Министерство науки и высшего образования Российской Федерации

Томский государственный университет
систем управления и радиоэлектроники

В.В. Кручинин

ЗАДАЧНИК ПО ИНФОРМАТИКЕ

Томск
2022

УДК 378.16:004
ББК 32.97я72
К–84

Рецензент:
Шульц Д.С., доцент кафедры технологий электронного обучения ТУСУР, канд. физ.-мат.наук

Кручинин, Владимир Викторович
К-84 Задачник по информатике. — Томск: Томский государственный университет систем управления и радиоэлектроники, 2022. — 102 с.

Предлагается сборник задач и заданий на разработку алгоритмов по курсам, связанным с информатикой, информационными технологиями и системами программирования. Представлено свыше 400 заданий и задач, сложность которых варьируется от простейших до очень сложных, относящихся к классу олимпиадных по информатике. Часть заданий сформулированы как шаблоны, из которых можно получить множество конкретных заданий. Предназначен для преподавателей, студентов и программистов, заинтересованных в повышении уровня компетенций по разработке алгоритмов.

Одобрено на заседании каф. ТЭО протокол № __8__от_29.04.2022_

УДК 378.16:004
ББК 32.97я72

Оглавление

Введение

Информатика – наука о методах сбора, хранения, преобразования, передачи, анализа и оценки информации с применением компьютеров. Важнейшими понятиями информатики являются алгоритмы и структуры данных. Получение знаний, умений и навыков по разработке алгоритмов является одной из основных составляющих подготовки студентов по дисциплинам, связанным с информатикой, такие, как основы информатики и программирования, информационные технологии и языки программирования и др.

Для закрепления теоретических знаний в таких курсах используются практические занятия, лабораторные и курсовые работы, на которых решаются задачи на построение алгоритма и его реализации на некотором языке программирования. Часто дисциплины по основам информатики ведутся на младших курсах бакалавриата, когда специальных знаний по направлению подготовки не достаточно, поэтому набор заданий на программирование является ограниченным, и в основном, связан с математическими задачами. С другой стороны, индивидуальный подход к обучению студентов требует наличие как простых заданий, так и заданий повышенной сложности. Поэтому в предлагаемом задачнике собраны задачи, которые требуют небольшой объем знаний в области комбинаторики, но имеют сложность от простейших до очень сложных, относящихся к классу олимпиадных по информатике. Особенностью данного задачника является его нацеленность на построение рекурсивных алгоритмов. Задания, представленные в этом задачнике, могут быть расширены путем добавления условий на исследование полученных алгоритмов на предмет вычислительной сложности, как в теоретическом, так и в экспериментальном плане.

Предлагаемый задачник основан на опыте преподавания для студентов ТУСУР по информатике, программированию, организации НИРС и ГПО, а также опыте привлечения студентов к проектам лаборатории инструментальных систем моделирования и обучения (ЛИСМО ТУСУР).

Данный задачник может быть использован как преподавателями информатики при разработке методического обеспечения или проведении соответствующих занятий, так и студентами самостоятельно в процессе изучения соответствующих дисциплин. Кроме того, данный задачник может служить источником заданий для тестирования программистов при приеме на работу, а также для всех желающих повысить компетенции в разработке алгоритмов повышенной сложности.

1 Задачи для систем счисления

Раздел курса информатики «Системы счисления» является одной из важных изучаемых тем. Математические основы хорошо и доступно описаны в работе [1]. Рассмотрим задачи, связанные с этой темой.

1.1 Двоичная система счисления

1. Задача о числе конвертов для суммы выдачи. Вычислить сколько надо конвертов заготовить, чтобы выдать любую сумму денег от 1 до 1000000000 руб.
2. Задача о награде магараджи за шахматы. Подсчитать вес зерна или объем для данной задачи, при условии размера доски [n×n]. Вес зерна представлен в таблице 1.1, вес 1 литра зерна – в таблице 1.2.

Таблица 1.1. Вес зерна

Зерно	Абсолютный вес зерна
рожь озимая	32 *мг*
пшеница озимая	40 *мг*
пшеница яровая	35 *м*
гречиха	20 *мг*
овес	24 *м*

Таблица 1.2. Вес 1 литра зерна

1 литр	Вес
пшеница	770 г
ржи	670 г
ячменя	540 г
овса	460 г

Для решения задачи необходимо воспользоваться [1].
Задачу можно усложнить, задав тип зерна для клеток шахматной доски.

1.2 Подсчет числа дней между датами.

Заданы календари:
1. Календарь Нумы Помпилия

март (31 день),
апрель (29 дней),
май (31 день),
июнь (29 дней),
квинтилис (31 день),
секстилис (29 дней),

сентябрь (29 дней),
октябрь (31 день),
ноябрь (29 дней),
декабрь (29 дней),
январь (29 дней),
февраль (28 дней)

2. Юлианский календарь

Високосный год кратен четырем

1	Январь	31
2	Февраль	28 (в високосном году — 29)
3	Март	31
4	Апрель	30
5	Май	31
6	Июнь	30
7	Июль	31
8	Август	31
9	Сентябрь	30
10	Октябрь	31
11	Ноябрь	30
12	Декабрь	31

3. Григорианский календарь

Високосный	год, номер которого кратен 400
Невисокосный	другие годы, номер которых кратен 100
Високосный	другие годы, номер которых кратен 4
Невисокосный	остальные годы

Задание. Дана исходная дата [*число*, *месяц*, *год*] и календарь (один из выше предложенных и [https://ru.wikipedia.org/wiki/календарь]).

1) подсчитать число дней до другой даты [$число_1$, $месяц_1$, $год_1$];
2) найти дату, разность между данной и исходной, равной N дней.

2 Вычисление специальных чисел

Задание. Записать рекуррентный алгоритм для вычисления специальных чисел, представленных ниже.

1. Вычисление факториала

$$n! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot \cdots . n$$

$$n! = \begin{cases} 1, \ \ n = 0, \\ n \cdot f(n-1), \ \ n > 0. \end{cases}$$

2. Вычисление биномиальных коэффициентов

$$C_n^m = \frac{n!}{m!\,(n-m)!}$$

$$C_n^m = \begin{cases} 0, \ \ n < m \\ 1, \ \ \ \ n = m \\ C_{n-1}^{m-1} + C_{n-1}^m \end{cases}$$

3. Вычисление чисел Бернулли

$$B_n = \begin{cases} 1, \ \ n = 0, \\ -\frac{1}{n+1} \sum_{k=1}^{n} C_{n+1}^{k+1} B_{n-k}\,, \ \ n > 0. \end{cases}$$

4. Вычисление чисел Эйлера

$$E_n = \begin{cases} 1, & n = 0, \\ 1 - \frac{1}{2} \sum_{k=1}^{n} C_n^k E_{n-k}\,, & n > 0. \end{cases}$$

5. Числа Фибоначчи

$$F_n = \begin{cases} 1, \ \ \ \ n = 1 \\ 1, \ \ \ \ n = 2 \\ F_{n-1} + F_{n-2} \end{cases}$$

6. Числа Люка

$$L_n = \begin{cases} 2, \ \ \ \ n = 0 \\ 1, \ \ \ \ n = 1 \\ L_{n-1} + L_{n-2} \end{cases}$$

7. Числа Перрена

$$P_n = \begin{cases} 3, \ \ \ \ n = 0 \\ 0, \ \ \ \ n = 1 \\ 2, \ \ \ \ n = 2 \\ P_{n-2} + P_{n-3}\,, n > 2 \end{cases}$$

8. Числа Подована

$$P_n = \begin{cases} 1, \ \ \ \ n = 0 \\ 1, \ \ \ \ n = 1 \\ 1, \ \ \ \ n = 2 \\ P_{n-2} + P_{n-3}\,, n > 2 \end{cases}$$

9. Числа Трибоначчи

$$T_n = \begin{cases} 0, & n = 0 \\ 0, & n = 1 \\ 1, & n = 2 \\ T_{n-1} + T_{n-2} + T_{n-3}, & n > 2 \end{cases}$$

10. Числа Каталана

$$C_n = \begin{cases} 1, & n = 0 \\ \sum_{i=0}^{n-1} C_i C_{n-i-1}, & n > 0 \end{cases}$$

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

11. Числа Риордана

$$R_n = \begin{cases} 1, & n = 0 \\ \sum_{i=0}^{n-1} R_i R_{n-i-1} + (-1)^n, & n > 0 \end{cases}$$

$$R_n = \begin{cases} 1, & n = 0 \\ 0, & n = 1 \\ \dfrac{(n-1)(2R_{n-1} + 3R_{n-2})}{(n+1)} \end{cases}$$

12. Числа Белла

$$B_n = \begin{cases} 1, & n = 0 \\ \sum_{i=0}^{n-1} B_i \binom{n-1}{i}, & n > 0 \end{cases}$$

13. Числа Стирлинга первого рода

$$s(n,k) = \begin{cases} 0, \quad n < k, & k < 0, \quad n < 0 \\ 1, & n = k \\ -(n-1) \cdot s(n-1,k) + s(n-1,k-1) \end{cases}$$

14. Числа Стирлинга второго рода

$$S(n,k) = \begin{cases} 0, \quad n < k, & k < 0, \quad n < 0 \\ 1, & n = k \\ kS(n-1,k) + S(n-1,k-1) \end{cases}$$

15. Числа Моцкина

$$M_n = \begin{cases} 1, & n = 0 \\ 1, & n = 1 \\ \dfrac{3(n-1)M_{n-2} + (2n+1)M_{n-1}}{(n+2)} \end{cases}$$

16. Числа Шредера

$$Sh_n = \begin{cases} 1, & n = 0 \\ Sh_{n-1} + \sum_{i=0}^{n-1} Sh_i Sh_{n-i-1} \end{cases}$$

17. Числа Галуа

$$G_n^q = \begin{cases} 1, & n = 0 \\ 2, & n = 1 \\ 2G_{n-1}^q + (q^{n-1} - 1)2G_{n-2}^q \end{cases}$$

18. Числа Деланноя

$$D_{n,k} = \begin{cases} 1, & n = 0 \; or \; k = 0 \\ D_{n,k-1} + D_{n-1,k-1} + D_{n-1,k} \end{cases}$$

Для усложнения задания можно дополнить его требованием быстрого вычисления, путем использования специального массива для хранения предыдущих значений специального числа.

3 Числовые треугольники

Числовые треугольники являются важными математическими объектами, имеющими множество разнообразных применений. Можно выделить такие треугольники: Паскаля, Эйлера, Стирлинга, Каталана, Моцкина, Нараянны и др.

Задание. Записать рекуррентный алгоритм для вычисления специальных чисел, представленных ниже.

1. Треугольник Паскаля

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k} + T_{n-1,k-1}, & n > k. \end{cases}$$

2. Треугольник Каталана 1

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + T_{n-1,k+1}, & n > k. \end{cases}$$

3. Треугольник Каталана 2

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + 2T_{n-1,k} + T_{n-1,k+1}, & n > k. \end{cases}$$

4. Треугольник Фибоначчи

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + T_{n-1,k} + T_{n-2,k}, & n > k. \end{cases}$$

5. Треугольник Трибоначчи

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + T_{n-1,k} + T_{n-2,k} + T_{n-3,k}, & n > k. \end{cases}$$

6. Треугольник Моцкина

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + T_{n-1,k} + T_{n-1,k+1}, & n > k. \end{cases}$$

7. Треугольник для больших чисел Шредера

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + T_{n-1,k} + T_{n,k+1}, & n > k. \end{cases}$$

8. Треугольник для малых чисел Шредера

$$T_{n,k} = \begin{cases} 0, & n < k \quad, n < 0, k < 0, \\ 1, & n = k\,, \\ T_{n-1,k-1} - T_{n-1,k} + 2T_{n,k+1}, & n > k. \end{cases}$$

9. Треугольник для чисел Стирлинга первого рода

$$s(n,k) = \begin{cases} 0, \quad n < k, \qquad k < 0, \qquad n < 0 \\ 1, \qquad\qquad\qquad\qquad\qquad n = k \\ -n \cdot s(n-1,k) + s(n-1,k-1) \end{cases}$$

10. Треугольник для чисел Стирлинга второго рода

$$S(n,k) = \begin{cases} 0, \quad n < k, \qquad k < 0, \qquad n < 0 \\ 1, \qquad\qquad\qquad\qquad\qquad n = k \\ k \cdot S(n-1,k) + S(n-1,k-1) \end{cases}$$

11. Треугольник для чисел Эйлера первого рода

$$E(n,k) = \begin{cases} 0, \qquad n < k, \qquad k \le 0, \qquad n \le 0 \\ 1, \qquad\qquad\qquad\qquad n = 1, k = 1 \\ k \cdot E(n-1,k) + (n-k+1) \cdot E(n-1,k-1) \end{cases}$$

12. Треугольник для чисел Эйлера второго рода

$$E_2(n,k) = \begin{cases} 0, \qquad n < k, \qquad k < 0, \qquad n < 0, \\ 1, \qquad\qquad\qquad\qquad n = 1, k = 1, \\ k \cdot E_2(n-1,k) + (2n-k) \cdot E_2(n-1,k-1). \end{cases}$$

13. Треугольник Нараяны

$$N_{n,k} = \frac{1}{k+1}\binom{n}{k}\binom{n}{k+1}$$

$$N_{n,k} = \begin{cases} 1, & n = k \text{ или } k = 1, \\ \sum_{i=1}^{k-1} \binom{n+i-1}{2k-2} N_{k-1,i} & \end{cases}$$

14. Треугольник тангенса

$$T(n,k) = \begin{cases} 0, & n < k, \quad k < 0, \quad n < 0, \\ 1, & n = k, \\ k \cdot (k+1) \cdot T(n-1,k+1) + T(n-1,k-1) & \end{cases}$$

15. Треугольник центральных факториальных чисел

$$T(n,k) = \begin{cases} 0, & n < k, \quad k < 0, \quad n < 0, \\ 1, & k = 1, \\ k^2 \cdot T(n-1,k+1) + T(n-1,k-1) & \end{cases}$$

16. Треугольник синуса

$$T(n,k) = \begin{cases} 0, & n < k, \quad k \le 0, \quad n < 0, \\ 1, & n = k, \\ k^2 \cdot T(n-2,k) + T(n-2,k-2), & n > 0 \end{cases}$$

17. Треугольник арктангенса

$$T(n,k) = \begin{cases} 0, & n < k, \quad k \le 0, \quad n < 0, \\ 1, & n = k, \\ (n-1) \cdot (n-2) \cdot T(n-2,k) + T(n-1,k-1), & n > 0 \end{cases}$$

Для усложнения задания можно дополнить его требованием быстрого вычисления, путем использования специального массива для хранения предыдущих значений специального числа.

4 Сложные виды рекурсии

Задание. Записать алгоритмы подсчета рекурсивных функций.

1. Подсчет числа деревьев с четным и нечетным числом сыновей узлов.

$$O(n) = \begin{cases} 0, & n \leq 1 \\ 1, & n = 1, \\ \sum_{i=1}^{n-1} C(n-1)E(n-i+1) + C(n-1), & n > 1 \end{cases}$$

где C(n) – числа Каталана.

$$E(n) = \begin{cases} 0, & n \leq 1 \\ 1, & n = 1, \\ \sum_{i=1}^{n-1} C(n-1)O(n-i-1)\,, & n > 1 \end{cases}$$

2. Система рекурсивных формул. Задача мажордома

$$A(n) = \begin{cases} 0, & n < 2 \\ 2, & n = 2, \\ 2(2n-3) \cdot \big((n-1) \cdot A(n-1) + B(n-1)\big) + 2C(n-1), & n > 2 \end{cases}$$

$$B(n) = \begin{cases} 0, & n \leq 2, \\ 2n \cdot (2(n-1) \cdot A(n-1) + B(n-1)), & n > 2 \end{cases}$$

$$C(n) = \begin{cases} 0, & n < 2 \\ 4, & n = 2, \\ 2n \cdot \big((n-1) \cdot A(n-1) + B(n-1)\big), & n > 2 \end{cases}$$

3. Дважды рекурсивные функции

$$s(n) = \begin{cases} 0, & n \leq 0 \\ s\big(s(n-1) - s(n-2)\big) + 1, & n > 0 \end{cases}$$

4. Функция Аккермана

$$A(m,n) = \begin{cases} n+1, & m = 0 \\ A(m-1, n), & n = 0, \\ A(m-1, A(m, n-1)), & n > 2 \end{cases}$$

Задание: записать алгоритм для схемы Горнера, представленной ниже.

Пусть задан многочлен

$$P(x) = \sum_{i=0}^{n} a_i x^i$$

Его можно представить

$$P(x) = (a_0 + x(a_1 + x(a_2 + \cdots x(a_n) \ \cdots \ )))$$

Тогда, можно предложить следующую схему вычислений многочлена $P(x)$ при заданном значении $x = x_0$:

1. $s = 0$
2. $s = a_n + x_0 s$
3. $s = a_{n-1} + x_0 s$
4. $s = a_{n-2} + x_0 s$
5. …
6. $s = a_0 + x_0 s$

Если $x, a_0, a_1, \dots, a_n$ принимают целые положительные значения тогда можно предложить обратную схему при которой можно найти $a_0, a_1, \dots, a_n$ зная значения x и s. Для этого нужно задать операцию получения остатка от деления на целое. Операция определения остатка от деления будем записывать

$$r = mod(n, m),$$

где r-остаток, n-частное, m- делитель.

1. $a_0 = mod(s, x_0)$
2. $s = s/x_0$
3. $a_1 = mod(s, x_0)$
4. $s = s/x_0$
5. …
6. $a_n = mod(s, x_0)$

5 Вычисление констант

5.1 Методы и алгоритмы вычисления числа π

Методы и алгоритмы вычисления числа π имеют тысячелетнюю историю. Одним из самых подробных описанием этой истории является статья американского профессора индийского происхождения Рави Агарвала в соавторстве [9], в которой описано свыше 200 методов и алгоритмов вычисления числа π . Ниже представлено несколько методов вычисления числа π:

1) $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} + \cdots$

2) $\frac{\pi}{4} = 1 - 2\left(\frac{1}{3\cdot.5} + \frac{1}{7\cdot 9} + \frac{1}{9\cdot 11} + \frac{1}{11\cdot 13} + \cdots\right)$

3) $\pi = 3 + \left(\frac{4}{2\cdot 3\cdot.5} - \frac{4}{4\cdot 5\cdot 6} + \frac{4}{6\cdot 7\cdot 8} - \frac{4}{8\cdot 9\cdot 10} + \cdots\right)$

4) $\frac{2}{\pi} = \frac{\sqrt{2}}{2}\frac{\sqrt{(2+\sqrt{2})}}{2}\frac{\sqrt{(2+\sqrt{(2+\sqrt{2})}}}{2}\cdots$

5) $\pi = 2\cdot\frac{2\cdot 2}{1\cdot 3}\cdot\frac{4\cdot 4}{3\cdot 5}\cdot\frac{6\cdot 6}{5\cdot 7}\cdot\frac{8\cdot 8}{7\cdot 9}\cdots$

6) $\pi = \cfrac{4}{1+\cfrac{1}{3+\cfrac{2^2}{5+\cfrac{3^2}{7+\cfrac{4^2}{9+\cdots}}}}}$

7) $\pi = 3 + \cfrac{1}{6+\cfrac{3^2}{6+\cfrac{5^2}{6+\cfrac{7^2}{6+\cfrac{9^2}{6+\cdots}}}}}$

8) $\frac{\pi^2}{6} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$

9) $\frac{\pi^2}{8} = 1 + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \cdots$

10) $\frac{\pi^2}{12} = 1 - \frac{1}{2^2} + \frac{1}{3^2} - \frac{1}{4^2} + \frac{1}{5^2} - \cdots$

11) $\frac{\pi^2}{6} = 3\sum_{n=1}^{\infty}\frac{1}{n^2 C_{2n}^n}$, где C_{2n}^n- биномиальный коэффициент

12) $\pi^2 = 9\left(1 + \frac{1^2}{3\cdot.5} + \frac{1^2\cdot 2^2}{3\cdot 4\cdot 5\cdot 6} + \frac{1^2\cdot 2^2\cdot 3^2}{3\cdot 4\cdot 5\cdot 6\cdot 7\cdot 8} + \cdots\right)$

13) $\frac{\pi}{2} = \left(1 + \frac{1!}{3} + \frac{2!}{3\cdot 5} + \frac{3!}{3\cdot 5\cdot 7} + \frac{4!}{3\cdot 5\cdot 7\cdot 9} + \cdots\right)$

14) $\pi = 2\left(1 + \frac{1^2}{3!} + \frac{1^2 \cdot 3^2}{5!} + \frac{1^2 \cdot 3^2 \cdot 5^2}{7!} + \frac{1^2 \cdot 3^2 \cdot 5^2 \cdot 7^2}{9!} + \cdots\right)$

15) $\frac{\pi}{4} = \left(\frac{1}{2} + \frac{1}{3}\right) - \frac{1}{3}\left(\frac{1}{2^3} + \frac{1}{3^3}\right) + \frac{1}{5}\left(\frac{1}{2^5} + \frac{1}{3^5}\right) - \frac{1}{7}\left(\frac{1}{2^7} + \frac{1}{3^7}\right) + \cdots$

16) $\frac{\pi^2}{16} = \sum_{n=0}^{\infty} \frac{(-1)^n}{n+1}\left(1 + \frac{1}{3} + \frac{1}{5} + \cdots + \frac{1}{2n+1}\right)$

17) $\frac{\pi^4}{90} = \frac{36}{17}\sum_{n=1}^{\infty} \frac{1}{n^4 C_{2n}^n}$,

где C_{2n}^n- биномиальный коэффициент

18) Алгоритм быстрого вычисления π

$a_0 = 1$, $b_0 = \frac{1}{\sqrt{2}}$, $s_0 = \frac{1}{2}$,

$$a_k = \frac{a_{k-1} + b_{k-1}}{2}$$

$$b_k = \sqrt{a_{k-1} \cdot b_{k-1}}$$

$$c_k = {a_k}^2 - {b_k}^2$$

$$s_k = s_{k-1} - 2^k \cdot c_k,$$

$$p_k = \frac{2{a_k}^2}{s_k}$$

19) Алгоритм быстрого вычисления π

$x_0 = \sqrt{2}$, $y_0 = 0$, $\alpha_0 = 2 + \sqrt{2}$,

$$x_{k+1} = \frac{1}{2}\left(\sqrt{x_k} + \frac{1}{\sqrt{x_k}}\right)$$

$$y_{k+1} = \sqrt{x_k}\left(\frac{1 + y_k}{x_k + y_k}\right)$$

$$\alpha_{k+1} = \alpha_k y_{k+1}\left(\frac{1 + x_{k+1}}{1 + y_{k+1}}\right)$$

20) Быстрое вычисление $1/\pi$

$a_0 = 6 - 4\sqrt{2}$, $y_0 = \sqrt{2} - 1$,

$$y_{k+1} = \frac{1 - (1 - {y_k}^4)^{\frac{1}{4}}}{1 + (1 - {y_k}^4)^{\frac{1}{4}}}$$

$$a_{k+1} = a_k(1 + {y_{k+1}}^4) - 2^{2k+3}(1 + y_{k+1} + {y_{k+1}}^2)$$

Задание. Выбрать метод из перечисленных выше и вычислить n-первых значений π.

Задания повышенной сложности:

1. Разработать алгоритм генерации изображений на основании цифр числа π.
2. Разработать алгоритм генерации звуков (музыки) на основании цифр числа π.
3. Разработать алгоритм кодирования.

5.2 Алгоритмы и методы вычисления числа е

1) $e = \sum_{n=0}^{\infty} \frac{1}{n!}$

2) $e = \sum_{n=0}^{\infty} \frac{2n+1}{(2n)!}$

3) $e = \sum_{n=0}^{\infty} \frac{2n+2}{(2n+1)!}$

4) $e = \sum_{n=0}^{\infty} \frac{3-4n^2}{(2n+1)!}$

5) $e = \sum_{n=0}^{\infty} \frac{1+3n^2}{(3n)!}$

6) $e = \sum_{n=0}^{\infty} \frac{1+(3n+1)^2}{(3n+1)!}$

7) $\frac{1}{e} = \sum_{n=0}^{\infty} \frac{2n+1}{(2n)!}$

8) $\sqrt{e} = \sum_{n=0}^{\infty} \frac{4n+3}{2^{2n+1}(2n+1)!}$

9) $e = \frac{2}{1}\left(\frac{4}{3}\right)^{\frac{1}{2}}\left(\frac{6\cdot 8}{5\cdot 7}\right)^{\frac{1}{4}}\left(\frac{10\cdot 12\cdot 14\cdot 16}{9\cdot 11\cdot 13\cdot 15}\right)^{\frac{1}{8}}\cdots$

10) $$e = 2\left(\frac{2}{1}\right)^{\frac{1}{2}}\left(\frac{2\cdot 4}{3\cdot 3}\right)^{\frac{1}{4}}\left(\frac{4\cdot 6\cdot 6\cdot 8}{5\cdot 5\cdot 7\cdot 7}\right)^{\frac{1}{8}}\cdots$$

11) $$\frac{e-1}{2} = \cfrac{1}{1+\cfrac{1}{6+\cfrac{1}{10+\cfrac{1}{14+\cfrac{1}{18+\cdots}}}}}$$

где в знаменателях записана последовательность [6, 10, 14, 18, …, 4n-2, …].

Задание.

1. Выбрать метод и вычислить n-первых значений e.
2. Посчитать число различных цифр n-первых значений e.

Задания повышенной сложности.

1. Разработать алгоритм генерации изображений на основании цифр числа e.
2. Разработать алгоритм генерации звуков (музыки) на основании цифр числа e.
3. Разработать алгоритм кодирования.

5.3 Вычисление постоянной Гельфонда

Задание. Выбрать метод и вычислить n-первых значений постоянной Гельфонда:

$$e^{\pi} = 23.140692632779269005\ldots$$

Пояснения: Алгоритм приближенного вычисления

1. $k_0 = \frac{1}{\sqrt{2}}$
2. $k_n = \frac{1-\sqrt{1-{k_{n-1}}^2}}{1+\sqrt{1-{k_{n-1}}^2}}$
3. $e^{\pi} \approx k_n$

5.4 Алгоритмы вычисления золотого сечения

Задание. Выбрать метод из ниже приведённых и вычислить n-первых значений золотого сечения.

1) $\frac{1+\sqrt{5}}{2} = \frac{1}{1+\frac{1}{1+\frac{1}{1+\frac{1}{1+\frac{1}{1+\cdots}}}}}$

2) $\frac{1+\sqrt{5}}{2} = \sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\cdots}}}}}$

3) $\frac{1+\sqrt{5}}{2} = \frac{13}{8} + \sum_{n=0}^{\infty} \frac{(-1)^{n+1}(2n+1)!}{4^{2n+3} n! (n+1)!}$

5.5 Методы и алгоритмы вычисления функций

Вычисление функций основано на разложении функции в ряд Тейлора.

Перечислим ряды для основных элементарных функций:

1) $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$

2) $\ln x = \sum_{n=1}^{\infty} \frac{x^n}{n}$

3) $\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)!}$

4) $\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!}$

5) $\operatorname{tg} x = \frac{x}{1-\frac{x^2}{3-\frac{x^2}{5-\frac{x^2}{7-\frac{x^2}{9-\cdots}}}}}$

6) $\operatorname{tg} x = \sum_{n=1}^{\infty} \frac{2^{2n}(2^{2n}-1)|B_{2n}| x^{2n-1}}{(2n)!}$

7) $\operatorname{ctg} x = \frac{1}{x} - \sum_{n=1}^{\infty} \frac{2^{2n}|B_{2n}|x^{2n-1}}{(2n)!}$

8) $\sec x = \sum_{n=0}^{\infty} \frac{|E_{2n}|x^{2n}}{(2n)!}$

9) $\operatorname{cosec} x = \sum_{n=1}^{\infty} \frac{2(2^{2n-1}-1)|B_{2n}|x^{2n-1}}{(2n)!}$

10) $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \sum_{n=1}^{\infty} \frac{(-1)^n x^{2n+1}}{n!(2n+1)}$

11) $\operatorname{Erfi}(x) = \sum_{n=1}^{\infty} \frac{x^{2n+1}}{n!(2n+1)}$

Задание. Записать алгоритм приближенного вычисления значения функции (из приведённых выше) на основе вычисления ограниченного ряда при заданном значении n.

6 Поиск

Поисковые задачи являются одной из самых распространенных задач программирования.

6.1 Поиск в массивах.

6.1.1 Поиск в одномерных массивах

Задание. Построить алгоритм для задачи, предложенной ниже.

1. Найти максимальный элемент в одномерном массиве, все элементы различны.
2. Найти минимальный элемент в одномерном массиве, все элементы различны.
3. Найти все максимальные элементы в одномерном массиве, элементы могут быть одинаковы.
4. Найти все минимальные элементы в одномерном массиве, элементы могут быть одинаковы.
5. Найти сумму различных элементов в одномерном массиве, часть элементов массива одинаковы. Например,

   [1,3,7,3,4,7,9,4,2] сумма различных элементов 1+3+7+4+2+9.
6. Подсчитать число одинаковых элементов в одномерном массиве.
7. Найти сумму элементов, имеющих копию в одномерном массиве, часть элементов массива одинаковы. Например,

   [1,3,7,3,4,7,9,4,2] сумма имеющих копию элементов 3+7+4.
8. Подсчитать число условий a_i>a_i+1 в одномерном массиве.
9. Подсчитать число условий a_i<a_i+1 в одномерном массиве.
10. Подсчитать число условий a_i<a_i+1<a_i+2 в одномерном массиве.
11. Подсчитать число условий a_i>a_i+1>a_i+2 в одномерном массиве.
12. Проверить условие, что числа a_i не являются числами Фибоначчи.
13. Проверить условие, что числа a_i не являются числами Каталана.
14. Проверить условие, что числа a_i не являются числами Моцкина.
15. Проверить условие, что числа a_i не являются числами Люка.
16. Проверить условие, что числа a_i не являются малыми числами Шредера.

17. Проверить условие, что числа a_i не являются большими числами Шредера.
18. Найти максимальную суммы двух элементов одномерного массива.
19. Найти минимальную суммы двух элементов одномерного массива.
20. Найти максимальную суммы трех элементов одномерного массива.
21. Найти минимальную суммы трех элементов одномерного массива.
22. Найти два элемента в одномерном массиве, сумма которых равна некоторому числу s.
23. Найти три элемента в одномерном массиве, сумма которых равна некоторому числу s.
24. Найти два элемента в одномерном массиве, сумма квадратов которых равна некоторому числу s.
25. Найти три элемента в одномерном массиве, сумма квадратов которых равна некоторому числу s.
26. Найти максимальный элемент среди четных элементов в одномерном массиве, все элементы различны.
27. Найти минимальный элемент среди четных элементов в одномерном массиве, все элементы различны.
28. Найти все максимальные элементы среди четных элементов в одномерном массиве, элементы могут быть одинаковы.
29. Найти все минимальные элементы среди четных элементов в одномерном массиве, элементы могут быть одинаковы.
30. Найти сумму различных элементов среди четных элементов в одномерном массиве, часть элементов массива одинаковы.
31. Подсчитать число одинаковых элементов среди четных элементов в одномерном массиве.
32. Найти сумму элементов, имеющих копию среди четных элементов в одномерном массиве, часть элементов массива одинаковы.
33. Проверить условие, что числа a_i не являются числами Фибоначчи среди четных элементов одномерного массива.
34. Проверить условие, что числа a_i не являются числами Каталана среди четных элементов одномерного массива среди четных элементов одномерного массива
35. Проверить условие, что числа a_i не являются числами Моцкина среди четных элементов одномерного массива.

36. Проверить условие, что числа a_i не являются числами Люка среди четных элементов одномерного массива.

37. Проверить условие, что числа a_i не являются малыми числами Шредера среди четных элементов одномерного массива.

38. Проверить условие, что числа a_i не являются большими числами Шредера среди четных элементов одномерного массива.

39. Найти максимальную суммы двух элементов среди четных элементов одномерного массива.

40. Найти минимальную суммы двух элементов среди четных элементов одномерного массива.

41. Найти максимальную суммы трех элементов среди четных элементов одномерного массива.

42. Найти минимальную суммы трех элементов среди четных элементов одномерного массива.

43. Найти два элемента среди четных элементов одномерного массива, сумма которых равна некоторому числу s.

44. Найти три элемента среди четных элементов одномерного массива, сумма которых равна некоторому числу s.

45. Найти два элемента среди четных элементов одномерного массива, сумма квадратов которых равна некоторому числу s.

46. Найти три элемента среди четных элементов одномерного массива, сумма квадратов которых равна некоторому числу s.

47. Найти минимальный элемент среди нечетных элементов в одномерном массиве, все элементы различны.

48. Найти все максимальные элементы среди нечетных элементов в одномерном массиве, элементы могут быть одинаковы.

49. Найти все минимальные элементы среди нечетных элементов в одномерном массиве, элементы могут быть одинаковы.

50. Найти сумму различных элементов среди нечетных элементов в одномерном массиве, часть элементов массива одинаковы.

51. Подсчитать число одинаковых элементов среди нечетных элементов в одномерном массиве.

52. Найти сумму элементов, имеющих копию среди нечетных элементов в одномерном массиве, часть элементов массива одинаковы.

53. Проверить условие, что числа a_i не являются числами Фибоначчи среди нечетных элементов одномерного массива.
54. Проверить условие, что числа a_i не являются числами Каталана среди нечетных элементов одномерного массива среди нечетных элементов одномерного массива.
55. Проверить условие, что числа a_i не являются числами Моцкина среди нечетных элементов одномерного массива.
56. Проверить условие, что числа a_i не являются числами Люка среди нечетных элементов одномерного массива.
57. Проверить условие, что числа a_i не являются малыми числами Шредера среди нечетных элементов одномерного массива.
58. Проверить условие, что числа a_i не являются большими числами Шредера среди нечетных элементов одномерного массива.
59. Найти максимальную суммы двух элементов среди нечетных элементов одномерного массива.
60. Найти минимальную суммы двух элементов среди четных элементов одномерного массива.
61. Найти максимальную суммы трех элементов среди четных элементов одномерного массива.
62. Найти минимальную суммы трех элементов среди четных элементов одномерного массива.
63. Найти два элемента среди четных элементов одномерного массива, сумма которых равна некоторому числу s.
64. Найти три элемента среди четных элементов одномерного массива, сумма которых равна некоторому числу s.
65. Найти два элемента среди четных элементов одномерного массива, сумма квадратов которых равна некоторому числу s.
66. Найти три элемента среди четных элементов одномерного массива, сумма квадратов которых равна некоторому числу s.
67. Найти все суммы из элементов одномерного массива, равные некоторому заданному числу s.

6.1.2Поиск в двухмерных массивах

Задание. Построить алгоритм для задачи, предложенной ниже.

1. Найти максимальное значение элемента матрицы и минимальное значение элемента матрицы.
2. Найти максимальное значение элемента среди нечетных строк матрицы.
3. Найти максимальное значение элемента среди нечетных столбцов матрицы.
4. Найти максимальное значение элемента среди четных строк матрицы.
5. Найти максимальное значение элемента среди четных столбцов матрицы.
6. Найти минимальное значение элемента среди нечетных строк матрицы.
7. Найти минимальное значение элемента среди нечетных столбцов матрицы.
8. Найти минимальное значение элемента среди четных строк матрицы.
9. Найти минимальное значение элемента среди четных столбцов матрицы.
10. Найти максимальное значение произведения двух элементов среди нечетных строк матрицы.
11. Найти максимальное значение произведения двух элементов среди нечетных столбцов матрицы.
12. Найти максимальное значение произведения двух элементов среди четных строк матрицы.
13. Найти максимальное значение произведения двух элементов среди четных столбцов матрицы.
14. Найти минимальное значение произведения двух элементов среди нечетных строк матрицы.
15. Найти минимальное значение произведения двух элементов среди нечетных столбцов матрицы.
16. Найти минимальное значение произведения двух элементов среди четных строк матрицы.
17. Найти минимальное значение произведения двух элементов среди четных столбцов матрицы.
18. Найти максимальное значение произведения двух элементов среди нечетных строк матрицы.
19. Найти максимальное значение суммы двух элементов среди нечетных столбцов матрицы.
20. Найти максимальное значение суммы двух элементов среди четных строк матрицы.
21. Найти максимальное значение суммы двух элементов среди четных столбцов матрицы.
22. Найти минимальное значение суммы двух элементов среди нечетных строк матрицы.
23. Найти минимальное значение суммы двух элементов среди нечетных столбцов матрицы.
24. Найти минимальное значение суммы двух элементов среди четных строк матрицы.

25. Найти минимальное значение суммы двух элементов среди четных столбцов матрицы.
26. Найти два элемента матрицы, сумма которых равна заданному числу s.
27. Найти все пары элементов матрицы, сумма которых равна заданному числу s.
28. Найти три элемента матрицы, сумма которых равна заданному числу s.
29. Найти все тройки элементов матрицы, сумма которых равна заданному числу s.
30. Найти два элемента матрицы с четными номерами столбцов, сумма которых равна заданному числу s.
31. Найти все пары элементов матрицы с четными номерами столбцов, сумма которых равна заданному числу s.
32. Найти три элемента матрицы с четными номерами столбцов, сумма которых равна заданному числу s.
33. Найти все тройки элементов матрицы с четными номерами столбцов, сумма которых равна заданному числу s.
34. Найти два элемента матрицы с нечетными номерами столбцов, сумма которых равна заданному числу s.
35. Найти все пары элементов матрицы с нечетными номерами столбцов, сумма которых равна заданному числу s.
36. Найти три элемента матрицы с нечетными номерами столбцов, сумма которых равна заданному числу s.
37. Найти все тройки элементов матрицы с нечетными номерами столбцов, сумма которых равна заданному числу s.
38. Найти два элемента матрицы с нечетными номерами строк, сумма которых равна заданному числу s.
39. Найти все пары элементов матрицы с нечетными номерами строк, сумма которых равна заданному числу s.
40. Найти три элемента матрицы с нечетными номерами строк, сумма которых равна заданному числу s.
41. Найти все тройки элементов матрицы с нечетными номерами строк, сумма которых равна заданному числу s.
42. Найти два элемента матрицы с четными номерами строк, сумма которых равна заданному числу s.
43. Найти все пары элементов матрицы с четными номерами строк, сумма которых равна заданному числу s.
44. Найти три элемента матрицы с четными номерами строк, сумма которых равна заданному числу s.
45. Найти все тройки элементов матрицы с четными номерами строк, сумма которых равна заданному числу s.

6.2 Поиск в треугольных матрицах

Треугольная матрица имеет особый вид и выглядит следующим образом

$$\begin{bmatrix} T_{1,1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ T_{n,1} & \cdots & T_{n,n} \end{bmatrix}$$

В таких матрицах выполняется условие: все $T_{n,k}$ при $n < k$ в вычислениях не участвуют.

Задание. Построить алгоритм для задачи, предложенной ниже.

1. Найти минимальный элемент в треугольной матрице.
2. Найти максимальный элемент в треугольной матрице.
3. Найти сумму элементов треугольной матрицы.
4. Найти сумму четных элементов треугольной матрицы.
5. Найти сумме нечетных элементов треугольной матрицы.
6. Найти сумму элементов четных строк треугольной матрицы.
7. Найти сумму элементов нечетных строк треугольной матрицы.
8. Найти сумму элементов четных столбцов треугольной матрицы.
9. Найти сумму элементов нечетных столбцов треугольной матрицы.
10. Найти сумму диагональных элементов треугольной матрицы (диагональный элемент имеет T(2n-1,n)).

7 Шахматные задачи

Задание. Построить алгоритм задачи, записанной ниже - расстановка фигур на шахматной доске, при условии, что ни одна не атакует другую.

1. 2 ферзя и доска 2×n.
2. 3 ферзя и доска 3×n.
3. 4 ферзя и доска 4×n.
4. 5 ферзей и доска 5×n.
5. n ферзей и доска n×n
6. 2 короля и доска 2×n.
7. 3 короля и доска 3×n.
8. 4 короля и доска 4×n.
9. 5 королей и доска 5×n.
10. 2 ладьи и доска 2×n.
11. 3 ладьи и доска 3×n.
12. 4 ладьи и доска 4×n.
13. 5 ладьи и доска 5×n.
14. 2 слона и доска 2×n.
15. 3 слона и доска 3×n.
16. 4 слона и доска 4×n.
17. 5 слонов и доска 5×n.
18. 2 коня и доска 2×n.
19. 3 коня и доска 3×n.
20. 4 коня и доска 4×n.
21. 5 коней и доска 5×n.

Задание: Построить алгоритм задачи, записанной ниже на получение вариантов обхода конем шахматной доски.

1. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу построить 1 путь.
2. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу. Построить несколько путей.
3. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу построить 1 путь. Использовать рамочный метод Мунка и Коллини [10].
4. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу построить 1 путь. Использовать метод Полиньяка и Роже [10].
5. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу построить 1 путь. Использовать метод Эйлера и Вандермонда [10].

6. Обойти конем все поля шахматной доски 8×8, посетив каждое из них по одному разу построить 1 путь. Правило Варнсдорфа [10]. Указание. Правило Варнсдорфа заключается в следующем:

   1) при обходе доски коня следует всякий раз ставить на поле, из которого он может сделать наименьшее число ходов на еще не пройденные поля;

   2) если таких полей несколько, то разрешается выбирать любое из них.

Задания повышенной сложности.

1. Задача нахождения пути получения заданной позиции из начальной.
2. Задача нахождения всех путей обхода конем.
3. Задана позиция белого коня и черного короля, часть клеток для коня запрещены (мины), конь должен побить черного короля и вернуться в исходную позицию не используя клетки предыдущего пути.

8 Обработка строк символов.

Задание. Дана строка символов. Построить алгоритм задачи, записанной ниже.

1. Определить длину строки.
2. Определить число предложений в строке.
3. Определить число слов в строке.
4. Определить знаков препинания.
5. Подсчитать число гласных.
6. Подсчитать число согласных.
7. Найти слово с максимальным числом символов.
8. Найти слово с минимальным числом символов.
9. Найти слово с максимальным числом согласных.
10. Найти слово с минимальным числом согласных.
11. Найти слово с максимальным числом гласных.
12. Найти слово с минимальным числом гласных.
13. Получить текст с обратным порядком слов.
14. Получить список слов текста по возрастанию числа букв в слове.
15. Получить список слов текста по убыванию числа букв в слове.
16. Получить список слов в алфавитном порядке.

9 Транслятор.

Задание. Дана строка символов. Построить алгоритм задачи, записанной ниже.

1. Подсчитать число констант.
2. Подсчитать число идентификаторов.
3. Подсчитать число ключевых слов.
4. Пусть задано арифметическое выражение, состоящее из целых цифр, знаков операций суммы, произведения и скобок, вычислить его значение.
5. Пусть задано арифметическое выражение, состоящее из целых цифр, знаков операций сумы, произведения и скобок, построить польскую запись вычислений выражения.
6. Пусть задан некоторый многочлен в виде строки, записать алгоритм получения производной.
7. Пусть задан некоторый многочлен в виде строки, записать алгоритм получения неопределенного интеграла с константой равной 0.

10 Полиндром

Палиндром - это любой текст, который одинаково читается слева направо, и справа налево.

Задание. Построить алгоритм задачи, записанной ниже.

1. Проверка является ли слово палиндромом.
2. Подсчитать число палиндромов в строке.
3. Найти самый длинный палиндром.
4. Найти самый короткий палиндром

11 Комбинаторная генерация

Методы и алгоритмы комбинаторной генерации приобретают важное научное и практическое значение в следующих отраслях: в программировании, в проектировании информационных систем и баз данных, в теории кодирования и сжатия информации, в химии, при исследовании различных химических структур, в биологии при моделировании ДНК-структур. В общем случае, для получения процедур генерации и нумерации комбинаторного множества необходимо задать биективное отображение:

$$Generate: N_m \to A_m$$

где N_m - конечное подмножество натуральных чисел; A_m - комбинаторное множество. Тогда обратное отображение задает процедуру нумерации:

$$Rank:\ A_m \to N_m$$

На рис. 1 показаны основные элементы и схема взаимодействия алгоритмов Generate и Rank. Алгоритм Generate на основании числа $num \in N$ и некоторого описания D создает элемент $a \in A_m$. Алгоритм Rank производит обратное действие, на основании $a \in A_m$. и некоторого описания D, формирует номер $num \in N$

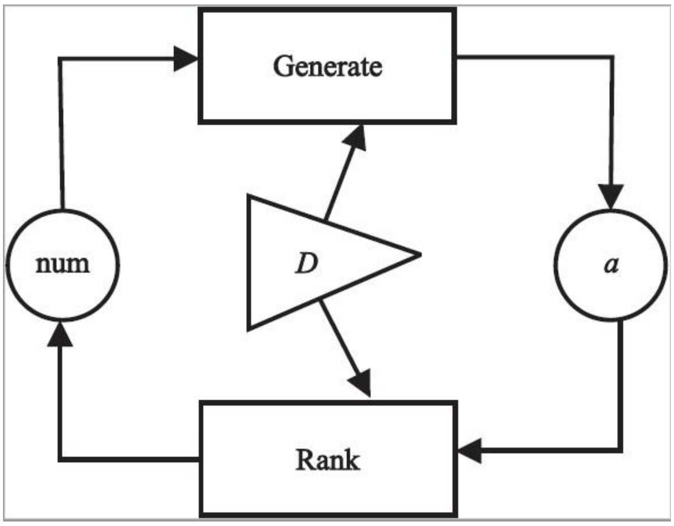


Рис. 1 Основная схема алгоритмов генерации и нумерации элементов комбинаторных множеств

Методы и алгоритмы комбинаторной генерации представлены в работах [8,16,25].

В разделе 11 приведены примеры разработки алгоритмов комбинаторной генерации разработанный для множеств строк символов с правильной скобочной записью, которые описываются числами Каталана.

11.1 Генерация перестановок

Основные формулы для вычисления факториала:

1. Формула по определению:

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$$

2. Рекуррентная формула 1:

$$f(n) = \begin{cases} 1, & n = 0 \\ n \cdot f(n-1) \end{cases}$$

3. Рекуррентная формула 2:

$$f(n) = \begin{cases} 1, & n = 0 \\ \sum_{k=0}^{n-1} \binom{n-1}{k} f(k)\, f(n-1-k) \end{cases}$$

4. Рекуррентная формула на основе чисел Стирлинга первого рода:

$$f(n) = \sum_{k=0}^{n} S_1\,(n, k)$$

5. Рекуррентная формула на основе чисел Эйлера:

$$f(n) = \sum_{k=0}^{n} E_1\,(n, k)$$

6. Формула на основе мультиноминальных коэффициентов:

$$n! = \binom{n}{\lambda_1, \lambda_2, \ldots, \lambda_k} \lambda_1!\, \lambda_2! \ldots \lambda_k!$$

$$\lambda_1 + \lambda_2 + \cdots + \lambda_k = n$$

7. Формула для мультиноминальных коэффициентов:

$$\binom{n}{\lambda_1, \lambda_2, \ldots, \lambda_k} = \binom{n}{\lambda_1} \binom{n-\lambda_1}{\lambda_2} \ldots \binom{\lambda_1}{\lambda_1}$$

8. Схема генерации перестановок (рис. 2)

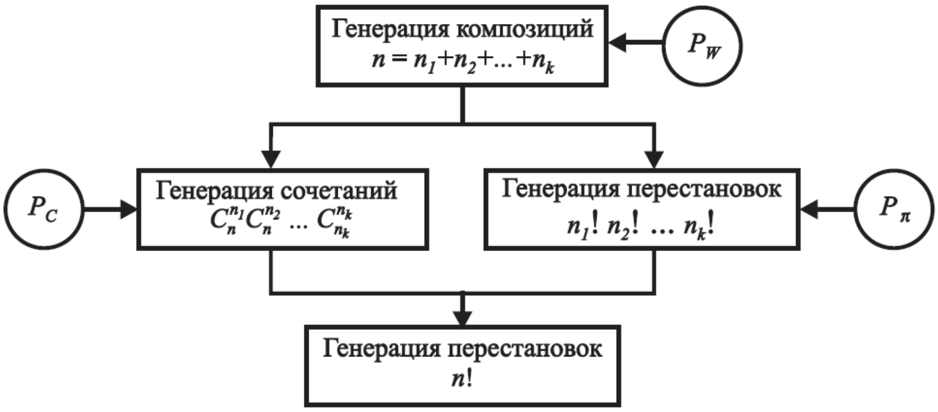


Рис. 2 Схема генерации перестановок на основе комбинации композиций, сочетаний и перестановок.

Задание. Построить алгоритм задачи, записанной ниже.

1. Генерация перестановок на основе инверсий.
2. Генерация перестановок на основе чисел Эйлера первого рода.
3. Генерация перестановок на основе чисел Стирлинга первого рода.
4. Генерация перестановок на основе возрастающих деревьев.
5. Генерация беспорядков.
6. Генерация мультиперестановок.
7. Генерация перестановок на основе схемы (рис. 2). См. A000142, A000166 OEIS (приложение 1).

11.2 Генерация разбиений числа *n*

Задание. Построить алгоритм задачи, записанной ниже.

1. Генерация всех разбиений числа n.
2. Генерация разбиений, имеющих k частей
3. Генерация разбиений, ограниченных на множество принимаемых значений частей.

См. A000041 OEIS (Приложение 1).

11.3 Генерация разбиений множеств

Задание. Построить алгоритм задачи, записанной ниже.

1. Генерация всех подмножеств.
2. Генерация всех разбиений множества.
3. Генерация разбиений, имеющих ровно k членов.
4. Генерация ограниченных разбиений множества.

См. A000110, A008277 OEIS (Приложение 1).

11.4 Множества, заданные факториальными числами (2n)!! , (2n-1)!!

Двойной факториал для нечетных чисел определяется формулой:

$$(2n-1)!! = 1 \cdot 3 \cdot 5 \ldots \cdot 2n-1, \quad (-1)!!=1$$

Легко увидеть рекуррентную формулу:

$$f(n) = \begin{cases} 1, & n = 0 \\ (2n-1) \cdot f(n-1) \end{cases}$$

Имеется явная формула вычисления двойных факториалов для нечетных чисел

$$f(n) = \frac{(2n)!}{2^n n!}$$

$$f(n) = \sum_{k=0}^{n} E_2\,(n,k)$$

где $E_2(n,k)$ – числа Эйлера второго рода

Имеется также рекуррентная формула вида

$$f(n) = \begin{cases} 1, & n = 0 \\ \sum_{i=1}^{n-1} \sum_{j=0}^{n-j-1} \binom{n-1}{i} \binom{n-i-1}{j} f(i)\, f(j) f(n-i-j-1) \end{cases}$$

Множества, описываемые двойными факториалами нечетного числа:

1. Множество совершенных паросочетаний полного графа G n+1 вершин, n нечетное число.

2. Тернарные возрастающие деревья с *n* вершинами.
3. Разбиения множества 2*n* элементов на n блоков, в каждом из которых ровно 2 элемента. Например, *n*=3.

   0 [[1, 2], [3, 4], [5, 6]]

   1 [[3, 2], [1, 4], [5, 6]]

   2 [[1, 3], [2, 4], [5, 6]]

   3 [[5, 2], [3, 4], [1, 6]]

   4 [[1, 5], [3, 4], [2, 6]]

   5 [[1, 2], [5, 4], [3, 6]]

   6 [[1, 2], [3, 5], [4, 6]]

   7 [[5, 2], [1, 4], [3, 6]]

   8 [[3, 5], [1, 4], [2, 6]]

   9 [[3, 2], [5, 4], [1, 6]]

   10 [[3, 2], [1, 5], [4, 6]]

   11 [[5, 3], [2, 4], [1, 6]]

   12 [[1, 5], [2, 4], [3, 6]]

   13 [[1, 3], [5, 4], [2, 6]]

   14 [[1, 3], [2, 5], [4, 6]]
4. Перестановки Стирлига. Например, все перестановки n=3

   0 [3, 3, 2, 2, 1, 1]

   1 [2, 3, 3, 2, 1, 1]

   2 [2, 2, 3, 3, 1, 1]

   3 [2, 2, 1, 3, 3, 1]

   4 [2, 2, 1, 1, 3, 3]

   5 [3, 3, 1, 2, 2, 1]

   6 [1, 3, 3, 2, 2, 1]

   7 [1, 2, 3, 3, 2, 1]

   8 [1, 2, 2, 3, 3, 1]

   9 [1, 2, 2, 1, 3, 3]

10 [3, 3, 1, 1, 2, 2]

11 [1, 3, 3, 1, 2, 2]

12 [1, 1, 3, 3, 2, 2]

13 [1, 1, 2, 3, 3, 2]

14 [1, 1, 2, 2, 3, 3]

См. A006882, A001147 OEIS (Приложение 1).

Задание. Построить алгоритм задачи, записанной выше.

11.5 Генерация комбинаторных объектов описываемы биномиальными коэффициентами

Задание. Построить алгоритм задачи, записанной ниже.

1. Сгенерировать двоичное число длинной *n* и имеющее ровно *k* единиц. Например,

   n=5, *k*=3.

   [1, 1, 1, 0, 0]

   [1, 1, 0, 1, 0]

   [1, 0, 1, 1, 0]

   [0, 1, 1, 1, 0]

   [1, 1, 0, 0, 1]

   [1, 0, 1, 0, 1]

   [0, 1, 1, 0, 1]

   [1, 0, 0, 1, 1]

   [0, 1, 0, 1, 1]

   [0, 0, 1, 1, 1]

2. Сгенерировать подмножество из множества размером n и имеющее ровно k членов. Например, *n*=5, *k*=3.

   [1, 2, 3]

   [1, 2, 4]

   [1, 3, 4]

   [2, 3, 4]

[1, 2, 5]

[1, 3, 5]

[2, 3, 5]

[1, 4, 5]

[2, 4, 5]

[3, 4, 5]

3. Сгенерировать композиции числа n имеющие ровно *k* частей. Например, *n*=5, *k*=3

   [1, 1, 3]

   [1, 2, 2]

   [2, 1, 2]

   [1, 3, 1]

   [2, 2, 1]

   [3, 1, 1]

4. Сгенерировать число путей в прямоугольнике (0,0) (*n*,*k*), с шагами (0,1) и (1,0) (см. рис. 3)

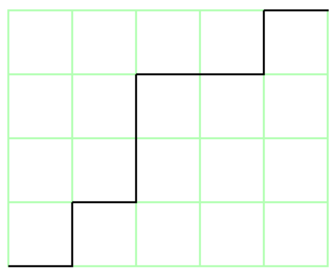


Рис. 3 Пример дорожки RURUURRUR

5. Сгенерировать строку символов с алфавитом (a,b) в которой (*n* букв a) и (*k* букв b). Пример решения см. задание 1 (раздел 11.5).

6. Сгенерировать разложения числа n имеющих ровно *k* частей. Например, *n*=4, *k*=3

   [0, 0, 4]

   [0, 1, 3]

   [1, 0, 3]

[0, 2, 2]

[1, 1, 2]

[2, 0, 2]

[0, 3, 1]

[1, 2, 1]

[2, 1, 1]

[3, 0, 1]

[0, 4, 0]

[1, 3, 0]

[2, 2, 0]

[3, 1, 0]

[4, 0, 0]

7. Сгенерировать все размещения n неразличимых шаров по k ящикам. (см. задачу 6). См. Треугольник Паскаля, A007318 OEIS (Приложение 1).

11.6 Генерация множеств, заданных треугольными числами

Треугольные числа являются суммой натуральных чисел

$$t_n = \sum_{k=1}^{n} k$$

Имеется рекуррентная формула

$$t_n = t_{n-1} + n$$

И явная формула

$$t_n = \binom{n}{2}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Перечислить все дуги в полном графе.
2. Множество костяшек домино.
3. Вставка скобок () в строку n символов: (a)bc, (ab)c, (abc), a(b)c, a(bc), ab(c).

См. A000217 OEIS (Приложение 1).

11.7 Тетраэдрические (или треугольные пирамидальные) числа

Тетраэдрические (или треугольные пирамидальные) числа определяются формулой

$$p_n = \sum_{k=1}^{n} t_k$$

Где t_k – треугольное число.

Имеется рекуррентная формула

$$p_n = p_{n-1} + t_n$$

И явная формула

$$p_n = \binom{n+2}{3}$$

Задание. Построить алгоритм задачи, записанной ниже.

Перечислить все упорядоченные произведения $\lambda_1 \cdot \lambda_2$ целых положительных чисел, таких, что

$$p_n = \sum_{\lambda_1+\lambda_2=n+1} \lambda_1 \cdot \lambda_2$$

Например, $\lambda_1 + \lambda_2 = 6$

1+5

2+4

3+3

4+2

5+1

$$p_n = 1 \cdot 5 + 2 \cdot 4 + 3 \cdot 3 + 4 \cdot 2 + 5 \cdot 1 = 35$$

См. A000292 OEIS (Приложение 1).

11.8 Тритриангулярные числа

Тритриангулярные числа – это числа в форме

$$T(n) = \binom{\binom{n+2}{2}}{2} = \frac{1}{8}n(n+1)(n+2)(n+3)$$

Задание. Построить алгоритмы комбинаторной генерации для выбора двух пар объектов из набора из (n+1) объектов, когда порядок не имеет значения.

<u>Например</u>, для n = 4 (4 объекта) имеется 3 случая выбора (12)(34), (13)(24), and (14)(23).

См. A050534 OEIS (Приложение 1).

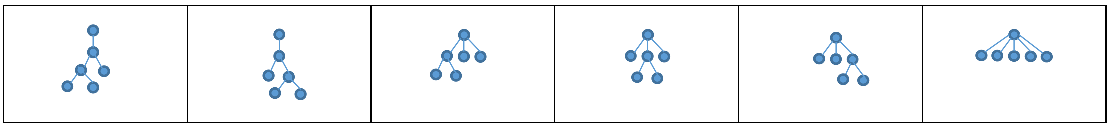
11.9 Множества, описываемые центральными биномиальными коэффициентами

Центральными биномиальными коэффициентами являются числа вида

$$p_n = \binom{2n}{n}$$

Задание. Построить алгоритм генерации для следующих множеств:

1. Корневые упорядоченные деревья с числом дуг (2n+1), имеющие корень с нечетной степенью, а остальные узлы имеют степень (0 или 2). Например, n=2, дуг-5:



2. Множество слов алфавита {a,b} длинной 2n, таких что в слове w нет префикса z, содержащего больше букв b, чем a.

   <u>Пояснение</u>: Центральные биномиальные коэффициенты можно рассматривать как частный случай биномиальных коэффициентов вида $\binom{n+m}{n}$, при $n = m$. Поэтому множества, описываемые p_n, являются частным случаем множеств, описываемые $\binom{n+m}{n}$.

Связь с числами Каталана C_n.

$$p_n = (n+1)C_n$$

Таким образом, множество p_n можно рассматривать как декартово произведение множества Каталана C_n и множества с (n+1) элементов.

Имеется тождество

$$p_n = \binom{2n}{n} = \sum_{k=0}^{n} \binom{n}{k}^2$$

которое позволяет рассматривать множество p_n как объединение множеств $B_{n,k}$ декартовых произведений описываемых биномиальными коэффициентами

$$B_{n,0} \times B_{n,0} + B_{n,1} \times B_{n,1} + \cdots + B_{n,k} \times B_{n,k}$$

11.10 Множества, описываемые биномиальными коэффициентами C^{n+1}_{2n+1}

Задание. Построить алгоритм задачи, записанной ниже.

1. Перечислить все варианты размещения (n + 1) неразличимых шаров в (n + 1) различимых коробок.
2. Корневые упорядоченные деревья с числом дуг (2n+2), имеющие корень с четной степенью, а остальные узла имеют степень 0 или 2.
3. Множество разложений числа n + 1 на n + 1 частей. Например, a(2) = 10: {0+0+3, 0+3+0, 3+0+0, 0+1+2, 0+2+1, 1+0+2, 1+2+0, 2+1+0, 2+0+1, 1+1+1}
4. Перечислить все нечетные графы.

11.11 Генерация комбинаторных объектов на основе чисел Фибоначчи

1. Сгенерировать двоичное число длинной *n* бит, в котором нет последовательности нулей.

   <u>Например</u>, битовые числа: 010, 011, 101, 110, 111.
2. Число подмножеств множества целых чисел {1,..,n}, не имеющих двух последовательных целых. Например, {1,2,3,4}:

   {},{1}{2},{3},{4},{1,3},{1,4}{2,4}
3. Последовательность домино (пример, рис. 4)

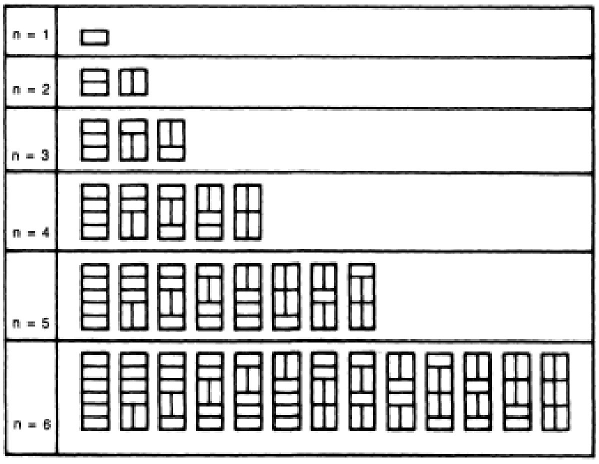


Рис. 4. Последовательности домино

4. Сгенерировать композиции числа (n+1), не имеющих частей, равных 1.

   Например, {2+2+2},{2+4},{4+2},{3+3},{6}

5. Сгенерировать композиции числа n-1, состоящие из частей равных 1 или 2.

   Например, число 4 может быть представлено {1+1+1+1}, {2+1+1}, {1+2+1}, {1+1+2}, {2+2}

6. Сгенерировать композиции числа n, состоящие из четного числа частей и нечетных чисел.

   Например, {1+1+1+1+1+1}, {1+1+1+3}, {1+1+3+1}, {1+3+1+1}, {1+5}, {3+1+1+1}, {3+3}, {5+1}.

   См. A000045 OEIS (Приложение 1)

11.12 Задание на генерацию множеств Каталана

Для генерации множеств, заданных числами Каталана имеется множество публикаций [2-7,12,13,16,20,21,24].

Задание. Построить алгоритм задачи, записанной ниже.

1. Все триангуляции выпуклого (n+2)-угольника. Примеры приведены на рис. 5

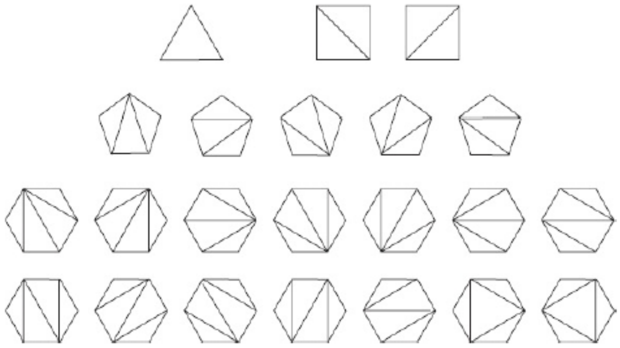


Рис. 5. Триангуляции выпуклого (n+2) - угольника

2. Бинарные расстановки скобок в строке, состоящей из (n+1) букв.

Например, все расстановки скобок в строке *xxxx*:

(*x*(*x*(*xx*)))
(*x*((*xx*)*x*))
((*xx*)(*xx*))
((*x*(*xx*))*x*)
(((*xx*)x)*x*)

Указание (см.раздел 12 Пример реализации алгоритмов комбинаторной генерации для множеств, описываемых числами Каталана). Необходимо модифицировать алгоритм GenCatalan. При n=0 печатать *x*. Перед первым вызовом GenCatalan печатать открывающую скобку. После второго вызова GenCatalan печатать закрывающую скобку.

3. Двоичные деревья с n вершинами (рис.6).

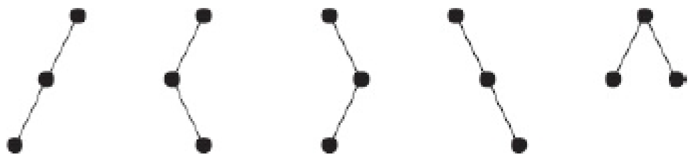


Рис. 6. Все двоичные деревья с 3 вершинами

Указание **(**см.раздел12**)**. Получить скобочною строку с помощью алгоритма GenCatalan, затем заменить открывающую скобку на 01, и закрывающую скобку на 10.
В полученной бинарной строке убрать первый и последние символы (0 или 1).
Затем использовать рекурсивный алгоритм генерации двоичного дерева:

вход: строка C содержащая последовательность
нулей или единиц

выход: двоичное дерево
Создать корень и сделать его текущим
Генерация(C):
 получить два бита ab из стоки C, и удалить их из C
  если ab = 01 то записать новый узел к текущему как левый,
   сделать его текущим.
   вызывать Генерация(c)
  если ab = 10 то записать новый узел к текущему как правый,
   сделать его текущим.
   вызывать Генерация(c)
  если ab = 00 то занести в стек текущий узел,
   записать новый узел к
   текущему как левый, сделать его текущим.
   вызывать Генерация(c)
  если ab = 11 то взять из стека текущий узел,
   записать новый узел к
   текущему как правый, сделать его текущим.
   вызывать Генерация(c)
 конец

4. Полные двоичные деревья с (2n+1) вершинами или (n+1) концевыми вершинами. Полным двоичным деревом называется двоичное дерево, у которого узлы содержат ровно два сына. Ниже представлены все полные двоичные деревья с 4 концевыми вершинами (или с семью вершинами). Скобки () обозначают узел. Итого: 4 листа и 3 узла, всего 7 вершин.

(()(()(()())))
(()((()())()))
((()())(()()))
((()(()()))())
(((()())())())

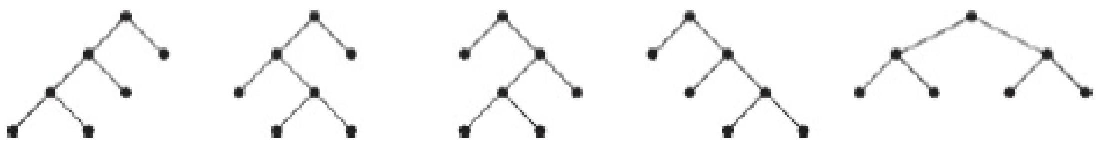


Рис. 7 Полные двоичные деревья с числом узлов 7

Указание (см.раздел 12 Пример реализации алгоритмов комбинаторной генерации для множеств, описываемых числами Каталана). Необходимо модифицировать алгоритм GenCatalan. При входном перед каждым вызовом GenCatalan печатать открывающую скобку. После каждого вызова GenCatalan печатать закрывающую скобку. Преобразовать поученную скобочную запись в полное двоичное дерево.

5. Плоские деревья с (n+1) вершинами.

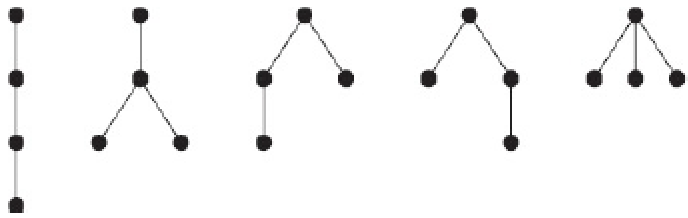


Рис. 8. Плоские деревья с 4 вершинами.

Указание. Получить скобочное представление, используя алгоритм GenCatalan. Создать узел, к этому узлу присоединить все узлы, порождаемые парой скобок (T) первого уровня и т.д (рис. 8).

6. Трехвалентные плоские деревья с корнем степени один и с (2n+2) вершинами (рис. 9)

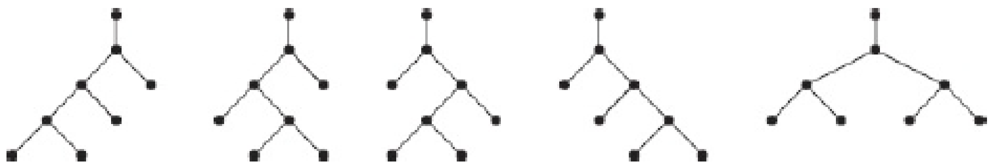


Рис.9 Все трехвалентные плоские деревья с корнем степени один и с 8 узлов.

7. Плоские деревья с (n+2) вершинами, такие, что крайняя правая цепь каждого главного поддерева имеет четную длину (рис.10)

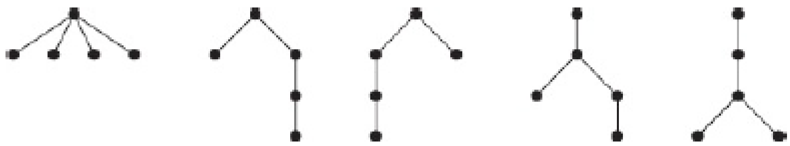


Рис. 10.

8. Решетчатые пути из точки (0,0) в точку (n,n) с шагами вида (0,1) или (1,0), не поднимающихся выше прямой y=x (рис. 11)

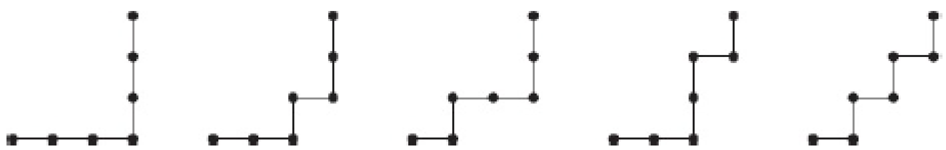


Рис. 11. Пути Дика с числом шагов 8

Биекция: открывающая скобка соответствует шагу (1,0) (шаг вправо), закрывающая скобка соответствует шагу (0,1) (шаг вверх)

9. Пути Дика из точки (0,0) в точку ($2n$,0), т.е. решетчатые пути с шагами вида (1,1) и (1,-1), не опускающиеся ниже оси x. (рис. 12)

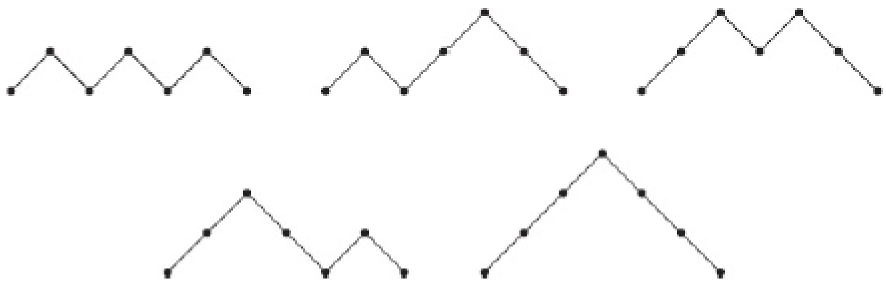


Рис. 12. Пути Дика из (0,0) в (8,0)

Биекция: открывающая скобка соответствует шагу (1,1) (шаг вверх по диагонали), закрывающая скобка соответствует шагу (1,-1) (шаг вниз по диагонали).

10. Пути Дика (см. пункт 9) из точки (0,0) в точку (2n+2,0), такие, что любая максимальная последовательность соседних шагов вида (1,-1), заканчивающаяся на оси x, имеют нечетную длину (рис. 13)

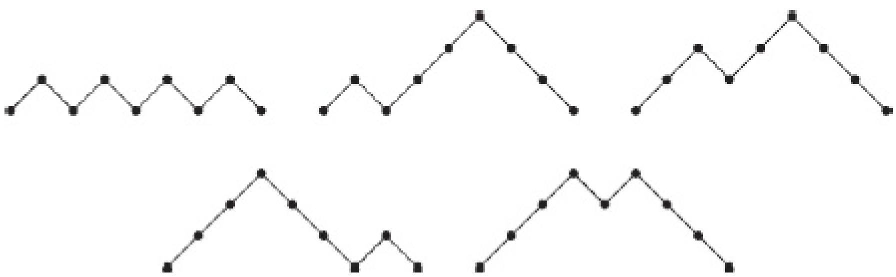


Рис. 13. Все пути Дика

11. Пути Дика (см. пункт 9) из точки (0,0) в точку ($2n$+2,0), не имеющие пиков на высоте два (рис 14).

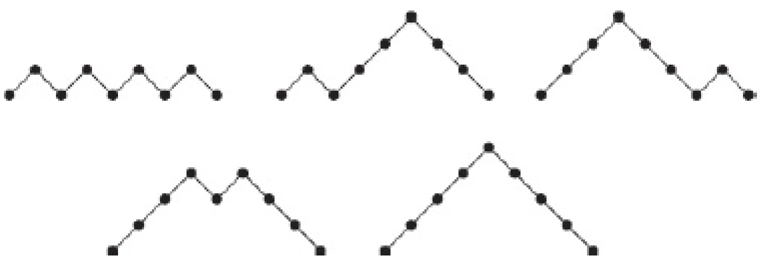


Рис. 14. Все пути Дика

12.Неупорядоченные пары решетчатых путей, состоящих их (n+1) шагов, которые начинаются в точке (0,0), имеют шаги вида (1,0) или (0,1), заканчиваются в одной и той же точке и пересекаются только в начальной и конечной точках (рис. 15)

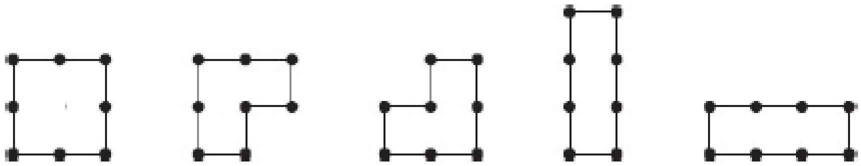


Рис. 15. Все решетчатые пути для задачи 12

13.Неупорядоченные пары решетчатых путей, состоящих их n-1 шагов, которые начинаются в точке (0,0), имеют шаги вида (1,0) или (0,1), заканчиваются в одной и той же точке, причем один путь никогда не поднимается выше другого (рис. 16).

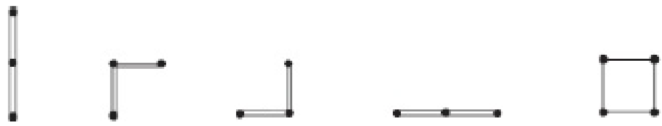


Рис. 16. Все решетчатые пути для задачи 13

14. Дано n непересекающихся хорд, состоящих $2n$ точек на окружности (рис. 17)

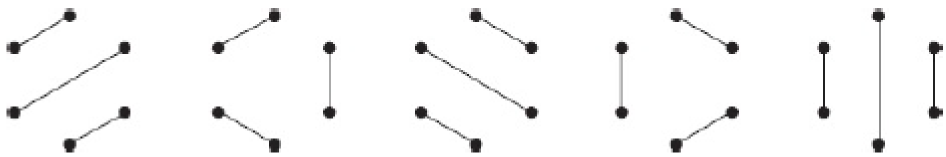


Рис. 17. Все множество хорд для n=3

15. Последовательности из n единиц и n минус единиц, все частичные суммы которых неотрицательны.

Например,

111---, 11-1--, 11—1-, 1-11--, 1-1-1-

Биекция: открывающая скобка - единица, закрывающая скобка -1.

16. Последовательности $1 \leq a_1 \leq a_2 \leq \cdots \leq a_n$ целых чисел, такие что $a_i \leq i$.
Например,

111 112 113 122 123

Биекция: в начале 1, скобка открывающая печать, скобка закрывающая +1.

17. Последовательности $a_1 \le a_2 \le \dots \le a_{n-1}$ целых чисел, удовлетворяющих условию $1 \le a_i \le 2i$.

Указание. Вычитая (i-1) из a_i и добавляя в начало 1, получим предыдущую последовательность.

Например,

12, 13, 14, 23, 24

18. Последовательности $a_1 \le a_2 \le \dots \le a_n$ целых чисел, удовлетворяющих условию $a_1 = 0,\quad 0 \le a_{i+1} \le a_i$

Например,

000, 001, 010, 011, 012

Указание. Пусть $b_i = a_i - a_{i+1} + 1$. Заменив a_i на последовательность, состоящую из одной единицы, за которой следуют b_i минус единиц, при $1 \le i \le n$, где $a_{n+1} = 0$, получим последовательность (см. задачу 15)

Например, 010 - b_1=0, b_2=2, b_3=1 -> 11--1-

19. Последовательности $a_1 \le a_2 \le \dots \le a_{n-1}$ целых чисел, такие что $a_i \le 1$ и все частичные суммы неотрицательны

0,0 0,1 1,-1 1,0 1,1

Указание. Взять первые разности из пункта 18.

Например,

a_2-a_1 и a_3-a_2

000 -> 0-0=0, 0-0=0

001 -> 0-0=0, 1-0=1

010 -> 1-0=1, 0-1=-1

011 -> 1-0=1, 1-1=0

012 -> 1-0=1, 2-1=1

20. Последовательности $a_1 \le a_2 \le \dots \le a_n$ целых чисел, такие что $a_i \ge -1$ и все частичные суммы неотрицательны и $a_1 + a_2 + \dots + a_n = 0$.

Например,

0,0,0 0,1,-1 1,0,-1 1,-1,0 2,-1,-1

Указание. Будем осуществлять поиск в глубину на плоском дереве с (n+1) вершинами. При первом вхождении вершины запишем число, на единицу меньшее, чем число ее сыновей, а последнюю вершину проигнорируем. Это даст биекцию.

21. Перестановки $a_1 \dots a_{2n}$ мультимножества $\{1^1, 2^2 \dots n^2\}$ такие что: 1) первые вхождения элементов {1,2,…,n} упорядочены по возрастанию; 2) не существует подпоследовательности вида $\alpha\beta\alpha\beta$.
Например,

112233 112332 122331 123321 122133

Указание. Генерируем перестановку (см. пункт 23), вставляем числа [1,…,n] с условиями 1 и 2.

22. Непересекающиеся разбиения множества [n], т.е. разбиения вида

$$\pi = \{ B_1, \dots, B_k\} \in \mathrm{P}_n$$

такие, что если $a < b < c < d$ и $a, c \in B_i$ и $b, d \in B_j$, то $i = j$.

Например,

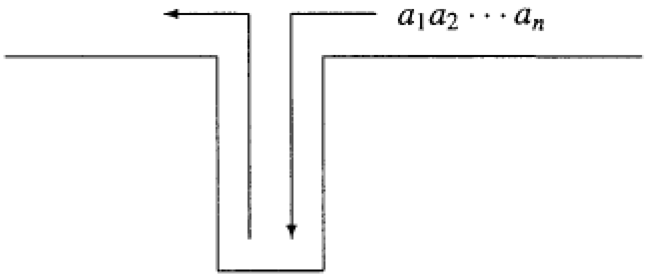
[123] [12]-[3] [12]-[3] [13]-[2] [23]-[1] [1]-[2]-[3]

Указание**.** По скобочной записи, получаемой алгоритмом GenCatalan (см раздел 12), строим плоское дерево с число вершин (n+1), нумеруем узлы дерева от 1 до n в соответствии с левосторонним обходом, корень дерева не нумеруется. Для всех узлов не являющиеся листьями находим множество сыновей.

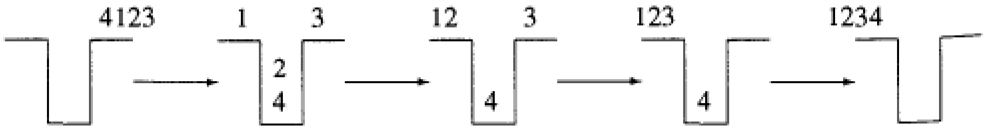
Например,

(())() нумерация (1(2))(3) множества [1,3]-[2]

23. Перестановки $a_1, a_2, \dots, a_n$ множества $[n]$, которые можно упорядочить по возрастанию при помощи единственного стека. Эти перестановки определяются рекурсивно следующим образом. Пусть $\emptyset$- пустая последовательность, положим $S(\emptyset) = \emptyset$. Если $w = unv$ w последовательность различных целых чисел, наибольший член которого равен n, то положим $S(w) = S(u)S(v)n$. Перестановка называется стек-сортируемой, если $S(w) = w$.



Например,



123 132 213 312 321

Указание**.** Нужно воспользоваться алгоритмом (см. пункт 23), затем преобразовать выходной массив List[j]=n-List[j]+1, отпечатать с конца.

23. Перестановки $a_1, a_2, \dots, a_n$ множества [n], такие что, не существует тройки индексов $i < j < k$, удовлетворяющей условию $a_j < a_k < a_i$, (такие перестановки называются 312-избегающими)

123 132 213 231 321

Указание. Получить скобочную строку на основе алгоритма GenCatalan. Положить *N*=*J*=0. При просмотре данной строки использовать следующие правила: если текущий символ «открывающая скобка», то *N*=*N*+1, положить *N* в стек. Если текущий символ «)», то вынуть число из стека, j=j+1 и записать его в выходной массив $List[j]$.

24. Перестановки $a_1, a_2, \dots, a_n$ множества $[n]$ такие, что длина максимальной убывающей подпоследовательности не превосходит двух (т.е. такие, что не существует тройки индексов $i < j < k$, удовлетворяющей условию $a_i > a_j > a_k$ такие перестановки называются 321-избегающими):

123 213 132 312 231

Указание**.** Получить скобочную строку на основе алгоритма GenCatalan. Положить *w*=0, организовать цикл по просмотру символов этой строки. Если текущий символ «открывающаяся скобка», то в стек *Stk* заносится *w*, иначе *w*=*w*+1. По окончанию цикла записывается массив *Lid*, содержащий последовательность от 1 до n, записанную по убыванию. Далее, организуется цикл от 1 до (n-1). Если *Stk*[i]>*Stk*[i+1], то Lid[*n*-*Stk*[i]+1]=0, в стек *Per* заносится значение *Stk*[i], иначе в стек *Per* заносится значение функции GetMax(*Lid*). По окончанию данного цикла стек *Per* содержит искомую перестановку. Функция GetMax ищет в массиве *Lid* первый ненулевой элемент m, на его место записывается 0 и возвращается значение m.

25. Перестановки $a_1, a_2, \dots, a_n$ множества [2n], такие что:
1) элементы $[1{,}3, \dots, 2n-1]$ упорядочены по возрастанию;
2) элементы $[2{,}4, \dots, 2n]$ упорядочены по возрастанию;
3) элемент $(2i-1)$ предшествует $2i,\ 1 \le i \le n$

123456 123546 132456 132546 135246

Указание**.** Получить скобочную запись с помощью алгоритма GenCatalan. Положить o=1, e=2. Организовать цикл для скобочной строки, и использовать правила:

если текущий символ – «открывающаяся скобка», то печатать o и o=o+2 иначе печатать e и e=e+2.

26. Стандартные таблицы Юнга вида (n,n) или эквивалентные таблицы вида (n,n-1) (см. [2]).

T(3,3)

123 124 125 134 135
456 356 346 256 246

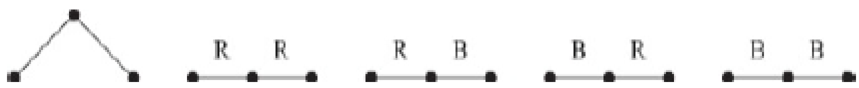
T(3,2)

123 124 125 134 135
45 35 34 25 24

Указание**.** Получить скобочную запись с помощью алгоритма GenCatalan. Запишем n=length(s)/2, j=1, k=1. Записать два массива M_1[n], M_2[n]. Организовать цикл для просмотра строки по i от 1 до 2n по следующим правилам: если текущий символ скобка открывающая, то M_1[j]=i и j=j+1 иначе M_2[k]=i и k=k+1. После цикла печать M_1, M_2.

27. Путь Моцкина это решетчатый путь из точки (0,0) в точку (n,0), с шагами вида (1,0) (1,1) и (1,-1), не опускающиеся ниже оси x. Необходимо сгенерировать все пути Моцкина из (0,0) в точку (n-1,0) с учетом, что шаг вида (0,1) окрашен в синий или красный цвет.

Например, (см. рис. 18)



/\, RR, RB, BR, BB

Рис. 18. Пути Моцкина и соответствующая строка символов

Указание. Необходимо сгенерировать скобочную строку, удалить первый и последний символы. Далее организовать цикл с шагом 2 по следующим правилам:

если два подряд символа равны «((», то печать шага (1,1),

если два подряд символа равны «))», то печать шага (1,-1),

если два подряд символа равны «()», то печать красного шага (1,0),

если два подряд символа равны «)(», то печать синего шага (1,0).

28. Пути Дика, имеющие (*n*-1) пиков и не содержащие трех последовательных шагов, направленных вверх или трех последовательных шагов, направленных вниз.

Например,

"/\/\" "/\/\/\" "//\\/\" "//\\/\\" "//\/\\"

Указание. Необходимо сгенерировать скобочную строку, удалить первый и последний символы. Далее организовать цикл с шагом 2 по следующим правилам:

если два подряд символа равны « ((» , то печать шагов (1,1)+(1,1)+(1,-1),

если два подряд символа равны «))», то печать шагов (1,1)+(1,-1)+(1,-1),

если два подряд символа равны «()», то печать шагов (1,1)+(1,-1),

если два подряд символа равны «)(», то печать шагов (1,1)+(1,1)+(1,-1)+(1,-1).

29. Левые факторы путей Дика, состоящие из (n-1) шагов, направленных верх (рис. 19).

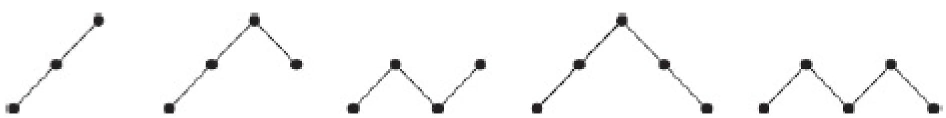


Рис. 19. Левые факторы путей Дика

Биекция: между скобочной записью и левыми факторами путей Дика

```
["(",")","(",")"]
"/\/\"
["(",")","("]
"/\/"
["(","(",")",")"]
"//\\"
["(","(",")"]
"//\"
["(","("]
"//"
```

<u>Указание</u>**.** Необходимо сгенерировать скобочную строку, удалить все последние закрывающие скобки, удалить последнею открывающую скобку. Каждую оставшуюся открывающую скобку заменить на шаг (0,1), каждую закрывающую скобку заменить на шаг (1,-1).

30. Все пути Моцкина, не имеющие пиков и состоящие из n шагов вверх (1,1) и влево (1,0) (рис. 20).

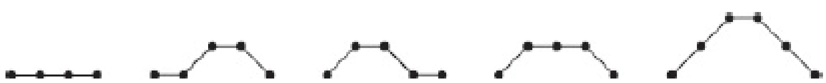


Рис. 20. Все пути Моцкина

31. Число вариантов соединений (n+1) точек на прямой n непересекающимися дугами так, чтобы не было циклов и из каждой точки дуги выходили только влево или только вправо:

<u>Указание</u>**.** Занумеруйте точки числами 1, 2,…, $n + 1$ слева направо. Пусть точка с номером i соединена с a_i точками, номера которых больше, чем i. Рассмотрим последовательность из (n+1) фрагментов, i-й из которых имеет вид ‘1^a_i -1’ [24].

11.13 Генерация множеств, заданных числами Моцкина.

Числа Моцкина описывают большое множество комбинаторных объектов: классы деревьев, решетчатые пути, скобочные структуры и д.р.

Числа Моцкина задаются следующими формулами:

$$M_n = \begin{cases} 1, & n = 0 \\ 1, & n = 1 \\ \dfrac{3(n-1)M_{n-2} + (2n+1)M_{n-1}}{(n+2)} \end{cases}$$

Рекуррентная формула

$$M_n = \begin{cases} 1, & n = 0 \\ 1, & n = 1 \\ M_{n-1} + \sum_{i=0}^{n-2} M_i M_{n-i-2}, & n > 1 \end{cases}$$

Явная формула для чисел Моцкина

$$M_n = \frac{1}{n+1} \sum_{k=0}^{\frac{n+1}{2}} \binom{n+1}{k} \binom{n+1-k}{k-1}$$

Связь с числами Каталана

$$M_n = \sum_{k=0}^{n} \binom{n}{2k} C_k$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Перечислить варианты соединения *n* различающихся точек на окружности непересекающимися хордами (хорды могут выходить не из каждой точки см. рис. 21)

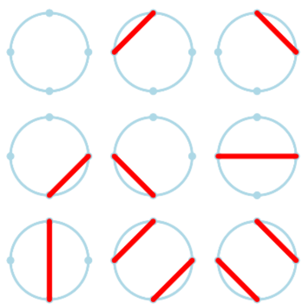


Рис. 21. Варианты соединения непересекающихся хорд для 4 точек

2. Пути из *n* шагов на оси, шаги –1, 0, или 1, начинающиеся и заканчивающиеся в 0.

3. Пути на прямоугольной решетке из (0,0) в (n,n), с шагами (0,2), (2,0), и (1,1), никогда не пересекающую прямую y = x.

4. Пути из (0,0) в (n,0) с шагами (1,0), (1,1), and (1,–1), никогда не переходящую ось x. Такие пути называются дорожки Моцкина. На рис. 22 показаны все варианты дорожек для 4 шагов.

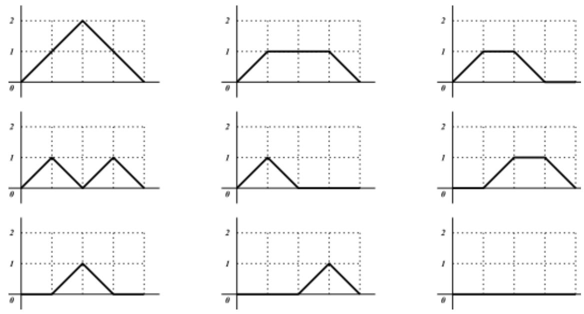


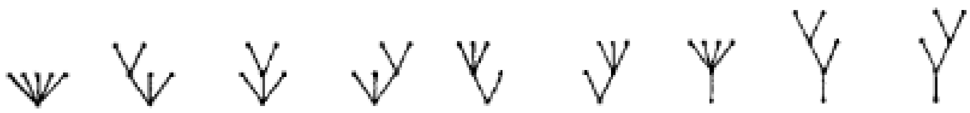
Рис. 22. Пути Моцкина для n=4

5. Пары последовательностей чисел $1 < a_1 < \cdots < a_k < n$ и $1 < b_1 < \cdots < b_k < n$ такие, что $a_k < b_k$ и каждое число из множества чисел появляется по крайне мере один раз среди a_k и b_k

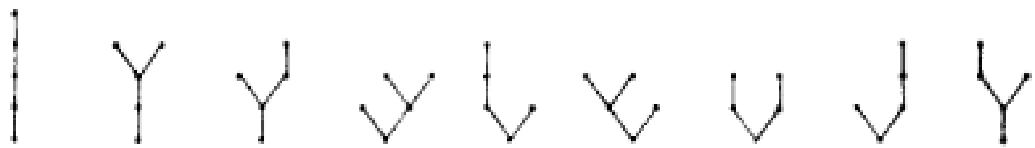
6.Баллотировочная последовательность $a_1, \ldots, a_{2n+2}$ такая, что нет последовательности трех чисел $a_{i-1}, a_i, a_{i+1} = 1, -1, 1$

7.Корневые деревья с n+1 дугами у которых корень не имеет сыновей, имеющих одного сына.

Например, все деревья



8.Корневые деревья с n дугами имеет узлы, у которых не больше двух сыновей. Например, все деревья имеющие 4 дуги.



9. Множество правильных скобок, у которых нет скобок вида ((x)).

Например,

( )( )( )( )( ), (( )( ))( )( ), ( )(( )( ))( ), ( )( )(( )( )), (( )( )( ))( ),

( )(( )( )( )), (( )( )( )( )), ((( )( ))( )), (( )(( )( ))).

10.Двоичные деревья (n-1) дуг такие, что нет последовательных дуг, наклоненных вправо.

<u>Например</u>, все деревья из 3 дуг, у которых нет последовательно наклонённых дуг вправо.

11.Получить все слова длинной в *n* символов алфавита {a,b,c} такие, что для каждого префикса z слова w выполняется условие: #(z,a) >= #(z,b) >= #(z,c), где #(z,x) считает x букв в z.

<u>Например</u>,

M(2)={aa,ab}

M(3)={aaa,aab,aba,abc}

M(4) ={aaaa, aaab, aaba, abaa, aabb, abab, aabc, abac, abca}.

11.14 Множества, заданные числами Шредера

Задание. Построить алгоритм задачи, записанной ниже.

Малые числа Шредера

1. Сгенерировать варианты расстановки скобок в строке символов.

<u>Например</u>, для строки “xxxx”

0 xxxx

1 x(xx)x

2 xx(xx)

3 (xx)xx

4 (xx)(xx)

5 (xxx)x

6 x(xxx)

7 ((xx)x)x

8 x((xx)x)

9 (x(xx))x

10 x(x(xx))

2. Сгенерировать деревья, у которых ровно *n*- листьев.

   Например, см. рис. 23.

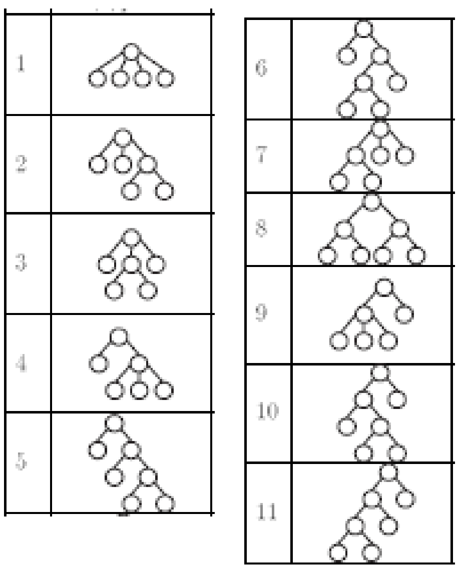


Рис. 23. Деревья имеющие 4 листа

3. Сгенерировать деревья Шредера с заданным числом узлов *n*.

   Например, см рис. 24.

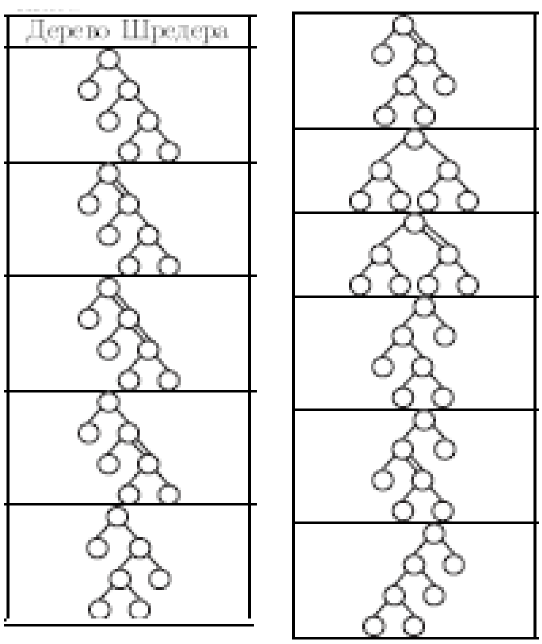


Рис. 24. Деревья имеющие 4 листа и правые ветви имеют два цвета

4. Сгенерировать дорожки Шредера из точки (0,0) в точку (2n,0) используя шаги H=(2,0), U=(1,1) и D(1,-1). Например, все дорожки из (0,0) в точку (4,0).

   HH, UHD, UUDD, HUD, UDH, UDUD.

Большие числа Шредера

1. Сгенерировать дорожки Шредера на решетке из точки (0,0) в точку (n,n).

   Используя шаги: (0, 1), (1, 0), (1, 1). Пример показан на рис. 25

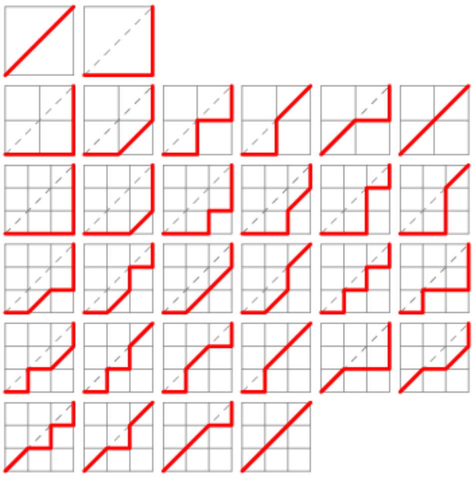


Рис. 25. Дорожки Шредера

11.15. Генерация комбинаторных объектов на основе чисел Риордана

Числа Риордана описывают некоторое множество комбинаторных объектов: деревья, дорожки, классы композиций натурального числа и др.

Для чисел Риордана имеется несколько рекуррентных формул

$$R_n = \frac{n-1}{n+1}(2 \cdot R_{n-1} + 3 \cdot R_{n-2})$$

$$R_n = \begin{cases} 1, & n = 0 \\ \sum_{i=0}^{n-1} R_i R_{n-i-1} + (-1)^n, & n > 0 \end{cases}$$

$$R_n = \begin{cases} 1, & n = 1 \\ \sum_{i=1}^{n-2} R_i (R_{n-i} + R_{n-i-1}) & n > 0 \end{cases}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Сгенерировать деревья с n узлами, не имеющие узлов с одним сыном. (пример показан на рис. 26)

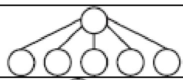
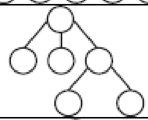
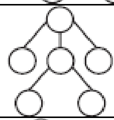
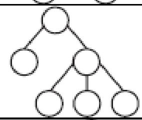
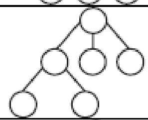
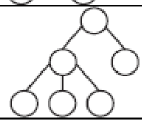
№	Композиция	Дерево Риордана
1	1+[1+1+1+1]	
2	1+1+3[1+1+1]	
3	1+3[1+1+1]+1	
4	1+4[1+1+1+1]	
5	3[1+1+1]+3[1+1+1]	
6	4[1+1+1+1]+1	

Рис. 26. Деревья Риордана и композиции

2. Сгенерировать пути Моцкина из *n* шагов, не имеющих шагов горизонтальных шагов уровня 0.

11.16. Генерация множеств, заданных числами Фина

Числа Фина записаны в энциклопедии целых последовательностей под номером A000957. Числа Фина описывают множество разнообразных объектов.

Например, они описывают число планарных корневых деревьев у которых корень имеет четное число сыновей.

Для чисел Фина имеется ряд явных и рекуррентных формул. Система рекуррентных формул описания чисел Фина.

$$E(n) = \begin{cases} 0 & n \le 1 \\ 1, & n = 1, \\ \sum_{i=1}^{n-1} C(i-1)O(n-i-1)\,, & n > 1 \end{cases}$$

Где C(n) – числа Каталана, $E(n)$ – числа Фина. $O(n)$ – число планарных корневых деревьев, у которых корень имеет нечетное число сыновей

$$O(n) = \begin{cases} 0, & n \leq 1 \\ 1, & n = 1, \\ \sum_{i=1}^{n-1} C(i-1)E(n-i+1) + C(n-1), & n > 1 \end{cases}$$

Имеется рекуррентная формула для чисел Фина

$$E(n) = \begin{cases} 0 & n \leq 1 \\ 1, & n = 1, \\ \sum_{i=2}^{n-1} C(n-i-1)(E(i) + E(i-1)), & n > 1 \end{cases}$$

Явная формула для чисел Фина имеет вид

$$E(n) = \frac{1}{n} \sum_{k=0}^{\frac{n-1}{2}} \binom{n}{k} \binom{n-k-2}{n-2k-1} 2^{n-2k-1}$$

Для чисел Фина имеется связь с числами Каталана

$$\mathrm{C}(n) = 2E(n) + E(n-1)$$

Эта связь обеспечивает получение заданий, для генерации подмножеств из множеств, описываемых числами Каталана. Некоторые задания приведены ниже.

Задание. Построить алгоритм задачи, записанной ниже.

1. Получить все планарные корневые деревья с n узлами, имеющими четное число сыновей у корня (см. таблицу).

   0 (()(()()))

   1 (()((())))

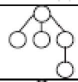
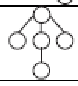
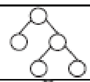
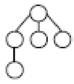
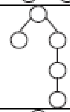
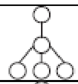
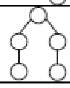
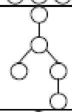
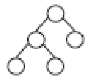
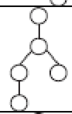
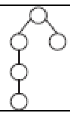
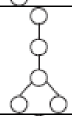
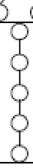
   2 (()()()())

   3 ((())(()))

   4 ((()())())

   5 (((()))())

Таблица Корневых деревьев с четным и нечетным числом деревьев у корня.

№	Композиция	Нечетное дерево	Композиция	Четное дерево
1	1+1+2		1+1+1+1	
2	1+2+1		1+3[1+1]	
3	2+1+1		1+3[2]	
4	4[1+1+1]		2+2	
5	4[1+2]		3[1+1]+1	
6	4[2+1]		3[2]+1	
7	4[3[1+1]			
8	4[3[2[1]]]			

2. Дорожки Дика Dyck, не имеющие холмов /\ высотой равной 1 (рис. 27).

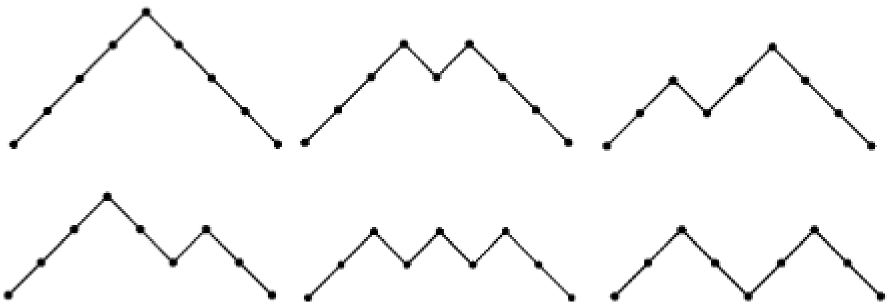


Рис. 27. Пример дорожек Дика для задания 2.

3. Непересекающиеся разбиения множества (см. числа Каталана), имеющие четное число элементов в первом блоке. <u>Например</u>, представлено на рис. 28.

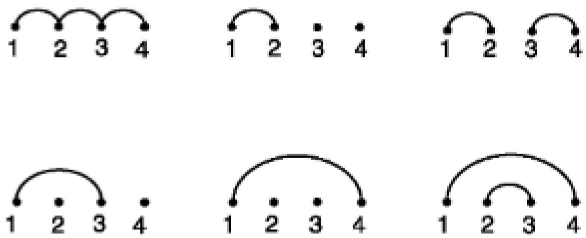


Рис. 28. Все непересекающиеся разбиения с четным числом элементов в первом блоке, множество содержит 4 элемента

4. Двоичные деревья, состоящие из n узлов, где корень и все правые узлы имеют степень 2 или 0 (рис. 29).

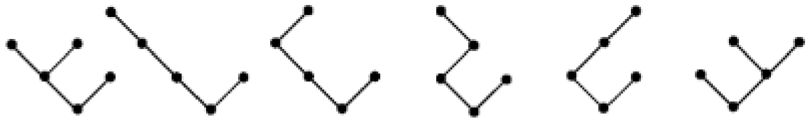


Рис. 29. Двоичные деревья из 4 узлов с указанными свойствами

5. Диаграммы Юнга (n; n), где нет колонки в форме

$$\begin{array}{|c|}\hline k \\ \hline k+1 \\ \hline\end{array}.$$

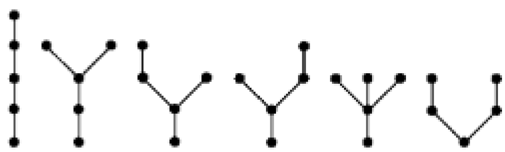
Например,

$$\begin{array}{|c|c|c|c|}\hline 1&2&3&4\\\hline 5&6&7&8\\\hline\end{array},\quad \begin{array}{|c|c|c|c|}\hline 1&2&3&5\\\hline 4&6&7&8\\\hline\end{array},\quad \begin{array}{|c|c|c|c|}\hline 1&2&3&6\\\hline 4&5&7&8\\\hline\end{array};$$

$$\begin{array}{|c|c|c|c|}\hline 1&2&4&5\\\hline 3&6&7&8\\\hline\end{array},\quad \begin{array}{|c|c|c|c|}\hline 1&2&4&6\\\hline 3&5&7&8\\\hline\end{array},\quad \begin{array}{|c|c|c|c|}\hline 1&2&5&6\\\hline 3&4&7&8\\\hline\end{array}.$$

6. Планарные деревья, не имеющие листья у корневой вершины.

Например,



11.15 Генерация множеств, заданных сопряженными числами Фина

Сопряженные числа Фина описываются рекуррентной формулой следующего вида

$$O(n)=\begin{cases}0, & n\le 1\\ 1, & n=1,\\ \sum_{i=1}^{n-1} C(i-1)E(n-i+1)+C(n-1), & n>1\end{cases}$$

где $E(n)$ – числа Фина, $C(n)$ - числа Каталана.

Существует явная связь между числами Каталана, Фина и сопряжёнными числами Фина

$$\mathrm{C}(n)=E(n)+O(n)$$

Множества, описываемые числами Фина и сопряженными числами Фина при объединении образуют множество Каталана. Таким образом, все задачи, представленные в разделе 11.12, могут быть сформулированы для чисел Фина и сопряженных чисел Фина.

Например, получить все планарные корневые деревья с n узлами, имеющими нечетное число сыновей у корня (см. Таблица Корневых деревьев с четным и нечетным числом деревьев у корня)

Задание. Построить алгоритм задачи, записанной выше.

11.16 Генерация множеств, описываемых числами Белла

Числа Белла описываются рекуррентным уравнением:

$$B_n=\begin{cases}1, & n=0\\ \sum_{i=0}^{n-1} B_i\binom{n-1}{i}, & n>0\end{cases}$$

Они описывают несколько комбинаторных множеств, одним из них являются разбиения множеств.

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество всех разбиений множества (рис. 30)

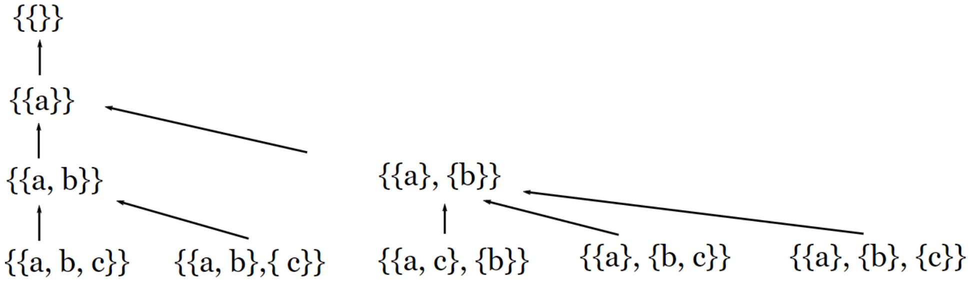


Рис. 30. Все разбиения множества {abc}

2. Множество вариантов размещения n матрешек. Например, 3 матрешки (рис. 31)

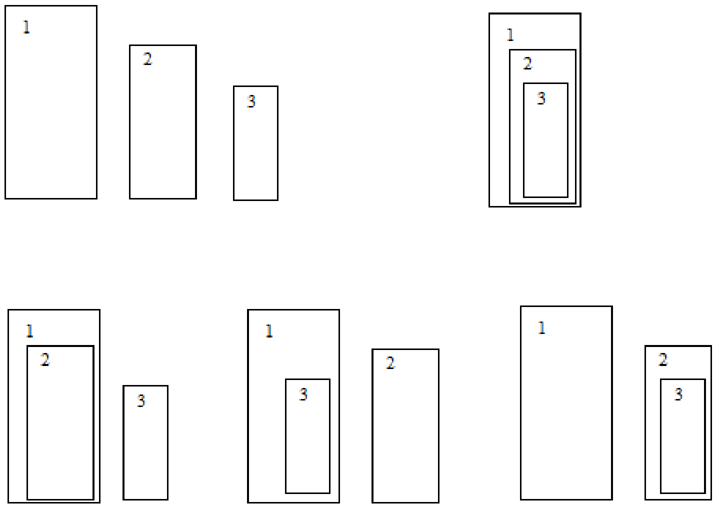


Рис. 31. Все варианты размещения 3 матрешек

A000110 – Статья в OEIS содержит описание чисел Белла и их комбинаторные интерпретации.

11.17 Генерация множеств, описываемых числами Фубини

Числа Фубини описываются следующим рекуррентным уравнением

$$Fu(n) = \begin{cases} 1, & n = 0 \\ \sum_{k=1}^{n} \binom{n}{k} Fu(n-k), & n > 0 \end{cases}$$

или

$$Fu(n) = \sum_{k=0}^{n} S_2\,(n,k)k!$$

Здесь $S_2(n,k)$ - числа Стирлинга второго рода.

См. OEIS A000670 – описание чисел Фубини и их комбинаторная интерпретация.

См. [12-14]

Задание. Построить алгоритм задачи, записанной выше.

11.18 Генерация множеств, описываемые формулой Фусса-Каталана

Задание. Построить алгоритм задачи, записанной ниже.

Числа Фусса-Каталана определяются явной формулой

$$T_n(m,k) = \frac{k}{mn+k}\binom{mn+k}{n}$$

Имеется также рекуррентная формула

$$T_n(m,k) = \begin{cases} 0, & n < 0, k \le 0 \\ 1, & n = 0 \\ T_n(m,k-1) + T_{n-1}(m,m+k+1) & \end{cases}$$

Представленные формулы обобщают достаточно много разнообразных чисел.

Например, при m=0, получим треугольник Паскаля. При m=2, k=1 получим числа Каталана.

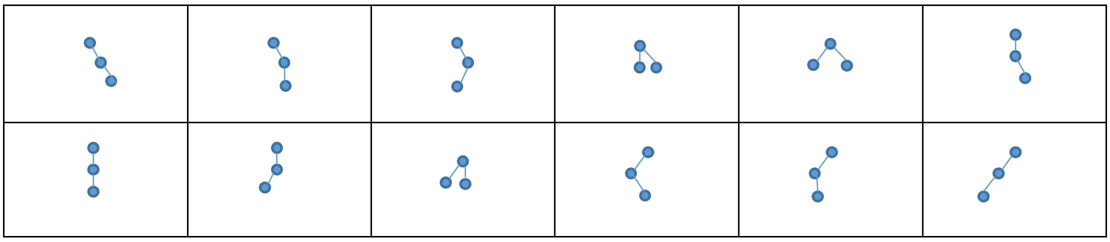
Рассмотрим формулу Фусса-Каталана при m=3, k=1

$$T_n = \frac{1}{3n+1}\binom{3n+1}{n}$$

Эта формула описывает следующие классы множеств:

1. Множество тернарных деревьев с заданным числом узлов n.

Например, T(3)=12, все тернарные деревья из 3 узлов.



Для тернарных деревьев имеется рекуррентная формула, основанная на разложениях

$$T_n = \begin{cases} 1, & n = 0 \\ \sum\limits_{i+j+k=n} T_i T_j T_k\,, & n > 0 \end{cases}$$

2. Множество полных тернарных деревьев

   0 (()()(()()(()()())))

   1 (()()(()(()()())()))

   2 (()()((()()())()()))

   3 (()(()()())(()()()))

   4 ((()()())()(()()()))

   5 (()(()()(()()()))())

   6 (()(()(()()())())())

   7 (()((()()())()())())

   8 ((()()())(()()())())

   9 ((()()(()()()))()())

   10 ((()(()()())())()())

   11 (((()()())()())()())

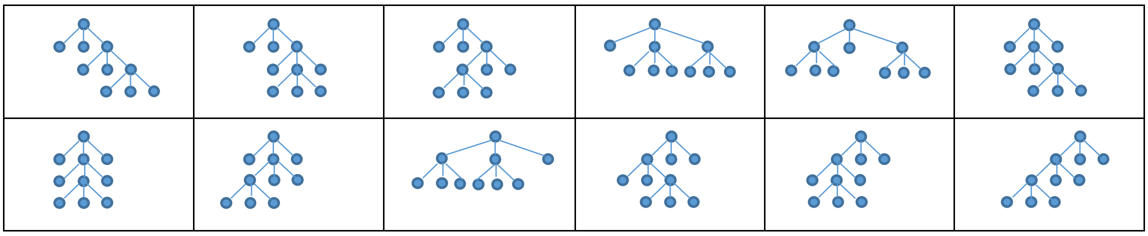


Таблица полных тернарных деревьев с (3*n*+1) узлами, *n*=3

3. Множество путей из (0,0) в (2*n*,0), используя шаги U шаг - (1,1) и D шаг (0,-2) и не пересекая ось x.

   <u>Например</u>, a(2)=3;

   UUUUDD,

   UUUDUD,

   UUDUUD.

Числа Фусса-Каталана вида

$$T_n = \frac{1}{4n+1}\binom{4n+1}{n}$$

Для данного вида имеется рекуррентная формула, основанная на разложениях

$$T_n = \begin{cases} 1, & n = 0 \\ \sum\limits_{i+j+k+ь=n} T_i T_j T_k T_m\,, & n > 0 \end{cases}$$

Эта формула описывает следующие классы множеств.

1. Множество четверичных деревьев с заданным числом узлов n.
2. Множество полных четверичных чисел (4n+1) узлами.

11.19 Генерация множеств, описываемых треугольником Каталана 1

Для треугольника Каталана имеется замечательная формула

$$T(n,k) = \sum_{\pi_k \in P_n} C(\lambda_1)\,C(\lambda_2)\ldots C(\lambda_k),$$

где $\pi_k = \{\lambda_1 + \lambda_2 + \cdots + \lambda_k\}$ – композиция числа n имеющая ровно k частей, P_n – множество композиций числа n. Здесь $C(m) - (m-1)$ число Каталана. Суммирование ведется по всем композициям π_k множества P_n.

Например,

$$T(n,3) = \sum_{\lambda_1+\lambda_2+\lambda_3=n} C(\lambda_1)\,C(\lambda_2)C(\lambda_3)$$

Пусть n=5, тогда композиций будет у которых ровно 3 части будет 6:

{1+1+3}

{1+2+2}

{1+3+1}

{2+1+2}

{2+2+1}

{3+1+1}

Откуда

$$T(5,3) = C(1)C(1)C(3) + C(1)C(2)C(2) + C(1)C(3)C(1) + C(2)C(1)C(2) +$$
$$+C(2)C(2)C(1) + C(3)C(1)C(1).$$

Зная, что $C(1) = 1$, $C(2) = 1, C(3) = 2$, получим

$$T(5,3) = 2 + 1 + 2 + 1 + 1 + 2 = 9.$$

Для треугольника Каталана имеется явная формула

$$T(n,k) = \frac{k}{n}\binom{2n-k-1}{n-k}$$

Для треугольника Каталана имеется рекуррентная формула

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k} + T_{n-1,k-1}, & n > k. \end{cases}$$

Рассмотрим комбинаторную интерпретацию $T(n,k)$. Произведение чисел Каталана $Cat(\lambda_1)Cat(\lambda_2)\dots Cat(\lambda_k)$

Можно рассматривать как декартово произведение множеств Каталана, заданного числами $Cat(\lambda_i)$. Например, числа Каталана описывают число корневых упорядоченных деревьев. Тогда $T(n,k)$ описывает число лесов корневых упорядоченных деревьев имеющих n узлов и k деревьев. Если числа Каталана описывают число дорожек Дика, то $T(n,k)$ - число дорожек, получаемых из k дорожек Дика с общим числом n шагов.

Задание. Построить алгоритм задачи, записанной ниже.

1. Для множества планарных корневых деревьев с заданным числом узлов n, $T(n,k)$ перечисляет упорядоченный лес k планарных корневых деревьев, содержащих суммарно n узлов (рис. 32)

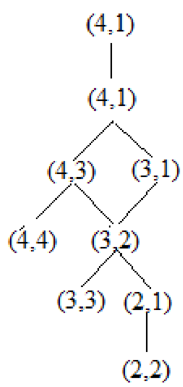


Рис. 32. Схема построения всех элементов множества T(4,2)

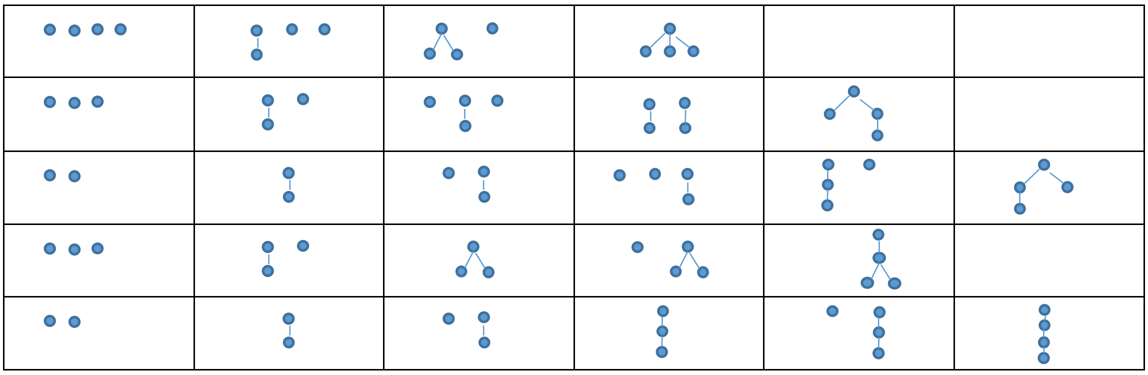


Таблица построения всех лесов для T(4,2)

2. Для скобочных структур с заданным числом пар скобок “(,)” $T(n,k)$ перечисляет множество k-кортежей правильных скобочных структур, содержащих суммарно n пар скобок.
3. Для дорожек Дика с заданным числом пар шагов (1,1), (1,-1), $T(n,k)$ перечисляет множество дорожек имеющих ровно k точек касания оси x.

11.20 Генерация множеств, заданных треугольником Каталана 2

Треугольник Каталана 2 задается следующей рекуррентной формулой

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k, \\ T_{n-1,k-1} + 2T_{n-1,k} + T_{n-1,k+1} & , n > k. \end{cases}$$

Для треугольника Каталана 2 также справедлива формула, основанная на композициях

$$T(n,k) = \sum_{\pi_k \in P_n} C(\lambda_1)\, C(\lambda_2) \ldots C(\lambda_k),$$

где $\pi_k = \{\lambda_1 + \lambda_2 + \cdots + \lambda_k\}$ – композиция числа n имеющая ровно k частей, P_n – множество композиций числа n. $C(n) - n$_ое число Каталана. Суммирование ведется по всем композициям π_k множества P_n. Как видно из формулы, треугольники Каталана 1 и 2 различны. В первом треугольнике учитывается нулевой член последовательности, а во втором не учитывается.

Для треугольника Каталана 2 имеется явная формула

$$T(n,k)=\frac{k}{n}\binom{2n}{n-k}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Рассмотрим пример, пусть треугольник Каталана 2 описывает упорядоченный лес из полных двоичных деревьев

   Пусть кортеж деревьев

   $T(n,k)=\{T_1,T_j,\dots T_k\}$

   Правила построения:

1) *n=k*

   T={(()()), (()()), …,(()())}

2) T(n-1,k-1)

   {(()()),T(n-1,k-1)

3) 2T(n-1,k)

   $\{((T_1)()),T_2,\dots,T_k\}$

   $\{(()(T_1)),T_2,\dots,T_k\}$

4) T(n-1,k+1)

   $\{((T_1)(T_2)),T_3,\dots,T_k\}$

   Все упорядоченные леса полных двоичных деревьев для T(4,2):

   [['(()())'], ['(((()())())())']]

   [['(()())'], ['(()((()())()))']]

   [['(()())'], ['((()(()()))())']]

   [['(()())'], ['(()(()(()())))']]

   [['(()())'], ['((()())(()()))']]

   [['((()())())'], ['((()())())']]

   [['(()(()()))'], ['((()())())']]

   [['((()())())'], ['(()(()()))']]

   [['(()(()()))'], ['(()(()()))']]

   [['(((()())())())'], '(()())']

   [['(()((()())()))'], '(()())']

   [['((()(()()))())'], '(()())']

[['(()(()(()()))'], '(()())']

[['((()())(()()))'], '(()())']

2. Сгенерировать все пути, имеющие (2*n*+1)-шагов из (0,0) в (2*n*+1,2*k*+1), используя шаги *u*=(1,1) и *d*=(1,-1) где все пути находятся в неотрицательной плоскости.

   Например,

   T(2,0)=5 имеем {uuudd}, {uudud}, {uuddu}, {uduud}, {ududu};

   T(2,1)=4 имеем {uuuud}, {uuudu}, {uuduu}, {uduuu};

   T(2,2)=1 имеем {uuuuu}.

3. Сгенерировать все пути из (0,0) в (*n*,*k*) такие, что не опускаются ниже *y*=0 и используются шаги *U*=(1,1), *D*=(1,-1) два типа шагов *H*=(1,0) (имеет два цвета)

   Например, T(3,1)=14

   UDU, UUD,

   4 HHU paths,

   4 HUH paths

   4 UHH paths.

11.21 Генерация множеств, заданных треугольником Моцкина

Для треугольника Моцкина имеется замечательная формула

$$T(n,k) = \sum_{\pi_k \in P_n} M(\lambda_1)\, M(\lambda_2) \dots M(\lambda_k),$$

где $\pi_k = \{\lambda_1 + \lambda_2 + \cdots + \lambda_k\}$ – композиция числа *n* имеющая ровно *k* частей, P_n – множество композиций числа *n*. $\mathrm{M}(n)$ – число Моцкина. Суммирование ведется по всем композициям π_k множества P_n.

Например,

$$T(n,2) = \sum_{\lambda_1+\lambda_2=n} M(\lambda_1)\, M(\lambda_2)$$

Пусть *n*=5, тогда композиций будет 4:

{1+4}

{2+3}

{3+2}

{4+1}

Откуда

$$T(5{,}3) = M(1)M(4) + M(2)M(3) + M(3)M(2) + M(4)M(1)$$

Зная, что $C(1) = 1$, $C(2) = 1, C(3) = 2$, $C(4) = 4$ получим

$$T(5{,}2) = 4 + 2 + 2 + 4 = 12.$$

T(5,1)

0 /----

1 //\--

2 /-/\-

3 //-\-

4 /--/\

5 //\/\

6 /-/-\

7 //--\

8 ///\\

T(5,2)

0 /---/

1 //\-/

2 /-/\/

3 //-\/

4 /--/-

5 //\/-

6 /-/--

7 //---

8 ///\-

9 /-//\

10 //-/\

11 ///-\

T(5,3)

0 /--//

1 //\//

2 /-/-/

3 //--/

4 ///\/

5 /-//-

6 //-/-

7 ///--

8 ////\

T(5,4)

0 /-///

1 //-//

2   ///-/

4 ////-

Задание. Построить алгоритм задачи, записанной выше.

11.22 Генерация множеств, описываемых треугольниками Шредера

Для треугольника заданного малыми числами Шредера также имеется формула, основанная на композициях

$$T(n,k) = \sum_{\pi_k \in P_n} s(\lambda_1)\, s(\lambda_2) \dots s(\lambda_k),$$

где $\pi_k = \{\lambda_1 + \lambda_2 + \dots + \lambda_k\}$ – композиция числа n имеющая ровно k частей, P_n – множество композиций числа n. $\mathrm{s}(n) -$ малые числа Шредера. Суммирование ведется по всем композициям π_k множества P_n.

Например,

$$T(n,2) = \sum_{\lambda_1+\lambda_2=n} s(\lambda_1)\, s(\lambda_2)$$

Пусть n=5, тогда композиций будет 4:

{1+4}

{2+3}

{3+2}

{4+1}

Для данного треугольника чисел существует рекуррентная формула

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k \\ T_{n-1,k-1} + \sum_{i=1}^{n-k} T_{n,k+i}, & n > k. \end{cases},$$

Явная формула имеет вид

$$T_{n,k} = \frac{k}{n} \sum_{j=0}^{n-k} \binom{n}{j} 2^{n-k-j} (-1)^j \binom{2n-k-j-1}{n-1}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество лесов деревьев.

['()', '(()(()()))']

['()', '((()())())']

['()', '(()()())']

['(()())', '(()())']

['(()(()()))', '()']

['((()())())', '()']

['(()()())', '()']

2. Множество строк с расстановкой скобок

   ['x', '(x(xx))']

   ['x', '((xx)x)']

   ['x', '(xxx)']

   ['(xx)', '(xx)']

   ['(x(xx))', 'x']

   ['((xx)x)', 'x']

   ['(xxx)', 'x']

11.23 Генерация множеств, описываемых числами Нараяна

Задание. Построить алгоритм задачи, записанной ниже.

1. Генерация дорожек Дика длинной *n* с заданным числом пиков *k*.

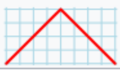
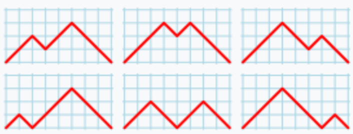
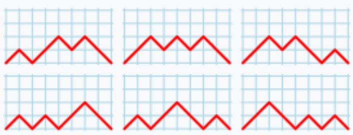
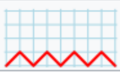
$N(4,k)$	Пути
$N(4,1)=1$ путь с одним максимумом:	
$N(4,2)=6$ путей с двумя максимумами:	
$N(4,3)=6$ путей с тремя максимумами:	
$N(4,4)=1$ путь с четырьмя максимумами:	

Рис. 33. Пример дорожек Дика *n*=4.

2. Генерация непересекающихся разбиений множества n, имеющих ровно k блоков. Например, N

3. Генерация 123-перестановок с k

11.24 Генерация множеств, заданных треугольником Риордана

Для треугольника заданного числами Риордана имеется формула, основанная на композициях

$$T(n,k)=\sum_{\pi_k\in P_n} r(\lambda_1)\, r(\lambda_2) \dots r(\lambda_k),$$

где $\pi_k=\{\lambda_1+\lambda_2+\cdots+\lambda_k\}$ – композиция числа n имеющая ровно k частей, P_n – множество композиций числа n. $\mathrm{r}(n)$ — числа Риордана. Суммирование ведется по всем композициям π_k множества P_n.

Например,

$$T(n,2)=\sum_{\lambda_1+\lambda_2=n} r(\lambda_1)\, r(\lambda_2)$$

Пусть n=5, тогда композиций будет 4:

{1+4}

{2+3}

{3+2}

{4+1}

Для данного треугольника чисел существует рекуррентная формула

$$T_{n,k}=\begin{cases}0, & n<k, n<0, k<0,\\ 1, & n=k\\ T_{n-1,k-1}+\sum\limits_{i=1}^{n-k-1} T_{n,k+i}\,, & n>k.\end{cases},$$

Явная формула имеет вид

$$T_{n,k} = \frac{k}{n}\sum_{j=0}^{n-k}\binom{n}{j+k}(-1)^{n-k-j}\binom{2j+k-1}{j}$$

Связь с числами Моцкина

$$M_n = \sum_{k=0}^{n} T_{n,k}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество лесов деревьев Риордана (см. раздел 11.16)

   Например,

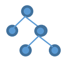
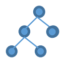
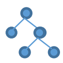
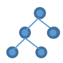
1		
2		
3		
4		
5		
6		
7		

Таблица лесов деревьев Риордана T(6,2) из 6 узлов, 2 дерева.

2. Множество дорожек Моцкина T(n,k), состоящий из n шагов и имеющие ровно k горизонтальных шагов уровня 0

   Например, 4 шага

0	1	2	3	4

//\\\\, /\\/\\ , /__\\	_/_\\, /__	/__, __/\\		_____

11.25 Генерация множеств, заданных треугольником Фибоначчи.

Для треугольника Фибоначчи выполняется формула основанная на композициях

$$T(n,k) = \sum_{\pi_k \in P_n} F(\lambda_1)\,F(\lambda_2)\dots F(\lambda_k),$$

где $\pi_k = \{\lambda_1 + \lambda_2 + \cdots + \lambda_k = n\}$ – композиция числа n имеющая ровно k частей, P_n – множество композиций числа n. $\mathrm{F}(n) -$ числа Фибоначчи. Суммирование ведется по всем композициям π_k множества P_n. A037027.

Для треугольника Фибоначчи имеется явная формула

$$T(n,k) = \sum_{i=0}^{n-k} \binom{k+i}{k} \binom{i}{n-i-k}$$

Имеется также рекуррентная формула

$$T_{n,k} = \begin{cases} 0, & n < k, n < 0, k < 0, \\ 1, & n = k \\ T_{n-1,k-1} + T_{n-1,k} + T_{n-2,k}, & n > k. \end{cases},$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Получить все пути на решетке из точки (0,0) в точку (n,k) используя шаги (0,1), (1,0), (2,0).

   Например, T(4,2)=5

   ['-|', '-', '-', '-|']

   ['-|', '--', '-| ']

   ['-|', '-', '-|', '-']

   ['-|', '-|', '-', '-']

   ['-|', '-|', '--']

11.26 Генерация множеств, заданных треугольником Фина

Задание. Построить алгоритм задачи, записанной ниже.

Явная формула для чисел Фина имеет вид

$$E(n,m)=\frac{m}{n}\sum_{k=0}^{\frac{n-m}{2}}\binom{n}{k}\binom{n-m-k-1}{n-m-2k}2^{n-m-2k}$$

T(n,k) = T(n-1,k-1) + Sum_{i>=0} T(n-1,k+1+i)*2^i.

См. работу [30].

11.27 Пирамида Нараяны

Пирамида Нараяны $N_p(n,m,k)$ имеет три параметра и задается следующей формулой, основанной на композициях и треугольнике Нараяны $N(n,m)$ [22].

$$N_p(n,m,k)=\sum_{\lambda_1+\lambda_2+\cdots+\lambda_k=n}\ \sum_{\mu_1+\mu_2+\cdots+\mu_k=m}\ \prod_{i=1}^{k}N(\lambda_i,\mu_i)$$

Рекуррентная формула для пирамиды Нараяны

$$N_p(n,m,k)=\begin{cases}0, & n<0, m<0, k<0, n<m, m<k,\\ 1, & n=m=k,\\ N_p(n-1,m-1,k-1)+N_p(n-1,m-1,k)+ & \\ \quad +N_p(n-1,m,k)+N_p(n-1,m,k+1) & \end{cases}$$

Явная формула пирамиды Нараяны

$$N_p(n,m,k)=\frac{k}{n}\binom{n}{m}\binom{n}{m-k}$$

Связь с треугольником Каталана

$$\sum_{m=1}^{n}N_p(n,m,k)=T_2(n,k)$$

Где $T_2(n,k)$ – треугольник Каталана 2.

Эта формула обеспечивает связь между числами Каталана и пирамидой Нараяны, которая позволяет генерировать классы подмножеств, описываемых числами Каталана.

Задание. Построить алгоритм задачи, записанной ниже.

1. Получить все k- множества путей Дика имеющих в сумме m пиков и 2n шагов.

   Указание.

   $N_p(n-1, m-1, k-1)$ – добавить множество с одним пиком и 2 шагами [UD],s.

   $N_p(n-1, m-1, k)$ – добавить в первое множество с одним пиком и 2 шагами UDs.

   $N_p(n-1, m, k)$ – добавить в первое множество 2 шага UsD.

   $N_p(n-1, m, k+1)$ – добавить во второе один шаг и объединить первое и второе подмножества s1+Us2S.

2. Получить все дорожки из точки (0,0) с шагами (-1,0) L, (1,0) – R, (0,1)- U, (0,-1)-D в точки (x,k-1),

   Например,

   1) T(3,1,2) = 3: 3 дорожки (0,0) в (-2,1) варианты {LLU, LUL, ULL}.

   2) T(3,2,2) = 8: 8 дорожек (0,0) в (0,1) варианты {UDU, UUD, ULR, URL, RLU, LRU, RUL LUR}.

   3) T(3,3) = 3: дорожки из (0,0) в (2,1) варианты {RRU, RUR, URR}.

3. Получить все варианты двух множеств путей Дика, имеющих 1 или 2 пика и $2n$ шагов.

4. Получить все варианты k-множеств путей Дика, имеющих 1 или 2 пика и $2n$ шагов.

5. Получить все варианты k-множеств путей Дика, не имеющих пика 1 и каждый путь состоит из $2n$ шагов.

6. Получить все варианты k-множеств путей Дика, не имеющих пиков 1, 3 и каждый путь состоит из $2n$ шагов.

11.28 Пирамида Эйлера первого рода

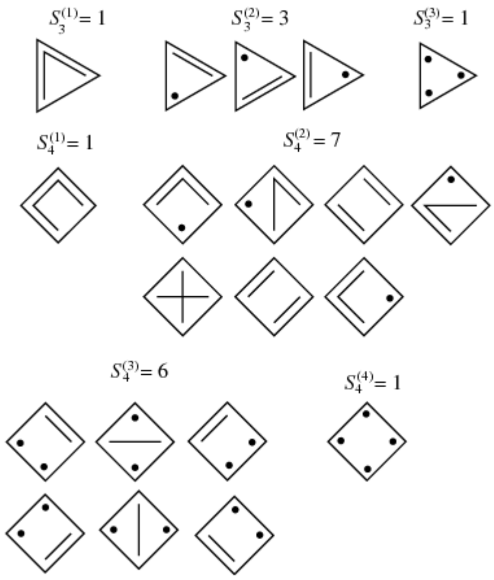
Задание. Построить алгоритм задачи, записанной ниже.

$$E(n,m,k) = \begin{cases} 0, & n < m, \quad m < 0, \quad n < 0 \\ k, & n = k \\ (k+m)\cdot E(n-1,m,k) + (n-m+1)\cdot E(n-1,m-1,k) & \end{cases}$$

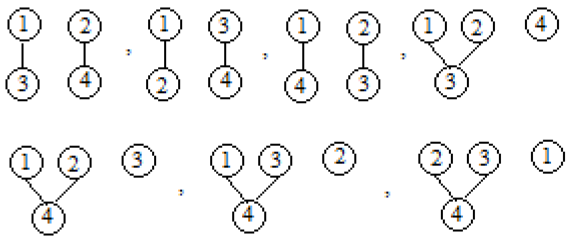
11.29 Множества, описываемые числами Стирлинга второго рода.

Задание. Построить алгоритм задачи, записанной ниже.

1. Задача о матрешках. Имеется *n* матрешек, необходимо получить *k* матрешек $k \le n$.
2. Множество диаграмм Дикау (Dickau)



3. Множество *k* стопок игральных карт колодой *n*.
4. Множество неупорядоченных лесов *k* помеченных деревьев из n вершин, у которых метка родителя больше чем метки всех его потомков высотой 0 и 1. <u>Например</u>, все леса *n*=4, *k*=2



11.30 Множества, описываемые числами Стирлинга первого рода

Числа Стирлинга первого рода имеют рекуррентную формулу

$$s(n,k)=\begin{cases}0, & n<k, \quad k<0, \quad n<0\\ 1, & n=k\\ (n-1)\cdot s(n-1,k)+s(n-1,k-1) & \end{cases}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество лесов k помеченных деревьев из (n+1) вершин, у которых метка родителя больше чем метки все его потомков.
2. Множество перестановок, имеющих $к$ циклов

11.31 Множества, описываемые треугольником Эйлера первого рода

Треугольник Эйлера первого рода описывается рекуррентным уравнением вида

$$E(n,k)=\begin{cases}0, & n<k, \quad k\le 0, \quad n\le 0\\ 1, & n=1, k=1\\ k\cdot E(n-1,k)+(n-k+1)\cdot E(n-1,k-1) & \end{cases}$$

Задание. Построить алгоритм задачи, записанной ниже.

Множества перестановок размерности n имеющие k подъемов

$E(4,1)$	$E(4,2)$	$E(4,3)$	$E(4,4)$
[4, 3, 2, 1]	[3, 4, 1, 2] [3, 1, 4, 2] [4, 1, 3, 2] [1, 4, 3, 2]	[4, 1, 2, 3] [1, 4, 2, 3] [1, 2, 4, 3] [3, 4, 1, 2]	[1, 2, 3, 4]

	[4, 2, 3, 1] [2, 4, 3, 1] [2, 4, 1, 3] [2, 1, 4, 3] [3, 4, 2, 1] [3, 2, 4, 1] [3, 2, 1, 4]	[3, 1, 2, 4] [1, 3, 4, 2] [1, 3, 2, 4] [2, 3, 4, 1] [2, 3, 1, 4] [2, 4, 1, 3] [2, 1, 3, 4]	

11.32 Множества, описываемые треугольником Эйлера второго рода

Треугольник Эйлера второго рода описывается рекуррентным уравнением вида

$$E_2(n,k) = \begin{cases} 0, & n < k, \quad k < 0, \quad n < 0, \\ 1, & n = 1, k = 1, \\ k \cdot E_2(n-1,k) + (2n-k) \cdot E_2(n-1,k-1). & \end{cases}$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Перестановки Стирлига *n*, имеющие *k* падений.

   Например, все перестановки *n*=3

$E_2(3,1)$	$E_2(3,2)$	$E_2(3,3)$
[1, 1, 2, 2, 3, 3]	[2, 2, 3, 3, 1, 1] [2, 2, 1, 1, 3, 3] [1, 2, 2, 3, 3, 1] [1, 2, 2, 1, 3, 3] [3, 3, 1, 1, 2, 2] [1, 3, 3, 1, 2, 2] [1, 1, 3, 3, 2, 2] [1, 1, 2, 3, 3, 2]	[3, 3, 2, 2, 1, 1] [2, 3, 3, 2, 1, 1] [2, 2, 1, 3, 3, 1] [3, 3, 1, 2, 2, 1] [1, 3, 3, 2, 2, 1] [1, 2, 3, 3, 2, 1]

11.33 Множества, описываемые тангенциальным числами

Тангенциальные числа описаны в работе [15].

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество змеевых перестановок

a) a<b>c<…>x

0 [1, 5, 2, 4, 3]
1 [2, 5, 1, 4, 3]
2 [3, 5, 1, 4, 2]
3 [4, 5, 1, 3, 2]
4 [1, 5, 3, 4, 2]
5 [2, 5, 3, 4, 1]
6 [3, 5, 2, 4, 1]
7 [4, 5, 2, 3, 1]
8 [1, 3, 2, 5, 4]
9 [1, 4, 2, 5, 3]
10 [1, 4, 3, 5, 2]
11 [2, 4, 3, 5, 1]
12 [2, 3, 1, 5, 4]
13 [2, 4, 1, 5, 3]
14 [3, 4, 1, 5, 2]
15 [3, 4, 2, 5, 1]

b) a>b<c>…<x

0 [2, 1, 4, 3, 5]
1 [3, 1, 4, 2, 5]
2 [4, 1, 3, 2, 5]
3 [5, 1, 3, 2, 4]
4 [2, 1, 5, 3, 4]
5 [3, 1, 5, 2, 4]
6 [4, 1, 5, 2, 3]
7 [5, 1, 4, 2, 3]
8 [3, 2, 4, 1, 5]
9 [3, 2, 5, 1, 4]
10 [4, 2, 5, 1, 3]
11 [4, 3, 5, 1, 2]

12 [4, 2, 3, 1, 5]
13 [5, 2, 3, 1, 4]
14 [5, 2, 4, 1, 3]
15 [5, 3, 4, 1, 2]

2. Генерация полных двоичных возрастающих деревьев (2n-1) узлов

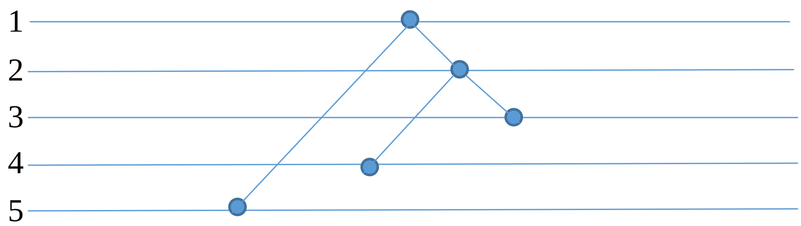


Рис. 34. Полное двоичное возрастающее дерево

11.34 Множества, описываемые треугольником Эйлера-Бернулли

Для треугольника Эйлера-Бернулли имеется рекуррентная формула

$$T(n,m) = \begin{cases} 0, & n < 0, m < 0, n < m \\ 1, & n = 0, \quad m = 1 \\ \sum_{k=0}^{m-1} T(n-1,k), & n \text{ нечетное} \\ \sum_{k=0}^{n-m} T(n-1,k+m), & n \text{ четное} \end{cases}$$

Также существует явная формула, связанная с числами Эйлера

$$T(n,m) = \left| \sum_{k=0}^{m} \binom{m}{k} E(n-m+k) \right|$$

Задание. Построить алгоритм задачи, записанной ниже.

1. Множество зигзаг перестановок (n+1), имеющие последним число (m+1). <u>Например</u>,

   Все зигзаг перестановки T(4,0).

[4, 5, 2, 3, 1]

[3, 5, 2, 4, 1]

[2, 5, 3, 4, 1]

[3, 4, 2, 5, 1]

[2, 4, 3, 5, 1]

Все зигзаг перестановки T(4,1).

[4, 5, 1, 3, 2]

[3, 5, 1, 4, 2]

[1, 5, 3, 4, 2]

[3, 4, 1, 5, 2]

[1, 4, 3, 5, 2]

Все зигзаг перестановки T(4,2).

[2, 5, 1, 4, 3]

[1, 5, 2, 4, 3]

[2, 4, 1, 5, 3]

[1, 4, 2, 5, 3]

Все зигзаг перестановки T(4,3).

[2, 3, 1, 5, 4]

[1, 3, 2, 5, 4]

2. Множество двоичных возрастающих деревьев, с самым правым листом равным (m+1)

12 Пример реализации алгоритмов комбинаторной генерации для множеств, описываемых числами Каталана.

Задание. Построить алгоритм задачи, записанной ниже.

Числом Каталана называется число различных правильных скобочных структур из n пар скобок.

Например, все правильные скобочные структуры из 3 пар скобок:

$$()()(),\quad ()(()),\quad (()()),\quad (())(),\quad ((())).$$

Первые 11 членов последовательности чисел Каталана (последовательность A000108 в онлайн-энциклопедии целых последовательностей) следующая:

$$1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, \ldots$$

Имеется выражение для чисел Каталана, записанное в явном виде:

$$C_n = \frac{1}{n+1}\binom{2n}{n}.$$

В рекуррентном виде:

$$C_n = \begin{cases} 1, & n = 0, \\ \sum_{i=0}^{n-1} C_i C_{n-i-1}, & n > 0. \end{cases}$$

Представленное рекуррентное выражение является рекурсивной функцией алгебры $\{N,+,\times,R\}$, где N – множество натуральных чисел, R – оператор рекурсии. Тогда для данной рекуррентной формулы C_n можно построить схему рекурсивной композиции дерева И/ИЛИ. Правила построения следующие:

1. Корнем будет узел C_{n+1}, это будет ИЛИ узел, у которого ровно (n+1) сыновей.

2. Каждый i-й сын T^n_i ИЛИ-узла является И-узлом, имеющим двух сыновей и соответствующих поддеревьям C_i и C_{n-i}.

Схема рекурсивной композиции представлена на рис 35.

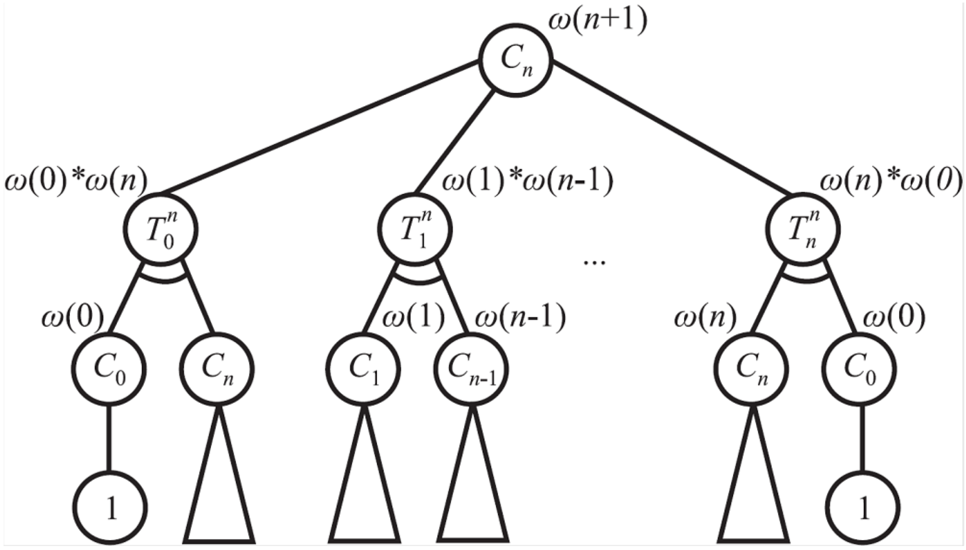


Рис. 35. Схема рекурсивной композиции для рекуррентной формулы для чисел Каталана.

Вариантом для дерева И/ИЛИ, основанного на данной схеме рекурсивной композиции, будет являться двоичное дерево, т.к. все И-узлы имеют двух сыновей. Для получения скобочной записи необходимо построить соответствующий алгоритм. Правило построения следующее: (см. рис 36) в левостороннем обходе варианта при просмотре каждого левого сына, кроме листа, при входе печатать открывающую скобку, при возврате - печатать закрывающую скобку.

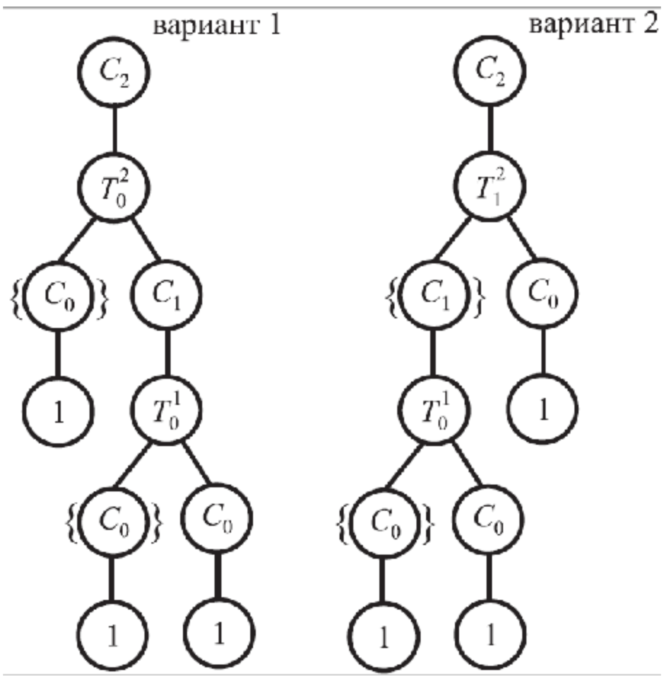


Рис. 36. Варианты для дерева И/ИЛИ

В основе этого алгоритма лежит алгоритм генерации варианта по схеме рекурсивной композиции, представленной для чисел Каталана (рис. 37). Корнем дерева является ИЛИ-узел, помеченный как C_n, число вариантов равно $\omega(n)$. Поэтому первым шагом определяем номер ветви и узел T_i^n, используя алгоритм определения сына ИЛИ-узла по номеру *num*. Каждый узел T_i^n является И-узлом и содержит два сына. Используя алгоритм вычисления номеров сыновей для И-узла, определяем номера l_1 и l_2 для сыновей C_i и C_{n-i-1}. Далее производим рекурсивный спуск для сыновей C_i и C_{n-i-1}

Algorithm 1: Алгоритм GenCatalan для генерации множеств Каталана.

algorithm `GenCatalan` (num,n) **begin**
    if $n = 0$ **then return**

    S:=0
    for ($i := 0$ *to* $n - 1$) **do**
        if $num < S + C_i \cdot C_{n-1-i}$ **then** {определяем T_i^n}
            num:=num-S {вычисляем l_1 и l_2 для узлов C_i и C_{n-i} }
            $l_1 := num \mod C_i$
            print("(") { печатаем открывающую скобку }
            `GenCatalan` (l_1,i) { совершаем рекурсивный спуск для левого сына}
            print(")") { печатаем закрывающую скобку }
            $l_2 := num/C_i$
            `GenCatalan` ($l_2, n - 1 - i$) { совершаем рекурсивный спуск для правого сына}
            return
        end
        $S := S + C_i \cdot C_{n-1-i}$
    end
end

Рис. 37. Алгоритм генерации скобочной записи.

Для построения алгоритма нумерации необходимо анализировать строку, состоящую из скобок. Считаем, что скобочное представление правильное. Для этого используем функцию match(s), которая проверяет текущий символ строки и если он равен символу s, то двигает входной указатель строки на следующий символ и возвращает номер позиции символа в строке, в противном случае возвращает -1 (рис. 38).

Algorithm 2: Алгоритм нумерации элементов множеств Каталана.

algorithm `RankCatalan` (n) **begin**

$poso$:=match("(")

if $poso$=-1 **then return** 0

l_1:=`RankCatalan` (n-1)

$posz$:=match(")")

$k := (posz - poso - 1)/2$

l_2:=`RankCatalan` (n-1)

$S := 0$

for $(i := 0$ *to* $k - 1)$ **do** $S := S + C_i * C_{k-i+1}$

return $S + l_1 + l_2 \cdot C_k$

end

Рис. 38. Алгоритм нумерации скобочной записи.

В таблице представлена биекция между строками правильной скобочной записи и натуральными числами для n=4, C_4=14.

Таблица соответствия между номером и скобочной строкой

Номер строки (num)	Скобочная запись
0	()()()()
1	()()(())
2	()(())()
3	()(()())
4	()((()))
5	(())()()
6	(())(())
7	(()())()
8	((()))()
9	(()()())
10	(()(()))
11	((())())
12	((()()))
13	(((())))

Полученные алгоритмы GenCatalan и RankCatalan можно использовать для построения алгоритмов комбинаторной генерации для других множеств, описываемых числами Каталана. В настоящее время известно свыше 300 множеств [6].

Заключение

Предлагаемый задачник обеспечивает преподавателей, студентов и программистов широким спектром заданий разных уровней, а в некоторых случаях имеющих научную ценность. Многие представленные задачи не требуют специальных знаний и вполне могут использоваться на начальных курсах бакалавриата по направлению информатики и программирования.

Наличие сложных задач, связанных с комбинаторной генерацией, требуют подготовки в области комбинаторики и рекурсивного программирования. Наличие списка литературы позволяет восполнить соответствующие пробелы знаний и успешно решать предлагаемые задания. Важно также отметить, что все задачи на комбинаторную генерацию имеют математические описания в онлайн энциклопедии числовых последовательностей (приложение 1).

Освоив методологию решения задач комбинаторной генерации, решив несколько задач на эту тему, студент может решать более трудные задачи самостоятельно. Огромное число таких задач имеется в онлайн-энциклопедии целых последовательностей. Кроме того, алгоритмы комбинаторной генерации могут быть использованы при решении задач дискретной оптимизации, организации ассоциативной памяти, кодирования и защиты сложных информационных объектов.

Для дальнейшего повышения компетенций по информатике и программированию предлагается ознакомиться с другими задачниками [26-29].

Литература

1.Гашков С. Б. Системы счисления и их применение / С. Б. Гашков :

2-е изд., испр. и доп. — М. : изд-во МЦНМО, 2012.— 68 с. : ил.

2. М. А. Берштейн, Г. А. Мерзон, Диаграммы Юнга, пути на решётке и метод отражений, Матем. просв., сер. 3, 18, Изд-во МЦНМО, М., 2014, 112–141

3. М.Гарднер. Числа Каталана. Квант, 1978, №7. – С.21-26

4. Виленкин Н. Я. Комбинаторика. М.: Наука, 1969. – 328 с

5. Дональд Э. Кнут Искусство программирования, том 4, выпуск 4. Генерация всех деревьев. История комбинаторной генерации-Изд-во Вильямс. – 160с.

6. Richard Stanley, Catalan numbers, Cambridge Universiy Press (2015), with an appendix by Igor Pak. – 222p

7. T. Koshy, Catalan Numbers with Applications, Oxford University Press, 2009. – 440 p

8. Кручинин, В.В. Комбинаторика композиций и ее приложения Изд-во: Томск: В-Спектр, 2010 г. – 156 с

9. Agarwal, R.P., Agarwal, H. & Sen, S.K. Birth, growth and computation of pi to ten trillion digits. Adv Differ Equ 2013, 100 (2013). https://doi.org/10.1186/1687-1847-2013-100

10. Гик Е.Я. Математика на шахматной доске. – Изд-во: Астрель, 2009. – 321с.

11. Cheon G. S., Lee S. G., Shapiro L. W. The Fine numbers refined //European Journal of Combinatorics. – 2010. – Т. 31. – №. 1. – С. 120-128.

12. Дональд Кнут, Роналд Грэхем, Орен Паташник. Конкретная математика. Математические основы М.: «Вильямс», 2009. — 784 с.

13. Silvia Heubach and Toufik Mansour, Combinatorics of Compositions and Words, CRC Press, 2010. – 504 p

14. Richard P. Stanley, Enumerative Combinatorics, Wadsworth, Vol. 1, 1986; see Example 3.15.10, p. 146.

15. В. И. Арнольд, “Исчисление змей и комбинаторика чисел Бернулли, Эйлера и Спрингера групп Кокстера”, УМН, 47:1(283) (1992), 3–45; 47:1 (1992), 1–51

16. Кручинин В.В. Методы построения алгоритмов генерации и нумерации комбинаторных объектов на основе деревьев И/ИЛИ Томск: Изд-во «В-Спектр», 2007. – 200с.

17. Кнут Д.Э. Искусство программирования для ЭВМ. Том 1. Основные алгоритмы - М.: Мир, 1976. - 735 с.

18. Кнут Д.Э. Искусство программирования для ЭВМ. Том 2. Получисленные алгоритмы - М.: Мир, 1977. - 724 с.

19. Кнут Д.Э. Искусство программирования для ЭВМ. Том 3. Сортировка и поиск - М.: Мир, 1978. - 844 с.

20. Кнут Д. Э. Искусство программирования, том 4, A. Комбинаторные алгоритмы, часть 1 = The Art of Computer Programming, Volume 4A: Combinatorial Algorithms, Part 1 / под ред. Ю. В. Козаченко. — 1. — Москва: Вильямс, 2013. — Т. 4. — 960 с.

21. Frank R. Bernhart, Catalan, Motzkin, and Riordan numbers, Discrete Mathematics, Volume 204, Issues 1–3, 1999, P. 73-112.

22. Kruchinin Dmitry, Kruchinin Vladimir, Shablya Yuriy On some properties of generalized Narayana numbers// Quaestiones Mathematicae 44(9), 2021- 15pp

23. Shapiro, Louis W.; Sulanke, Robert A. (2000), "Bijections for the Schröder numbers", Mathematics Magazine, 73 (5): 369–376,

24. В. В. Доценко Числа Каталана и естественные отображения// Летние конференции Турнира городов: Избранные материалы. Вып. 1/ Под общ. ред. Н. Н. Константинова. Сост. Б. Р. Френкин. — М.: МЦНМО, 2009. — 264с (139-165).

25. Кручинин, В. В. Метод кодирования информационных объектов на основе деревьев И/ИЛИ / В. В. Кручинин Б. А. Люкшин // Доклады ТУСУР. - 2010. - № 1(21), ч. 1. - С. 170-172.

26. Абрамов С. А., Гнездилова Г. Г., Капустина Е. Н. и др. Задачи по программированию. М.: Наука, 1988.

27. Златопольский Д. М. Задачи по программированию. 7–11 классы: книга для учителя. М.: Первое сентября, 2000.

28. Златопольский Д. М. Основы программирования на языке Python. 2-е изд. М.: ДМК-Пресс, 2018. 4. Пильщиков В. Н. Сборник упражнений по языку Паскаль. М.: Наука, 1988. 5. Светозарова Г. И., Мельников А. А., Козловский А. В. Практикум по программированию на языке Бейсик. М.: Наука, 1988.

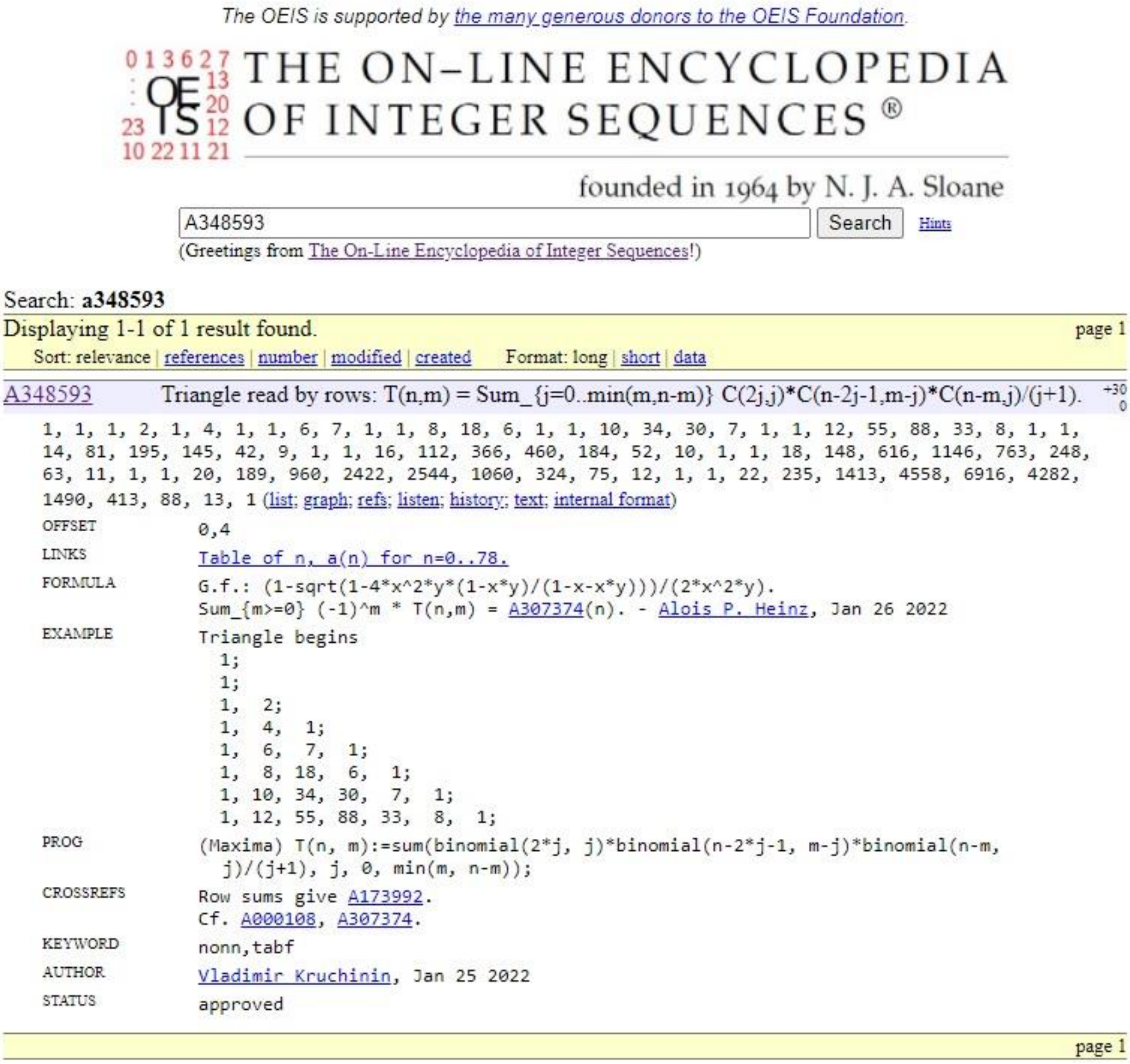
29. Златопольский Д. 1400 задач по программированию. – М.: ДМК Пресс, 2020. – 192 с.: ил.

30. E. Deutsch and L. Shapiro, A survey of the Fine numbers, Discrete Math., 241 (2001), 241-265.

Приложение 1

Онлайн энциклопедия целых последовательностей

Онлайн энциклопедия целых последовательностей (www.oeis.org) предназначена для математиков, инженеров и студентов, которые используют эту энциклопедию в своей работе при проведении исследований и учебных занятий. На рис. 39 показан пример статьи с номером A348593.



The OEIS is supported by the many generous donors to the OEIS Foundation.

THE ON-LINE ENCYCLOPEDIA OF INTEGER SEQUENCES ®

founded in 1964 by N. J. A. Sloane

A348593 Search Hints

(Greetings from The On-Line Encyclopedia of Integer Sequences!)

Search: **a348593**

Displaying 1-1 of 1 result found. page 1

Sort: relevance | references | number | modified | created Format: long | short | data

A348593 Triangle read by rows: T(n,m) = Sum_{j=0..min(m,n-m)} C(2j,j)*C(n-2j-1,m-j)*C(n-m,j)/(j+1). +30 0

1, 1, 1, 2, 1, 4, 1, 1, 6, 7, 1, 1, 8, 18, 6, 1, 1, 10, 34, 30, 7, 1, 1, 12, 55, 88, 33, 8, 1, 1, 14, 81, 195, 145, 42, 9, 1, 1, 16, 112, 366, 460, 184, 52, 10, 1, 1, 18, 148, 616, 1146, 763, 248, 63, 11, 1, 1, 20, 189, 960, 2422, 2544, 1060, 324, 75, 12, 1, 1, 22, 235, 1413, 4558, 6916, 4282, 1490, 413, 88, 13, 1 (list; graph; refs; listen; history; text; internal format)

OFFSET 0,4

LINKS Table of n, a(n) for n=0..78.

FORMULA G.f.: (1-sqrt(1-4*x^2*y*(1-x*y)/(1-x-x*y)))/(2*x^2*y).

Sum_{m>=0} (-1)^m * T(n,m) = A307374(n). - Alois P. Heinz, Jan 26 2022

EXAMPLE Triangle begins

```
  1;
  1;
  1,  2;
  1,  4,  1;
  1,  6,  7,  1;
  1,  8, 18,  6,  1;
  1, 10, 34, 30,  7,  1;
  1, 12, 55, 88, 33,  8,  1;
```

PROG (Maxima) T(n, m):=sum(binomial(2*j, j)*binomial(n-2*j-1, m-j)*binomial(n-m, j)/(j+1), j, 0, min(m, n-m));

CROSSREFS Row sums give A173992.

Cf. A000108, A307374.

KEYWORD nonn,tabf

AUTHOR Vladimir Kruchinin, Jan 25 2022

STATUS approved

page 1

Рис. 39 Пример статьи онлайн энциклопедии OEIS.

Каждая статья этой энциклопедии содержит:

1. Название и номер.

2. Числовую последовательность.

3. Комментарии, в которых описаны математические объекты, связанные с этой последовательностью.

4. Формулы, задающие эту последовательность (производящая функция, явная или рекуррентная формула, уравнение и т.д.).

5. Список литературы, связанный с этой последовательностью.

6. Реализацию формул или алгоритмов вычисления этой последовательности в различных системах программирования (Maxima, Mathematica, Maple и др..

7. Среду ссылок на интернет ресурсы.

8. Примеры использования.

9. Связь с другими последовательностями.

10. Автора этой последовательности и авторов отдельных элементов этой последовательности.

Основной функцией этой базы знаний является поиск и редактирование числовых последовательностей, которые могут быть двух типов: линейная последовательность и треугольник.