $C, N > 0$, такие что $f(n) \leq C \cdot g(n)$ для всех $n \geq N$
- $f(n) = \Omega(g(n)) \Leftrightarrow$ существуют константы $C, N > 0$, такие что $f(n) \geq C \cdot g(n)$ для любого $n \geq N$.

Конечно, $f(n) = \Omega(g(n))$ тогда и только тогда, когда $g(n) = O(f(n))$. Символы $O(g(n))$ и $\Omega(g(n))$ читаются соответственно: «порядка не более чем $g(n)$» и «порядка не менее чем $g(n)$». Если сложность какого-либо алгоритма есть $O(g(n))$, то мы говорим, что этот алгоритм «затрачивает порядка $O(g(n))$ времени»[9]. Подобным же образом определяются символы $O(g(n_1, \ldots, n_k))$ и $\Omega(g(n_1, \ldots, n_k))$ для функции многих переменных, например:

$f(n_1, \ldots, n_k) = O(g(n_1, \ldots, n_k)) \Leftrightarrow$ существуют константы $C, N \geq 0$, такие что $f(n_1, \ldots, n_k) \leq g(n_1, \ldots, n_k)$ для всех $n_1, \ldots, n_k \geq N$

Определенную таким образом сложность иногда называют *временной сложностью* в отличие от *сложности по памяти*, определяющей величину объема памяти, использованного алгоритмом, как функцию размерности задачи.

1.2 Функции и размещения

Классической задачей комбинаторики является задача определения числа способов размещения некоторых объектов в каком-то количестве «ящиков» так, чтобы были выполнены заданные ограничения. Эту задачу можно сформулировать несколько более формально следующим образом. Даны множества X, Y, причем $|X| = n$, $|Y| = m$. Сколько существует функций $f : X \to Y$, удовлетворяющих заданный ограничениям? Элементы множества X соответствуют

[9]Символьную запись $f(n) = O(g(n))$ не следует трактовать как равенство; например, из $f(n) = O(g(n))$ и $h(n) = O(g(n))$, конечно, не вытекает $f(n) = h(n)$.

объектам, элементы множества Y — ящикам, а каждая функция $f : X \to Y$ определяет некоторое размещение, указывая для каждого объекта $x \in X$ ящик $f(x) \in Y$, в котором данный объект находится. Другую традиционную интерпретацию получим, трактуя Y как множество «цветов», а $f(x)$ как «цвет объекта x». Наша задача, таким образом, эквивалентна вопросу, сколькими способами можно покрасить объекты так, чтобы были соблюдены некоторые ограничения.

Заметим, что без потери общности можем всегда считать, что $X = 1, \ldots, n$ и $Y = 1, \ldots, m$. Каждую функцию f можно тогда отождествить с последовательностью $\langle f(1), \ldots, f(n)\rangle$.

Наша задача имеет самый простой вид, если не накладывается никаких ограничений на размещения. Имеет место следующая теорема.

Теорема 1.1. *Если $|X| = n$, $|Y| = m$, то число всех функций $f : X \to Y$ равно m^n.*

Доказательство. Считая, что $X = 1, \ldots, n$, сводим нашу задачу к вопросу о числе всех последовательностей $\langle y_1, \ldots, y_n\rangle$ с членами из m-элементного множества Y. Каждый член последовательности y_i мы можем выбрать m способами, что дает m^n возможностей выбора последовательности $\langle y_1, \ldots, y_n\rangle$. ■

Легко также найти число размещений, для которых каждый ящик содержит не более одного объекта — такие размещения соответствуют взаимно однозначным функциям. Обозначим через $[m]_n$ число всех взаимно однозначных функций из n-элементного множества в m-элементное множество.

Теорема 1.2. *Если $|X| = n$, $|Y| = m$, то число всех взаимно однозначных функций $f : X \to Y$ равно*

$$[m]_n = m(m-1)\ldots(m-n+1) \tag{1.1}$$

(полагаем $[m]_0 = 1$).

Доказательство. Будем определять на этот раз число инъективных (т.е. имеющих все различные члены) последовательностей $\langle y_1, \ldots, y_n\rangle$ с членами из множества Y. Элемент y_1 такого множества мы можем выбрать m способами, элемент y_2 — $m-1$ способами, в общем случае если уже выбраны элементы $y_1, \ldots, y_{i-1}$, то в качестве y_i можем выбрать любой из $m-i+1$ элементов множества $Y\backslash\{y_1, \ldots, y_{i-1}\}$ (принимаем $n \le m$, если $n > m$, то очевидно, что и $[m]_n$ и искомое число функций равны нулю). (Это дает $m(m-1)\ldots(m-n+1)$ возможностей выбора инъективных последовательностей $\langle y_1, \ldots, y_n\rangle$) ■

Приведем в качестве примера $[4]_3 = 24$ последовательности длины 3 с эле-

ментами из множества $X = \{1, 2, 3, 4\}$:

$$\begin{array}{cccc}
\langle 1,2,3\rangle & \langle 2,1,3\rangle & \langle 3,1,2\rangle & \langle 4,1,2\rangle \\
\langle 1,2,4\rangle & \langle 2,1,4\rangle & \langle 3,1,4\rangle & \langle 4,1,3\rangle \\
\langle 1,3,2\rangle & \langle 2,3,1\rangle & \langle 3,2,1\rangle & \langle 4,2,1\rangle \\
\langle 1,3,4\rangle & \langle 2,3,4\rangle & \langle 3,2,4\rangle & \langle 4,2,3\rangle \\
\langle 1,4,2\rangle & \langle 2,4,1\rangle & \langle 3,4,1\rangle & \langle 4,3,1\rangle \\
\langle 1,4,3\rangle & \langle 2,4,3\rangle & \langle 3,4,2\rangle & \langle 4,3,2\rangle
\end{array}$$

Если $m = n$, то каждая взаимно однозначная функция $f : X \to Y$ является взаимно однозначным отображением множества X на множество Y. В таком случае $[n]_n = n(n-1) \cdot \ldots \cdot 1$ обозначаем $n!$ (n факториал). Каждое взаимно однозначное отображение $f : X \to X$ называется *перестановкой* множества X. Как частный случай теоремы 1.2 получаем следующую теорему.

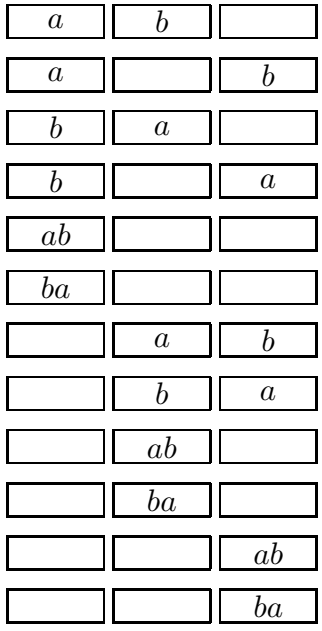

Рис. 1.2: Размещение (упорядоченное) элементов a,b в трех ящиках

Теорема 1.3. *Число перестановок n-элементного множества равно $n!$*

Перестановки мы будем обсуждать в последующих разделах, сейчас же остановимся еще на одном типе размещения объектов по ящикам. Предположим, что мы размещаем n объектов по m ящикам так, чтобы каждый ящик содержал бы последовательность, а не множество, как прежде, помещенных в нем объектов. Два размещения назовем равными, если в каждом ящике содержится одна и та же последовательность объектов. Размещения такого типа будем называть упорядоченными размещениями n объектов по m ящикам. Обозначим число таких упорядочений через $[m]^n$.

Теорема 1.4. *Число упорядоченных размещений n объектов по m ящикам равно*

$$[m]^n = m(m+1)\ldots(m+n-1) \qquad (1.2)$$

(полагаем $[m]^0 = 1$).

Доказательство. Будем строить упорядоченное размещение, добавляя по очереди новые объекты. Первый объект мы можем разместить m способами, второй — $m+1$ способами, ибо его можно разместить в одном из $m-1$ пустых ящиков или в ящике, содержащем первый объект, перед ним или после него. В общем случае предположим, что уже размещено $i-1$ объектов, причем для $k = 1, 2, \ldots, m$ в k-м ящике находятся r_k объектов. Тогда

i-й объект можем добавить в k-й ящик $r_k + 1$ способами, что дает в сумме

$$(r_1 + 1) + \ldots + (r_m + 1) = (r_1 + \ldots + r_m) + m = m + i - 1$$

возможностей. Таким образом, всех упорядоченных размещений будет $m(m + 1) \ldots (m + n - 1)$. ■

На рис. 1.2 представлены $[3]^2 = 12$ упорядоченных размещений элементов a, b в трех ящиках.

Приведем в заключение следующие простые зависимости:

$$[m]_n = (m - n + l)[m]_{n-1}, \tag{1.3}$$

$$[m]_n = m!/n!, \tag{1.4}$$

$$[m]^n = [m + n - 1]_n. \tag{1.5}$$

1.3 Перестановки: разложение на циклы, знак перестановки

Напомним, что перестановкой[10] n-элементного множества X называется произвольная взаимно однозначная функция $f : X \to X$. Обычно перестановка определяется с помощью таблицы с двумя строками, из которых каждая содержит все элементы множества X, причем элемент $f(x)$ помещается под элементом x. Для примера рассмотрим такую перестановку f множества $\{a, b, c, d\}$, что

$$f(a) = d; f(b) = a; f(\mathrm{c}) = b; f(d) = b;$$

она записывается в виде

$$f = \begin{pmatrix} a & b & c & d \\ d & a & b & c \end{pmatrix}$$

Если порядок элементов в верхней строке фиксирован, то каждой перестановке однозначно соответствует последовательность, содержащаяся в нижней строке, например для перестановки f это есть $\langle d, a, b, c\rangle$. Поэтому будем называть иногда произвольную инъективную последовательность длины n с элементами из множества X перестановкой n-элементного множества X.

В наших исследованиях природа элементов множества X несущественна — примем для простоты $X = \{1, \ldots, n\}$. Обозначим множество всех перестановок этого множества через S_n. Произвольная перестановка $f \in S_n$ будет обычно

[10] Обычно функция $f : X \to X$ называется подстановкой, а перестановкой называется вторая строка таблицы, определяющей подстановку. В этом разделе и далее термин «перестановка» используется для обозначения обоих понятий, что, однако, не приводит к какой-либо двусмысленности. — *Прим. перев.*

отождествляться с последовательностью $\langle a_1, \ldots, a_n \rangle$, где $a_i = f(i)$. Под суперпозицией перестановок f и g мы будем понимать перестановку fg, определяемую следующим образом:

$$fg(i) = f(g(i)).$$

Отметим, что для суперпозиции двух перестановок, скажем

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 3 & 2 & 1 & 4 \end{pmatrix} \quad g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 5 & 3 & 1 & 4 \end{pmatrix},$$

достаточно изменить порядок столбцов в перестановке f таким образом, чтобы в первой строке получить последовательность, имеющуюся во второй строке перестановки g, тогда вторая строка перестановки f дает суперпозицию fg. В нашем случае

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 3 & 2 & 1 & 4 \end{pmatrix}$$

$$g = \begin{pmatrix} 5 & 3 & 2 & 1 & 4 \\ 3 & 4 & 2 & 5 & 1 \end{pmatrix},$$

$$fg = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 5 & 1 \end{pmatrix}$$

Перестановку

$$e = \begin{pmatrix} 1 & 2 & \ldots & n \\ 1 & 2 & \ldots & n \end{pmatrix}$$

будем называть *тождественной перестановкой*. Очевидно, что $ef = fe = f$ для произвольной перестановки $f \in S_n$. Легко также заметить, что каждая перестановка $f \in S_n$ однозначно определяет перестановку f^{-1}, такую что $ff^{-1} = f^{-1}f = e$. Будем называть ее *перестановкой, обратной к* f. Чтобы ее определить, достаточно поменять местами строки в записи перестановки f. Например, для

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 1 & 5 \end{pmatrix}$$

получаем

$$f^{-1} = \begin{pmatrix} 3 & 4 & 2 & 1 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 1 & 2 & 5 \end{pmatrix}$$

Из наших рассуждений следует, что для произвольных перестановок $f, g, h \in S_n$ выполняются условия

$$(fg)h = f(gh) \tag{1.6}$$

$$fe = ef = f \tag{1.7}$$