

Элементарная теория чисел

Делимость

Пусть $a, b \in \mathbb{Z}$. Тогда a — **делитель** b , когда

$$ax = b, x \in \mathbb{Z} \iff a \mid b \iff |a| \leq |b|$$

Отношение делимости *транзитивно*, такое выражение можно *перемножить* с другим:

$$\times \begin{cases} a \mid b \\ c \mid d \end{cases} \implies ac \mid bd$$

Общий делитель чисел делит их *линейную комбинацию*:

$$a \mid b, c \implies a \mid bx + cy, \quad x, y \in \mathbb{Z}$$

Заметим, что $a = bx + cy^*$, когда $(b, c) \mid a$.

Доказательство. Пусть $d := (b, c)$, тогда:

$$d \mid b, c \implies d \mid (bx + cy) \implies d \mid a \quad \blacksquare$$

Коэффициенты Безу (x, y) *неуникальны* и легко выражаются (*доказывается подстановкой в соотношение*):

$$(x + mk, y - ak), \quad k \in \mathbb{Z}$$

* Такое уравнение называют *соотношением Безу*, а x и y — *коэффициентами Безу* (*Э. Безу*).

Наибольший общий делитель

*Наибольший общий делитель** для $\{a_k\}_{k \in \mathbb{N}}$ — такое $\gcd(\{a_k\})$, что

$$\exists d: d \mid \gcd(\{a_k\}) \mid \{a_k\}.$$

Упрощённая запись $\gcd(\{a_k\}) = (\{a_k\})$.

Этот бинарный оператор *коммутативен, ассоциативен и дистрибутивен*.

Наименьшее общее кратное

*Наименьшее общее кратное*** для $\{a_k\}_{k \in \mathbb{N}}$ — такое $\text{lcm}(\{a_k\})$, что

$$\exists m: \{a_k\} \mid \text{lcm}(\{a_k\}) \mid m.$$

Упрощённая запись $\text{lcm}(\{a_k\}) = [\{a_k\}]$.

Этот бинарный оператор *коммутативен и ассоциативен, однако не дистрибутивен*.

ДВОЙСТВЕННОСТЬ

НОД и НОК двойственны друг другу:

$$(a, b) \cdot [a, b] = ab$$

Доказательство. Пусть $m := [a, b]$, тогда:

$$a, b \mid m \iff ab \mid am, bm \iff ab \mid (am, bm) \iff ab \mid (a, b)m$$

Так как $(a, b) \mid [a, b] \mid ab$, то $ab/(a, b) \mid [a, b]$.

Значит, $ab/(a, b) \leq [a, b]$. Но $[a, b]$ — *наименьшее* общее кратное a, b . Следовательно, $ab/(a, b) \nless [a, b]$, поэтому:

$$ab/(a, b) = [a, b] \iff ab = (a, b) \cdot [a, b] \blacksquare$$

* Сокращённо *НОД*, или *Greatest Common Divisor (GCD)*.

** Сокращённо *НОК*, или *Least Common Multiple (LCM)*.

Элементарная алгебра

Свойства неравенств

Отношение сравнения *транзитивно*; неравенства можно *складывать (не вычитать)*, а также *перемножать* и возводить в натуральную степень k (*без смены знака*):

$$\begin{cases} a < b \\ c \leq d \end{cases} \Rightarrow \begin{cases} a + c < b + d \\ ac < bd \\ a^k < b^k \end{cases}$$

При умножении на отрицательное число m знак неравенства *инвертируется*:

$$a < b \iff am > bm$$

Неравенство Коши

Пусть $a, b \in \mathbb{R}^+$. Тогда верно (*О.Л. Коши*):

$$\frac{a+b}{2} \geq \sqrt{ab}$$

Доказательство.

$$\begin{aligned} \frac{a+b}{2} \geq \sqrt{ab} &\iff a+b \geq 2\sqrt{ab} \iff a-2\sqrt{ab}+b \geq 0 \iff \\ &(\sqrt{a}-\sqrt{b})^2 \geq 0 \blacksquare \end{aligned}$$

Неравенство Бернулли

Пусть $n \geq 2$, $x > 0$. Тогда верно (*Я. Бернулли*):

$$(1+x)^n > 1+nx$$

Доказательство. Проверим базис индукции $n = 2$:

$$(1+x)^2 > 1+2x \iff 1+2x+x^2 > 1+2x \quad \square$$

Проверим индукционный шаг $n + 1$. Пусть утверждение верно для некоторого $n > 2$, тогда:

$$\begin{aligned} (1+x)^n > 1+nx &\iff (1+x)^{n+1} > (1+nx)(1+x) \iff \\ (1+x)^{n+1} > 1+(n+1)x+nx^2 &\iff (1+x)^{n+1} > 1+(n+1)x \blacksquare \end{aligned}$$

Свойства функций

Функция f *возрастает*, когда

$$\forall x_1, x_2 \in D_f, x_1 < x_2 \implies f(x_1) < f(x_2).$$

Максимумом функции f называется такая точка x_0 , что

$$\forall \varepsilon > 0 \exists U_\varepsilon(x_0) : \forall x \in U f(x) < f(x_0).$$

Функция f *убывает*, когда

$$\forall x_1, x_2 \in D_f, x_1 < x_2 \implies f(x_1) > f(x_2).$$

Минимумом функции f называется такая точка x_0 , что

$$\forall \varepsilon > 0 \exists U_\varepsilon(x_0) : \forall x \in U f(x) > f(x_0).$$

Функция f *чётна*, когда

$$\forall x \in D_f \implies f(-x) = f(x).$$

Функция f *нечётна*, когда

$$\forall x \in D_f \implies f(-x) = -f(x).$$

Функция f *периодична*, когда

$$\forall x \in D_f \exists T \neq 0 : f(x) = f(x \pm T),$$

где T — **период** функции; наименьший положительный период называется *основным*.

Функция модуля

Абсолютная величина (*модуль*) — чётная функция $f: \mathbb{R} \rightarrow \mathbb{R}_0^+$, которая задаётся формулой:

$$f(x) = |x| = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases}$$

Она *дистрибутивна* относительно умножения, отчасти — относительно сложения: $|a + b| \leq |a| + |b|$.

Степенная функция

Возведение в чётную степень — чётная функция; график — *парабола*:

$$f: \mathbb{R} \xrightarrow{x \mapsto x^n} \mathbb{R}_0^+, n \in \mathbb{N}$$

Обратная функция к $f|_{\mathbb{R}_0^+}$ — **арифметический корень**:

$$f^{-1}: \mathbb{R}_0^+ \xrightarrow{x \mapsto \sqrt[n]{x}} \mathbb{R}_0^+$$

Возведение в нечётную степень — нечётная функция;
график — *кубическая парабола*:

$$g: \mathbb{R} \xrightarrow{x \mapsto x^n} \mathbb{R}, n \in \mathbb{N}$$

Обратная функция к g — **арифметический корень**:

$$g^{-1}: \mathbb{R} \xrightarrow{x \mapsto \sqrt[n]{x}} \mathbb{R}$$

Функция знака

Функция знака (*сигнум-функция*) — нечётная функция $\operatorname{sgn}: \mathbb{R} \rightarrow \{-1; 0; 1\}$, которая определяет знак аргумента:

$$\operatorname{sgn} x = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

Функция натурального логарифма

Функция натурального логарифма — значение интеграла:

$$\ln x = \int_1^x \frac{dt}{t}, \quad \ln: \mathbb{R}^+ \rightarrow \mathbb{R}$$

Свойства:

$$\ln ax = \ln a + \ln x$$

$$\ln x^{\frac{m}{n}} = \frac{m}{n} \ln x$$

Логарифмическая функция

Логарифмическая функция по основанию a — отношение:

$$\log_a x = \frac{\ln x}{\ln a}, \quad \log_a: \mathbb{R}^+ \rightarrow \mathbb{R}, \quad a \neq 1$$

Логарифмические тождества:

$$\log_a a^\beta = \beta \quad a^{\log_a b} = b$$

Переход к новому основанию:

$$\log_a b = \frac{\log_c b}{\log_c a}$$

Свойства:

$$\log_a b = \frac{1}{\log_b a}$$

$$\log_a b \log_c d = \log_c b \log_a d$$

$$\log_a b \log_b c = \log_a c$$

Теория функций

Монотонность

Встречная монотонность. Да.

Теорема. Выражение вида $f(f(x))\dots$

Метод рационализации. Пусть f — монотонно возрастающая функция. Тогда справедливо:

$$f(a) - f(b) \vee 0 \implies a - b \vee 0$$

Его можно применять к отдельному *множителю* или в составе *дроби*.

Теория многочленов

Многочлен

Многочлен $f(x)$ от переменной x — выражение вида:

$$\sum_{k=0}^n a_k x^k$$

Коэффициенты многочлена — действительные числа a_0, \dots, a_n :

$\mathbb{R}[x]$ — *множество* таких многочленов

Старшим называется *коэффициент* многочлена $a_n \neq 0$.

Степень многочлена — натуральное число n :

$$n \equiv \deg f \text{ — обозначение}$$

Виды многочленов:

- *нулевой* — *ноль* ненулевых коэффициентов
- *одночлен* — *один* ненулевой коэффициент
- *двучлен* — *два* ненулевых коэффициента
- *трёхчлен* — *три* ненулевых коэффициента

Деление с остатком

Теорема. Пусть $f(x), g(x)$ — многочлены, причём $g(x)$ ненулевой. Тогда существуют многочлены $u(x)$ и $r(x)$:

$$f(x) = u(x)g(x) + r(x), \quad \deg r < \deg u$$

Схема Горнера — алгоритм, при помощи которого многочлен $f(x)$ можно разделить с остатком на линейный двучлен $x - c$.

Коэффициенты частного рассчитываются по формулам:

$$b_0 = a_0$$

$$b_1 = a_1 + cb_0$$

...

$$b_n = a_n + cb_{n-1}$$

Корни многочлена

Теорема. Значение многочлена $f(x)$ при $x = c$ совпадает с остатком от деления $f(x)$ на $x - c$:

$$f(c) = f(x) \bmod (x - c)$$

Доказательство. По теореме о делении с остатком:

$$f(x) = u(x)(x - c) + r(x) \implies f(c) = r(c) \blacksquare$$

Теорема Безу. Число c — корень многочлена тогда и только тогда, когда $x - c$ делит $f(x)$:

$$f(c) = 0 \iff (x - c) \mid f(x)$$

Доказательство. По определению делимости:

$$(x - c) \mid f(x) \iff f(x) = u(x)(x - c) \implies f(c) = 0 \blacksquare$$

Кратность корня

Корень кратности k многочлена $f(x)$ — такое число c , что:

$$(x - c)^k \mid f(x) \quad (k \text{ максимально})$$

Теорема. Пусть $f(x)$ — многочлен степени $n \geq 1$. Тогда $f(x)$ имеет не более n корней с учётом их кратности.

Доказательство. Докажем по индукции:

- 1) $n = 1 \implies f(x)$ — линейный многочлен \implies 1 корень. \square
- 2) $n > 1 \implies$ допустим, теорема верна для кратности $n - 1$:
 - $> f(x)$ не имеет корней; \square
 - $> f(x)$ имеет корень $c \implies f(x) = (x - c)g(x)$, где $g(x)$ имеет не более $n - 1$ по допущению. \blacksquare

Теорема. Пусть c — кратный корень многочлена $f(x)$. Тогда справедливо:

$$(x - c)^k \mid f(x) \iff (x - c)^{k-1} \mid f'(x)$$

Доказательство \Rightarrow . По дистрибуции производной:

$$\begin{aligned} f(x) = (x - c)^k g(x) &\Rightarrow f'(x) = (x - c)^{k-1} [g(x) + (x - c)g'(x)] \\ &\Rightarrow (x - c)^{k-1} \mid f'(x) \quad \square \end{aligned}$$

Доказательство \Leftarrow . Пусть c — корень кратности m многочлена $f(x)$ и кратности $k - 1$ его производной $f'(x)$.

По прямой теореме:

$$(x - c)^m \mid f(x) \Rightarrow \begin{cases} (x - c)^{m-1} \mid f'(x) \\ (x - c)^{n-1} \mid f'(x) \end{cases} \Rightarrow m = n \quad \blacksquare$$

Рациональные корни

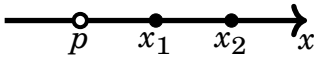
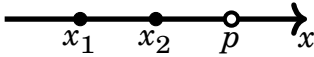
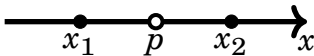
Теорема. Пусть $f(x) \in \mathbb{Z}[x]$. Если несократимая дробь p/q — корень, то справедливо:

$$p \mid a_n, \quad q \mid a_0, \quad p - mq \mid f(m), \quad m \in \mathbb{Z}$$

Доказательство. Да.

Квадратный трёхчлен

Расположение корней относительно числа p :

Расположение корней	Равносильно
	$\begin{cases} D > 0 \\ af(p) > 0 \\ p < x_0 \end{cases}$
	$\begin{cases} D > 0 \\ af(p) > 0 \\ x_0 < p \end{cases}$
	$af(p) < 0$

Метод неопределённых коэффициентов

Метод неопределённых коэффициентов применяется для вычисления коэффициентов выражений, вид которых *заранее известен*.

Теорема. Пусть $P(x)$, $G(x)$ — многочлены одинаковой степени. Они *равны*, если:

$$p_i = g_i, \quad \forall i \in [0; n]$$

Формулы Виета

Формулы Виета. Для многочлена n -ой степени и его n корней справедливы соотношения:

$$\begin{aligned} x_1 + \dots + x_n &= -\frac{a_1}{a_0} \\ x_1 x_2 + \dots + x_{n-1} x_n &= \frac{a_2}{a_0} \end{aligned} \quad x_1 \dots x_n = (-1)^n \frac{a_n}{a_0}$$

Идея. Разложим многочлен на *линейные множители* с неизвестными корнями.

После упрощения воспользуемся *методом неопределённых*

коэффициентов.

Избавление от иррациональности

Факт. Дробь с иррациональностью в знаменателе можно представить в виде *комбинации* иррациональностей:

$$\frac{1}{1 + \sqrt{2} + \sqrt{3}} = A + B\sqrt{2} + C\sqrt{3} + D\sqrt{6}$$

$$\frac{1}{2 - \sqrt[3]{2}} = A + B\sqrt[3]{2} + C\sqrt[3]{4}$$

Этот факт имеет сложное доказательство, которое косвенно есть на сайте МГУ.

Модульная арифметика

Конгруэнтность

Два целых числа **конгруэнтны** (*сравнимы*) по модулю m , когда их разность кратна m (*К.Ф. Гаусс*):

$$a \equiv b \pmod{m} \iff m \mid (a - b) \iff a = b + mk, k \in \mathbb{Z}$$

Отношение конгруэнтности *транзитивно*, поэтому числа образуют *систему остаточных классов* \mathbf{Z}_m по модулю m . Например, \mathbf{Z}_3 :

$$\{\dots, -6, -3, \mathbf{0}, 3, 6, \dots\} \text{ класс } r_0$$

$$\{\dots, -5, -2, \mathbf{1}, 4, 7, \dots\} \text{ класс } r_1$$

$$\{\dots, -4, -1, \mathbf{2}, 5, 8, \dots\} \text{ класс } r_2$$

Свойства сравнения

Конгруэнтные числа можно *складывать, перемножать* и передавать *многочлену* $f \in \mathbb{Z}[x]$:

$$\begin{cases} a \equiv b \pmod{m} \\ c \equiv d \pmod{m} \end{cases} \implies \begin{cases} a + c \equiv b + d \pmod{m} \\ ac \equiv bd \pmod{m} \\ f(a) \equiv f(b) \pmod{m} \end{cases}$$

Конгруэнтные числа можно *умножать (делить)* на одно число с *увеличением (сокращением)* модуля:

$$a \equiv b \pmod{m} \iff ad \equiv bd \pmod{md}$$

$$ad \equiv bd \pmod{m} \iff a \equiv b \pmod{\frac{m}{(m,d)}}$$

Из транзитивности делимости следует:

$$a \equiv b \pmod{m}, n \mid m \implies a \equiv b \pmod{n}$$

Признаки делимости

Для вывода признаков делимости *лучше* использовать *десятичное представление* числа $\overline{a_1 a_2 \dots a_n}$:

$$\overline{a_1 a_2 \dots a_n} = \sum_{i=0}^{n-1} a_{n-i} 10^i$$

- При модуле $m = 2^k; 5^k; 10^k$ одночлены $a_{n-i}10^i \equiv a_{n-i}0 = 0 \ (i \geq k)$. Значит, число $\overline{a_1a_2 \dots a_n}$ кратно m , когда последние k цифр кратны m :

$$\overline{a_1a_2 \dots a_n} \equiv 0 \iff \overline{a_{n-k+1} \dots a_{n-1}a_n} \equiv 0$$

- При модуле $m = 3; 9$ одночлены $a_{n-i}10^i \equiv a_{n-i}1^i = a_{n-i}$. Значит, число $\overline{a_1a_2 \dots a_n}$ кратно m , когда сумма цифр кратна m :

$$\overline{a_1a_2 \dots a_n} \equiv 0 \iff a_1 + a_2 + \dots + a_n \equiv 0$$

- При модуле $m = 11$ одночлены $a_{n-i}10^i \equiv a_{n-i}(-1)^i$. Значит, число $\overline{a_1a_2 \dots a_n}$ кратно 11, когда знакопеременная сумма цифр кратна 11:

$$\overline{a_1a_2 \dots a_n} \equiv 0 \iff a_1 - a_2 + \dots - a_n \equiv 0$$

- При модуле $m = 7$ вычтем из числа n последнюю цифру; останется $\lfloor n/10 \rfloor$. Последняя цифра равна $n - 10\lfloor n/10 \rfloor$. Вычтем из числа удвоенную последнюю цифру:

$$\left\lfloor \frac{n}{10} \right\rfloor - 2(n - 10 \left\lfloor \frac{n}{10} \right\rfloor) \equiv 0 \iff 21 \left\lfloor \frac{n}{10} \right\rfloor - 2n \equiv 0$$

Одночлен $21\lfloor n/10 \rfloor \equiv 0$. Значит, число $\overline{a_1a_2 \dots a_n}$ кратно 7, когда удвоенная разность последней цифры числа и самого числа без этой цифры кратна 7:

$$\overline{a_1a_2 \dots a_n} \equiv 0 \iff \overline{a_1a_2 \dots a_{n-1}} - 2a_n \equiv 0$$

Функция Эйлера

Функция $\phi(m)$ считает количество положительных целых чисел, меньших m и взаимно простых с ним (для малых и простых m целесообразно перебрать вручную):

$$\phi(m) = m \prod_{p|m} \left(1 - \frac{1}{p}\right)$$

p — простой делитель m ;

$1/p$ — часть чисел, кратных p ;

$1 - 1/p$ — часть чисел, взаимно простых с p .

Функция Эйлера мультипликативна (только для взаимно простых натуральных чисел).

Теорема Эйлера

Теорема. Пусть $a \in \mathbb{Z}$, $(a, m) = 1$. Тогда: (Л. Эйлер)

$$a^{\phi(m)} \equiv 1 \pmod{m}, \quad a \not\equiv 0 \pmod{m}$$

Доказательство. Введём систему остаточных классов \mathbb{Z}_m . В ней есть m классов: r_0, r_1, \dots, r_{m-1} .

Пусть множество Φ содержит в себе $\phi(m)$ остатков, взаимно простых с m . Домножим каждый элемент на a и образуем новое множество Φ_a . Заметим, что:

Элементы Φ_a из разных классов. Φ и Φ_a конгруэнтны.

Допустим, это не так. Тогда:

Пусть $ar_k \equiv r_l, r_l \in \mathbb{Z}_m$.

$$ar_k \equiv ar_l \implies r_k \equiv r_l$$

Так как $m \nmid ar_k$, то:

Но $r_k \not\equiv r_l \implies ar_k \not\equiv ar_l \square$

$$r_l \in \Phi \implies \Phi \equiv \Phi_a \square$$

Перемножим элементы множеств Φ и Φ_a :

$$r_0 r_1 \dots r_{\phi(m)} \equiv ar_0 ar_1 \dots ar_{\phi(m)} \implies$$

$$r_0 r_1 \dots r_{\phi(m)} \equiv a^{\phi(m)} r_0 r_1 \dots r_{\phi(m)} \implies a^{\phi(m)} \equiv 1 \blacksquare$$

Следствие. Пусть $a \in \mathbb{Z}$, $b \in \mathbb{N}$, $(m, a) = 1$. Тогда:

$$a^b \equiv a^{b \bmod \phi(m)} \pmod{m}, \quad a \not\equiv 0 \pmod{m}$$

Доказательство. Представим b в арифметическом виде:

$$b = \phi(m) \left\lfloor \frac{b}{\phi(m)} \right\rfloor + b \bmod \phi(m)$$

$\phi(m)$ — модуль деления.

$\lfloor b/\phi(m) \rfloor$ — целое частное.

$b \bmod \phi(m)$ — остаток.

Подставим полученное выражение:

$$a^{\phi(m) \lfloor b/\phi(m) \rfloor + b \bmod \phi(m)} = (a^{\phi(m)})^{\lfloor b/\phi(m) \rfloor} a^{b \bmod \phi(m)}$$

Так как $a^{\phi(m)} \equiv 1$, получается $a^b \equiv a^{b \bmod \phi(m)}$. ■

Алгоритм Евклида

Пусть $a, b \in \mathbb{N}^0$ ($a > b$), тогда:

$$(a, b) = (a \bmod b, b)$$

Доказательство. Пусть $m \mid a - b, b$:

$$+ \begin{cases} a - b \equiv 0 & (\bmod m) \\ b \equiv 0 & (\bmod m) \end{cases} \Rightarrow \begin{cases} a \equiv 0 & (\bmod m) \\ b \equiv 0 & (\bmod m) \end{cases}$$

Значит, любой общий делитель $a - b, b$ имеется у a, b .

По определению НОД:

$$(a, b) = (a - b, b)$$

По определению конгруэнтных чисел:

$$(a, b) = (a \bmod b, b)$$

Мультипликативная инверсия

Пусть $ab \equiv 1 \pmod{m}$ — линейное сравнение, где b — **мультипликативная инверсия** числа a по модулю m :

$$b \equiv a^{-1} \equiv \frac{1}{a} \pmod{m}, \quad (a, m) = 1$$

«Дробные» числа можно складывать, перемножать и сокращать как рациональные:

$$\begin{cases} \frac{a}{b} + \frac{c}{d} \equiv \frac{ad + bc}{cd} & (\bmod m) \\ \frac{a}{b} \times \frac{c}{d} \equiv \frac{ac}{bd} & (\bmod m) \\ \frac{eg}{fg} \equiv \frac{e}{f} & (\bmod m) \end{cases}$$

Линейное сравнение

Линейное сравнение вида $ax \equiv b \pmod{m}$ разрешимо относительно x , когда $(m, a) \mid b$. (по соотношению Безу)

План решения:

- упростить линейное сравнение
- рассчитать (m, a) по алгоритму Евклида
- выразить (m, a) через полученные остатки
- домножить соотношение Безу на b

Пример. Решить линейное сравнение: $4x \equiv 4 \pmod{6}$.

Упростим сравнение:

$$4x \equiv 4 \pmod{6} \mid \cdot 1/2$$

$$2x \equiv 2 \pmod{3}$$

Применим алгоритм Евклида в алгебраическом виде:

«Прямой» алгоритм:

$$3 = 2 \cdot 1 + 1$$

$$2 = 1 \cdot 2 + 0$$

«Обратный» алгоритм:

$$1 = 3 \cdot 1 + 2 \cdot (-1) \mid \cdot 2$$

$$2 = 3 \cdot 2 + 2 \cdot (-2)$$

Итак, коэффициенты Безу найдены: $x = -2$, $y = 2$.

Ответ: $x = -2$.

Китайская теорема об остатках

Сравнения можно объединять в систему:

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ \dots \\ x \equiv a_n \pmod{m_n} \end{cases}$$

Она разрешима относительно x по модулю $[m_1, \dots, m_n]$, когда разрешима каждая пара сравнений, в частности $(m_1, m_2) \mid a_1 - a_2$.

Доказательство. Рассмотрим пару сравнений из системы:

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \end{cases} \iff \begin{cases} x = a_1 + m_1 j, & j \in \mathbb{Z} \\ x = a_2 + m_2 k, & k \in \mathbb{Z} \end{cases} \iff$$

$$m_1j + m_2k = a_2 - a_1$$

Данное соотношение Безу имеет целые коэффициенты j, k , когда $(m_1, m_2) \mid (a_1 - a_2)$. \square

По индукции, система будет разрешима относительно x , когда будет разрешима каждая пара сравнений.

Допустим, $x \equiv y \equiv a_i \pmod{m_i}$, $i \in \{i\}_{i=1}^n$ — решение всей системы. Значит, $m_i \mid x - y \implies [m_1, \dots, m_n] \mid x - y \iff x \equiv y \pmod{[m_1, \dots, m_n]}$. ■

План решения каждой пары сравнений:

- упростить линейные сравнения;
- преобразовать их в соотношения Безу, приравнять их;
- решить полученное выражение как линейное сравнение.

Пример. Решить систему сравнений:

$$\begin{cases} x \equiv 2 & (\text{mod } 3) \\ x \equiv 2 & (\text{mod } 4) \\ 2x \equiv -3 & (\text{mod } 5) \end{cases}$$

Упростим последнее сравнение:

$$2x \equiv -3 \pmod{5} \iff x \equiv 1 \pmod{5}$$

Преобразуем первую пару сравнений в соотношения Безу:

$$\begin{cases} x \equiv 2 \pmod{3} \\ x \equiv 2 \pmod{4} \end{cases} \iff \begin{cases} x = 2 + 3j, & j \in \mathbb{Z} \\ x = 2 + 4k, & k \in \mathbb{Z} \end{cases}$$

Приравняем их и решим как сравнение:

$$2 + 3j = 2 + 4k \iff 2 \equiv 2 + k \pmod{3} \iff k \equiv 0 \pmod{3}$$

Значит, $x = 2 + 4k \equiv 2 \pmod{12}$ — решение первой пары.

Аналогично решив следующую (и последнюю) пару, получим решение всей системы: $x \equiv 26 \pmod{60}$.

Ответ: $x \equiv 26 \pmod{60}$.

Сравнение по составному модулю

Пусть $f \in \mathbb{Z}[x]$. Тогда для $m = p_1^{\alpha_1} \dots p_r^{\alpha_r}$ разрешимо

$$f(x) \equiv 0 \pmod{m},$$

если разрешимы $f(x) \equiv 0 \pmod{p_i^{\alpha_i}}, i \in [1; r] \cap \mathbb{Z}$.

Доказательство \Rightarrow . Пусть $x \in \mathbb{Z}$ — решение

$$f(x) \equiv 0 \pmod{m}, p_i^{\alpha_i} \mid m \Rightarrow f(x) \equiv 0 \pmod{p_i^{\alpha_i}}. \blacksquare$$

Доказательство \Leftarrow . Пусть x_i — решение

$$f(x_i) \equiv 0 \pmod{p_i^{\alpha_i}}$$

По китайской теореме об остатках:

$$\forall i_1, i_2 \in [1; r], i_1 \neq i_2 (p_{i_1}^{\alpha_{i_1}}, p_{i_2}^{\alpha_{i_2}}) = 1 \Rightarrow$$

$$\exists x: x \equiv x_i \pmod{p_i^{\alpha_i}} \Rightarrow f(x) \equiv 0 \pmod{[p_1^{\alpha_1}, \dots, p_r^{\alpha_r}]} \Rightarrow \\ f(x) \equiv 0 \pmod{m} \blacksquare$$

Сравнение по степени простого модуля

Пусть $f \in \mathbb{Z}[x]$. Тогда для простого p разрешимо

$$f(x) \equiv 0 \pmod{p^\alpha},$$

если разрешимы $f(x) \equiv 0 \pmod{p^i}, i \in [1; \alpha] \cap \mathbb{Z}$.

Доказательство. Аналогично прошлому пункту.

Лемма Гензеля

Пусть для $f \in \mathbb{Z}[x]$ верно (*К. Гензель*):

$$f(a) \equiv 0 \pmod{p^\alpha}, \quad f'(a) \not\equiv 0 \pmod{p}$$

Тогда существует такое уникальное t , что:

$$f(a + tp^\alpha) \equiv 0 \pmod{p^{\alpha+1}}$$

Доказательство. Пусть a — решение $f(x) \equiv 0 \pmod{p^\alpha}$, которое можно представить в виде $x = a + tp^\alpha$.

По теореме Тейлора:

$$f(a + tp^\alpha) = f(a) + tp^\alpha f'(a) + t^2 p^{2\alpha} f''(a)/2! + \dots \\ + t^n p^{n\alpha} f^{(n)}(a)/n! \equiv f(a) + tp^\alpha f'(a) \pmod{p^{\alpha+1}} \blacksquare$$

Следствие. Пусть для $f \in \mathbb{Z}[x]$ верно

$$f(x_\alpha) \equiv 0 \pmod{p^\alpha}, \quad f'(x_\alpha) \not\equiv 0 \pmod{p^\alpha}.$$

Тогда решение сравнения по модулю $p^{\alpha+1}$ имеет вид:

$$x_{\alpha+1} \equiv x_\alpha - \frac{f(x_\alpha)}{f'(x_\alpha)} \pmod{p^{\alpha+1}}$$

Доказательство. По лемме Гензеля:

$$f(x_\alpha) + tp^\alpha f'(x_\alpha) \equiv 0 \pmod{p^{\alpha+1}} \iff$$

$$tp^\alpha \equiv -\frac{f(x_\alpha)}{f'(x_\alpha)} \pmod{p^{\alpha+1}} \iff$$

$$x_\alpha + tp^\alpha \equiv x_{\alpha+1} \equiv x_\alpha - \frac{f(x_\alpha)}{f'(x_\alpha)} \pmod{p^{\alpha+1}} \quad \blacksquare$$

Тригонометрия

Основные функции

Единичной называется окружность, которая задаётся уравнением $x^2 + y^2 = 1$.

Тригонометрические функции соотносят *координаты* точки единичной окружности и *градусную меру дуги*, образуемой ей с начальным радиусом.

Синус — нечётная функция с периодом 2π ; график — *синусоида*:

$$\sin: \mathbb{R} \xrightarrow{\alpha \mapsto y} [-1; 1]$$

Обратная нечётная функция к $\sin|_{[-\pi/2; \pi/2]}$ — **арксинус**:

$$\sin^{-1} = \arcsin: [-1; 1] \xrightarrow{\alpha \mapsto y} [-\pi/2; \pi/2]$$

Косинус — чётная функция с периодом 2π ; график — *синусоида* со смещением влево на $\pi/2$ («*косинусоида*»):

$$\cos: \mathbb{R} \xrightarrow{\alpha \mapsto x} [-1; 1]$$

Обратная функция к $\cos|_{[0; \pi]}$ — **арккосинус**:

$$\cos^{-1} = \arccos: [-1; 1] \xrightarrow{\alpha \mapsto x} [0; \pi]$$

Тангенс — нечётная функция с периодом π ; график — *тангенсоида*:

$$\operatorname{tg}: \mathbb{R} \setminus \{\pi/2 + \pi n \mid n \in \mathbb{Z}\} \xrightarrow{\alpha \mapsto y/x} \mathbb{R}$$

Обратная нечётная функция к $\operatorname{tg}|_{(-\pi/2; \pi/2)}$ — **арктангенс**:

$$\operatorname{tg}^{-1} = \operatorname{arctg}: \mathbb{R} \xrightarrow{y/x \mapsto \alpha} (-\pi/2; \pi/2)$$

Котангенс — нечётная функция с периодом π ; график — *тангенсоида* с симметрией относительно оси Ox и смещением вправо на $\pi/2$ («*котангенсоида*»):

$$\operatorname{ctg}: \mathbb{R} \setminus \{\pi n \mid n \in \mathbb{Z}\} \xrightarrow{\alpha \mapsto x/y} \mathbb{R}$$

Обратная функция к $\operatorname{ctg}|_{(0; \pi)}$ — **арккотангенс**:

$$\operatorname{ctg}^{-1} = \operatorname{arcctg}: \mathbb{R} \xrightarrow{x/y \mapsto \alpha} (0; \pi)$$

Основные тождества

Из определений тригонометрических функций следует:

$$\begin{aligned}\sin^2 \alpha + \cos^2 \alpha &= 1 & \arccos x &= \arcsin(\sqrt{1-x^2}) \\ 1 + \operatorname{tg}^2 \alpha &= 1/\cos^2 \alpha & \arccos x &= \operatorname{arctg}(\sqrt{1-x^2}/x) \\ 1 + \operatorname{ctg}^2 \alpha &= 1/\sin^2 \alpha & \arcsin y &= \operatorname{arcctg}(\sqrt{1-y^2}/y)\end{aligned}$$

Сумма и разность углов

Из скалярного произведения векторов следует:

$$\begin{aligned}\cos(\alpha \pm \beta) &= \cos \alpha \cos \beta \mp \sin \alpha \sin \beta \\ \sin(\alpha \pm \beta) &= \sin \alpha \cos \beta \pm \sin \beta \cos \alpha \\ \operatorname{tg}(\alpha \pm \beta) &= \frac{\operatorname{tg} \alpha \pm \operatorname{tg} \beta}{1 \mp \operatorname{tg} \alpha \operatorname{tg} \beta} & \operatorname{ctg}(\alpha \pm \beta) &= \frac{\operatorname{ctg} \alpha \operatorname{ctg} \beta \mp 1}{\operatorname{ctg} \alpha \pm \operatorname{ctg} \beta}\end{aligned}$$

Доказательство. Пусть $\vec{A} = \langle \cos \alpha; \sin \alpha \rangle$, $\vec{B} = \langle \cos \beta; \sin \beta \rangle$.

Рассмотрим их скалярное произведение:

$$\begin{aligned}+ \begin{cases} \vec{A} \cdot \vec{B} = \cos \alpha \cos \beta + \sin \alpha \sin \beta \\ \vec{A} \cdot \vec{B} = \|\vec{A}\| \|\vec{B}\| \cos(\alpha - \beta) = \cos(\alpha - \beta) \end{cases} &\Rightarrow \\ \cos(\alpha - \beta) &= \cos \alpha \cos \beta + \sin \alpha \sin \beta \quad \square\end{aligned}$$

Затем полезно применить эти четыре формулы:

$$\begin{aligned}\alpha + \beta &= \alpha - (-\beta) \\ \sin(\alpha - \beta) &= \cos((\pi/2 - \alpha) + \beta) \\ \operatorname{tg} \alpha &= \sin \alpha / \cos \alpha & \operatorname{ctg} \alpha &= \cos \alpha / \sin \alpha \quad \blacksquare\end{aligned}$$

Двойной угол

Из формул суммы и разности двух углов следует:

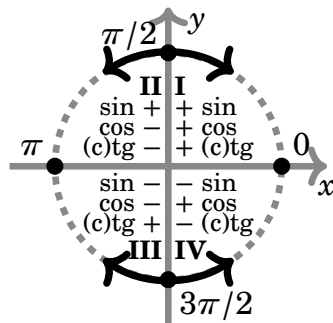
$$\begin{aligned}\cos 2\alpha &= \cos^2 \alpha - \sin^2 \alpha & \sin 2\alpha &= 2 \sin \alpha \cos \alpha \\ \operatorname{tg} 2\alpha &= \frac{2 \operatorname{tg} \alpha}{1 - \operatorname{tg}^2 \alpha} & \operatorname{ctg} 2\alpha &= \frac{\operatorname{ctg}^2 \alpha - 1}{2 \operatorname{ctg} \alpha} \\ (\sin \alpha \pm \cos \alpha)^2 &= 1 \pm \sin 2\alpha\end{aligned}$$

Формулы приведения

Из формул суммы и разности двух углов следуют *формулы приведения*, которые имеют вид:

$$f(\pi n/2 \pm \alpha) = \pm \text{cof}(\alpha), \quad n \in \mathbb{Z}$$

Конечная функция и её знак определяются по графику; стрелками обозначены места смены функции на *кофункцию*.



Следствие. Для обратных функций верно:

$$\arcsin x = \pi/2 - \arccos x \quad \arccos(-x) = \pi - \arccos x$$

$$\text{arctg} x = \pi/2 - \text{arcctg} x \quad \text{arcctg}(-x) = \pi - \text{arcctg} x$$

Понижение степени

Из формул двойного угла и основного тригонометрического тождества следует:

$$2 \cos^2 \alpha = 1 + \cos 2\alpha \quad \text{tg}^2 \alpha = \frac{1 - \cos 2\alpha}{1 + \cos 2\alpha}$$

$$2 \sin^2 \alpha = 1 - \cos 2\alpha \quad \text{ctg}^2 \alpha = \frac{1 + \cos 2\alpha}{1 - \cos 2\alpha}$$

Из них легко выводятся формулы *половинного угла*.

Сумма и разность функций

Из формул суммы и разности двух углов следует:

$$\sin \alpha \pm \sin \beta = 2 \sin \frac{\alpha \pm \beta}{2} \cos \frac{\alpha \mp \beta}{2}$$

$$\cos \alpha + \cos \beta = 2 \cos \frac{\alpha + \beta}{2} \cos \frac{\alpha - \beta}{2}$$

$$\cos \alpha - \cos \beta = -2 \sin \frac{\alpha + \beta}{2} \sin \frac{\alpha - \beta}{2}$$

Из них можно вывести формулы *произведения двух функций*.

Доказательство. Рассмотрим сумму синусов:

$$\sin(x+y) + \sin(x-y) =$$

$$\sin x \cos y + \sin y \cos x + \sin x \cos y - \sin y \cos x = 2 \sin x \cos y$$

Введём обозначения:

$$\begin{cases} x+y = \alpha \\ x-y = \beta \end{cases} \iff \begin{cases} 2x = \alpha + \beta \\ 2y = \alpha - \beta \end{cases} \iff \begin{cases} x = \frac{\alpha + \beta}{2} \\ y = \frac{\alpha - \beta}{2} \end{cases}$$

Таким образом:

$$\sin \alpha + \sin \beta = 2 \sin \frac{\alpha + \beta}{2} \cos \frac{\alpha - \beta}{2} \quad \square$$

Остальные формулы доказываются аналогично. ■

Для $c = \sqrt{a^2 + b^2}$ справедливо:

$$a \sin \alpha + b \cos \alpha = c \sin(\alpha + \phi) = c \cos(\alpha - \phi)$$

$$|a \sin \alpha + b \cos \alpha| \leq c$$

Доказательство. Рассмотрим синус суммы двух углов:

$$c \sin(\alpha + \phi) = c \sin \alpha \cos \phi + c \sin \phi \cos \alpha$$

Обозначим $a = c \cos \phi$, $b = c \sin \phi$ и найдём сумму квадратов:

$$a^2 + b^2 = c^2(\sin^2 \phi + \cos^2 \phi) = c^2 \iff c = \sqrt{a^2 + b^2} \quad \square$$

Случай с косинусом доказывается аналогично. ■

Подстановка Вейерштрасса

Тригонометрические функции от 2α можно выразить через тангенс от α (*К. Вейерштрасс*):

$$\sin 2\alpha = \frac{2 \operatorname{tg} \alpha}{1 + \operatorname{tg}^2 \alpha} \quad \cos 2\alpha = \frac{1 - \operatorname{tg}^2 \alpha}{1 + \operatorname{tg}^2 \alpha}$$

Доказательство. Распишем каждую функцию:

$$\sin 2\alpha = \frac{2 \sin \alpha \cos \alpha}{\sin^2 \alpha + \cos^2 \alpha} = \frac{2 \operatorname{tg} \alpha}{1 + \operatorname{tg}^2 \alpha} \quad \square$$

$$\cos 2\alpha = \frac{\cos^2 \alpha - \sin^2 \alpha}{\sin^2 \alpha + \cos^2 \alpha} = \frac{1 - \operatorname{tg}^2 \alpha}{1 + \operatorname{tg}^2 \alpha} \quad \blacksquare$$

Теория множеств

Открытое множество

ε -окрестность точки $x_0 \in X$ метрического пространства $\langle X, d \rangle$ — такое множество точек $x \in X$, что $d(x_0, x) < \varepsilon$.

Упрощённая запись $\{x \mid d(x_0, x) < \varepsilon\} =: U_\varepsilon(x_0)$.

Особые случаи:

$$U_\varepsilon(+\infty) := (1/\varepsilon; +\infty)$$

$$U_\varepsilon(-\infty) := (-\infty; -1/\varepsilon)$$

Проколотой называется ε -окрестность точки x_0 без неё:

$$\mathring{U}_\varepsilon(x_0) := U_\varepsilon(x_0) \setminus \{x_0\}$$

Правосторонней (левосторонней) называется ε -окрестность точки x_0 без левой (правой) половины:

$$U_{\varepsilon+}(x_0) := [x_0; \varepsilon) \quad U_{\varepsilon-}(x_0) := (\varepsilon; x_0]$$

Ограниченное множество

Множество M ограничено *сверху*, если

$$\forall m \in M \exists C \in \mathbb{R} : m \leq C.$$

Точной (минимальной, англ. *supremum*) называется такая *верхняя* граница множества M — $\sup M$, что

$$\forall \varepsilon > 0 \exists m \in M : m \in U_{\varepsilon-}(\sup M).$$

Множество M ограничено *снизу*, если

$$\forall m \in M \exists C \in \mathbb{R} : m \geq C.$$

Точной (максимальной, англ. *infimum*) называется такая *нижняя* граница множества M — $\inf M$, что

$$\forall \varepsilon > 0 \exists m \in M : m \in U_{\varepsilon+}(\inf M).$$

Принцип Кантора

Последовательность вложенных отрезков содержит точки ξ , которые принадлежат им всем:

$$\forall n \in \mathbb{N} \exists \xi \in [a_n; b_n] \subset [a_{n-1}; b_{n-1}]$$

Если $n \rightarrow \infty$, $(b_n - a_n) \rightarrow 0$, то ξ единственна:

$$\lim_{n \rightarrow \infty} a_n = \sup\{a_n\} = \lim_{n \rightarrow \infty} b_n = \inf\{b_n\} = \xi$$

Доказательство. По теореме Вейерштрасса:

$$\lim_{n \rightarrow \infty} a_n = \sup\{a_n\} \quad \lim_{n \rightarrow \infty} b_n = \inf\{b_n\}$$

Значит, $\forall (n \in \mathbb{N}, \xi \in [\sup\{a_n\}; \inf\{b_n\}]) \xi \in [a_n; b_n]$. \square

Если $\inf\{b_n\} = \sup\{a_n\}$, то ξ единственна:

$$0 = \inf\{b_n\} - \sup\{a_n\} = \lim_{n \rightarrow \infty} b_n - \lim_{n \rightarrow \infty} a_n = \lim_{n \rightarrow \infty} (b_n - a_n) \blacksquare$$

Определение

Множеством называется объединение *различных* объектов — *элементов* множества — в единое целое.

Способы задания множеств.

- 1) Перечислением (*списком элементов*)
- 2) Порождающей процедурой
- 3) Разрешающей процедурой

Множество A есть **подмножество** B , если все элементы A являются элементами B :

$$A \subseteq B$$

Множество A есть **надмножество** B , если все элементы B являются элементами A :

$$A \supseteq B$$

Подмножество **собственное** (*строгое*), если оно не равно исходному множеству:

$$A \subseteq B \wedge A \neq B \implies A \subset B$$

Булеан множества $\mathcal{B}(A)$ — множество всех его подмножеств:

$$\mathcal{B}(A) = \{X \mid X \subseteq A\}$$

Мощность булеана определяется формулой:

$$|\mathcal{B}(A)| = 2^{|A|}$$

Метод взаимного включения. Множества A и B равны, если они содержат одни и те же элементы:

$$A \subseteq B \wedge A \supseteq B$$

Пустое множество \emptyset не содержит ни одного элемента и есть подмножество любого множества.

Универсум U — широкое множество, которое состоит из всех элементов *исследуемой области*.

Конечным множество состоит из *конечного* числа элементов, а **бесконечное** множество — из *бесконечного*.

Бесконечные множества делятся на два вида:

- **счётное**: равномощно множеству \mathbb{N} (*его можно пронумеровать*)
- **несчётное**: не равномощно \mathbb{N} (*его нельзя пронумеровать*)

Мощность *конечного* множества — число его элементов.

Алгебра логики

Определение

Алгебра логики — алгебраическая структура, которая образована двухэлементным множеством $\{0; 1\}$.

Высказывание — повествовательное предложение, о котором можно сказать в данный момент, *истинно* ли оно или *ложно*.

Предикат (одноместный) — функция переменной x , которое принимает значение из множества $\{0; 1\}$.

Высказывание — нуль-местный предикат.

Логическая связка — операция алгебры логики:

- 1) *Инверсия* « \neg » — логическое «не».
- 2) *Конъюнкция* « \wedge » — логическое «и».
- 3) *Дизъюнкция* « \vee » — логическое «или».
- 4) *Строгая дизъюнкция* « $\dot{\vee}$ » — логическое «искл. или».
- 5) *Импликация* « \rightarrow » — логическое « \Rightarrow ».
- 6) *Эквиваленция* « \equiv » — логическое « \Leftrightarrow ».

Квантор — логическая операция:

- квантор всеобщности « \forall »
- квантор существования « \exists »

Свойства

Конъюнкция и дизъюнкция коммутативны, ассоциативны и дистрибутивны относительно друг друга.

Идемпотентность. (Тавтология)

$$A \wedge A = A \quad A \vee A = A$$

Инволютивность.

$$\bar{\bar{A}} = A$$

Закон противоречия и исключённого третьего.

$$A \wedge \bar{A} = 0 \quad A \vee \bar{A} = 1$$

Закон поглощения.

$$A \wedge 1 = A \quad A \wedge 0 = 0$$

$$A \vee 1 = 1 \quad A \vee 0 = A$$

$$A \wedge (A \vee B) = A \quad A \vee (A \wedge B) = A$$

$$A \wedge (\bar{A} \vee B) = A \wedge B \quad A \vee (\bar{A} \wedge B) = A \vee B$$

Закон де Моргана. (*Двойственность*)

$$\overline{A \wedge B} = \bar{A} \vee \bar{B} \quad \overline{A \vee B} = \bar{A} \wedge \bar{B}$$

Закон склеивания.

$$(A \vee B) \wedge (A \vee \bar{B}) = A$$

$$(A \wedge B) \vee (A \wedge \bar{B}) = A$$

Нормальная форма

Терм — компонент логической функции:

— **макстерм** — переменные прямой и инверсной форм связаны *дизъюнкцией*

— **минтерм** — переменные прямой и инверсной форм связаны *конъюнкцией*

Ранг терма — число переменных, которые в него входят.

Нормальная дизъюнктивная форма (DNF) — дизъюнкция минтермов любого ранга.

Нормальная конъюнктивная форма (CNF) — конъюнкция макстермов любого ранга:

$$A \dot{\vee} B = (A \vee B) \wedge (\bar{A} \vee \bar{B})$$

$$A \equiv B = (\bar{A} \vee B) \wedge (A \vee \bar{B})$$

$$A \rightarrow B = \bar{A} \vee B$$

Нормальная импликативная форма (INF) — конъюнкция макстермов любого ранга, которые заменены импликацией:

$$A \vee B = (\bar{A} \rightarrow B) \wedge (\bar{B} \rightarrow A)$$

Общая алгебра

Вектор

Вектор (*кортеж, или упорядоченная n -ка*) — упорядоченный набор элементов — *координат (компонент)* вектора.

Размерность вектора — число его координат.

Декартово (прямое) произведение множеств X_1, \dots, X_n — множество векторов вида:

$$X_1 \times \dots \times X_n = \{(x_1, \dots, x_n) \mid x_1 \in X_1, \dots, x_n \in X_n\}$$

Свойства:

- дистрибутивность относительно *пересечения* \cap
- дистрибутивность относительно *разности* \setminus
- *не коммутативность
- *не ассоциативность
- $(A \cap B) \times (C \cap D) = (A \times C) \cap (B \times D)$

Декартова степень множества — прямое произведение одинаковых множеств:

$$A^3 = A \times A \times A$$

Теорема. Мощность декартова произведения конечных множеств равна *произведению мощностей* данных множеств:

$$|A_1 \times \dots \times A_n| = |A_1| \times \dots \times |A_n|$$

Отношение

Отношение ρ между множествами X_1, \dots, X_n — подмножество их *декартова произведения*.

Бинарное отношение включает два множества, что можно упрощённо записать:

$$x \in X, y \in Y, \langle x, y \rangle \in \rho =: x \rho y$$

Унарное отношение (*свойство*) включает в себя...?

$X \supseteq D_\rho$ — область определения (*прообраз*) отношения;
 $Y \supseteq E_\rho$ — область значений (*образ*) отношения.

Виды отношений

Отношение ρ *инъективно*, когда

$$\forall x_1, x_2 \in D_\rho \exists y \in E_\rho : x_1 \rho y, x_2 \rho y \iff x_1 = x_2.$$

Отношение ρ *функционально*, когда

$$\forall x \in D_\rho \exists! y \in E_\rho : x \rho y.$$

Функциональное отношение называется **отображением** (*функцией*) и обозначается:

$$\rho : X \xrightarrow{x \mapsto y} Y$$

Отношение ρ *сюръективно*, когда

$$\forall y \in Y \exists x \in D_\rho : x \mapsto y.$$

Отношение ρ *всюду определено*, когда

$$\forall x \in X \exists y \in E_\rho : x \mapsto y.$$

Композиция (*суперпозиция*) бинарных отношений $f \subseteq X \times Y$, $g \subseteq Y \times Z$ — такое $h \subseteq X \times Z \iff f \circ g$, что:

$$\forall x \in X, z \in Z \exists y \in Y x(f \circ g)z \iff xfy \wedge ygz$$

Свойства:

- ассоциативность
- *не коммутативность

Обратное бинарное отношение ρ^{-1} получается перестановкой исходных множеств в декартовом произведении:

$$\rho \subseteq X \times Y \iff \rho^{-1} \subseteq Y \times X$$

Свойства:

- идемпотентность
- дистрибутивность относительно *пересечения* \cap
- дистрибутивность относительно *объединения* \cup
- дистрибутивность относительно *композиции* \circ :

$$(P \circ Q)^{-1} = Q^{-1} \circ P^{-1}$$

СВОЙСТВА ОТНОШЕНИЙ

Пусть $* \subseteq X^2$ — произвольное бинарное отношение.

Отношение $*$ *симметрично*, когда

$$\forall x, y \in X \quad x * y = y * x.$$

Отношение $*$ *антисимметрично*, когда

$$\forall x, y \in X \quad x * y \wedge y * x \implies x = y$$

Отношение $*$ *транзитивно*, когда

$$\forall x, y, z \in X \quad x * y \wedge y * z \implies x * z$$

Отношение $*$ *рефлексивно*, когда

$$\forall x \in X \quad x * x$$

Отношение $*$ *антирефлексивно*, когда

$$\forall x \in X \quad \neg(x * x)$$

Ограничение и продолжение

Ограничением отображения $f: X \rightarrow Y$ на $S \subseteq D_f$ называется такое $f|_S: S \rightarrow Y$, что

$$\forall s \in S: f|_S(s) = f(s).$$

В свою очередь, f является *продолжением* отображения $f|_S$.

Алгебраическая операция

n -местная **алгебраическая операция** — отображение:

$$*: X^n \rightarrow X$$

Нейтральным называется такой элемент $e \in X$, что:

$$\forall x \in X \quad e * x = x * e = x$$

Поглощающим называется такой элемент $w \in X$, что:

$$\forall x \in X \quad w * x = x * w = w$$

Симметричным (*противоположным, обратным*) к элементу $x \in X$ называется такой элемент $x^{-1} \in X$, что:

$$x * x^{-1} = x^{-1} * x = e \text{ (нейтр.)}$$

Если $y: X \rightarrow Y$ и $x: Y \rightarrow X$ — отображения, то y *инъективно*, а x *сюръективно*.

Доказательство. По условию, множество X накладывается на себя. Значит, f *всюду определено*.

Так как g функционально, то

$$\forall x_1, x_2 \in X \quad \exists y \in E_f: x_1 f y, x_2 f y \iff x_1 = x_2,$$

то есть f *инъективно*. \square

Когда X накладывается на себя, то

$$\forall x \in E_g \quad \exists y \in D_g: x \mapsto y,$$

то есть g *сюръективно*. \blacksquare

Свойства операций

Пусть $*: X^2 \rightarrow X$, $\circ: X^2 \rightarrow X$ — бинарные алгебраические операции.

Операция $*$ *коммутативна*, когда

$$\forall x, y \in X \quad x * y = y * x$$

Операция $*$ *ассоциативна*, когда

$$\forall x, y, z \in X \quad (x * y) * z = x * (y * z)$$

Операция $*$ *дистрибутивна* относительно \circ , когда

$$\forall x, y, z \in X \begin{cases} x * (y \circ z) = (x * y) \circ (x * z) \text{ (дистр. справа)} \\ (x \circ y) * z = (x * z) \circ (y * z) \text{ (дистр. слева)} \end{cases}$$

Алгебраическая структура

Алгебраическая структура (*система*) — множество X с введёнными на нём алгебраическими операциями:

$$\langle X; *, *_1, *_2, \dots, *_n \rangle$$

Полугруппа — алгебраическая структура $\langle X; * \rangle$ с двухместной ассоциативной операцией $*$.

Группа — полугруппа, для которой существуют нейтральный и обратный элементы.

Кольцо — коммутативная аддитивная группа, мультипликативная полугруппа, где \times дистрибутивно относительно $+$.

Поле — коммутативное кольцо с обратным элементом для \times .

Числовые системы

Система натуральных чисел — коммутативная аддитивная и мультипликативная полугруппа $\langle \mathbb{N}; +, \times \rangle$.

Система целых чисел — коммутативное кольцо $\langle \mathbb{Z}; +, \times \rangle$.

Система рациональных чисел — упорядоченное поле $\langle \mathbb{Q}; +, \times \rangle$.

Система действительных чисел — непрерывное упорядоченное поле $\langle \mathbb{R}; +, \times \rangle$.

Проективно расширенная числовая прямая — расширение множества действительных чисел $\widehat{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$:

$$a \pm \infty = \infty \pm a = \infty, \quad a \neq \infty$$

$$b \cdot \infty = \infty \cdot b = \infty, \quad b \neq 0$$

$$\frac{a}{\infty} = 0 \quad \frac{b}{0} = \infty$$

Комплексные числа

Система комплексных чисел — непрерывное поле:

$$\langle \mathbb{C} = \mathbb{R}^2; +, \cdot \rangle$$

Комплексное число — элемент $z = (a, b) \in \mathbb{C}$.

Свойство. Множество \mathbb{C} включает в себя множество \mathbb{R} :

$$\forall a \in \mathbb{R} \quad (a, 0) \leftrightarrow a \implies \mathbb{R} \subset \mathbb{C}$$

Мнимая единица — комплексное число вида:

$$(0, 1) \leftrightarrow i, \quad i^2 = -1$$

Плоскость комплексных чисел — декартова система координат Oab с биекцией вида:

$$z = (a, b) \leftrightarrow M(a, b)$$

— Oa есть действительная ось ($a = \Re z$)

— Ob есть мнимая ось ($b = \Im z$)

Свойство. Комплексные числа *равны*, если соответственно равны их действительные и мнимые части:

$$(a_1, b_1) = (a_2, b_2) \iff a_1 = a_2 \wedge b_1 = b_2$$

Формы записи. Пусть $z = (a, b) \in \mathbb{C}$.

— алгебраическая форма: $z = a + bi$

— тригонометрическая форма: $z = |z| (\cos \varphi + i \sin \varphi)$

Операции комплексных чисел

Сложение и умножение. Пусть $z_1 = (a, b), z_2 = (c, d)$:

$$z_1 + z_2 = (a, b) + (c, d) = (a + c, b + d)$$

$$z_1 \cdot z_2 = (a, b) \cdot (c, d) = (ac - bd, ad + bd)$$

Свойства операций *индуцируются* из множества действительных чисел \mathbb{R} .

Сопряжение — операция смены знака $\Im z$:

$$z = (a, b) \rightarrow \bar{z} = (a, -b)$$

Свойства:

- дистрибутивность относительно сложения +
- дистрибутивность относительно умножения ·

Модуль комплексного числа z — расстояние $\rho(O; M)$:

$$|z| = r = |OM| = \sqrt{a^2 + b^2} = \sqrt{z\bar{z}}$$

Свойства:

- дистрибутивность относительно умножения ·
(деления :)

Аргумент комплексного числа — угол, образованный \overrightarrow{OM} с действительной осью:

$$\arg z = \varphi = \operatorname{arctg} \frac{b}{a}, \quad \varphi \in (-\pi; \pi]$$

Свойства:

- $\arg(z_1 \cdot z_2) = \arg z_1 + \arg z_2$
- $\arg(z_1 / z_2) = \arg z_1 - \arg z_2$
- $\arg z^n = n \arg z, \quad n \in \mathbb{Z}$

Формула Муавра. Возведение в степень числа $z \in \mathbb{C}$:

$$z^n = |z|^n (\cos n\varphi + i \sin n\varphi), \quad n \in \mathbb{Z}$$

Следствие. Корень n -й степени из числа $z \in \mathbb{C}$:

$$\sqrt[n]{z} = \sqrt[n]{|z|} \left(\cos \frac{\varphi + 2\pi k}{n} + i \sin \frac{\varphi + 2\pi k}{n} \right), \quad k \in \mathbb{Z}$$

Квадратный корень. Корень из числа $z \in \mathbb{C}$:

$$\sqrt{z} = \pm \left(\sqrt{\frac{|z| + a}{2}} + \operatorname{sgn}(b) i \sqrt{\frac{|z| - a}{2}} \right)$$

Метрическое пространство

Метрическое пространство — алгебраическая структура $\langle M; d \rangle$, где d — метрика.

Метрика d множества M — функция $d: M \times M \rightarrow R_0^+$, которая определяет *расстояние* между его двумя элементами.

Например, *евклидова метрика* использует теорему Пифагора в n -мерном пространстве:

$$d(x, y) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2}$$

Для метрического пространства $\langle M; d \rangle$, $x, y, z \in M$ выполняются следующие *аксиомы*:

- $d(x, y) = 0 \iff x = y$ — *тождество*;
- $d(x, y) = d(y, x)$ — *симметрия*;
- $d(x, y) \leq d(x, z) + d(y, z)$ — «*неравенство треугольника*».

Длина отрезка

Расстояние между точками $A\langle a \rangle$ и $B\langle b \rangle$ —

$$|\overrightarrow{AB}| = |a - b| \implies AB^2 = (a - b)(\bar{a} - \bar{b})$$

Уравнение окружности с центром $A\langle a \rangle$ радиуса r —

$$(z - a)(\bar{z} - \bar{a}) = r^2$$

Скалярное произведение векторов

Скалярное произведение радиус-векторов —

$$2\overrightarrow{OA} \cdot \overrightarrow{OB} = a\bar{b} + \bar{a}b$$

Доказательство. Пусть $A\langle a \rangle$, $B\langle b \rangle$, $a = x_1 + y_1i$, $b = x_2 + y_2i$. Тогда:

$$\begin{aligned} a\bar{b} + \bar{a}b &= (x_1 + y_1i)(x_2 - y_2i) + (x_1 - y_1i)(x_2 + y_2i) = \\ &= 2(x_1x_2 + y_1y_2) = 2\overrightarrow{OA} \cdot \overrightarrow{OB} \blacksquare \end{aligned}$$

Пусть $A\langle a \rangle$, $B\langle b \rangle$, $C\langle c \rangle$, $D\langle d \rangle$ — четыре различные точки.

Тогда скалярное произведение произвольных векторов —

$$2\overrightarrow{AB} \cdot \overrightarrow{CD} = (b - a)(\bar{d} - \bar{c}) + (\bar{b} - \bar{a})(d - c)$$

Доказательство. По условию:

$$\begin{aligned} 2\overrightarrow{AB} \cdot \overrightarrow{CD} &= 2(\overrightarrow{OB} - \overrightarrow{OA})(\overrightarrow{OD} - \overrightarrow{OC}) = \\ &= 2(\overrightarrow{OB} \cdot \overrightarrow{OD} - \overrightarrow{OB} \cdot \overrightarrow{OC} - \overrightarrow{OA} \cdot \overrightarrow{OD} + \overrightarrow{OA} \cdot \overrightarrow{OC}) = \\ &= 2 \cdot \frac{1}{2}(b\bar{d} + \bar{b}d - b\bar{c} - \bar{b}c - a\bar{d} - \bar{a}d + a\bar{c} + \bar{a}c) = \\ &= (a - b)(\bar{c} - \bar{d}) + (\bar{a} - \bar{b})(c - d) \blacksquare \end{aligned}$$

Коллинеарность

Коллинеарными называются:

- *точки*, которые лежат на одной прямой;
- *векторы*, которые лежат на одной прямой или на параллельных прямых.

Критерий коллинеарности точек A, B с O:

$$\frac{a}{b} = \overline{\left(\frac{a}{b}\right)} \quad \text{или} \quad \begin{cases} a = 0 \\ b = 0 \end{cases}$$

Доказательство. Очевидно, что:

$$\begin{cases} \arg a - \arg b = 0 \\ \arg a - \arg b = \pm\pi \end{cases} \Rightarrow \arg \frac{a}{b} = 0; \pm\pi$$

По определению аргумента комплексного числа:

$$\frac{a}{b} \text{ — действительное число} \Rightarrow \frac{a}{b} = \overline{\left(\frac{a}{b}\right)} \blacksquare$$

Критерий коллинеарности векторов \overrightarrow{AB} , \overrightarrow{CD} :

$$\frac{b - a}{d - c} = \overline{\left(\frac{b - a}{d - c}\right)} \quad \text{или} \quad \begin{cases} \overrightarrow{AB} = \vec{0} \\ \overrightarrow{CD} = \vec{0} \end{cases}$$

Доказательство. По определению комплексных чисел:

$$\overrightarrow{AB} \sim b - a, \quad \overrightarrow{CD} \sim d - c$$

По критерию коллинеарности двух точек с O :

$$\frac{b-a}{d-c} = \overline{\left(\frac{b-a}{d-c}\right)} \blacksquare$$

Если A, B, C, D лежат на одной окружности, то:

$$\overrightarrow{AB} \parallel \overrightarrow{CD} \iff \frac{b}{d} = \frac{a}{c}$$

Критерий коллинеарности трёх точек:

$$\frac{b-a}{c-a} = \overline{\left(\frac{b-a}{c-a}\right)} \quad \text{или} \quad \begin{cases} \overrightarrow{AB} = \vec{0} \\ \overrightarrow{AC} = \vec{0} \end{cases}$$

Доказательство. Очевидно, что:

$$\overrightarrow{AB} \parallel \overrightarrow{AC} \iff A, B, C \text{ коллинеарны}$$

По критерию коллинеарности векторов:

$$\frac{b-a}{c-a} = \overline{\left(\frac{b-a}{c-a}\right)} \blacksquare$$

Уравнение секущей AB :

$$(\bar{a} - \bar{b})z + (b - a)\bar{z} + a\bar{b} - b\bar{a} = 0$$

Доказательство. Нет и не будет: раздел будет снесён.

Вычислительная геометрия

Деление отрезка в отношении

Точка C делит отрезок AB в отношении $\lambda \in \mathbb{R}$, если:

$$\begin{cases} C \in AB \\ \overrightarrow{AC} = \lambda \overrightarrow{CB} \\ \lambda \neq -1 \end{cases}$$

Теорема. Пусть C делит AB в отношении $\lambda \in \mathbb{R}$. Тогда координаты точки C равны:

$$x_C = \frac{x_A + \lambda x_B}{1 + \lambda} \quad y_C = \frac{y_A + \lambda y_B}{1 + \lambda}$$

Доказательство. По условию:

$$\overrightarrow{AC} = \lambda \overrightarrow{CB} \iff \overrightarrow{OC} - \overrightarrow{OA} = \lambda (\overrightarrow{OB} - \overrightarrow{OC})$$

По теореме Фалеса:

$$\begin{cases} x_C - x_A = \lambda (x_B - x_C) \\ y_C - y_A = \lambda (y_B - y_C) \end{cases} \iff \begin{cases} x_C = \frac{x_A + \lambda x_B}{1 + \lambda} \\ y_C = \frac{y_A + \lambda y_B}{1 + \lambda} \end{cases} \quad \blacksquare$$

Направляющий вектор

Направляющим называется вектор, параллельный данной прямой:

$$\vec{l} = \begin{pmatrix} A \\ B \end{pmatrix} \implies \begin{cases} x = x_0 + At \\ y = y_0 + Bt \end{cases}$$

Теорема. Уравнение прямой по точке и направляющему вектору:

$$\frac{A}{B} = \frac{x - x_0}{y - y_0}$$

Уравнение легко выводится из определения направляющего вектора.

Свойство. Связь угла между прямыми и их направляющими векторами:

$$\cos \angle(a, b) = |\cos \angle(\vec{a}, \vec{b})|$$

Коллинеарность

Коллинеарными называются:

- *точки*, которые лежат на одной прямой;
- *векторы*, которые лежат на одной прямой или на параллельных прямых.

Критерий коллинеарности двух векторов:

$$\vec{a} = \lambda \vec{b}, \lambda \in \mathbb{R} \iff \begin{cases} x_a = \lambda x_b \\ y_a = \lambda y_b \end{cases}$$

В частности, нулевой вектор коллинеарен *любому* вектору:

$$\vec{0} = 0\vec{a}$$

Уравнение секущей по двум известным точкам:

$$A(x_a, y_a), B(x_b, y_b) \implies \frac{x - x_a}{x_b - x_a} = \frac{y - y_a}{y_b - y_a}$$

Доказательство. Пусть \overrightarrow{AX} , \overrightarrow{AB} — коллинеарные векторы.

По критерию коллинеарности двух векторов:

$$\begin{cases} x - x_a = \lambda(x_b - x_a) \\ y - y_a = \lambda(y_b - y_a) \end{cases} \iff \lambda = \frac{x - x_a}{x_b - x_a} = \frac{y - y_a}{y_b - y_a} \blacksquare$$

Скалярное произведение

Теорема. Косинус угла α между векторами \vec{a} и \vec{b} равен:

$$\cos \alpha = \frac{x_a x_b + y_a y_b}{|\vec{a}| |\vec{b}|}$$

Доказательство. Отложим векторы $\overrightarrow{AB} = \vec{a}$, $\overrightarrow{AC} = \vec{b}$ от

начала координат.

По теореме косинусов:

$$BC^2 = AB^2 + AC^2 - 2 |\vec{AB}| |\vec{AC}| \cos \alpha \Rightarrow$$
$$\cos \alpha = \frac{AB^2 + AC^2 - BC^2}{2 |\vec{AB}| |\vec{AC}|}$$

По теореме Пифагора:

$$\begin{cases} AB^2 = x_a^2 + y_a^2 \\ AC^2 = x_b^2 + y_b^2 \\ BC^2 = (x_a - x_b)^2 + (y_a - y_b)^2 \end{cases} \Rightarrow$$
$$AB^2 + AC^2 - BC^2 = 2(x_a x_b + y_a y_b) \Rightarrow$$
$$\cos \alpha = \frac{x_a x_b + y_a y_b}{|\vec{a}| |\vec{b}|} \blacksquare$$

Скалярное произведение векторов \vec{a}, \vec{b} — величина:

$$x_a x_b + y_a y_b =: \vec{a} \cdot \vec{b}$$

Теорема. Скалярное произведение двух векторов равно:

$$\vec{a} \cdot \vec{b} = |\vec{a}| |\vec{b}| \cos \alpha,$$

α — угол между векторами.

Следствие. Если векторы \vec{a} и \vec{b} коллинеарны, то:

- $\vec{a} \cdot \vec{b} > 0 \Rightarrow$ векторы сонаправлены
- $\vec{a} \cdot \vec{b} < 0 \Rightarrow$ векторы несонаправлены
- $\vec{a} \cdot \vec{b} = 0 \Rightarrow$ один из векторов нулевой

Ориентированный угол

Ориентированным называется угол α между \vec{a} и \vec{b} , на который нужно повернуть \vec{a} , чтобы он был сонаправлен с \vec{b} :

$$\angle(\vec{a}, \vec{b}) \text{ — обозначение, } \alpha \in (-\pi; \pi]$$

Знак ориентированного угла:

- **положительный**, если поворот происходит в *положительном* направлении системы координат;
- **отрицательный**, если поворот происходит в *отрицательном* направлении системы координат;
- **нуль**, если вектора *сонаправлены*.

Косое произведение

Косое произведение векторов \vec{a}, \vec{b} — величина:

$$x_a x_b - y_a y_b =: \vec{a} \wedge \vec{b}$$

Теорема. Косое произведение двух векторов равно:

$$\vec{a} \wedge \vec{b} = |\vec{a}| |\vec{b}| \sin \alpha,$$

α — угол между векторами.

Теорема. Знак косого произведения векторов *совпадает* со знаком ориентированного угла.

Доказательство вытекает из *чётности* синуса угла между векторами.

Взаимное расположение

Расположение *точки* A относительно *прямой, луча или отрезка* BC :

- $\angle(\overrightarrow{BA}, \overrightarrow{BC}) > 0 \Rightarrow A$ лежит в **левой** полуплоскости;
- $\angle(\overrightarrow{BA}, \overrightarrow{BC}) < 0 \Rightarrow A$ лежит в **правой** полуплоскости;
- $\angle(\overrightarrow{BA}, \overrightarrow{BC}) = 0 \Rightarrow A$ **коллинеарна** прямой BC .

Взаимное расположение *двух отрезков или лучей* AB и CD :

- концы *обоих* отрезков лежат в *разных* полуплоскостях относительно друг друга \Rightarrow отрезки **пересекаются**;
- концы *одного* отрезка лежат в *одной* полуплоскости относительно другого \Rightarrow отрезки **не пересекаются**;
- концы *одного* отрезка лежат *на* прямой другого отрезка:

- > конец одного отрезка лежит *на* другом \Rightarrow отрезки имеют **общий подотрезок**;
- > концы одного отрезка *не* лежат на другом \Rightarrow отрезки **не пересекаются**.

Ориентированная площадь

Ориентированной называется площадь многоугольника, которая обладает знаком его ориентированных углов.

Теорема. Ориентированная площадь треугольника равна половине *косого произведения* векторов ориентированного угла.

Ориентированная площадь — *аддитивная* величина, к основным методам её расчёта относят:

- метод трапеций;
- метод треугольников.

Метод трассировки луча

Задача. На плоскости даны многоугольник и точка. Решить вопрос о принадлежности точки многоугольнику.

Алгоритм трассировки луча:

- 1) Проверить принадлежность точки стороне многоугольника: если *истина*, остановить алгоритм.
- 2) Выпустить из точки в случайном направлении луч.
- 3) Посчитать число n пересечений луча со сторонами многоугольника:

$$\begin{cases} n \equiv 0 \pmod{2} \Rightarrow \text{точка снаружи} \\ n \equiv 1 \pmod{2} \Rightarrow \text{точка внутри} \end{cases}$$

Метод заметающей прямой

Да.

Уравнения кривых

Алгебраическая кривая — это...

Кривые первого порядка — прямая.

Уравнение ромба

Кривые второго порядка (*коники*) — эллипс, парабола, гипербола.

ГМТ

Биссектриса — это

Биссекторная плоскость — это...

Свойство равноудалённости, расстояния и пр..

Гомотетия также

Геометрические тела

Правильный тетраэдр, пирамида...

Прямые в пространстве

По взаимному расположению прямые бывают трёх видов:

- **пересекающиеся**: имеют *общую точку*
- **параллельные**: *лежат* в одной плоскости и *не имеют* общих точек
- **скрещивающиеся**: *не лежат* в одной плоскости

Свойство. Угол между скрещивающимися прямыми равен углу между двумя *пересекающимися* прямыми, которые соответственно им параллельны.

Сонаправленные лучи лежат в одной полуплоскости на параллельных прямых.

Свойство. Углы с сонаправленными сторонами *равны*.

Предел последовательности

Определение

Предел последовательности $\{x_n\}$ — такое a , что

$$\forall \varepsilon > 0 \exists N: \forall n > N \ x_n \in U_\varepsilon(a).$$

Упрощённая запись $\lim_{n \rightarrow \infty} x_n = a$ или $n \rightarrow \infty, x_n \rightarrow a$.

Этот оператор «дистрибутивен» относительно сложения, умножения и деления (*предел знаменателя не равен нулю*).

Частичным называется предел подпоследовательности.

Свойства

Сходимость \implies ограниченность.

Доказательство. Пусть $\lim_{n \rightarrow \infty} x_n = a$. По определению:

$$\forall \varepsilon > 0 \exists N: \forall n > N \ x_n \in U_\varepsilon(a)$$

По «дистрибуции» модуля относительно сложения:

$$|x_n| = |x_n - a + a| \leq |x_n - a| + |a| < \varepsilon + |a|$$

Положим, что $\forall m \leq N \ L = \max(|\{x_m\}|, \varepsilon + |a|) \implies |x_n| \leq L$. ■

Предельный переход. Пусть $n \rightarrow \infty, x_n \rightarrow a, y_n \rightarrow b$.

Тогда справедливо следствие:

$$x_n \leq y_n \text{ или } x_n < y_n \implies a \leq b$$

Доказательство. По определению предела:

$$\forall \varepsilon > 0 \exists N: \forall n > N \ x_n \in U_\varepsilon(a), y_n \in U_\varepsilon(b)$$

Следовательно,

$$+ \begin{cases} x_n \leq y_n \\ a - x_n < \varepsilon \\ y_n - b < \varepsilon \end{cases} \iff \begin{cases} y_n - x_n \geq 0 \\ y_n - x_n < 2\varepsilon + b - a \end{cases} \iff \frac{a - b}{2} < \varepsilon$$

Так как ε — сколь угодно малое положительное число, то $a - b \leq 0 \iff a \leq b$. □

При $x_n < y_n$ доказательство аналогично. ■

Теорема о промежуточной функции. Пусть $n \rightarrow \infty$, $x_n, y_n \rightarrow a$. Тогда справедливо следствие:

$$\forall \{z_n\}: x_n \leq z_n \leq y_n \implies z_n \rightarrow a$$

Доказательство. По определению предела:

$$\forall \varepsilon > 0 \exists N: \forall n > N \ x_n, y_n \in U_\varepsilon(a)$$

Следовательно,

$$a - \varepsilon < x_n \leq z_n \leq y_n < a + \varepsilon \implies z_n \in U_\varepsilon(a) \implies \lim_{n \rightarrow \infty} z_n = a. \blacksquare$$

Условие Коши

Последовательность $\{x_n\}$ удовлетворяет **условию Коши** (является *фундаментальной*), если

$$\forall \varepsilon > 0 \exists N: \forall n, m > N \ |x_n - x_m| < \varepsilon.$$

Фундаментальность \implies ограниченность.

Доказательство. По условию Коши:

$$\forall \varepsilon > 0 \exists N: \forall n, m > N \ |x_n - x_m| < \varepsilon$$

По «дистрибуции» модуля относительно сложения:

$$\begin{aligned} \begin{cases} |x_n - x_m| < \varepsilon \\ |x_n| = |x_n - x_m + x_m| \end{cases} &\iff \begin{cases} |x_n - x_m| < \varepsilon \\ |x_n| \leq |x_n - x_m| + |x_m| \end{cases} \iff \\ &|x_n| < \varepsilon + |x_m| \end{aligned}$$

Положим, что $\forall k \leq N \ L = \max(|\{x_k\}|, \varepsilon + |x_m|) \implies |x_n| \leq L. \blacksquare$

Принцип компактности отрезка

Ограниченность \implies частичная сходимость:

$$\forall \{x_n\} \in [a; b] \exists \{n_k\} \uparrow: \lim_{k \rightarrow \infty} x_{n_k} = \xi$$

Доказательство. По принципу Кантора:

$$\forall k \in \mathbb{N} \exists! \xi \in [a_k; b_k] \subset [a_{k-1}; b_{k-1}] \iff$$

$$\lim_{k \rightarrow \infty} a_k = \lim_{k \rightarrow \infty} b_k = \xi$$

Образуем подпоследовательность:

$$\{x_{n_k} \mid \{n_k\} \uparrow, x_{n_k} \in [a_k; b_k]\}$$

По теореме о промежуточной функции:

$$a_k \leq x_{n_k} \leq b_k \implies \lim_{k \rightarrow \infty} x_{n_k} = \xi \blacksquare$$

Частичный предел фундаментальной последовательности является её пределом.

Доказательство. Пусть $\{x_n\}$ фундаментальна \implies она ограничена.

По принципу компактности отрезка $\lim_{k \rightarrow \infty} x_{n_k} = a$.

По условию Коши:

$$\forall \varepsilon/2 > 0 \exists N: \forall n, m > N \quad |x_n - x_m| < \varepsilon/2$$

Зафиксируем n . При $x_m = x_{n_k} > N$ перейдём к пределу:

$$|x_n - a| \leq \varepsilon/2 < \varepsilon \iff \lim_{k \rightarrow \infty} x_{n_k} = a \blacksquare$$

Критерий Коши

Сходимость \iff фундаментальность.

Доказательство \implies . По определению предела:

$$\forall \varepsilon > 0 \exists N: \forall n > N \quad x_n \in U_{\varepsilon/2}(a)$$

Значит, $\forall n, m > N \quad |x_n - x_m| = |(x_n - a) + (a - x_m)|$.

По «дистрибуции» модуля относительно сложения:

$$|x_n - x_m| \leq |x_n - a| + |x_m - a| < \varepsilon/2 + \varepsilon/2 = \varepsilon \blacksquare$$

Доказательство \Leftarrow . Пусть $\{x_n\}$ фундаментальна \implies она ограничена \implies по принципу компактности отрезка она частично сходится к $c \implies$ по условию Коши и принципу компактности отрезка она сходится к c . \blacksquare

Теорема Вейерштрасса

Монотонность \implies сходимость:

$$\left[\begin{array}{l} \forall \{x_n\} \nearrow \lim_{n \rightarrow \infty} x_n = \sup\{x_n\} \\ \forall \{y_n\} \searrow \lim_{n \rightarrow \infty} y_n = \inf\{y_n\} \end{array} \right.$$

Доказательство. По определению точной верхней границы:

$$\forall n \in \mathbb{N} \ x_n \leq \sup\{x_n\}$$

Так как последовательность неубывает, то

$$\forall \epsilon > 0 \ \exists N: \forall n > N \ x_n \in U_\epsilon(\sup\{x_n\}) \implies$$

$$\lim_{n \rightarrow \infty} x_n = \sup\{x_n\}. \quad \square$$

Для $\{y_n\} \searrow$ доказательство аналогично. ■

Предел функции

Определение

Предел функции f в точке x_0 — такое a , что (*О.Л. Коши*)

$$\forall \varepsilon > 0 \exists \delta > 0: \underbrace{\forall x \in \overset{\circ}{U}_{\delta}(x_0)}_I, \underbrace{f(x) \in U_{\varepsilon}(a)}_{II}.$$

I — функция f определена в какой-либо проколотой δ -окрестности точки x_0 ;

II — функция f имеет образ в какой-либо проколотой ε -окрестности точки a .

Предел функции f в точке x_0 — такое a , что (*Э. Гейне*)

$$\forall \{x_n\} \in D_f: \lim_{n \rightarrow \infty} x_n = x_0 \ (x_n \neq x_0) \implies \lim_{n \rightarrow \infty} f(x_n) = a.$$

Упрощённая запись $\lim_{x \rightarrow x_0} f(x) = a$ или $x \rightarrow x_0, f(x) \rightarrow a$.

Бесконечно малая и большая

Бесконечно малой (*б.м.*) называется такая функция $\alpha(x)$ при $x \rightarrow x_0$, что:

$$\lim_{x \rightarrow x_0} \alpha(x) = 0$$

Связь предела и б.м. Если функция f имеет предел $\lim_{x \rightarrow x_0} f(x) = a$, то справедливо:

$$f(x) = a + \underset{x \rightarrow x_0}{\alpha(x)}, \ \alpha \text{ — б.м.}$$

Бесконечно большой (*б.б.*) называется такая функция $y(x)$ при $x \rightarrow x_0$, что:

$$\lim_{x \rightarrow x_0} y(x) = \infty$$

Связь бесконечно малой и большой. Верен факт:

$$\underset{x \rightarrow x_0}{\alpha(x)} \text{ — б.м., } \forall x \in \overset{\circ}{U}_{x_0} \ \alpha(x) \neq 0 \iff \frac{1}{\alpha} \text{ — б.б.}$$

Композиция функций

Пусть f, g — функции. Тогда:

$$\begin{cases} \lim_{x \rightarrow x_0} f(x) = y_0 \\ \lim_{y \rightarrow y_0} g(y) = z_0 \end{cases} \implies \begin{cases} \lim_{x \rightarrow x_0} (g \circ f)(x) = z_0 \\ f(x) \neq y_0 \end{cases}$$

Доказательство. Пусть $g \circ f = \varphi$; по определению предела:

$$\begin{cases} \forall \varepsilon > 0 \exists \delta > 0: \forall y \in \mathring{U}_\delta(y_0) \subseteq D_g, f(x) \in U_\varepsilon(z_0) \\ \forall \delta > 0 \exists \sigma > 0: \forall x \in \mathring{U}_\sigma(x_0) \subseteq D_f, g(y) \in U_\delta(y_0) \end{cases}$$

Из $\mathring{U}_\delta(y_0) \cap U_\delta(y_0) = \mathring{U}_\delta(y_0)$ следует:

$$\begin{cases} \forall \varepsilon > 0 \exists \sigma > 0: \forall x \in \mathring{U}_\sigma(x_0) \subseteq D_f, \varphi(x) \in U_\varepsilon(z_0) \\ y \neq y_0 \implies f(x) \neq y_0 \end{cases} \iff$$

$$\lim_{x \rightarrow x_0} \varphi(x) = z_0, f(x) \neq y_0 \blacksquare$$

Односторонний предел

Односторонним (*правым или левым*) называется предел функции, который определён в терминах односторонних окрестностей (*монотонных последовательностей*):

$$\lim_{x \rightarrow x_0 + 0} f(x) = a \quad \text{или} \quad x \rightarrow x_0 + 0, f(x) \rightarrow a$$

$$\lim_{x \rightarrow x_0 - 0} f(x) = a \quad \text{или} \quad x \rightarrow x_0 - 0, f(x) \rightarrow a$$

Существование предела равносильно существованию *равных* односторонних пределов:

$$\lim_{x \rightarrow x_0} f(x) \iff \lim_{x \rightarrow x_0 + 0} f(x) = \lim_{x \rightarrow x_0 - 0} f(x)$$

Асимптота

Асимптота — прямая, к которой *неограниченно* приближается кривая, но не сливается с ней.

Горизонтальная асимптота для графика функции f задаётся уравнением:

$$y = \lim_{x \rightarrow \infty} f(x)$$

Наклонная асимптота для графика функции f задаётся уравнением $y = kx + b$, где

$$k = \lim_{x \rightarrow \infty} \frac{f(x)}{x},$$
$$b = \lim_{x \rightarrow \infty} (f(x) - kx).$$

Вертикальная асимптота для графика функции f задаётся уравнением $x = a$, где

$$\lim_{x \rightarrow a+0} f(x) = \infty \text{ или } \lim_{x \rightarrow a-0} f(x) = \infty.$$

Непрерывность

Пусть $f: X \rightarrow Y$ — функция. Тогда:

$x - x_0 =: \Delta x$ — *приращение аргумента* в точке x_0

$f(x) - f(x_0) =: \Delta f$ — *приращение функции* в точке x_0

Функция f **непрерывна** в точке x_0 , если

$$\lim_{x \rightarrow x_0} f(x) = f(x_0) \quad \text{или} \quad \Delta x \rightarrow 0, \Delta f \rightarrow 0.$$

Односторонняя непрерывность в точке x_0 определяется через односторонние пределы.

Непрерывными в точке x_0 являются *сумма, произведение, частное (предел знаменателя не равен нулю) и композиция* непрерывных в ней функций.

Функция f **непрерывна** на промежутке $[a; b]$, если она непрерывна в каждой точке этого промежутка:

$$f \in \mathbb{C}[a; b] \text{ — нотация}$$

Предел под непрерывной функцией. Пусть f, g — функции, g непрерывна в точке x_0 . Тогда:

$$\lim_{x \rightarrow x_0} f(x) = a \implies \lim_{x \rightarrow x_0} (g \circ f)(x) = g\left(\lim_{x \rightarrow x_0} f(x)\right)$$

Доказательство схоже с теоремой о пределе *композиции функций*.

Замечательные пределы

Когда-нибудь это будет пояснено (*я надеюсь*):

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1 \quad \lim_{x \rightarrow \infty} \left(1 - \frac{1}{x}\right)^x = e$$

Теорема о промежуточном значении

Пусть $f \in \mathbb{C}[a; b]$. Тогда справедливо:

$$\forall c \in [f(a); f(b)] \exists \xi \in [a; b]: c = f(\xi)$$

Доказательство. По принципу Кантора:

$$\forall n \in \mathbb{N} \exists \xi \in [a_n; b_n] \subset [a_{n-1}; b_{n-1}] \subseteq X \Rightarrow \\ n \rightarrow \infty, a_n, b_n \rightarrow \xi$$

По определению непрерывности функции на промежутке:

$$n \rightarrow \infty, f(a_n), f(b_n) \rightarrow f(\xi)$$

По теореме о промежуточной функции:

$$f(a_n) \leq c \leq f(b_n) \Rightarrow c = f(\xi) \blacksquare$$

Метод бисекции. Пусть $f \in \mathbb{C}[a; b]$. Тогда справедливо:

$$\operatorname{sgn} f(a) \neq \operatorname{sgn} f(b) \Rightarrow \exists c \in [a; b]: f(c) = 0$$

Используется, если нужно найти *примерный* нуль функции.

Критерий Коши

Сходимость \iff выполнение условия Коши:

$$\forall \varepsilon > 0 \exists \delta > 0: \forall x', x'' \in \mathring{U}_\delta(x_0) |f(x') - f(x'')| < \varepsilon$$

Доказательство \Rightarrow . По определению предела:

$$\forall \varepsilon > 0 \exists \delta > 0: \mathring{U}_\delta(x_0) \subseteq D_f, U_{\varepsilon/2}(a) \cap E_f \neq \emptyset$$

Пусть $x', x'' \in \mathring{U}_\delta(x_0)$; по неравенству треугольника:

$$|f(x') - f(x'')| \leq |f(x') - a| + |f(x'') - a| < \varepsilon/2 + \varepsilon/2 = \varepsilon \blacksquare$$

Доказательство \Leftarrow . По условию Коши:

$$\exists \{x_n\} \in D_f: \lim_{n \rightarrow \infty} x_n = x_0, x_n \neq x_0$$

Последовательности $\{f(x_n)\}$ фундаментальны \Rightarrow сходятся.

По фундаментальности и сходимости к одной точке x_0 :

$$\lim_{x \rightarrow x_0} f(x) = a \blacksquare$$

Теорема Вейерштрасса

Пусть $f \in C[a; b]$. Тогда в некоторых точках отрезка функция достигает своих точных верхней и нижней границ на $[a; b]$.

Доказательство. Пусть $\sup f([a; b]) =: M$, $\inf f([a; b]) =: m$.

По определению точных верхней и нижней границ:

$$\forall x \in [a; b] f(x) \in [m; M]$$

По принципу компактности отрезка:

$$\lim_{n \rightarrow \infty} f(x_n) = M \quad \lim_{k \rightarrow \infty} x_{n_k} = \xi$$

По определению непрерывности:

$$\lim_{k \rightarrow \infty} f(x_{n_k}) = f(\xi) \Rightarrow f(\xi) = M \blacksquare$$

Дифференциальное исчисление

Дифференцируемость

Дифференцируемой («линейной в малом») в точке x_0 называется такая функция f , для которой справедливо:

$$\Delta f = (k + \alpha(x)) \Delta x, \quad \alpha \text{ — б.м.} \\ \Delta x \rightarrow 0$$

Односторонняя дифференцируемость в точке x_0 определяется через односторонние пределы.

Дифференциал функции f — линейная часть Δf :

$$k \Delta x =: df$$

Производная в точке x_0 — предел вида: (Ж.Л. Лагранж)

$$k = \lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x} =: f'(x_0)$$

Свойства

Таблица «дистрибуции» производной:

$$\begin{array}{ll} (f + g)' = f' + g' & \left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2} \\ (f \cdot g)' = f'g + fg' & \\ (f \circ g)' = (f' \circ g)g' & (kf)' = kf', \quad k = \text{const} \end{array}$$

Дифференцируемость \Rightarrow непрерывность.

Доказательство. По определению производной:

$$\lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x} = f'(x_0) \iff \frac{\Delta f}{\Delta x} = f'(x_0) + \alpha(x) \Delta x \iff$$

$$\Delta f = \Delta x (f'(x_0) + \alpha(x) \Delta x) \implies \Delta x \rightarrow 0, \Delta f \rightarrow 0 \blacksquare$$

Производная обратной функции. Пусть $y = f(x)$ — дифференцируемая функция. Тогда справедливо:

$$f^{-1'}(y) = \frac{1}{f'(x)}, \quad f'(x) \neq 0$$

Доказательство. По условию запишем тождество:

$$\frac{\Delta f}{\Delta x} = 1 : \frac{\Delta x}{\Delta f}$$

По предельному переходу и непрерывности функций:

$$\begin{aligned} \lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x} = 1 : \lim_{\Delta f \rightarrow 0} \frac{\Delta x}{\Delta f} &\stackrel{\text{опр}}{\iff} f'(x) = 1 : f^{-1}'(y) \iff \\ &\iff f^{-1}'(y) = \frac{1}{f'(x)}, f'(x) \neq 0 \blacksquare \end{aligned}$$

Инвариантность. Пусть $f(x)$, $g(x)$ — дифференцируемые функции. Тогда верно:

$$df = f'(x)dx \implies d(f \circ g) = f'(g(x))dg$$

Доказательство. По правилам дифференцирования:

$$d(f \circ g) = f'(g(x))dx = f'(g(x))g'(x)dx = f'(g(x))dg \blacksquare$$

Элементарные производные

Таблица производных элементарных функций:

$$\begin{array}{lll} C' = 0 & (x^n)' = nx^{n-1}, n \neq 0 & \ln' x = 1/x \\ \sin' \alpha = \cos \alpha & & \cos' \alpha = -\sin \alpha \\ \operatorname{tg}' \alpha = 1/\cos^2 \alpha & & \operatorname{ctg}' \alpha = -1/\sin^2 \alpha \\ \arcsin' x = 1/\sqrt{1-x^2} & & \arccos' x = -1/\sqrt{1-x^2} \\ \operatorname{arctg}' x = 1/(1+x^2) & & \operatorname{arcctg}' x = -1/(1+x^2) \end{array}$$

Касательная

Касательная к кривой в точке x_0 — прямая, которая проходит через x_0 и представляет *предельное* положение секущей при $x \rightarrow x_0$, или $\Delta x \rightarrow 0$.

Геометрический смысл производной. Угловым коэффициентом (*тангенс*) касательной к графику функции f равен *производной* в этой точке:

$$k = \operatorname{tg} \alpha = f'(x_0)$$

Доказательство. По определению касательной:

$$\lim_{x \rightarrow x_0} \frac{\Delta f}{\Delta x} = \operatorname{tg} \alpha = k$$

По определению производной:

$$f'(x_0) = \operatorname{tg} \alpha = k \blacksquare$$

Уравнение касательной к графику функции f в точке x_0 имеет вид:

$$f'(x_0) = \frac{y - f(x_0)}{x - x_0}$$

Доказательство. По уравнению секущей графика f :

$$\frac{x - x_0}{x_1 - x_0} = \frac{y - f(x_0)}{f(x_1) - f(x_0)} \implies y - f(x_0) = \frac{f(x_1) - f(x_0)}{x_1 - x_0} (x - x_0)$$

По определению касательной:

$$y - f(x_0) = \lim_{x_1 \rightarrow x_0} \frac{f(x_1) - f(x_0)}{x_1 - x_0} (x - x_0) = f'(x_0) (x - x_0) \blacksquare$$

Нормаль

Нормаль к кривой в точке x_0 — прямая, которая проходит через x_0 и образует с касательной в x_0 *прямой угол*:

$$\vec{n} = \begin{pmatrix} A \\ B \end{pmatrix} \text{ для } Ax + By + C = 0$$

Доказательство. По скалярному произведению векторов:

$$\begin{aligned} (\vec{f}' \cdot \vec{n}) &= 0 \implies A(x - x_0) + B(y - y_0) = 0 \\ Ax - Ax_0 + By - By_0 &= 0 \end{aligned}$$

Пусть $C = -(Ax_0 + By_0)$, тогда:

$$Ax + By + C = 0 \blacksquare$$

Теорема. Прямые f_1 и f_2 перпендикулярны, если:

$$(\vec{n}_1 \cdot \vec{n}_2) = 0, \text{ или } k_1 = -\frac{1}{k_2}$$

Доказательство. По скалярному произведению векторов:

$$(\vec{n}_1 \cdot \vec{n}_2) = 0 \implies \cos \alpha = 0 \implies \alpha = \frac{\pi}{2} \quad \square$$

$$(\vec{n}_1 \cdot \vec{n}_2) = 0 \implies k_1 k_2 + 1 = 0 \implies k_1 = -\frac{1}{k_2} \quad \blacksquare$$

Теорема. Уравнение прямой по точке и нормальному вектору:

$$(\vec{n} \cdot \vec{l}) = 0, \text{ или } A(x - x_0) + B(y - y_0) = 0$$

Промежутки монотонности

Если функция f дифференцируема в точке x_0 , то

$$\begin{cases} f'(x_0) > 0 \implies f \uparrow \text{ около } x_0 \\ f'(x_0) < 0 \implies f \downarrow \text{ около } x_0 \end{cases}.$$

Доказательство. По определению производной:

$$f'(x_0) > 0 \iff \lim_{\Delta x \rightarrow 0} \frac{\Delta f}{\Delta x} > 0 \iff \frac{\Delta f}{\Delta x} > o(\Delta x)$$

При достаточно малом Δx верно:

$$\frac{\Delta f}{\Delta x} > 0 \iff \begin{cases} \Delta f, \Delta x > 0 \\ \Delta f, \Delta x < 0 \end{cases} \iff f \uparrow \text{ около } x_0 \quad \square$$

Для $f'(x_0) < 0$ доказательство аналогично. \blacksquare

Экстремум

Локальный максимум функции f — такая точка x_0 , что:

$$\exists \delta > 0: \sup U_\delta(x_0) = f(x_0)$$

Локальный минимум функции f — такая точка x_0 , что:

$$\exists \delta > 0: \inf U_\delta(x_0) = f(x_0)$$

Их объединяют в точки **экстремума**.

Критической называется такая точка x_0 , в которой:

$$\begin{cases} f'(x_0) = 0 \text{ (стационарна)} \\ f'(x_0) = \text{undefined} \end{cases}$$

Экстремум — *критическая точка* первого порядка.
(не наоборот)

Доказательство. По определению локального максимума:

$$\exists \delta > 0: \forall x \in \overset{\circ}{U}_\delta(x_0) \quad f(x_0) > f(x)$$

Производная в точке x_0 либо существует, либо нет. \square

Допустим, она существует; по определению производной:

$$\lim_{x \rightarrow x_0} \frac{\Delta f}{\Delta x} = f'(x_0)$$

По предельному переходу:

$$\begin{aligned} \begin{cases} \Delta x > 0 \Rightarrow \Delta f / \Delta x < 0 \Rightarrow f'(x_0) \leq 0 \\ \Delta x < 0 \Rightarrow \Delta f / \Delta x > 0 \Rightarrow f'(x_0) \geq 0 \end{cases} &\iff \\ 0 \leq f'(x_0) \leq 0 &\iff f'(x_0) = 0 \quad \square \end{aligned}$$

Для локального минимума доказательство аналогично. \blacksquare

Условие. Критическая точка — *экстремум*, если в ней первая производная *меняет знак*.

Доказательство. По определению критической точки:

$$\begin{cases} f'(x_0) = 0 \\ f'(x_0) = \text{undefined} \end{cases}$$

Допустим для определённости:

$$\begin{cases} \exists \delta > 0: \forall x \in \overset{\circ}{U}_{\delta-}(x_0) \quad f'(x) > 0 \\ \exists \delta > 0: \forall x \in \overset{\circ}{U}_{\delta+}(x_0) \quad f'(x) < 0 \end{cases}$$

По промежуткам монотонности:

$$\begin{cases} f \uparrow \text{ на } U_{\delta-}(x_0) \\ f \downarrow \text{ на } U_{\delta+}(x_0) \end{cases} \iff x_0 \text{ — локальный максимум } \square$$

Для локального минимума доказательство аналогично. ■

Условие. Критическая точка — *экстремум*, если в ней вторая производная *ненулевая*, причём:

$$f''(x_0) < 0 \implies x_0 \text{ — локальный максимум}$$

$$f''(x_0) > 0 \implies x_0 \text{ — локальный минимум}$$

Выпуклость

Кривая f **выпукла вверх** в точке M , если в окрестности точки f лежит *ниже* своей касательной в этой точке.

Кривая f **выпукла вниз** в точке M , если в окрестности точки f лежит *выше* своей касательной в этой точке.

Кривая f **выпукла** на интервале $(a; b)$, если она выпукла в *каждой* её точке.

Условие. Пусть f дважды дифференцируема на $(a; b)$:

$$\forall x \in (a; b) \ f''(x) \leq 0 \implies f \text{ выпукла вверх}$$

$$\forall x \in (a; b) \ f''(x) \geq 0 \implies f \text{ выпукла вниз}$$

В **точке перегиба** происходит смена *характера выпуклости* функции.

Точка перегиба — *критическая точка* второго порядка.
(не наоборот)

Условие. Критическая точка — *точка перегиба*, если в ней вторая производная *меняет знак*.

Теоремы о среднем

Теорема Ролля. Пусть f дифференцируема на $[a; b]$. Тогда:

$$f(a) = f(b) \implies \exists \xi \in (a; b) : f'(\xi) = 0$$

Доказательство. По теореме Вейерштрасса:

$$f(m) = \inf f([a; b]) \quad f(M) = \sup f([a; b])$$

По условию существования экстремума:

$$f(a) = f(b) = f(m) \implies f'(M) = 0 \quad \square$$

При $f(m) = f(M)$ функция — константа на $[a; b]$, производная которой равна нулю. ■

Теорема Лагранжа. Пусть f дифференцируема на $[a; b]$. Тогда верно:

$$\exists \xi \in (a; b) : f'(\xi) = \frac{\Delta f}{\Delta x}$$

Доказательство. Пусть $\varphi(x) := f(x) - \lambda x$.

Подберём λ так, чтобы $\varphi(a) = \varphi(b)$:

$$\begin{aligned} f(a) - \lambda a &= f(b) - \lambda b \iff (b - a)\lambda = f(b) - f(a) \iff \\ &\iff \lambda = \frac{f(b) - f(a)}{b - a} \end{aligned}$$

По теореме Ролля:

$$\begin{aligned} \exists \xi \in (a; b) : \varphi'(\xi) &= 0 \iff f'(\xi) - \lambda = 0 \iff \\ \iff \lambda &= f'(\xi) \implies f'(\xi) = \frac{f(b) - f(a)}{b - a} = \frac{\Delta f}{\Delta x} \quad \blacksquare \end{aligned}$$

Постоянство функции

Пусть $f \in \mathbb{C}[a; b]$ и состоит из стационарных точек на $(a; b)$. Тогда:

$$f([a; b]) = C$$

Доказательство. По теореме Лагранжа:

$$\forall x', x'' \in [a; b] \exists \xi \in (x'; x'') : f'(\xi) = \frac{f(x'') - f(x')}{x'' - x'}$$

По определению стационарной точки:

$$f'(\xi) = 0 \Rightarrow \frac{f(x'') - f(x')}{x'' - x'} = 0 \iff f(x'') = f(x') \blacksquare$$

Пусть $f, g \in \mathbb{C}[a; b]$ и $f' = g'$. Тогда:

$$\forall x \in [a; b] f(x) - g(x) = C$$

Доказательство. Пусть $\varphi := f - g$; по условию:

$$\forall x \in (a; b) \varphi'(x) = f'(x) - g'(x) = 0$$

По условию постоянства функции:

$$\varphi'(x) = 0 \iff \varphi(x) = C \iff f(x) - g(x) = C \blacksquare$$

Частные производные

Функция нескольких переменных x_1, \dots, x_n задана соответствием вида:

$$f: \mathbb{R}^n \xrightarrow{x_1, \dots, x_n \rightarrow y} \mathbb{R}$$

Частная производная функции $f(x_1, \dots, x_n)$ по x_i — производная f с переменной x_i и др. *фикс. аргументами*:

$$\frac{\partial f}{\partial x_i} \text{ — нотация}$$

Вторая частная производная по x :

$$\frac{\partial^2 f}{\partial x^2} \text{ — нотация}$$

Смешанная частная производная по x, y :

$$\frac{\partial f}{\partial x \partial y} \text{ — нотация}$$

Условие. Критическая точка M — экстремум $f(x, y)$, если верно:

$$B^2 - AC < 0$$
$$A = \left. \frac{\partial^2 f}{\partial x^2} \right|_M \quad B = \left. \frac{\partial^2 f}{\partial x \partial y} \right|_M \quad C = \left. \frac{\partial^2 f}{\partial y^2} \right|_M$$

Причём:

$A < 0 \Rightarrow M$ — локальный максимум

$A > 0 \Rightarrow M$ — локальный минимум

Неявная производная

Неявной называется функция...

Устойчивость

Пусть дано дифференциальное уравнение:

$$\frac{dx}{dt} = f(x), \quad x \in \mathbb{R}^n$$

Нейтрально устойчивым называется такое *стационарное* решение $\tilde{\varphi}(t)$ дифференциального уравнения, что:

$$\forall \varepsilon > 0 \exists \delta > 0: \forall x_t \in U_\delta(x^*) \quad x_t \in U_\varepsilon(x^*)$$

$$\forall \varepsilon > 0 \exists \delta > 0: \forall x_t \in U_\delta(x^*) \quad x_t \in U_\varepsilon(x^*)$$

Малые отклонения от решения не выводят систему из окрестности стационарного решения.

Асимптотически устойчивым называется такое *нейтрально устойчивое* решение x^* дифференциального уравнения, что:

$$t \rightarrow +\infty, \quad |x_t - x^*| \rightarrow 0$$

Малые отклонения от нейтрально устойчивого решения со временем затухают.

Решения дифференциального уравнения делятся на:

- **притягивающие** — асимптотически устойчивые;
- **отталкивающие** — неустойчивые.

Линеаризация Ляпунова. Устойчивость стационарного состояния уравнения определяется знаком производной правой части в стационарной точке:

$$\frac{dx}{dt} = f(x)$$

Интегральное исчисление

Неопределённый интеграл

Первообразная для функции f на множестве X — такая функция F , что:

$$\forall x \in X \quad F'(x) = f(x)$$

Если у функции f есть первообразная F , то для любой константы C функция $F + C$ тоже первообразная, причём других нет.

Доказательство. По определению первообразной:

$$F' = f$$

По дистрибуции производной:

$$(F + C)' = f \implies F + C \text{ — первообразная для } f \quad \square$$

Пусть Φ — другая первообразная для f :

$$\begin{cases} \Phi' = f \\ F' = f \end{cases} \implies \Phi' - F' = (\Phi - F)' = f - f = 0$$

По условию постоянства функции:

$$\Phi - F = C \iff \Phi = F + C \quad \blacksquare$$

Неопределённый интеграл — множество всех первообразных функции f :

$$F(x) + C =: \int f(x) dx$$

f — подынтегральная функция;

$f(x) dx$ — подынтегральное выражение;

x — переменная интегрирования;

C — постоянная интегрирования.

Свойства

Операция интегрирования *дистрибутивна* относительно сложения, а также:

$$\begin{aligned} \int F'(x) dx &= F(x) + C & d\left(\int F(x) dx\right) &= F(x) dx \\ \left(\int F(x) dx\right)' &= F(x) + C & \int kF(x) dx &= k \int F(x) dx, \quad k \neq 0 \end{aligned}$$

Интегрирование по частям. Пусть $u dv$ — подынтегральная функция. Тогда справедливо:

$$\int u dv = uv - \int v du$$

Доказательство. По «дистрибуции» производной:

$$(uv)' = u'v + uv'$$

По определению интеграла:

$$\int (u'v + uv') dx = uv + C \iff \int u'v dx + \int uv' dx = uv + C$$

По определению дифференциала:

$$\begin{aligned} \begin{cases} du = u' dx \\ dv = v' dx \end{cases} &\implies \int v du + \int u dv = uv + C \iff \\ &\iff \int u dv = uv - \int v du \blacksquare \end{aligned}$$

Дифференциальное уравнение

Дифференциальным называется уравнение с неизвестной функцией под знаком *производной* или *дифференциала*.

Интегральная кривая — график решения дифференциального уравнения.

Метод Фурье. Решение дифференциального уравнения $y' = \varphi(x) \psi(y)$ удовлетворяет условию: (*Ж. Фурье*)

$$\begin{cases} \int \frac{dy}{\psi(y)} = \int \varphi(x) dx, & \psi(y) \neq 0 \\ y = y_0, & \psi(y_0) = 0 \end{cases}$$

Доказательство. По условию:

$$y' = \varphi(x) \psi(y) \iff \frac{y'}{\psi(y)} = \varphi(x), \quad \psi \neq 0$$

Возьмём интеграл от обеих частей уравнения:

$$\frac{y' dx}{\psi(y)} = \varphi(x) dx \implies \int \frac{dy}{\psi(y)} = \int \varphi(x) dx \quad \square$$

По условию:

$$\psi(y_0) = 0 \Rightarrow (y_0)' = 0 \Rightarrow 0 = 0 \blacksquare$$

Площадь плоской фигуры

Вложенной в фигуру F называется такая фигура P , которая целиком лежит внутри F :

$S_*(F)$ — внутренняя площадь F

Объемлющей фигуру F называется такая фигура Q , которая целиком содержит F :

$S^*(F)$ — внешняя площадь F

Квадрируемой называется такая фигура F , у которой множества $S_*(F)$ и $S^*(F)$ имеют единую точную границу:

$$\sup S_*(F) = \inf S^*(F) = S(F) \text{ — площадь } F$$

Спряmlяемой называется кривая с конечной длиной.

Свойства квадрируемости

Критерий квадрируемости. Фигура F квадрируема тогда и только тогда, когда:

$$\forall \varepsilon > 0 \exists P \subseteq F \subseteq Q: S(Q) - S(P) < \varepsilon$$

Доказательство \Rightarrow . Зафиксируем $\varepsilon > 0$.

По определению точных границ множеств $S(P)$, $S(Q)$:

$$\begin{aligned} & \left\{ \begin{array}{l} \forall \varepsilon/2 > 0 \exists P \subseteq F: S(F) - S(P) < \varepsilon/2 \\ \forall \varepsilon/2 > 0 \exists Q \supseteq F: S(Q) - S(F) < \varepsilon/2 \end{array} \right. \Rightarrow \\ & \Rightarrow S(Q) - S(P) < \varepsilon \blacksquare \end{aligned}$$

Доказательство \Leftarrow . По определению точных верхних границ множеств $S(P)$, $S(Q)$:

$$S(Q) - S(P) < \varepsilon \Rightarrow 0 \leq \inf_{Q \supseteq F} S(Q) - \sup_{P \subseteq F} S(P) < \varepsilon$$

По определению квадратуемой фигуры:

$$\inf S^*(F) - \sup (S_*(F)) = 0 \iff \inf S^*(F) = \sup S_*(F) \implies \\ \implies F \text{ квадратуема} \blacksquare$$

Признак квадратуемости. Если граница фигуры F — спрямляемая кривая, то F квадратуема.

Доказательство. По условию:

$$S^*(F) - S_*(F) = S \text{ фигуры, объемлющей границу } F$$

По определению точных границ множеств $S^*(F)$, $S_*(F)$:

$$\inf S^*(F) - \sup S_*(F) = S_{\text{гр}} = 0 \implies F \text{ квадратуема} \blacksquare$$

Аддитивность. Пусть F_1 и F_2 квадратуемы, причём $F_1 \cup F_2 = F$, $F_1 \cap F_2 = \emptyset$. Тогда F тоже квадратуема.

Доказательство. По условию:

$$F = F_1 \cup F_2 \implies S_{\text{гр}} \leq S_{\text{гр}1} + S_{\text{гр}2}$$

По критерию квадратуемости:

$$S_{\text{гр}1} + S_{\text{гр}2} = 0 \implies S_{\text{гр}} = 0 \implies F \text{ квадратуема} \blacksquare$$

Пересечение квадратуемых фигур *квадратуемо*.

Доказательство *аналогично* предыдущему свойству.

Определённый интеграл

Разбиение отрезка $[a; b]$ — конечное упорядоченное множество $X \subseteq [a; b]$, причём $a, b \in X$.

Частичным называется отрезок, составленный из *соседних* элементов разбиения:

$$\Delta x_k = x_k - x_{k-1} \text{ — длина частичного отрезка } [x_{k-1}; x_k]$$

Интегральная сумма функции f на $[a; b]$ имеет вид:

$$\sum_{k=1}^n f(\xi_k) \Delta x_k, \quad \xi_k \in [x_{k-1}; x_k]$$

Нижней (верхней) называется такая интегральная сумма, в которой ξ_k *минимизирует (максимизирует)* значение f на частичном отрезке.

Определённый интеграл — предел интегральной суммы:

$$\lim_{n \rightarrow +\infty} \sum_{k=1}^n f(\xi_k) \Delta x_k =: \int_a^b f(x) dx$$

Криволинейная трапеция — подграфик неотрицательной и непрерывной функции на $[a; b]$.

Геометрический смысл. Пусть f задаёт криволинейную трапецию T на $[a; b]$. Тогда её площадь равна:

$$S(T) = \int_a^b f(x) dx$$

Доказательство. По признаку квадратуемости:

$$f \in \mathbb{C}[a; b] \implies T \text{ квадратуема}$$

По определению интегральных сумм:

$$S^*(T) \text{ — верхняя сумма; } S_*(T) \text{ — нижняя сумма}$$

По определению квадратуемости:

$$\begin{aligned} S^*(T) - S_*(T) < \varepsilon &\implies \inf S^*(T) = \sup S_*(T) = S(T) \implies \\ &\implies \lim_{n \rightarrow +\infty} \sum_{k=1}^n f(\xi_k) \Delta x_k = S(T) \end{aligned}$$

По определению определённого интеграла:

$$S(T) = \int_a^b f(x) dx \blacksquare$$

Непрерывность \implies интегрируемость.

Доказательство. Когда-нибудь...

Оценка определённого интеграла. Пусть функция f на отрезке $[a; b]$ принимает значения из $[m; M]$. Тогда:

$$m(b-a) \leq \int_a^b f(x) dx \leq M(b-a)$$

Очевидным является *геометрическое* доказательство через площади фигур.

Свойства

Операция интегрирования *дистрибутивна* относительно сложения, а также:

$$\begin{aligned} \int_a^a F(x) dx &= 0 & \int_a^b F(x) dx &= - \int_b^a F(x) dx \\ \int_a^b kF(x) dx &= k \int_a^b F(x) dx, \quad k \neq 0 \end{aligned}$$

Аддитивность. Пусть $f \in [a; b]$, $c \in [a; b]$. Тогда верно:

$$\int_a^b F(x) dx = \int_a^c F(x) dx + \int_c^b F(x) dx$$

Интегрировать можно *неравенства*, если они непрерывны на области интегрирования:

$$\begin{cases} f, g \in \mathbb{C}[a; b] \\ f(x) \leq g(x) \end{cases} \implies \int_a^b f(x) dx \leq \int_a^b g(x) dx$$

Интеграл с переменным пределом

Теорема о среднем. Пусть $f \in \mathbb{C}[a; b]$. Тогда верно:

$$\exists \xi \in [a; b]: \int_a^b f(x) dx = f(\xi)(b-a)$$

Доказательство. По оценке определённого интеграла:

$$\begin{aligned} m \leq f(x) \leq M &\implies m(b-a) \leq \int_a^b f(x) dx \leq M(b-a) \implies \\ &\implies m \leq \frac{\int_a^b f(x) dx}{b-a} \leq M, \quad a \neq b \end{aligned}$$

По теореме о промежуточном значении:

$$\exists \xi \in [a; b]: f(\xi) = \frac{\int_a^b f(x) dx}{b-a} \Rightarrow \int_a^b f(x) dx = f(\xi)(b-a) \blacksquare$$

Интеграл с переменным верхним пределом — функция вида:

$$S(x) = \int_a^x f(t) dt, \quad x \in [a; b]$$

Пусть функция f непрерывна в окрестности точки $t = x$. Тогда в x функция $S(x)$ дифференцируема, причём

$$S'(x) = \left(\int_a^x f(t) dt \right)' = f(x).$$

Доказательство. По геометрическому смыслу определённого интеграла:

$$\begin{aligned} \Delta S &= S(x + \Delta x) - S(x) = \int_a^{x+\Delta x} f(t) dt - \int_a^x f(t) dt = \\ &= \int_a^x f(t) dt + \int_x^{x+\Delta x} f(t) dt - \int_a^x f(t) dt = \int_x^{x+\Delta x} f(t) dt \end{aligned}$$

По теореме о среднем:

$$\exists \xi \in [x; x + \Delta x]: \int_x^{x+\Delta x} f(t) dt = f(\xi)(x + \Delta x - x) = f(\xi) \Delta x$$

По предельному переходу:

$$\begin{aligned} \Delta S = f(\xi) \Delta x &\iff \frac{\Delta S}{\Delta x} = f(\xi) \Rightarrow \lim_{\Delta x \rightarrow 0} \frac{\Delta S}{\Delta x} = \lim_{\Delta x \rightarrow 0} f(\xi) \Rightarrow \\ &\Rightarrow S'(x) = f(x) \blacksquare \end{aligned}$$

Формула Ньютона—Лейбница

Пусть F — первообразная для функции f . Тогда верно:

$$\int_a^b f(x) dx = F(x) \Big|_a^b = F(b) - F(a)$$

Доказательство. По свойству интеграла с переменным верхним пределом:

$$S'(x) = f(x) \Rightarrow S(x) = F(x) + C \text{ — первообразные}$$

По определению определённого интеграла:

$$S(a) = \int_a^a f(t) dt = 0 \implies C = -F(a) \implies S(x) = F(x) - F(a)$$

По условию:

$$x = b \implies S(b) = F(b) - F(a) \iff \int_a^b f(t) dt = F(b) - F(a) \blacksquare$$

Длина кривой

Пусть график функции f — кривая. Тогда её длина на промежутке $[a; b]$ равна:

$$L = \int_a^b \sqrt{1 + f'^2(x)} dx$$

Доказательство. Пусть задано разбиение $X \subseteq [a; b]$.

Тогда длина хорды в точках x_k, x_{k-1} равна:

$$l_k = \sqrt{(\Delta f_k)^2 + (\Delta x_k)^2} = \Delta x_k \sqrt{1 + \left(\frac{\Delta f_k}{\Delta x_k} \right)^2}$$

По теореме Лагранжа:

$$\exists \xi \in [x_{k-1}; x_k]: l_k = \Delta x_k \sqrt{1 + f'^2(\xi)}$$

По предельному переходу:

$$L \geq \sum_{k=0}^n \Delta x_k \sqrt{1 + f'^2(\xi)} \implies L = \lim_{n \rightarrow \infty} \sum_{k=0}^n \Delta x_k \sqrt{1 + f'^2(\xi)}$$

По определению определённого интеграла:

$$\begin{aligned} \lim_{n \rightarrow \infty} \sum_{k=0}^n \Delta x_k \sqrt{1 + f'^2(\xi)} &= \int_a^b \sqrt{1 + f'^2(x)} dx \implies \\ &\implies L = \int_a^b \sqrt{1 + f'^2(x)} dx \blacksquare \end{aligned}$$

Среднее значение

Теорема. Среднее значение $f(x)$ на $[a; b]$ равно:

$$f_{\text{avg}}(x) = \frac{\int_a^b f(x) dx}{b - a}$$

Доказательство. По определению среднего значения:

$$f_{\text{avg}}(x) \approx \frac{\sum_{k=1}^n f(\xi_k)}{n}, \quad \xi_k \in \Delta x_k$$

Допустим, что все частичные отрезки *равны*:

$$n = \frac{b - a}{\Delta x} \Rightarrow f_{\text{avg}}(x) \approx \frac{\sum_{k=1}^n f(\xi_k) \Delta x}{b - a}$$

По предельному переходу:

$$f_{\text{avg}}(x) = \frac{\int_a^b f(x) dx}{b - a} \blacksquare$$

Следствие. Среднее значение $f(x)$ на $[a; b]$ равно угловому коэффициенту прямой, которая проходит через точки $\langle a, f(a) \rangle$ и $\langle b, f(b) \rangle$.

Это видно, если расписать утверждение выше по формуле Ньютона-Лейбница.

Теория алгоритмов

Поиск с возвратом

Поиск с возвратом — метод нахождения решений задачи полным перебором допустимых расстановок элементов конечного множества:

- в качестве *частичного решения* используется пустое упорядоченное множество M , которое расширяется до полного по одному элементу за операцию;
- если решение *полное* или *не удовлетворяет условию*, алгоритм приступает к другому частичному решению.

Пусть $T_1 = \langle V_1, E_1 \rangle$, $T_2 = \langle V_2, E_2 \rangle$ — корневые деревья.

Кандидат для $v \in V_1$ — элемент множества

$$C_v := \{w \mid w \in V_2, \text{depth}_v = \text{depth}_w\} \cup \{\lambda\}.$$

Возвратное дерево для T_1 и T_2 — такое дерево $T = \langle V, E \rangle$ с мнимым корнем, что:

$$\left\{ \begin{array}{l} M \subseteq V_1 \times W \text{ (упорядочено, биективно)} \\ W = [\text{root}_T, \dots, w] \setminus \{\text{root}_T\} \subseteq V_2 \cup \{\lambda\} \subseteq V \\ \text{children}_w = \emptyset \end{array} \right\} \text{ I}$$
$$\left\{ \begin{array}{l} \forall \langle w_1, w_2 \rangle \subseteq W \text{ order}(w_1) < \text{order}(w_2) \\ w_1, w_2 \neq \lambda \end{array} \right\} \text{ II}$$
$$\left\{ \forall \langle v_i, w_j \rangle \in M \ w_j \in C_{v_i} \right\} \text{ III}$$

I — всякая простая цепь возвратного дерева от корня до листа без корня соответствует *уникальному* отображению T_1 в T_2 ;

II — индекс узлов одной простой цепи от корня до листа без корня *строго возрастает*;

III — всякий узел простой цепи от корня до листа без корня является *кандидатом* для соответствующего узла T_1 .

Итерация построения полного решения M для условия P :

$$\begin{cases} \forall c \in C_{W.\text{last}()} \quad W := W \cup \{c\} \\ T(M) := M \text{ — частичное решение} \\ M \wedge P(M) \wedge T(M) \neq \emptyset \implies \text{расширить } M \\ M \wedge P(M) \wedge T(M) = \emptyset \implies \text{следующее } M \end{cases}$$

Дерево ветвей и границ для T_1 и T_2 — такое возвратное дерево для T_1 и T_2 , что $P := P \wedge R$, где:

$$R(M_i) = \begin{cases} \alpha_{\min} = \emptyset \implies \alpha_{\min} := \max \\ \alpha_{\min} \geq \gamma(M_i) \implies \text{True}, \alpha_{\min} := \gamma(M_i) \\ \alpha_{\min} < \gamma(M_i) \implies \text{False} \end{cases}$$

«Разделяй и властвуй»

«**Разделяй и властвуй**» — метод рекурсивного нахождения решений задачи:

- задача делится на меньшие, *независимые* друг от друга подзадачи, пока они не будут сведены к *тривиальным*;
- решения тривиальных подзадач *комбинируются* в единое к исходной задаче.

Пусть $T_1 = \langle V_1, E_1 \rangle$, $T_2 = \langle V_2, E_2 \rangle$ — корневые деревья, $A_1 = T_{1W_1}$, $A_2 = T_{2W_2}$, $B_1 = T_1 \setminus A_1$, $B_2 = T_2 \setminus A_2$ — их поддеревья:

$$\begin{cases} W_1 = \{v_m \in V_1 \mid \text{order}(v_m) < \text{order}(v)\} \\ W_2 = \{w_n \in V_2 \mid \text{order}(w_p) < \text{order}(w)\} \\ v := \text{last}_{v_i}, \quad w := \text{last}_{w_k} \end{cases}$$

Дерево «разделяй и властвуй» для T_1 и T_2 — такое ордерено $T = \langle V, E \rangle$ с вершинами вида $v_i v_j w_k w_l$, что:

$$\begin{cases} v_i, v_j \in V_1, \quad w_k, w_l \in V_2 \\ \text{root}_T = v_1 v_{n_1} w_1 w_{n_2} \quad (T_1 \rightarrow T_2) \end{cases}$$

Шаг рекурсивного построения решения M :

$$\begin{cases} v_i = v_j, w_k = w_l \Rightarrow v_i \mapsto w_k, \text{комбинировать} \\ v_i \neq v_j, w_k = w_l \Rightarrow A_1 \rightarrow T_2 \ (B_1 \rightarrow \lambda) \\ v_i = v_j, w_k \neq w_l \Rightarrow T_1 \rightarrow A_2 \ (\lambda \rightarrow B_2) \\ v_i \neq v_j, w_k \neq w_l \Rightarrow \begin{cases} A_1 \rightarrow A_2 \text{ или } A_1 \rightarrow T_2 \\ B_1 \rightarrow B_2 \text{ или } T_1 \rightarrow A_2 \end{cases} \end{cases}$$

Динамика

Динамическое программирование — метод рекурсивного нахождения решений задачи:

- задача делится на меньшие, *зависимые* друг от друга подзадачи, пока они не будут сведены к *тривиальным*;
- решения тривиальных подзадач *комбинируются* в единое к исходной задаче.

Мемоизация («сверху вниз») — кеширование и повторное использование ранее подсчитанных результатов.

Табуляция («снизу вверх») — заполнение кеша на основе тривиальных подзадач.

Лучшее решение выбирается из матрицы лучших решений его подграфов (*у них по рекурсии есть свои матрицы*):

$$\begin{array}{cccc} \langle v_i, w_k \rangle & \langle v_i, w_k w_{k+1} \rangle & \cdots & \langle v_i, w_k \dots w_l \rangle \\ \langle v_i v_{i+1}, w_k \rangle & \langle v_i v_{i+1}, w_k w_{k+1} \rangle & \cdots & \langle v_i v_{i+1}, w_k \dots w_l \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle v_i \dots v_j, w_k \rangle & \langle v_i \dots v_j, w_k w_{k+1} \rangle & \cdots & \langle v_i \dots v_j, w_k \dots w_l \rangle \end{array}$$

$$\left\{ \begin{array}{l} v \in V_1, w \in V_2 \\ \text{depth}_v = \text{depth}_w \\ \{v_i, \dots, v_j\} = \text{children}_v \\ \{w_k, \dots, w_l\} = \text{children}_w \\ \langle v_i \dots v_j, w_k \dots w_l \rangle \sim \gamma_{\min}(G_1 \rightarrow G_2) \\ G_1 = T_{1w_i} \cup \dots \cup T_{1w_j} \\ G_2 = T_{2w_k} \cup \dots \cup T_{2w_l} \\ \forall s \in \{i, \dots, j\} \text{ root}_{T_{1w_s}} = v_s \\ \forall t \in \{k, \dots, l\} \text{ root}_{T_{2w_t}} = w_t \end{array} \right.$$

Алгоритм табуляции занимает $\mathcal{O}(n_1 n_2)$ места, используя $\mathcal{O}(n_1 n_2)$ времени.

Уравнение Беллмана

Введём задачу на оптимизацию вида:

$$\begin{array}{ll} d & \text{— выбор;} \\ \text{opt}_{d \in \Delta} \{H(d)\} & \Delta \text{— допустимое множество;} \\ & H \text{— целевая функция одной переменной.} \end{array}$$

Оптимум — оптимальное значение целевой функции (выбор d^* оптимизирует H):

$$H^* := H(d^*) \quad d^* := \arg \text{opt}_{d \in \Delta} \{H(d)\}$$

Пусть H — целевая функция нескольких переменных.

Оптимум такой задачи можно найти либо *полным перебором*, либо *последовательным принятием решений*:

$$\begin{aligned} H^* &= \text{opt}_{\langle d_1, \dots, d_n \rangle \in \Delta} \{H(d_1, \dots, d_n)\} \\ &= \text{opt}_{d_1 \in D_1} \{ \text{opt}_{d_2 \in D_2} \{ \dots \{ \text{opt}_{d_n \in D_n} \{h(d_1, \dots, d_n)\} \} \dots \} \} \\ &= \text{opt}_{d_1 \in D_1} \{H(d_1, d_2^*(d_1), \dots, d_n^*(d_1))\} \end{aligned}$$

$\Delta = D_1 \times \dots \times D_n$ — пространство решений;

$D_n(d_1, \dots, d_{n-1})$ — множество решений, которое зависит от предыдущих $\langle d_1, \dots, d_{n-1} \rangle$ решений;

$d_i^*(d_1, \dots, d_{i-1})$ — локальный выбор d , оптимизирующий H .

Распределение ресурсов

В задаче на *оптимальное распределение ресурсов* требуется разделить ограниченное число ресурсов на множество их потребителей, у которых есть стоимость.

Общая формула:

$$f(k, m) = \min_{d \in \{0, \dots, m\}} \{C(k, d) + f(k + 1, m - d)\}$$

Жадные алгоритмы

Список жадных алгоритмов:

- перевод числа из десятичной системы счисления в m -ичную:
 - > задача о размене монет.
- что-то...

Бинарный поиск

Бинарный поиск на отрезке $[a; b]$ — метод поиска корня монотонной функции $f \in \mathbb{C}[a; b]$.

Правым называется такой бинарный поиск, который ищет *верхнюю границу* монотонной функции (*максимальное x , при котором есть корни*):

$$mid = \left\lceil \frac{l + r}{2} \right\rceil$$

Левым называется такой бинарный поиск, который ищет *нижнюю границу* монотонной функции (*минимальное x , при котором есть корни*):

$$mid = \left\lfloor \frac{l + r}{2} \right\rfloor$$

Задача. Один принтер печатает лист раз в x мин., другой — раз в y мин. За сколько минут они напечатают N листов? (*задача имеет решение за константу*)

Идея. Решим обратную задачу бинарным поиском по j :

— сколько листов напечатают оба принтера за j мин.?

Значит, ищем такое минимальное j , что верно:

$$\left\lfloor \frac{j}{x} \right\rfloor + \left\lfloor \frac{j}{y} \right\rfloor \geq N$$

Задача. На прямой есть N стойл. Максимизировать минимальное расстояние между $K < N$ коровами.

Идея. Решим обратную задачу бинарным поиском по j :

— расположить K коров так, чтобы минимальное расстояние между ними было не больше j

Коровы расположим *жадно*: в самых левых свободных стойлах, расстояние между которыми *не больше j* .

Задача. Дано N отрезков различных длин. Получить разрезаниями K равных отрезков максимальной длины.

Идея. Решим обратную задачу бинарным поиском по j :

— разрезать N отрезков разных длин на K отрезков длины j

Значит, ищем такое максимальное j , что верно:

$$\sum_{i=1}^N \left\lfloor \frac{a_i}{j} \right\rfloor \geq K$$

Задача. Есть N дипломов $h \times w$. Минимизировать сторону квадратной стены, на которой они будут размещены в целых координатах без поворотов.

Идея. Решим обратную задачу бинарным поиском по j :

— Разместить N дипломов на квадратной стене со стороной j

Разместим дипломы *жадно*: в самых верхних левых свободных координатах, иначе спустимся на уровень ниже.

Задача. Исследовательские модули с защитой толщины d — прямоугольники $(a + 2d) \times (b + 2d)$. Максимизировать d для размещения модулей в поле $w \times h$.

Идея. Решим обратную задачу бинарным поиском по j :

— Разместить модули размеров $(a + 2j) \times (b + 2j)$ в поле $w \times h$.

Разместим модули *жадно*: в самых верхних левых свободных координатах, иначе спустимся на уровень ниже.

Задача. Два лесоруба срубают A и B деревьев в день, но отдыхают каждый K -ый и M -ый дни соответственно. За сколько дней они управятся с X деревьями?

Идея. Решим обратную задачу бинарным поиском по j :

— сколько деревьев лесорубы срубят за j дней?

Значит, ищем такое минимальное j , что верно:

$$(A + B)j - A \left\lfloor \frac{j}{K} \right\rfloor - B \left\lfloor \frac{j}{M} \right\rfloor \geq X \iff$$

$$A \left(j - \left\lfloor \frac{j}{K} \right\rfloor \right) + B \left(j - \left\lfloor \frac{j}{M} \right\rfloor \right) \geq X$$

Задача. В классе учатся N человек различного роста. Составить R бригад по C человек так, чтобы максимальная разница в росте одной бригады была минимальна.

Идея. Решим обратную задачу бинарным поиском по j :

— составить R бригад по C человек с максимальной разницей в росте j .

Выберем бригады *жадно*: отсортируем школьников по росту, выберем самую левую бригаду, удовлетворяющую условиям.

Продолжим цикл, пока не *наберётся* нужное число бригад, либо школьники не *закончатся*.

Задача. Бетси может делать печенье за t_C ед. времени и булочку за t_M . Заказ выполняется не дольше C_i . Бетси может улучшить свою печь за монету, чтобы она производила печенье или булочку на единицу быстрее (*за положительное время*). Минимизировать количество монет, чтобы выполнить все заказы.

Идея. Пусть в оптимальном случае было затрачено w монет, и печь производит печенье за x ед. времени, а булочку — за y .

По условию задачи составим систему:

$$\begin{cases} 1 \leq x \leq t_C \\ 1 \leq y \leq t_M \\ x + y = t_C + t_M - w \\ Ax + By \leq C \end{cases}$$

Из двух последних выражений следует:

$$(A - B)x \leq C - B(t_C + t_M - w)$$

При делении неравенства на $A - B$ с учётом знака получим

одну границу x (случай $A = B$ обрабатывается отдельно).

Из первых трёх выражений найдём другую границу для x :

$$\begin{cases} 1 \leq x \leq t_C \\ t_C - w \leq x \leq t_C + t_M - w - 1 \end{cases}$$

Задача решается бинарным поиском по w : если множество допустимых x непустое, то w подходит.

Задача. Найти медиану таблица умножения $N \times N$.

Идея. Пусть $f(x)$ — количество натуральных чисел, не больших x .

По определению медианы m :

$$f(m) \geq \left\lfloor \frac{n^2}{2} \right\rfloor$$

Количество элементов строки, кратных её номеру i и не больших медианы m , равно:

$$j_{\max} = \left\lfloor \frac{m}{i} \right\rfloor$$

Задача решается бинарным поиском по m .

Тернарный поиск

Бинпоиск по производной?

Унимодальной на отрезке $[a; b]$ называется такая функция $f \in \mathbb{C}[a; b]$, которая имеет на нём *один экстремум*.

Тернарный поиск — метод поиска экстремума унимодальной функции.

Теорема. Вдоль прямой дороги расположились друзья в некоторых координатах. Каждый из них может идти с максимальной скоростью v_i . Найти минимальное время встречи всех друзей в одной точке с точностью до 10^{-6} .

Идея. Переформулируем задачу:

— найти координату x_0 места встречи, до которой все друзья

дойдут за минимальное время

Тернарным поиском по x_0 с $\varepsilon < 10^{-6}$ найдём минимальное время:

$$v_i = \frac{|x_0 - x_i|}{\tau_i} \implies \tau_i = \frac{|x_0 - x_i|}{v_i} \implies \tau = \max\{\tau_i \mid \forall i\}$$

Структуры данных

Стек

Задача. Вычислить область самого большого прямоугольника в гистограмме, который находится на общей базовой линии.

Идея. Введём два фиктивных столбца: один отрицательной высоты (*в начало*), другой — нулевой (*в конец*).

Пока гистограмма *строго возрастает*, добавлять столбцы в стек.

В противном случае, если высота i -го столбца \leq вершины стека:

- 1) Убирать из стека столбцы, пока гистограмма не станет *строго возрастающей*.
- 2) Считать площадь прямоугольника от $i - 1$ до последнего убранного столбца x :

$$S = h_x(i - x)$$

- 3) По достижении строго возрастающей последовательности добавить в стек столбец с параметрами:
 - h текущего столбца
 - x последнего удалённого столбца

Ближайшее меньшее

Задача. Дан массив чисел. Для каждого элемента x найти такое ближайшее число y слева, что $x < y$.

Идея. Введём вспомогательный массив и *стек*, в который постепенно будем помещать все элементы массива, большие вершины.

Если элемент массива a_i меньше вершины стека a_j , то для элемента i ближайшим большим числом будет являться вершина стека.

За счёт вспомогательного массива после линейной обработки исходного массива можно добиться *константного* времени.

Префиксная сумма

Префиксная сумма — это...

Задача. Дан массив целых чисел. Найти подотрезок с максимальной суммой.

Идея. Составим массив префиксных сумм π .

Начнём перебирать правую границу r искомого отрезка, так что остаётся найти величину:

$$\min\{\pi_r - \pi_i \mid i \in [0; r)\}$$

Заметим, что оптимальный вариант левой границы искомого отрезка — *глобальный минимум* на интервале $[0; r)$.

Таким образом, задача решается за *линейное* время.

Стек рекордов

Стек рекордов — монотонная подпоследовательность массива, для любых элементов которой верно:

$$i \leq j, a_i < a_j \quad \text{или} \quad i \leq j, a_i > a_j$$

Задача. Дан массив чисел. Найти минимумы для всех отрезков длины K .

Идея. Введём стек минимумов, в который будем добавлять элементы по правилу:

- 1) Элемент $>$ вершины \implies добавить в стек
- 2) Элемент \leq вершины \implies убирать верхние элементы до достижения возрастающей последовательности

Если последний элемент стека не входит в K -отрезок, то убрать его (*для этого лучше подходит очередь*).

Дерево отрезков

Дерево отрезков — бинарное дерево для массива arr , на котором можно реализовать массовые ассоциативные операции f за *логарифмическое время*:

- 1) Листья — элементы массива arr
- 2) Родитель содержит *результат операции* от своих детей
- 3) Корень содержит *результат операции* от arr на $[0; n)$

Два вида:

- ДО сверху — рекурсивный вариант
- ДО снизу — итеративный вариант

Одиночное обновление

Построение. По принципу «разделяй и властвуй»:

```
def build(v, l, r):
    if l == r:
        verts[v].val = arr[l]
    else:
        mid = (l + r) // 2
        build(2 * v, l, mid)
        build(2 * v + 1, mid + 1, r)
        verts[v].val = f(verts[2 * v].val, \
                        verts[2 * v + 1].val)
```

Обновление. По принципу «разделяй и властвуй»:

```
def update(v, l, r, idx, val):
    if l == r:
        verts[v].val = val
    else:
        mid = (l + r) // 2
        if idx <= mid:
            update(2 * v, l, mid, idx, val)
        else:
            update(2 * v + 1, mid + 1, r, idx, val)
        verts[v].val = f(verts[2 * v].val, \
```

```
verts[2 * v + 1].val)
```

Запрос. По принципу «разделяй и властвуй»:

```
def get(v, l, r, L, R):
    if L > R:
        return # neutral element
    if l == L and r == R:
        return verts[v].val
    mid = (l + r) // 2
    return f(get(2 * v, l, mid, L, min(R, mid)), \
             get(2 * v + 1, mid + 1, r, \
                 max(L, mid+1), R))
```

Массовое обновление

Обновление. По принципу «разделяй и властвуй» с применением *отложенных операций*:

```
def update(v, l, r, L, R, push):
    if L > R:
        return
    if l == L and r == R:
        verts[v].val = push
    else:
        mid = (l + r) // 2
        update(2 * v, l, mid, L, min(R, mid), push)
        update(2 * v + 1, \
              mid + 1, r, max(L, mid + 1), R, push)
```

Запрос. По принципу «разделяй и властвуй» — это для одного элемента, а мне нужна сумма на отрезке!!!!!!!:

```
def get(v, l, r, idx):
    if l == r:
        return verts[v].val
    mid = (l + r) // 2
    if idx <= mid:
        return verts[v].val + get(2 * v, l, mid, idx)
```

```
else:
```

```
    return verts[v].val + get(2 * v + 1, \  
                               mid + 1, r, idx)
```

Алгоритмы сортировки

Merge Sort

Сортировка слиянием основана на принципе «разделяй и властвуй»:

- 1) Выбрать опорный элемент mid .
- 2) Разделить массив на две части:

$$\boxed{0 \dots mid - 1} \quad \boxed{mid \dots N - 1}$$

- 3) Запустить сортировку в обеих частях.
- 4) Объединить два отсортированных массива (*2P-метод*).

Инверсия

Инверсия — такая пара чисел a_i, a_j из массива a , что:

$$i < j, \quad a_i > a_j$$

Разделённой называется такая *инверсия*, которая относится к двум элементам *разных* массивов.

Модификация сортировки слиянием считает количество инверсий в массиве:

- алгоритм обрабатывает элемент из отсортированного правого массива;
- к счётчику инверсий добавляется количество необработанных элементов отсортированного левого массива (*разделённые инверсии*).

Quicksort

Быстрая сортировка основана на принципе «разделяй и властвуй»:

- 1) Выбрать опорный элемент $pivot$.
- 2) Разделить массив на три части:

$$\boxed{< pivot} \quad \boxed{pivot} \quad \boxed{> pivot}$$

- 3) Запустить сортировку в крайних частях.

Алгоритм реализации:

```
| def quick_sort(low, high):
```

```

i, j = low, high
pivot = arr[(i + j) // 2]
while i <= j:
    while arr[i] < pivot:
        i += 1
    while arr[j] > pivot:
        j -= 1
    if i <= j:
        arr[i], arr[j] = arr[j], arr[i]
        i += 1
        j -= 1
if j > low:
    quick_sort(low, j)
if i < high:
    quick_sort(i, high)

```

Задача. Дано $N \leq 10^5$ чисел длины $[1; 100]$. Составить из них конкатенацией максимальное число. *(одно из них точно начинается не с нуля)*

Идея. Отсортировать числа быстрой сортировкой с компаратором:

```

def compare(num1, num2):
    if str1 + str2 > str2 + str1:
        return True
    return False

```

k-Порядковая статистика

k-Порядковая статистика — элемент линейно упорядоченного множества, который стоит на k -ом месте.

Quickselect — модификация *быстрой сортировки*, которая ищет k -порядковую статистику за $\mathcal{O}(n)$:

— сортируется лишь та крайняя часть, в которую входит k .

Radix Sort

Поразрядная сортировка предназначена для больших объектов (*чисел, строк*), которые можно разбить на *разряды*:

- LSD (*least significant digit*) — от младших к старшим разрядам;
- MSD (*most significant digit*) — от старших к младшим разрядам.

Алгоритм LSD-сортировки:

| yes

Алгоритм MSD-сортировки:

| yes

Динамическое программирование

Модель динамики

Целевой называется функция, у которой нужно найти *экстремум (оптимальное значение)*.

Состояние системы зависит от конечного числа *параметров (часто от одного или двух)*.

Принцип оптимальности. Оптимальное решение зависит лишь от текущего состояния и цели, а не от предыстории. (*Р. Беллман*)

Сертификат решения — последовательность управляющих шагов, которые оптимизируют целевую функцию.

Типы задач на динамику:

- оптимизация целевой функции;
- подсчёт количества вариантов решения;
- составление сертификата решения.

Подходы динамики

Мемоизация — *рекурсивный* подход динамики, при котором подсчитанные результаты *кешируются* и используются повторно (*вычисления отложены*).

Табуляция — *итеративный* подход динамики, при котором кеш заполняется сразу, на основе тривиальных подзадач.

Также пояснить про одномерный и двумерный кеш, определение кеша?

Задача о рюкзаке

Задача о рюкзаке — множество задач *комбинаторной оптимизации*, которые сводятся к выбору подмножества:

- с максимальной *стоимостью*
- с соблюдением ограничения на *вес*

Типы задач о рюкзаке:

- 0/1 (*каждый предмет в одном экземпляре*)
- неограниченный (*каждый предмет бесконечен*)
- задача размена монет
- разбиение N -множества (*balanced / unbalanced*)

Задача. Дан рюкзак вместимостью $C \leq 10^9$ и N вещей, которые имеют *вес* и *стоимость*. Максимизировать стоимость рюкзака, если $\sum c_i \leq 10^4$.

Идея. Из-за ограничений на вместимость *обратим логику*.

Пусть $f(c, i)$ — минимальный суммарный вес первых i вещей, которые стоят не менее c .

Очевидно, что $f(0, i) = 0$ для любых i .

Тогда верно рекуррентное соотношение:

$$f(c, i) = \min\{f(c, i - 1), f(c - c_{i-1}, i - 1) + w_{i-1}\}$$

Задача. Дан набор N гирек, которые имеют *вес*. Можно ли разбить гири на две кучи, равные по *количеству гирек* и *массе*?

Идея. Пусть $f(w, n, i)$ — возможность составить кучу массой w из n гирек, используя первые i гирек из набора.

Очевидно, что $f(0, 0, i) = 1$ для любых i .

Тогда верно рекуррентное соотношение:

$$f(w, n, i) = f(w, n, i - 1) \vee f(w - w_{i-1}, n - 1, i - 1)$$

Ответ будет лежать в $f(\frac{\sum w_i}{2}, \frac{N}{2}, N)$.

Задача. Дан набор N гирек, которые имеют *вес*. Можно ли разбить гири на три кучи, равные по *массе*?

Идея. Пусть $f(w_1, w_2, i)$ — возможность составить две кучи массами w_1 и w_2 , используя первые i гирек из набора.

Очевидно, что $f(0, 0, i) = 1$ для любых i .

Тогда верно рекуррентное соотношение:

$$f(w_1, w_2, i) = \begin{cases} f(w_1, w_2, i-1) \\ f(w_1 - w_{i-1}, w_2, i-1) \\ f(w_1, w_2 - w_{i-1}, i-1) \end{cases}$$

Ответ будет лежать в $f(\frac{\sum w_i}{3}, \frac{\sum w_i}{3}, N)$.

Счастливые билеты

Задача. Дано натуральное число n . Найти количество $2n$ -значных счастливых билетов.

Идея. Пусть D_n^k — количество n -значных чисел с суммой цифр k .

Легко проверить, что счастливых билетов ровно D_{2n}^{9n} :

$$\overline{a_1 \dots a_n b_1 \dots b_n} \mapsto \overline{a_1 \dots a_n (9 - b_1) \dots (9 - b_n)}$$

Очевидно, что $D_0^0 = 1, D_0^k = 0, k > 0$.

Тогда D_n^k можно выразить через $(n - 1)$ -значное число, добавив любую цифру j :

$$D_n^k = \sum_{j=0}^9 D_{n-1}^{k-j}$$

Задача. Пусть натуральное число *красивое*, если сумма квадратов его цифр — полный квадрат. Найти количество красивых чисел в диапазоне $[1; N]$.

Идея. Пусть D_n^k — количество n -значных чисел с суммой квадратов цифр k .

Тогда верна рекуррентная формула:

$$D_n^{k+j^2} += D_{n-1}^k, j \in \{0, \dots, 9\}$$

Заметим, что для фиксированного n верно:

$$1 \leq k \leq 81n$$

Для каждого *полного квадрата* $k \in [1; 81n]$ найдём все числа из диапазона $[1; N]$, опираясь на определение D_n^k :

- в ответ пойдёт $D_{n-1}^{k-d^2}$, $d < d_1$ — первая цифра числа;
- в ответ пойдёт $D_{n-2}^{k-d^2}$, $d < d_2$ — вторая цифра числа;
- в ответ пойдёт $D_{n-i-1}^{k-d^2}$, $d < d_i$ — i -ая цифра числа.

LIS

Задача. Дана последовательность целых чисел. Найти длину её наибольшей возрастающей подпоследовательности (*longest increasing subsequence, LIS*).

Идея. Что-то...

Числа Фибоначчи

Задача. На прямой доске вбиты гвозди. Соединить пары гвоздей нитками так, чтобы к каждому гвоздю была привязана хотя бы одна нитка, а суммарная длина всех ниток была минимальна.

Идея. Пусть $f(i)$ — суммарная длина ниток для $1 \dots i$ гвоздей.

Определим базовые случаи:

$$f(2) = x_2 - x_1 \quad f(3) = x_3 - x_1$$

Добавим один гвоздь к текущим:

- 1) Оптимально соединяем первые $i - 1$ гвоздей, а последний гвоздь — с $i - 1$ -ым:

$$f(i - 1) + x_i - x_{i-1}$$

- 2) Оптимально соединяем первые $i - 2$ гвоздей, а последний гвоздь — с $i - 1$ -ым:

$$f(i - 2) + x_i - x_{i-1}$$

Тогда значение $f(i)$ — *минимум* среди всех случаев.

Наибольшая подматрица

Задача. В прямоугольной таблице $N \times M$ клетки раскрашены в белый и чёрный цвета. Найти наибольшую по площади прямоугольную область белого цвета.

Идея. Пусть $f(x, k)$ — наибольший номер строки из отрезка $[-1; x]$, на которой в k -ом столбце есть клетка *чёрного* цвета.

Для хранения этой динамики достаточно *одномерного* массива длины N .

Так, для прямоугольника уже определены *две границы*: верхняя и нижняя.

Пусть $\langle i, j \rangle$ — правая нижняя граница прямоугольника.

По условию, нужно расширить прямоугольник влево до *первой* клетки $\langle k, j \rangle$, для которой верно:

$$f(k, j) > f(i, j)$$

За счёт линейного алгоритма поиска *ближайшего большего* за константу конечный алгоритм станет *линейным*.

Теория графов

Ориентированный граф

Граф (*ориентированный граф* или *орграф*) — упорядоченная пара $G = \langle V, E \rangle$, где

V — непустое множество *вершин* (*узлов*);

E — конечное множество *рёбер*, $E \subseteq V \times V$.

Порядок графа — число его вершин.

Размер графа — число его рёбер.

Ребро $e = \langle v, w \rangle$ задаётся вершинами v, w , где v — начало ребра, а w — его конец; вершины v, w являются *соседними*.

Входящая валентность вершины v графа G — число рёбер, чей конец в v :

$$\text{indeg}(v) = |\{\langle u, v \rangle \mid \langle u, v \rangle \in E\}|$$

Исходящая валентность вершины v графа G — число рёбер, чьё начало в v :

$$\text{outdeg}(v) = |\{\langle v, u \rangle \mid \langle v, u \rangle \in E\}|$$

Валентность вершины v графа G — сумма входящей и исходящей валентностей вершины:

$$\text{deg}(v) = \text{indeg}(v) + \text{outdeg}(v)$$

Свойство. Пусть $G = \langle V, E \rangle$ — граф с n вершинами и m рёбрами. Тогда:

$$\sum_{i=1}^n \text{indeg}(v_i) = \sum_{i=1}^n \text{outdeg}(v_i) = m$$

Подграф $G = \langle V, E \rangle$, порождённый на $W \subset V$, — граф вида

$$G_W = \langle W, E \cap W \times W \rangle.$$

Неориентированный граф

Неорграф (*неориентированный граф*) — такой граф $G = \langle V, E \rangle$, что:

$$\forall v, w \in V \langle v, w \rangle \in E \implies \langle w, v \rangle \in E$$

Валентность вершины v неорграфа — число рёбер, которые связаны с v .

Кратными называются два и более рёбер, которые образованы *одинаковыми* вершинами.

Последовательность вершин

Путь от вершины v_i до вершины v_j графа G — последовательность вершин или рёбер:

$$\left\{ \begin{array}{l} [v_i, v_{i+1}, \dots, v_{j-1}, v_j] \text{ вершины} \\ [e_i, e_{i+1}, \dots, e_{j-1}, e_j] \text{ рёбра} \\ e_k = \langle v_{k-1}, v_k \rangle, k \in \{i+1, \dots, j\} \end{array} \right.$$

Закрытым называется такой путь, где начальная и конечная вершины совпадают.

Цепь — путь без повтора рёбер.

Простая цепь — путь без повтора рёбер и вершин (*кроме, возможно, первой и последней вершины*).

Цикл — закрытая простая цепь.

Паросочетание — множество попарно несмежных рёбер.

Эйлеровой называется такая последовательность вершин, которая проходит по всем *рёбрам* графа.

Критерий эйлеровости. Связный неорграф *эйлеров*, если валентность всех его вершин чётна.

Критерий полуэйлеровости. Связный неорграф *полуэйлеров*, если:

- валентность всеъ вершин *чётна*;
- ноль или две вершины имеют *нечётную* валентность.

Ациклическим (*лесом*) называется граф без циклов.

Виды графов

Полным называется такой неорграф $G = \langle V, E \rangle$, что:

$$E = V \times V$$

Однородным называется такой неорграф, у которого *валентности* всех вершин равны.

Транспонированным называется такой граф G^T по отношению к G , у которого все рёбра *инвертированы*.

Взвешенным называется такой граф, в котором каждому ребру сопоставляется число — *вес, длина, стоимость*.

Связность

Связным называется:

- *неорграф*, между любыми вершинами которого есть маршрут;
- *орграф*, у которого аналогичный неорграф *связный*.

Сильно связным называется такой *орграф* $G = \langle V, E \rangle$, что:

$$\forall v, w \in V \exists \begin{cases} \text{маршрут от } v \text{ до } w \\ \text{маршрут от } w \text{ до } v \end{cases}$$

Точка сочленения — вершина, удаление которой делает граф *несвязным*.

Мост — ребро, удаление которого делает граф *несвязным*.

Компонента связности неорграфа — связный подграф, который не входит в состав такого же подграфа.

Компонента сильной связности орграфа — сильно связный подграф, который не входит в состав такого же подграфа.

Дерево

Свободное дерево T — компонента связности леса.

Свойство. Пусть $T = \langle V, E \rangle$. Тогда $|E| = |V| - 1$.

Поддерево $T = \langle V, E \rangle$, порождённое на $W \subset V$, — дерево вида:

$$T_W = \langle W, E \cap W \times W \rangle$$

Корневое дерево (ориентированное дерево или *ордерево*) — такой орграф, у которого:

- аналогичный неорграф есть свободное дерево;
- есть **корень** — единственная вершина с нулевой входящей валентностью.

Остовное дерево — ациклический связный подграф неорграфа, в который входят все его вершины.

Минимальным (миностовом) называется такое *остовное дерево*, суммарный вес рёбёр которого минимален.

Вершины дерева

Пусть T — корневое дерево, причём $\langle v, w \rangle \in E_T$:

- **родитель** вершины w — это $v =: \text{parent}_w$;
- **ребёнок** вершины v — это $w \in \text{children}_v$.

Корневым называется узел без родителей (с нулевой входящей валентностью).

Листовым называется узел без детей (с нулевой исходящей валентностью).

Сиблинги — вершины с общими родителями.

Уровень вершины v — длина простой цепи от root_T до v :

depth_v — обозначение

Диаметр дерева T — максимальная длина (в рёбрах) кратчайшего пути в T между любыми двумя вершинами.

Рёбра леса

Пусть $G = \langle V, E \rangle$ — лес:

- **обратное ребро** соединяет вершину с её *предком*;

- **прямое** ребро соединяет вершину с её *потомком*;
- **перекрёстное** ребро принадлежит множеству $V \times V \setminus E$.

Способы представления графа

Матрица смежности для $G = \langle V, E \rangle$ — булева матрица V^2 , элементы которой равны логическому значению выражения:

$$\langle v, w \rangle \in E \mid v, w \in V$$

Матрица занимает $\mathcal{O}(V^2)$ места; проверка смежности проходит за $\mathcal{O}(1)$.

Список смежности — хеш-таблица вида:

$$\text{вершина} \mapsto \text{смежные узлы}$$

Список занимает $\mathcal{O}(|V| + |E|)$ места; проверка смежности проходит за $\mathcal{O}(\text{outdeg}(v))$.

Способы представления дерева

Массив родителей — хеш-таблица вида:

$$v \mapsto \text{children}_v$$

Массив занимает $\mathcal{O}(|V|)$ места; вывод родителя и порядка дерева проходят за $\mathcal{O}(1)$.

«**Первый ребёнок, следующий сиблинг**» — хеш-таблица вида:

$$v \mapsto \langle \text{first}_v, \text{next}_v \rangle$$

Первый в памяти ребёнок узла v — first_v , *последний* — last_v ; *следующий* в памяти родственник узла v — next_v .

Массив занимает $\mathcal{O}(|V|)$ места; вывод первого ребёнка, следующего родственника и порядка дерева проходят за $\mathcal{O}(1)$.

Редактирование дерева

К **элементарным операциям** редактирования дерева относятся:

- *удаление* листового узла v с ребром $\langle \text{parent}_v, v \rangle$: $v \mapsto \lambda$;
- *вставка* листового узла v с ребром $\langle \text{parent}_v, v \rangle$: $\lambda \mapsto v$;
- *замещение* вершины v другой вершиной w : $v \mapsto w$.

Пусть $T_1 = \langle V_1, E_1 \rangle$, $T_2 = \langle V_2, E_2 \rangle$ — корневые деревья.

Трансформация T_1 в T_2 — упорядоченное биективное отображение $E \subseteq V_1 \cup \{\lambda\} \times V_2 \cup \{\lambda\}$.

Биективное отображение T_1 в T_2 — такое $M \subseteq W_1 \times W_2$ для $W_1 \subseteq V_1$, $W_2 \subseteq V_2$, что:

$$\begin{cases} \langle \text{root}_{T_1}, \text{root}_{T_2} \rangle \in M \neq \emptyset \\ \langle \text{parent}_v, \text{parent}_w \rangle \in M \iff \langle v, w \rangle \in M \\ v_2 = \text{next}_{v_1}, w_2 = \text{next}_{w_1} \iff \langle v_1, w_1 \rangle, \langle v_2, w_2 \rangle \in M \end{cases}$$

Лемма. Пусть M — отображение T_1 в T_2 . Тогда:

$$\forall \langle v, w \rangle \in M \text{ depth}_v = \text{depth}_w$$

Стоимость элементарной операции над T_1 и T_2 задаётся метрикой $\gamma: V_1 \cup V_2 \cup \{\lambda\} \times V_1 \cup V_2 \cup \{\lambda\} \rightarrow \mathbb{R}_0^+$.

Стоимость трансформации T_1 в T_2 (E) задаётся метрикой:

$$\gamma(E) = \sum_{\langle v, w \rangle \in E} \gamma(v, w)$$

Редакционная дистанция между T_1 и T_2 — функция:

$$\gamma_{\min} = \min(\{\gamma(E) \mid \forall E\})$$

Редакционный граф для T_1 и T_2 — неорграф $G = \langle V, E \rangle$ с вершинами вида vw , $v \in V_1 \cup \{v_0\}$, $w \in V_2 \cup \{w_0\}$ (v_0, w_0 — мнимые узлы), рёбра которого определяются по правилу:

$$\begin{cases} \text{depth}_{v_{i+1}} \geq \text{depth}_{w_{j+1}} \iff \langle v_i w_j, v_{i+1} w_j \rangle \in E \quad (v_{i+1} \mapsto \lambda) \\ \text{depth}_{v_{i+1}} = \text{depth}_{w_{j+1}} \iff \langle v_i w_j, v_{i+1} w_{j+1} \rangle \in E \quad (v_{i+1} \mapsto w_{j+1}) \\ \text{depth}_{v_{i+1}} \leq \text{depth}_{w_{j+1}} \iff \langle v_i w_j, v_i w_{j+1} \rangle \in E \quad (\lambda \mapsto w_{j+1}) \end{cases}$$

Лемма. Пусть G — редакционный граф для T_1 и T_2 . Тогда маршрут P от $v_0 w_0$ до $v_{n_1} w_{n_2}$ задаёт трансформацию:

$$\begin{aligned} E = & \{ \langle v_{i+1}, \lambda \rangle \mid \langle v_i w_j, v_{i+1} w_j \rangle \in P \} \cup \dots \\ & \dots \{ \langle v_{i+1} w_{j+1} \rangle \mid \langle v_i w_j, v_{i+1} w_{j+1} \rangle \in P \} \cup \dots \\ & \dots \{ \langle \lambda, w_{j+1} \rangle \mid \langle v_i w_j, v_i w_{j+1} \rangle \in P \} \end{aligned}$$

Алгоритм редактирования дерева занимает $\mathcal{O}(n_1 n_2)$ места,

используя $\mathcal{O}(n_1 n_2)$ времени.

Обход дерева

Обход дерева $T = \langle V, E \rangle$ — биективное отображение:

$$\text{order}: V \rightarrow \{1, \dots, |V|\}$$

Прямым называется такой обход дерева $T = \langle V, E \rangle$, что:

$$\begin{cases} \text{order}(\text{root}_T) = 1 \\ \text{order}(\text{first}_v) = \text{order}(v) + 1, \text{ first}_v \neq \emptyset \\ \text{order}(\text{next}_v) = \text{order}(v) + \text{size}(v), \text{ next}_v \neq \emptyset \end{cases}$$

Алгоритм прямого обхода дерева занимает линейное место, используя линейное время.

Алгоритмы на графах

Обход в глубину

Обход в глубину (*Depth-First Search, DFS*) — метод, при котором граф обходят сначала по детям, потом по сиблингам.

Алгоритмы:

- поиск эйлерового пути, цикла
- тест ацикличности
- поиск мостов, точек сочленения
- топологическая сортировка
- построение компонент связности:
 - > обыкновенных (*грядки, водостоки*)
 - > сильных (*алгоритм Косараджу, конденсация*)
- 2-SAT
- алгоритм Куна

Маркировка. Когда нужно найти *циклы*, вершины маркируются в зависимости от типа графа:

- неорграф \Rightarrow *бинарные* метки
- орграф \Rightarrow *тернарные* метки

Обход в ширину

Обход в ширину (*Breadth-First Search, BFS*) — метод, при котором граф обходят сначала по сиблингам, потом по детям.

Алгоритмы:

- алгоритм Кана
- проверка графа на двудольность
- поиск кратчайших рёберных путей

Нет циклов \Rightarrow не требуется массив *visited*.

Также были задачи на потоки. Рассмотреть их!

Поиск эйлерового пути

Алгоритм поиска эйлерового *цикла*:

```

if is_eulerian(graph): #1
    ecirc = graph.get_ecirc(v) #2
else:
    ecirc = -1

def get_ecirc(v): # DFS
    ecirc = list()
    for n in edges[v]:
        if edges[v][n] > 0:
            edges[v][n] -= 1
            ecirc += get_ecirc(n)
    return ecirc + [v] #3

```

- 1) Проверить граф на *эйлеровость*.
- 2) Начать с любой вершины v .
- 3) Если дан оргграф, ответ нужно *инвертировать*.

Алгоритм поиска эйлерового пути:

```

if is_semi_eulerian(graph): #1
    epath = graph.get_epath(v) #2
else:
    epath = -1

def get_epath(v): # DFS
    epath = list()
    for n in edges[v]:
        if edges[v][n] > 0:
            edges[v][n] -= 1
            epath += get_epath(n)
    return epath + [v] #3

```

- 1) Проверить граф на *полуэйлеровость*.
- 2) Начать с вершины нечётной степени v .

НО: если дан оргграф, выбрать вершину большей *исходящей* валентности.

- 3) Если дан оргграф, ответ нужно *инвертировать*.

Применяется в **задачах**:

- китайский почтальон;
- домино, восстановление строки.

Топологическая сортировка

Топологическая сортировка — упорядочивание вершин ориентированного леса согласно *частичному порядку*, который задан рёбрами орграфа.

Алгоритм Тарьяна — реализация через *DFS*:

```
def topo_sort(v): # DFS
    vertices[v].visited = 1
    stack = list()
    for n in vertices[v].adjacent:
        if not vertices[n].visited:
            stack += topo_sort(n)
    return stack + [v]
```

Алгоритм Кана — реализация через *BFS*:

```
def topo_sort(queue=deque()): # BFS
    for v in vertices: #1
        if vertices[v].indegree == 0:
            queue.append(v)
    order = list()
    while queue:
        q = queue.popleft()
        order.append(q)
        for n in vertices[q].adjacent_out:
            vertices[n].indegree -= 1
            if vertices[n].indegree == 0: #2
                queue.append(n)
    return order
```

- 1) Найти *корни* графа, добавить их в очередь.
- 2) Если какая либо из соседних вершин стала *корнем*.

Применяется в **задачах**:

- лексикографическая сортировка;
- топологическое маркирование;
- тест ацикличности.

Задача. В игре есть N уровней, соединённых M телепортами. Сколько есть способов добраться от первого уровня до N -го? *(телепорты не образуют циклов)*

Идея. Представим уровни как вершины, а телепорты как рёбра графа.

Пусть каждая вершина маркирована числом путей, исходящих от первой вершины *(оно не всегда равно входящей валентности)*.

Запустим алгоритм Кана из первой вершины, и при обработке ребёнка очередной вершины будем добавлять к его маркировке родительскую.

Ответ к задаче — значение маркировки вершины N .

Задача. Требуется выполнить N курсов. Есть M требований вида «курс a должен быть выполнен до курса b ». Составить, если возможно, порядок прохождения курсов.

Идея. Представим курсы как вершины, а требования как рёбра графа.

Проверим, что граф *ацикличен*: иначе решение составить невозможно.

Запустим алгоритм Кана из корневой вершины — полученная последовательность удовлетворяет условию.

Задача. Леви нужно добраться от города 1 до N , но он хочет это сделать через наибольшее количество промежуточных городов. Составить, если возможно, такой маршрут.

Идея. Представим города как вершины, а рейсы как рёбра графа.

Проверим, что граф *ацикличен*: иначе решение составить невозможно.

Пусть каждая вершина маркирована максимальным путём, исходящих от первой вершины.

Запустим алгоритм Кана из первой вершины, и при обработке ребёнка очередной вершины будем добавлять к его маркировке максимальный путь из родительских с инкрементом.

Ответ к задаче — значение маркировки вершины N .

Алгоритм Косараджу

Алгоритм поиска компонент сильной связности:

```
reverse = graph.transpose() #1
order = reverse.topo_sort() #2
mark = 0
for v in order[::-1]: #3
    if not graph.vertices[v].visited:
        mark_component(v, mark)
        mark += 1

def mark_component(v, mark): # DFS
    vertices[v].visited = 1
    vertices[v].mark = mark
    for n in vertices[v].adjacent:
        if not vertices[n].visited:
            mark_component(n, mark)
```

- 1) Построить *транспонированный* граф *reverse*.
- 2) Применить *топологическую сортировку* к *reverse*.
- 3) Применить *DFS* на *graph* в обратном порядке топологической сортировки:

цикл поиска в глубину \equiv сильная компонента связности

2-SAT

Алгоритм решения:

- 1) Перевести CNF в INF.
- 2) Построить граф импликаций.

- 3) Найти компоненты сильной связности в графе.
- 4) Проверить, что для любой вершины x графа справедливо:

$$c[x] \neq c[\neg x]$$

- 5) Если требуется вывести ответ, то воспользоваться формулой:

$$x = \begin{cases} \text{true,} & c[x] < c[\neg x] \\ \text{false,} & c[x] > c[\neg x] \end{cases}$$

ПОИСК МОСТОВ

Алгоритм поиска мостов на неорграфе:

```
def get_bridges(v, parent=-1): # DFS
    vertices[v].visited = 1
    time += 1
    vertices[v].ord = time #1
    vertices[v].min = time
    bridges = set()
    for n in vertices.adjacent:
        if n == parent: #2
            continue
        if vertices[n].visited: #3
            vertices[v].min = min(vertices[v].min, \
                                   vertices[n].ord)
        else: #4
            bridges |= get_bridges(n, v) #5
            vertices[v].min = min(vertices[v].min, \
                                   vertices[n].min)
            if vertices[n].min > vertices[v].ord and \
                parent != -1: #6
                bridges.add((v, n))
    return bridges
```

- 1) Запись текущего порядка захода в вершину.
- 2) Если $\langle v, n \rangle$ — ребро в обратную сторону.
- 3) Если $\langle v, n \rangle$ — обратное ребро.
- 4) Если $\langle v, n \rangle$ — ребро дерева.

- 5) Мост может быть отмечен *несколько раз* за алгоритм.
 6) Критерий моста:

$$[\langle v, neigh \rangle \text{ — мост}] = \begin{cases} \text{true,} & \min[neigh] > \text{ord}[v] \\ \text{false,} & \min[neigh] \leq \text{ord}[v] \end{cases}$$

Поиск точек сочленения

Алгоритм поиска точек сочленения на неорграфе:

```
def get_cuts(v, parent=-1): # DFS
    vertices[v].visited = 1
    time += 1
    vertices[v].ord = time #1
    vertices[v].min = time
    children, cuts = 0, set()
    for n in vertices.adjacent:
        if n == parent: #2
            continue
        if vertices[n].visited: #3
            vertices[v].min = min(vertices[v].min, \
                                   vertices[n].ord)
        else: #4
            children += 1
            cuts |= get_cuts(n, v) #5
            vertices[v].min = min(vertices[v].min, \
                                   vertices[n].min)
            if vertices[n].min >= vertices[v].ord and \
                parent != -1: #6
                cuts.add(v)
    if children > 1 and parent == -1:
        cuts.add(v)
    return cuts
```

- 1) Запись текущего порядка захода в вершину.
- 2) Если $\langle v, n \rangle$ — ребро в *обратную сторону*.
- 3) Если $\langle v, n \rangle$ — *обратное ребро*.
- 4) Если $\langle v, n \rangle$ — *ребро дерева*.

- 5) Вершина может быть отмечена *несколько раз* за алгоритм.
- 6) Критерий точки сочленения:

$$[v \text{ — т. сочленения}] = \begin{cases} \text{true,} & \min[\text{neigh}] \geq \text{ord}[v] \\ \text{false,} & \min[\text{neigh}] < \text{ord}[v] \end{cases}$$

Алгоритм Куна

Алгоритм поиска максимального паросочетания на двудольном графе:

```
for v in vertices.part: #1
    if has_ichain(v):
        null_all_visited(vertices)

matching = dict() # empty == None
def has_ichain(v): # DFS
    if vertices[v].visited:
        return False
    vertices[v].visited = 1
    for n in vertices[v].adjacent:
        if matching[n] == None or \
           has_ichain(matching[n]): #2
            matching[n] = v
            return True
    return False
```

- 1) Рассматриваются все вершины *одной доли* двудольного графа.
- 2) Если пары нет или если есть *увеличивающаяся цепь*.

Алгоритм Дейкстры

Алгоритм Дейкстры ищет кратчайшие пути из заданной вершины до всех остальных вершин взвешенного графа (*вес рёбер неотрицательный*).

Наивная реализация за $\mathcal{O}(n^2 + m)$:

```

def dijkstra(root): # tabulation
    # base (default = INF)
    verts[root].min = 0
    while True:
        # minimization -  $O(N)$ 
        v = -1
        for i in verts:
            if not verts[i].visited and \
                (v == -1 or verts[i].min < verts[v].min):
                v = i
        if v == -1: # reachable vertices are visited
            break
        verts[v].visited = 1
        # relaxation
        for n in verts[v].adjacent:
            curr_cost = verts[v].min + edges[n][v].cost
            if curr_cost < verts[n].min:
                verts[n].min = curr_cost
                verts[n].parent = v # certificate

```

Кучная реализация за $O(m \log n)$:

```

def dijkstrta(root):
    # base (default = INF)
    verts[root].min = 0
    queue = [(0, root)]
    heapq.heapify(queue)
    while queue:
        # minimization -  $O(\log N)$ 
        dist, v = heapq.heappop(queue)
        if dist > verts[v].min:
            continue
        # relaxation
        verts[v].visited = True
        for n in verts[v].adjacent:
            if verts[n].visited:
                continue

```

```

curr_cost = verts[v].min + edges[v][n].cost
if curr_cost < verts[n].min:
    verts[n].min = curr_cost
    verts[n].parent = v # certificate
    heapq.heappush((verts[n].min, n))

```

Алгоритм A*

Алгоритм A* ищет кратчайший путь между двумя заданными вершинами взвешенного графа на основе эвристики (*вес рёбер неотрицательный*).

Кучная реализация алгоритма за $\mathcal{O}(m \log n)$:

```

def a_star(start, goal):
    # base (default = INF)
    verts[start].min = 0
    queue = [(eur(start), start)]
    heapq.heapify(queue)
    while queue:
        # minimization -  $\mathcal{O}(\log N)$ 
        heur, v = heapq.heappop(queue)
        if v == goal:
            break
        if heur != verts[v].heur:
            continue
        # relaxation
        verts[v].visited = True
        for n in verts[v].adjacent:
            if verts[n].visited:
                continue
            curr = verts[v].min + edges[v][n].cost
            if curr < verts[n].min:
                verts[n].min = curr
                verts[n].heur = verts[n].min + heur(n)
                verts[n].parent = v # certificate
                heapq.heappush(queue, (verts[n].heur, n))

```

Кратчайший путь

Волновой алгоритм — да.

Алгоритм Беллмана-Форда — да.

Алгоритм Прима

Алгоритм поиска *миностова*:

```
def prim(start): # greedy
    # base (default = INF)
    verts[start].min = 0
    while True:
        # minimization
        v = -1
        for i in verts:
            if not verts[i].visited and \
                (v == -1 or verts[i].min < v[1]):
                v = i
        if v == -1: # reachable vertices are visited
            break
        verts[v].visited = 1
        if verts[v].parent != -1: # got safe edge
            add_safe_edge(v, verts[v].parent)
        # relaxation
        for n in verts[v].adjacent:
            curr_cost = edges[v][n].cost
            if curr_cost < verts[n].min:
                verts[n].min = curr_cost
                verts[n].parent = v # certificate
```

Структура идентична *алгоритму Дейкстры*.

Disjoint Set Union

Система непересекающихся множеств (*CHM*, или *DSU*)
— дерево, которое обладает операциями:

— `make_set(v)` — создать новое множество за $\mathcal{O}(1)$

- `find_set(v)` — найти множество элемента за $\mathcal{O}(\log n)$
- `union(v, u)` — объединить два множества за $\mathcal{O}(\log n)$

Создание нового множества:

```
def make_set(v):
    prev[v] = v # tree root
    rank[v] = 0 # tree depth (~log n)
```

Определение множества элемента:

```
def find_set(v): # path compression heuristic
    if v == prev[v]:
        return v
    return find_set(prev[v])
```

Объединение двух множеств:

```
def unite(v, u):
    v, u = find_set(v), find_set(u)
    if v != u:
        if rank[v] < rank[u]: # rank heuristic
            v, u = u, v
        prev[u] = v
        if rank[v] == rank[u]:
            rank[v] += 1
```

Задача. Дан взвешенный неорграф $G(N, M)$. Цена пути между вершинами — вес его максимального ребра. Найти число пар с мин. ценой пути между ними, равной X .

Идея. Добавим «рёбра» весом $< X$ в СНМ, метрикой которого является количество вершин.

Искомое число — суммарная *разница* числа новых и старых связей* при добавлении рёбер весом X (так исключатся их внутренние связи).

* — число пар всех вершин в n -компоненте:

$$S_n = \frac{n(n-1)}{2}$$

Алгоритм Краскала

Алгоритм Краскала помогает составить *миностов* на основе *DSU*.

В СНМ хранятся вершины графа, которые объединяются *безопасными рёбрами* в порядке *возрастания веса*.

Безопасным называется ребро, которое соединяет *разные* компоненты связности.

Алгоритм поиска миностова:

```
def kraskal():
    span = list()
    edges.sort()
    for i in range(len(edges)):
        e = edges[i]
        if find_set(e.start) != find_set(e.end):
            union(e.start, e.end)
            span.append(i)
```

Задача. Дан архипелаг из N островов с M мостами. Стоимость постройки моста — *расстояние* между островами. Объединить архипелаг мостами за *минимальную* стоимость.

Идея. Представим архипелаг как несвязный неорграф, который нужно сделать связным.

Сделаем граф полным, добавив все отсутствующие рёбра.

Ответ на задачу — стоимость *миностова*, который можно составить *алгоритмом Краскала*.

Дерево отрезков

Дерево отрезков (*segment tree*) — бинарное дерево, которое обладает операциями:

- `build(arr)` — построить дерево на массиве arr за $\mathcal{O}(n)$
- `get(l, r)` — вернуть $f(arr[l:r])$ за $\mathcal{O}(\log n)$
- `get(i)` — вернуть $f(arr[i])$ за $\mathcal{O}(\log n)$

- `update(i, val)` — обновить $arr[i]$ за $\mathcal{O}(\log n)$
- `update(l, r, val)` — обновить $arr[l:r]$ за $\mathcal{O}(\log n)$

Требования к операции f :

- ассоциативность
- наличие нейтрального элемента \perp

Построение дерева «сверху»:

```
def build(root, tl, tr):
    if tl + 1 == tr: # we use [tl; tr)
        verts[root].value = arr[tl]
    else:
        tm = (tl + tr) // 2
        build(2 * root, tl, tm)
        build(2 * root + 1, tm, tr)
        verts[root].value = f(verts[2 * v].value, \
                               verts[2 * v + 1].value)
```

Получение значения функции на отрезке:

```
def get(root, tl, tr, ql, qr):
    if [tl;tr) ∩ [ql;qr) = ∅: # don't go further
        return ⊥
    if [tl;tr) ⊆ [ql;qr): # subtree is covered
        return verts[root].value
    tm = (tl + tr) // 2
    return f(get(2 * root, tl, tm, ql, qr), \
             get(2 * root + 1, tm, tr, ql, qr))
```

Обновление элемента массива:

```
def update(root, tl, tr, i, val):
    if tl + 1 == tr:
        verts[root] = val
        return
    tm = (tl + tr) // 2
    if i < tm:
        update(2 * root, tl, tm, i, val)
```

```

else:
    update(2 * root + 1, tm, tr, i, val)
verts[root].value = f(verts[2 * root].value, \
                      verts[2 * root + 1].value)

```

Несогласованные поддеревья

Отложенной (массовой) называется операция, которая применяется к *подотрезку* массива в дереве отрезков.

Несогласованным называется поддерево, которое хранит в своих вершинах *частичный* результат выполнения отложенной операции.

Проталкивание массовой операции g :

```

def push(root, tl, tr):
    if tl + 1 != tr:
        verts[2 * root] = g(verts[2 * root], \
                           verts[root].diff)
        verts[2 * root + 1] = g(verts[2 * root + 1], \
                               verts[root].diff)
    verts[root].diff = 0

```

Обновление отрезка массива:

```

def update(root, tl, tr, ql, qr, val):
    push(root, tl, tr) # ???
    if  $[t_l; t_r) \cap [q_l; q_r) = \emptyset$ : # don't go further
        return  $\perp$ 
    if  $[t_l; t_r) \subseteq [q_l; q_r)$ : # subtree is covered
        verts[root].diff = g(verts[root].diff, val)
        return
    push(root, tl, tr) # ???
    tm = (tl + tr) // 2
    update(2 * root, tl, tm, ql, qr)
    update(2 * root + 1, tm, tr, ql, qr)
    verts[root].val = f(g(verts[2 * root].val, \
                        verts[2 * root].diff), \
                      g(verts[2 * root + 1].val, \

```

```
verts[2 * root + 1].diff))
```

Получение значения функции на отрезке:

```
def get(root, tl, tr, ql, qr):  
    if qr <= tl or tr <= ql: # don't go further  
        return 1  
    if ql <= tl and tr <= qr: # subtree is covered  
        return g(verts[root].val, verts[root].diff)  
    push(root, tl, tr)  
    ans = f(get(2 * root, tl, tm, ql, qr), \  
            get(2 * root + 1, tm, tr, ql, qr))  
    verts[root].val = f(g(verts[2 * root].val, \  
                        verts[2 * root].diff), \  
                        g(verts[2 * root + 1].val, \  
                          verts[2 * root + 1].diff))  
    return ans
```

Корневая декомпозиция

Да.

Алгоритмы на строках

Префикс-функция

Собственным называется *префикс*, который не совпадает со всей строкой.

Грань — собственный префикс строки, который равен её суффиксу.

Префикс-функция строки S — массив, i -ый элемент которого равен длине максимальной грани подстроки $S[0 \dots i]$.

Алгоритм реализации *префикс-функциции*:

```
prefixes = [0] * len(S)
def fill_prefixes(S, prefixes):
    for i in range(len(S) - 1):
        j = 0
        while j and S[i + 1] != S[j]: # I
            j = prefixes[j - 1]
        if S[i + 1] == S[j]:
            j += 1
        prefixes[i + 1] = j
```

I



Цикл смены индекса j идёт до равенства с $S(i + 1)$ или до границ индексации.

Алгоритм КМП

Задача. Даны строки *text* и *search*. Найти позиции всех вхождений *search* в *text*.

Идея. Образовать новую строку конкатенацией:

$search + \# + text$

Вычислить *префикс-функцию* от новой строки: индексы элементов, которые численно равны длине *search*, будут ответом.

Вхождения префиксов

Задача. Дана строка S . Посчитать для каждого префикса S , сколько раз он встречается в S .

Идея. Вычислить префикс-функцию от S . Затем составить отдельный массив *occur* по принципу:

- 1) Посчитать для каждого значения π , сколько раз оно встречалось в π .
- 2) Посчитать *динамику*: вхождения большего префикса i добавить к наибольшему собственному суффиксу этого префикса $prefixes[i - 1]$.

```
fill_prefixes()
for i in range(len(S)):
    occur[prefixes[i]] += 1
for i in range(len(S) - 1, 0, -1):
    occur[prefixes[i - 1]] += occur[i]
```

Учёт различных подстрок

Задача. Дана строка S . Посчитать количество её различных подстрок.

Идея. Добавим один символ в конец неполной строки: появятся новые подстроки, которые оканчиваются в новом символе. Нужно найти *уникальные* среди них.

Инвертируем строку, чтобы символ стал префиксом. Тогда число уникальных подстрок равно числу элементов π с нулевым значением:

$$\text{len}(S) - \pi_{\max}$$

Аналогично можно дописывать символы в начало или удалять символы с конца или с начала.

Сжатие строки

Задача. Дана строка S . Найти такую строку наименьшей длины, что S можно представить в виде конкатенации её копий.

Идея. Пусть $n = \text{len}(S)$, $k = n - \pi[n - 1]$. Тогда верно:

$$k \mid n \implies k \text{ — длина искомой строки}$$

Иначе S невозможно сжать.

Z-функция

Блок строки S — подстрока S , которая равна её собственному префиксу.

Z-функция строки S — массив, i -ый элемент которого равен длине максимального блока с началом в i .

Алгоритм реализации *z-функции*:

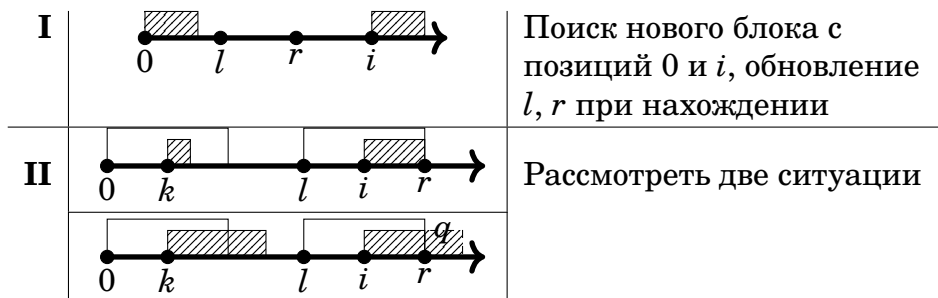
```
def get_block(S, l, r):
    idx = 0
    while S[l + idx] == S[r + idx]:
        if r + idx < len(S):
            break
        idx += 1
    return idx

blocks = [0] * len(S)
def fill_blocks(string, blocks):
    l, r = 0, 0
    for i in range(len(S)):
        if i > r: # I
            value = get_block(S, 0, i)
            if value:
                l, r = i, value
        else: # II
            k = i - l
            if blocks[k] < r - i + 1:
```

```

        blocks[i] = blocks[k]
    else:
        blocks[i] = r - i + 1
        l, q = i, get_block(S, blocks[i], r + 1)
        if q:
            blocks[i] += q
            r = i + blocks[i] - 1

```



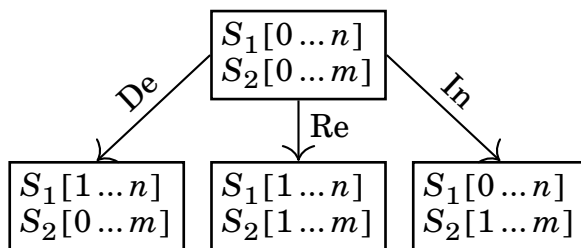
Редакционное расстояние

К элементарным операциям редактирования строки относятся:

- удаление символа;
- вставка символа;
- замещение символа.

Редакционное расстояние между строками S_1 и S_2 — минимальное количество *элементарных операций редактирования*, которые нужно совершить, чтобы перевести S_1 в S_2 .

Редакционный граф для строк S_1 и S_2 — орграф с вершинами-состояниями строк, у которых есть до трёх рёбер, эквивалентных элементарным операциям редактирования:



Алгоритм Вагнера-Фишера

Редакционное предписание — сертификат редакционного расстояния между двумя строками:

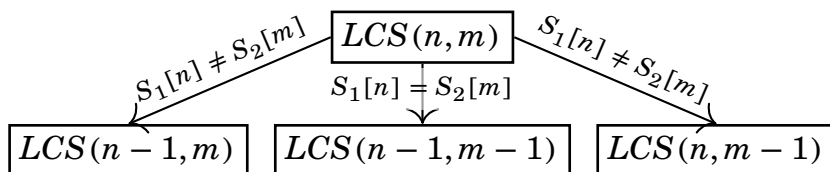
Delete, Insert, Replace, Match.

Алгоритм Вагнера-Фишера находит редакционное предписание по *матрице расстояний*, минимизируя выбор элементарных операций редактирования.

Общая подпоследовательность

Задача. Даны строки S_1 и S_2 . Определить длину их наибольшей общей подпоследовательности $LCS(S_1, S_2)$.

Идея. Используем принцип оптимальности на префиксе:



- 1) Если $S_1[n] = S_2[m] \Rightarrow$ инкрементируем прошлый узел.
- 2) Если $S_1[n] \neq S_2[m] \Rightarrow$ максимизируем прошлые узлы.

Алгоритм Нидлмана-Вунша

Задача. Даны две строки S_1 и S_2 . Составить их оптимальное выравнивание.

Идея. Составить *матрицу схожести* с весовой функцией:

- +1 — вес совпадения;
- $-\mu$ — штраф за замену;
- $-\delta$ — штраф за удаление, вставку.

Задача сводится к нахождению пути с *максимальным весом*.

Скобочная последовательность

Когда-нибудь...

Алгоритм сортировочной станции

Обратная польская нотация — форма записи математических выражений, в которой операторы расположены *после* операндов.

Задача. Дано математическое выражение в инфиксной нотации. Перевести его в постфиксную нотацию.

Идея. Сначала запарсить строку *регулярным выражением*:

$$\backslash d + \backslash . ? \backslash d * | [+ \backslash - * / ^ ()]$$

Затем ввести две структуры: *строку* с ответом и *стек* для операторов. Их заполнение происходит при считывании по токену за раз:

- 1) Токен — *число* \Rightarrow добавить к строке.
- 2) Токен — *бинарный оператор*:
 - > если приоритет последнего элемента в стеке \geq , чем у токена, то вытолкнуть его из стека в строку (*повторить при необходимости*);
 - > добавить токен в стек.
- 3) Токен — *открывающая скобка* \Rightarrow добавить в стек.
- 4) Токен — *закрывающая скобка* \Rightarrow вытолкнуть все операторы до «(» из стека в строку, а «(» удалить.

Для поддержки унарных операторов ввести флаги *next_unary* и *is_unary*.

DP по цифрам

Задача. Найти количество двоичных последовательностей из n элементов без k единиц подряд.

Идея. Пусть $f(x)$ — количество двоичных последовательностей из x элементов без k единиц подряд.

Определим базовые случаи:

$$\begin{aligned} f(0) &= 2^0 & f(k-1) &= 2^{k-1} \\ f(1) &= 2^1 & f(k) &= 2^k - 1 \end{aligned}$$

Рассмотрим случаи, когда k к последовательности

дописывается одна цифра:

$$011 \dots 010 + 0 = \underbrace{011 \dots 010}_f(x-1) \text{ seq. } 0$$

$$011 \dots 010 + 1 = \underbrace{011 \dots 010}_f(x-1) \text{ seq. } 1$$

Но в конце предыдущей последовательности может находиться $k - 1$ единиц, поэтому исключим её:

$$\underbrace{011 \dots 010}_f(x-k-1) \text{ seq. } 0 \underbrace{1 \dots 1}_{k-1}$$

Значит, конечная рекуррентная формула имеет вид:

$$f(x) = 2f(x-1) - f(x-k-1)$$

Задача. Найти количество n -значных чисел, у которых сумма любых двух соседних цифр — простое число.

Идея. Пусть $f(x, k)$ — количество x -значных чисел, у которых сумма любых двух соседних цифр — простое число, причём они оканчиваются на k .

Определим базовые случаи:

$$f(0, i) = 0 \quad f(1, j) = 1, \text{ но } f(1, 0) = 0$$

Тогда рекуррентная формула примет вид:

$$f(x, k) = \sum_{j+k - \text{простое}} f(x-1, j)$$

Задача. Найти количество строк длины n , которые состоят из символов a, b, c и не содержат подстроки ab .

Идея. Пусть $f(x, k)$ — количество строк длины x , которые оканчиваются символом k .

Определим базовые случаи:

$$f(0, k) = 1 \quad f(1, k) = 1$$

Тогда рекуррентная формула примет вид:

$$f(x, k) = \begin{cases} f(x-1, a) + f(x-1, c), & k = a \\ f(x-1, a) + f(x-1, b) + f(x-1, c), & k = b, c \end{cases}$$

Ответ — сумма $f(n, a) + f(n, b) + f(n, c)$.

Задача. Назовём число *интересным*, если его цифры идут в неубывающем порядке. Сколько интересных положительных чисел лежит в диапазоне $[L; R]$?

Идея. Пусть $f(x, k)$ — количество x -значных интересных чисел, которые оканчиваются на k (*ведущие нули допускаются*).

Представим R в виде $\overline{a_1 \dots a_n}$. Пусть y — интересное число в диапазоне $[1; R]$, причём $y = \overline{b_1 \dots b_n}$:

- 1) Если $b_i = a_i$, то $b_{i+1} \leq a_{i+1}$.
- 2) Если $b_i < a_i$, то последующие цифры интересного числа — любые.

Когда-нибудь... возможно...

Задача. Назовём число *гладким*, если его цифры идут в неубывающем порядке. Вывести N -ое гладкое число.

Идея. Пусть $f(x, k)$ — количество x -значных гладких чисел с цифрой k в x -ом разряде (*отсчёт справа*).

Определим базовый случай:

$$f(1, k) = 1 \text{ (нуль считается)}$$

Тогда рекуррентная формула имеет вид:

$$f(x, k) = \sum_{j=k}^9 f(x-1, j)$$

Определимся с числом разрядов у искомого числа:

$$\begin{cases} f(n, 0) \leq N \\ f(n+1, 0) > N \end{cases} \Rightarrow n+1 \text{ — число разрядов}$$

Каждую цифру N -го числа найдём при помощи цикла:

$$1) \text{ count} + f(x, k) \leq N \Rightarrow \text{count} += f(x, k)$$

$$2) \text{ count} + f(x, k) > N \Rightarrow k \text{ — } x\text{-ая цифра}$$

Задача. Назовём число *плавным*, если две соседние цифры различаются не более, чем на 1. Определить количество плавных натуральных чисел длины n .

Идея. Когда-нибудь...

Задача. Дан диапазон чисел $[L; R]$. Посчитать сумму *цифр* всех его натуральных чисел.

Идея. Пусть $f(x)$ — сумма цифр всех натуральных чисел из диапазона $[1; x]$.

Посчитаем некоторые значения:

$$f(9) = 1 + \dots + 9 = 45$$

$$\begin{aligned} f(99) &= f(9) + (10 + f(9)) + \dots + (90 + f(9)) \\ &= 10f(9) + 10 \cdot 45 \end{aligned}$$

$$\begin{aligned} f(999) &= f(99) + (100 + f(99)) + \dots + (900 + f(99)) \\ &= 10f(99) + 100 \cdot 45 \end{aligned}$$

Заметим, что $f(10^k - 1) = 10f(10^{k-1} - 1) + 10^{k-1} \cdot 45$.

Тут ещё добавить пример для $f(328)$...

Допустим, k — количество разрядов числа x , m — его *MSD*. Тогда искомая величина складывается из двух диапазонов:

$$[1; m10^{k-1} - 1] \cup [m10^{k-1}; x]$$

Посчитаем суммы цифр чисел на них:

$$f_1(x) = mf(10^{k-1} - 1) + \frac{m(m-1)}{2}10^{k-1}$$

$$f_2(x) = m(x \bmod 10^{k-1} + 1) + f(x \bmod 10^{k-1})$$

Комбинаторное исчисление

Принципы подсчёта

Правило сложения. Пусть S — конечное множество, образованное объединением подмножеств S_1, \dots, S_k . Тогда:

$$|S| = |S_1| + \dots + |S_k|$$

Правило умножения. Пусть S — конечное множество, которое есть декартово произведение $S_1 \times \dots \times S_k$. Тогда:

$$|S| = |S_1| \times \dots \times |S_k|$$

Правило вычитания. Пусть S — искомое подмножество T , \bar{S} — его дополнение. Тогда:

$$S = T \setminus \bar{S}$$

Задача. Сколько существует четырёхзначных чисел, в записи которых есть хотя бы одна цифра 7?

Идея. Пусть T — количество всех четырёхзначных чисел, S — количество четырёхзначных чисел, в записи которых *нет семёрки*.

Тогда искомое множество равно:

$$\bar{S} = T \setminus S$$

Принцип Дирихле. Пусть S_1, \dots, S_m — конечные непересекающиеся множества, причём:

$$|S_1| + \dots + |S_m| = n$$

Тогда существуют такие $i, j \in [1; m] \cap \mathbb{N}$, что:

$$|S_i| \geq \left\lceil \frac{n}{m} \right\rceil \quad |S_j| \leq \left\lfloor \frac{n}{m} \right\rfloor$$

Основные понятия

Пусть X — конечное множество, $n := |X|$, $[m] := [1; m] \cap \mathbb{N}$.

Упорядоченное разбиение m элементов из X — соответствие

$$s: [m] \rightarrow X.$$

Неупорядоченное разбиение m элементов из X — множество S мощностью m с элементами из X .

Перестановка — упорядоченное биективное разбиение:

$$P_n: [n] \rightarrow X, \quad P_n = n!$$

k-Размещение — упорядоченное инъективное разбиение:

$$A_n^k: [k] \rightarrow X, \quad A_n^k = \frac{P_n}{P_{n-k}}$$

k-Сочетание — неупорядоченное инъективное разбиение:

$$C_n^k: [k] \rightarrow X, \quad C_n^k = \frac{A_n^k}{P_k}, \quad C_n^k \equiv \binom{n}{k}$$

Полиномиальная теорема

Полиномиальными называются коэффициенты $\binom{n}{k_1, \dots, k_r}$ многочлена при $k_1, \dots, k_r \in \mathbb{N}_0$:

$$(x_1 + \dots + x_r)^n = \sum_{\substack{k_1, \dots, k_r \geq 0 \\ k_1 + \dots + k_r = n}} \binom{n}{k_1, \dots, k_r} x_1^{k_1} \dots x_r^{k_r}$$

Теорема. Для $k_1, \dots, k_r \geq 0$ с $k_1 + \dots + k_r = n$ справедливо:

$$\begin{aligned} \binom{n}{k_1, \dots, k_r} &= \binom{n}{k_1} \binom{n - k_1}{k_2} \dots \binom{n - k_1 - \dots - k_{r-1}}{k_r} \\ &= \frac{n!}{k_1! \cdot \dots \cdot k_r!} \end{aligned}$$

Доказательство. Раскроем скобки:

$$(x_1 + \dots + x_r)^n = \sum_{i_1=1}^r \dots \sum_{i_n=1}^r x_{i_1} \dots x_{i_n}$$

Одночлен $x_1 \dots x_r$ равен $x_1^{k_1} \dots x_r^{k_r}$, если среди индексов i_1, \dots, i_n ровно k_j равны $j \in [1; r] \cap \mathbb{Z}$.

Выбор k_j индексов происходит среди $n - k_1 - \dots - k_{j-1}$ оставшихся. Поэтому таких упорядоченных выборов

$$\binom{n-k_1-\dots-k_{j-1}}{k_j}:$$

$$\binom{n}{k_1, \dots, k_r} = \binom{n}{k_1} \binom{n-k_1}{k_2} \dots \binom{n-k_1-\dots-k_{r-1}}{k_r} \quad \square$$

По формуле сочетаний:

$$\begin{aligned} & \frac{n!}{k_1! \cancel{(n-k_1)!}} \cdot \frac{\cancel{(n-k_1)!}}{k_2! \cancel{(n-k_1-k_2)!}} \cdot \dots \cdot \frac{\cancel{(n-k_1-\dots-k_{r-1})!}}{k_r! (n-k_1-\dots-k_r)!} \\ &= \frac{n!}{k_1! \cdot \dots \cdot k_r! \cdot 0!} = \frac{n!}{k_1! \cdot \dots \cdot k_r!} \quad \blacksquare \end{aligned}$$

Формула Паскаля

Для $n \geq 1$ и $0 \leq k \leq n$ справедливо:

$$\binom{n}{k_1, \dots, k_r} = \sum_{i=1}^r \binom{n-1}{k_1, \dots, k_i-1, \dots, k_r}$$

Доказательство. Раскроем скобки:

$$(x_1 + \dots + x_r)^n = \sum_{\substack{k_1, \dots, k_r \\ k_1 + \dots + k_r = n}} \binom{n}{k_1, \dots, k_r} x_1^{k_1} \dots x_r^{k_r}$$

Раскроем скобки иначе:

$$\begin{aligned} (x_1 + \dots + x_r)^n &= (x_1 + \dots + x_r) (x_1 + \dots + x_r)^{n-1} \\ &= (x_1 + \dots + x_r) \cdot \sum_{\substack{k'_1, \dots, k'_r \\ k'_1 + \dots + k'_r = n-1}} \binom{n-1}{k'_1, \dots, k'_r} x_1^{k'_1} \dots x_r^{k'_r} \\ &= \sum_{i=1}^r \sum_{\substack{k'_1, \dots, k'_r \\ k'_1 + \dots + k'_r = n-1}} \binom{n-1}{k'_1, \dots, k'_r} x_1^{k'_1} \dots x_i^{k'_i+1} \dots x_r^{k'_r} \end{aligned}$$

Произведём замену индексов $k_i := k'_i + 1$, $k_j := k'_j$ ($i \neq j$):

$$(x_1 + \dots + x_r)^n =$$

$$\sum_{k_1, \dots, k_r} \sum_{i=1}^r \binom{n-1}{k_1, \dots, k_i-1, \dots, k_r} x_1^{k_1} \dots x_r^{k_r} \blacksquare$$

Принцип включения-исключения

Пусть A_1, \dots, A_n — конечные множества. Тогда верно:

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{k=1}^n (-1)^{k+1} \left| \bigcap_{j=1}^k A_j \right|$$

Доказательство. Пусть $x \in \bigcup_i^n A_i$, причём x содержится в k множествах A_1, \dots, A_k .

Левая часть формулы — 1. Докажем, что правая часть тоже:

$\binom{k}{1}$ раз x встречается во множествах мощностью 1;

...

$\binom{k}{k}$ раз x встречается во множествах мощностью k .

Подставляем биномиальные коэффициенты в формулу:

$$\binom{k}{1} - \binom{k}{2} + \dots + (-1)^{k+1} \binom{k}{k} = \sum_{i=1}^k \binom{k}{i} (-1)^{i+1}$$

По определению биномиальных коэффициентов:

$$\sum_{i=1}^k \binom{k}{i} (-1)^{i+1} = \binom{k}{0} - \sum_{i=0}^k (-1)^i 1^{k-i} = \binom{k}{0} - (1-1)^k = 1 \blacksquare$$

Беспорядок

Беспорядок — перестановка без инвариантов:

$$D_n = n! \sum_{k=0}^n \frac{(-1)^k}{k!}$$

Доказательство. Да.

Факт. Из разложения e в ряд Тейлора следует:

$$D_n = \left\lfloor \frac{n!}{e} \right\rfloor$$

Правило биекции

Пусть $f: X \rightarrow Y$ — биективное соответствие, где X, Y — конечные множества. Тогда:

$$|X| = |Y|$$

Задача. Сколько подмножеств имеет n -множество?

Решение. Пусть Y — n -множество.

Пусть $\overline{x_1 \dots x_n}$ — бинарная n -строка, где x_i указывает на наличие i -го элемента в произвольном множестве $\mathcal{P}(Y)$.

Пусть $f: X \rightarrow \mathcal{P}(Y)$ — соответствие, где X — множество всех возможных бинарных n -строк. Очевидно, что:

$$f \text{ — биекция} \implies |Y| = |X|$$

По правилу умножения:

$$|X| = 2^n \implies |\mathcal{P}(Y)| = 2^n$$

Ответ: $|\mathcal{P}(Y)| = 2^n$.

Биномиальные коэффициенты

Свойство 1. Для $n \in \mathbb{N}$ и $r \in [0; n] \cap \mathbb{Z}$ верно:

$$\binom{n}{r} = \binom{n}{n-r}$$

Доказательство. Пусть A — n -множество, из которого нужно выбрать B — r -подмножество.

По определению биномиальных коэффициентов:

$$\text{число неупорядоченных} \\ \text{выборок } B = \binom{n}{r}$$

С другой стороны, рассмотрим комплемент $A \setminus B$:

$$\text{число неупорядоченных} \\ \text{выборок } A \setminus B = \binom{n}{n-r}$$

Пусть $f: A_1 \rightarrow A_2$ — биективное соответствие, $A_1 = A_2 = A$.

Любой элемент $x \in B \subset A_1$ можно сопоставить $x \in A \setminus B \subset A_2$.
Значит, числа таких сопоставлений равны:

$$\binom{n}{r} = \binom{n}{n-r} \blacksquare$$

Свойство 2. Для $n \in \mathbb{N}$ верно:

$$\sum_{r=0}^n \binom{n}{r} = 2^n$$

Доказательство. Пусть A — n -множество, для которого посчитаем $|\mathcal{P}(A)|$.

С одной стороны, $|\mathcal{P}(A)| = 2^n$ по доказанному.

С другой стороны, посчитаем $|\mathcal{P}(A)|$ через биномиальные коэффициенты: есть $\binom{n}{r}$ способов выбрать r -подмножество.

По правилу сложения:

$$|\mathcal{P}(A)| = \sum_{r=0}^n \binom{n}{r} \Rightarrow \sum_{r=0}^n \binom{n}{r} = 2^n \blacksquare$$

Метод шаров и перегородок

Число способов составить r -мультимножество из n -множества равно:

$$\left(\binom{n}{r} \right) := \binom{n+r-1}{r} = \left(\binom{n}{k-1} \right) + \left(\binom{n-1}{k} \right)$$

Доказательство. Для подсчёта числа всех возможных r -мультимножеств введём $n-1$ *перегородок* — считается, что элементы между двумя соседними перегородками равны.

Таким образом, число способов заполнить $n+r-1$ позиций

с выбором r шаров (или вставкой $n-1$ перегородок) равно:

$$\binom{n+r-1}{r} \square$$

По формуле Паскаля:

$$\begin{aligned} \left(\binom{n}{k-1} \right) + \left(\binom{n-1}{k} \right) &= \binom{n+k-2}{k-1} + \binom{n+k-2}{k} \\ &= \binom{n+k-1}{k} \blacksquare \end{aligned}$$

Задача. Посчитать число неотрицательных целых решений

$$3x_1 + 3x_2 + 3x_3 + 7x_4 = 22.$$

Решение. Методом полного перебора, $x_4 \in \{0, 1, 2, 3\}$.

По методу шаров и перегородок:

$$\begin{aligned} \left[\begin{array}{l} x_4 = 0 \Rightarrow 3(x_1 + x_2 + x_3) = 22 \Leftrightarrow \text{решений нет} \\ x_4 = 1 \Rightarrow 3(x_1 + x_2 + x_3) = 15 \Leftrightarrow x_1 + x_2 + x_3 = 5 \\ x_4 = 2 \Rightarrow 3(x_1 + x_2 + x_3) = 8 \Leftrightarrow \text{решений нет} \\ x_4 = 3 \Rightarrow 3(x_1 + x_2 + x_3) = 1 \Leftrightarrow \text{решений нет} \end{array} \right. \\ \Leftrightarrow \left(\binom{3}{5} \right) = \binom{7}{5} = 21 \end{aligned}$$

Ответ: 21 решение.

Правило деления

Пусть $f: X \rightarrow Y$ — отображение k -к-одному, где X, Y — конечные множества. Тогда:

$$|X| = k |Y|$$

Задача. Сколько существует рассадок 4 рыцарей вокруг стола? Две рассадки эквивалентны, если одну можно получить из другой поворотом.

Решение. Пусть $A = \{x_1, x_2, x_3, x_4\}$ — множество рыцарей, X — множество 4-строк вида $x_j \dots x_k$, $1 \leq j, k \leq 4$, $i \neq j$.

Пусть $f: X \rightarrow Y$ — соответствие, где Y — множество всех

возможных расщадок для A . Очевидно, что:

$$f \text{ — } n\text{-к-одному} \Rightarrow 4|Y| = |X| \Leftrightarrow |Y| = |X|/4$$

По правилу умножения:

$$|X| = 4 \cdot 3 \cdot 2 \cdot 1 = P_4 = 24 \Rightarrow |Y| = 24/4 = 6$$

Ответ: $|Y| = 6$.

Разбиение множеств

Разбиение множества A — его представление в виде k непересекающихся непустых подмножеств:

$$A = \bigcup_{i=1}^k B_i \quad \begin{cases} B_i \neq \emptyset \\ B_i \cap B_j = \emptyset, i \neq j \end{cases}$$

Число Белла — количество разбиений n -множества.

Число Стирлинга второго порядка — количество разбиений n -множества на k подмножеств:

$$C(n, k) \equiv \begin{Bmatrix} n \\ k \end{Bmatrix} = k \begin{Bmatrix} n-1 \\ k \end{Bmatrix} + \begin{Bmatrix} n-1 \\ k-1 \end{Bmatrix}$$

Частные случаи:

$$\begin{Bmatrix} n \\ 0 \end{Bmatrix} = 0 \quad \begin{Bmatrix} n \\ 1 \end{Bmatrix} = 1 \quad \begin{Bmatrix} n \\ n \end{Bmatrix} = 1 \quad \begin{Bmatrix} n \\ n-1 \end{Bmatrix} = \begin{pmatrix} n \\ 2 \end{pmatrix}$$

Доказательство. Да

Материал к ЕГЭ

16 задание

Величина	Название
S	первоначальная сумма долга
B_i	размер долга на конец i -го периода
X_i, x	размер платежа в i -ый период
n	число платёжных периодов
r	учётная ставка (<i>в %</i>)
$p = 1 + 0.01r$	повышающий коэффициент

Аннуитетный платёж — долг выплачивается *равными платежами*.

Уравнение аннуитетного платежа:

$$p^n S = x \left(\frac{p^n - 1}{p - 1} \right)$$

Дифференцированный платёж — долг *уменьшается равномерно*, при этом платежи в каждый период *разные*.

Рекуррентные формулы дифференцированного платежа:

$$\begin{cases} B_i = pB_{i-1} - X_i \\ B_i = \frac{n-i}{n}S \end{cases}$$

X_i образует *арифметическую прогрессию*.