Аналитические и численные алгоритмы

Различие между аналитическими и численными алгоритмами

- **Аналитика** – четкая постановка задачи, вычисление точного решения
- **Численное решение** – приближенное решение, возможно предварительное предположение о характере и самом решении с последующей проверкой

Рассмотрим эти различия на примере классических задач,

связанных с **полиномами**

Полиномы – аналитические методы

$$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 = 0$$

$n = 2$: $x_{1,2} = \dfrac{-a_1 \pm \sqrt{a_1^2 - 4a_2 a_0}}{2a_2}$

$n = 3$: - формула Кардано

$n = 4$: - метод Феррари

Формула Кардано (1545)

$$y^3 + py + q = 0$$

$$ax^3 + bx^2 + cx + d = 0 \qquad x = y - \frac{b}{3a}$$

$$p = \frac{c}{a} - \frac{b^2}{3a^2} = \frac{3ac - b^2}{3a^2},$$

$$q = \frac{2b^3}{27a^3} - \frac{bc}{3a^2} + \frac{d}{a} = \frac{2b^3 - 9abc + 27a^2 d}{27a^3}$$

Формула Кардано

$$y^3 + py + q = 0$$

$$Q = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2$$

- Q >0 - один вещественный корень и два сопряжённых комплексных корня
- Q = 0 — один однократный вещественный корень и один двукратный, или, если p = q = 0, то один трёхкратный вещественный корень
- Q < 0 — три вещественных корня

Формула Кардано

$$y^3 + py + q = 0$$

$$Q = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2$$

$$\alpha\beta = -p/3$$

$$y_1 = \alpha + \beta,$$

$$y_{2,3} = -\frac{\alpha + \beta}{2} \pm i\frac{\alpha - \beta}{2}\sqrt{3},$$

где

$$\alpha = \sqrt[3]{-\frac{q}{2} + \sqrt{Q}},$$

$$\beta = \sqrt[3]{-\frac{q}{2} - \sqrt{Q}},$$

Метод Феррари

$$x^4 + ax^3 + bx^2 + cx + d = 0.$$

$$y^3 - by^2 + (ac - 4d)y - a^2d + 4bd - c^2 = 0$$

Пусть y_1 – любой корень

$$x^2 + \frac{a}{2}x + \frac{y_1}{2} = \pm\sqrt{\left(\frac{a^2}{4} - b + y_1\right)x^2 + \left(\frac{a}{2}y_1 - c\right)x + \frac{y_1^2}{4} - d},$$

Деление полиномов

$$x^9 + x^8 - 6x^7 + 2x^6 - 4x^5 + 3x^3 - x^2 - 9x - 2 = 0$$

Ряд и теорема Штурма

$$f_0(x), f_1(x), \ldots, f_s(x)$$

- $f_0(x) = f(x)$;
- $f_1(x) = f_0'(x)$;

$$f_{k+1}(x) = -f_{k-1}(x) \bmod f_k(x)$$

$$3x^4 + 2x^3 + 3x^2 + 5x + 1 = 0$$

Ряд и теорема Штурма (пример)

$3x^4 + 2x^3 + 3x^2 + 5x + 1$

$12x^3 + 6x^2 + 6x + 5$

$-\frac{5}{4}x^2 - \frac{7}{2}x - \frac{19}{24}$

$-\frac{1892}{25}x - \frac{562}{25}$

$-\frac{1478575}{20738992}$

- ∞	∞	0
+	+	+
-	+	+
-	-	-
+	-	-
-	-	-
3	1	1

Wolfram Mathematica

https://www.wolfram.com/language/fast-introduction-for-math-students/en

Библиотеки символьных вычислений

- SymPy - библиотека Python с открытым исходным кодом, используемая для символьных вычислений (компьютерная алгебра) https://www.sympy.org/en/index.html
- Библиотека GiNaC, линейная алгебра, дифференцирование https://www.ginac.de/
- Библиотека ViennaMath http://viennamath.sourceforge.net/viennamath-about.html

Численные алгоритмы

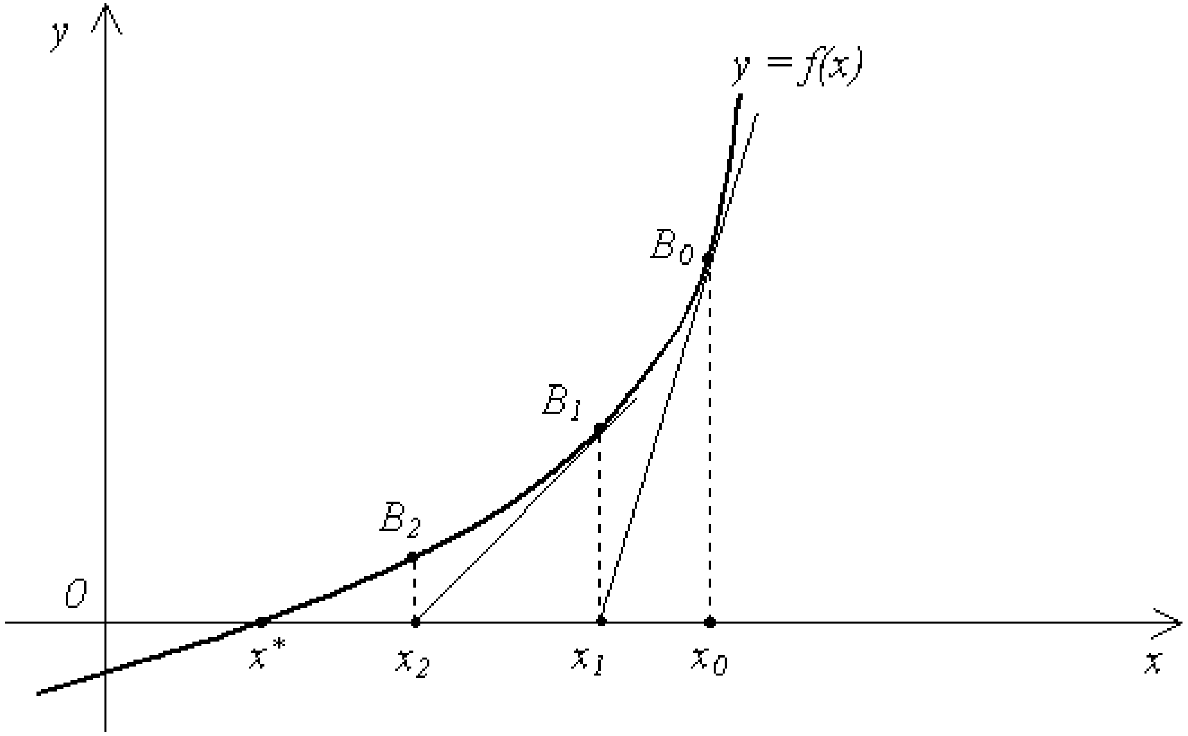
аналоги

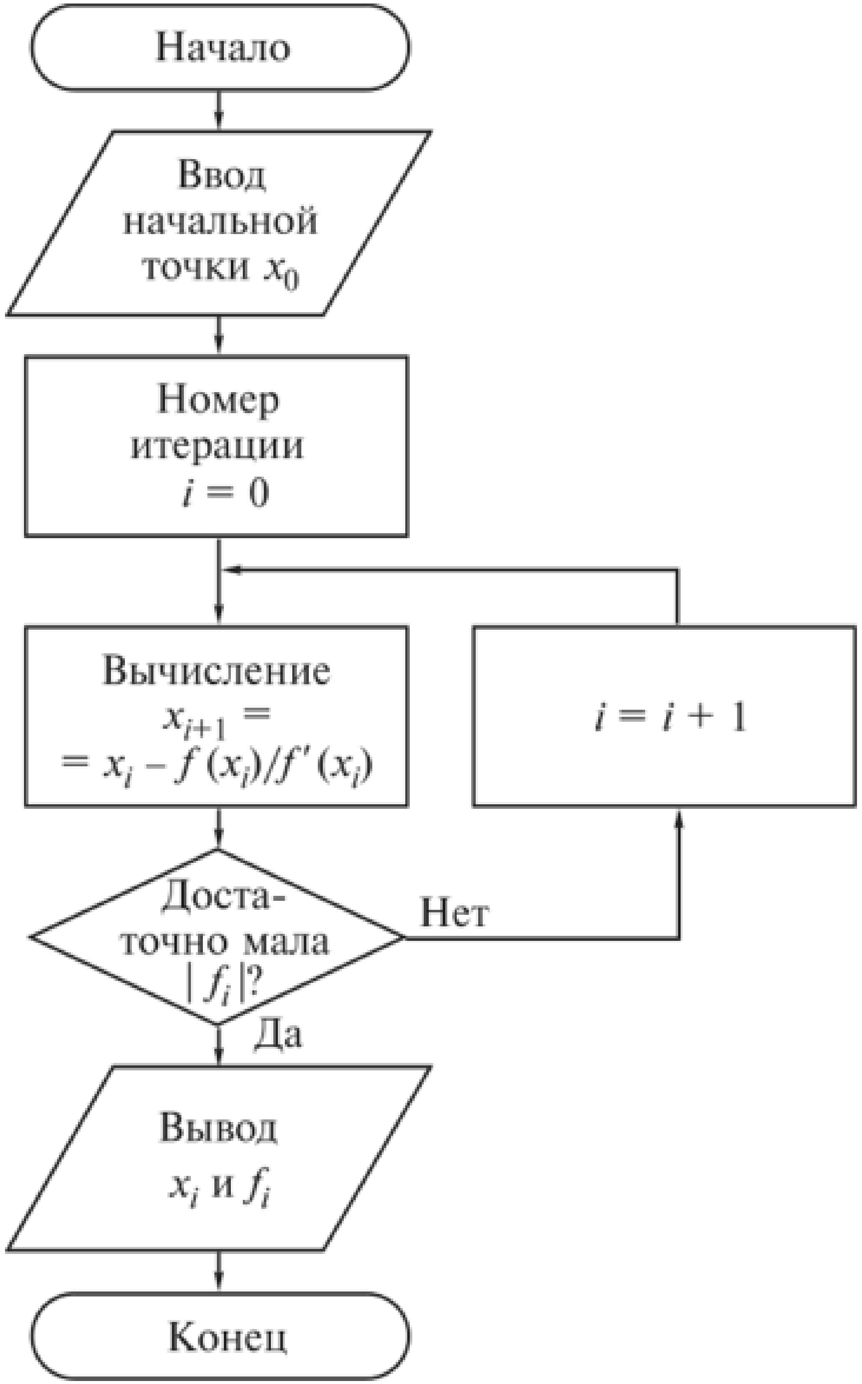
http://vmath.ru/vf5/polynomial/newton

http://vmath.ru/vf5/polynomial/zero_local

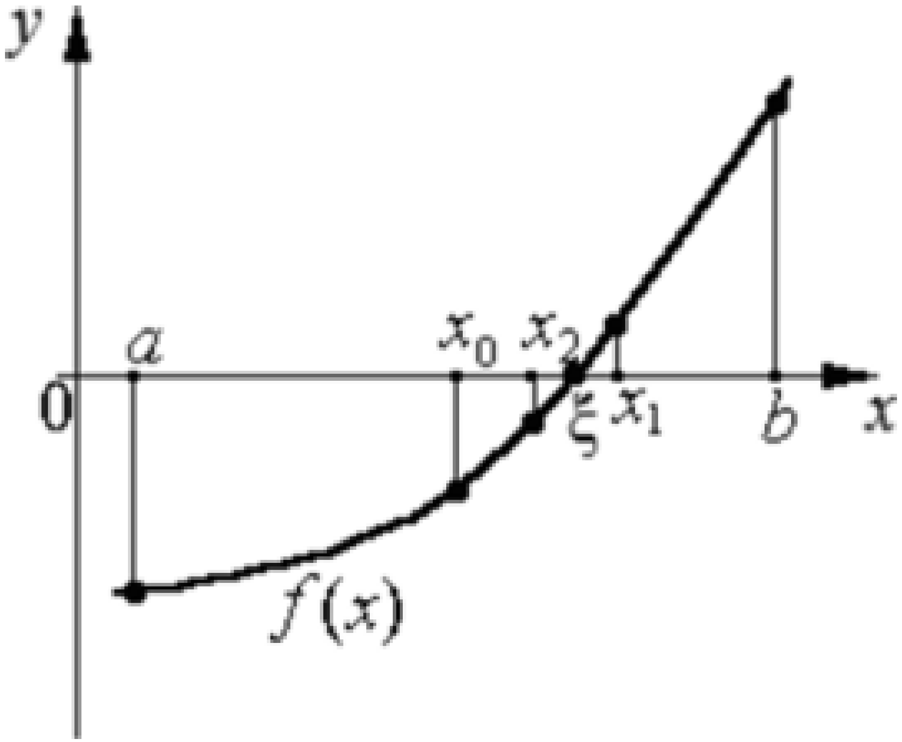
Алгоритм метода Ньютона

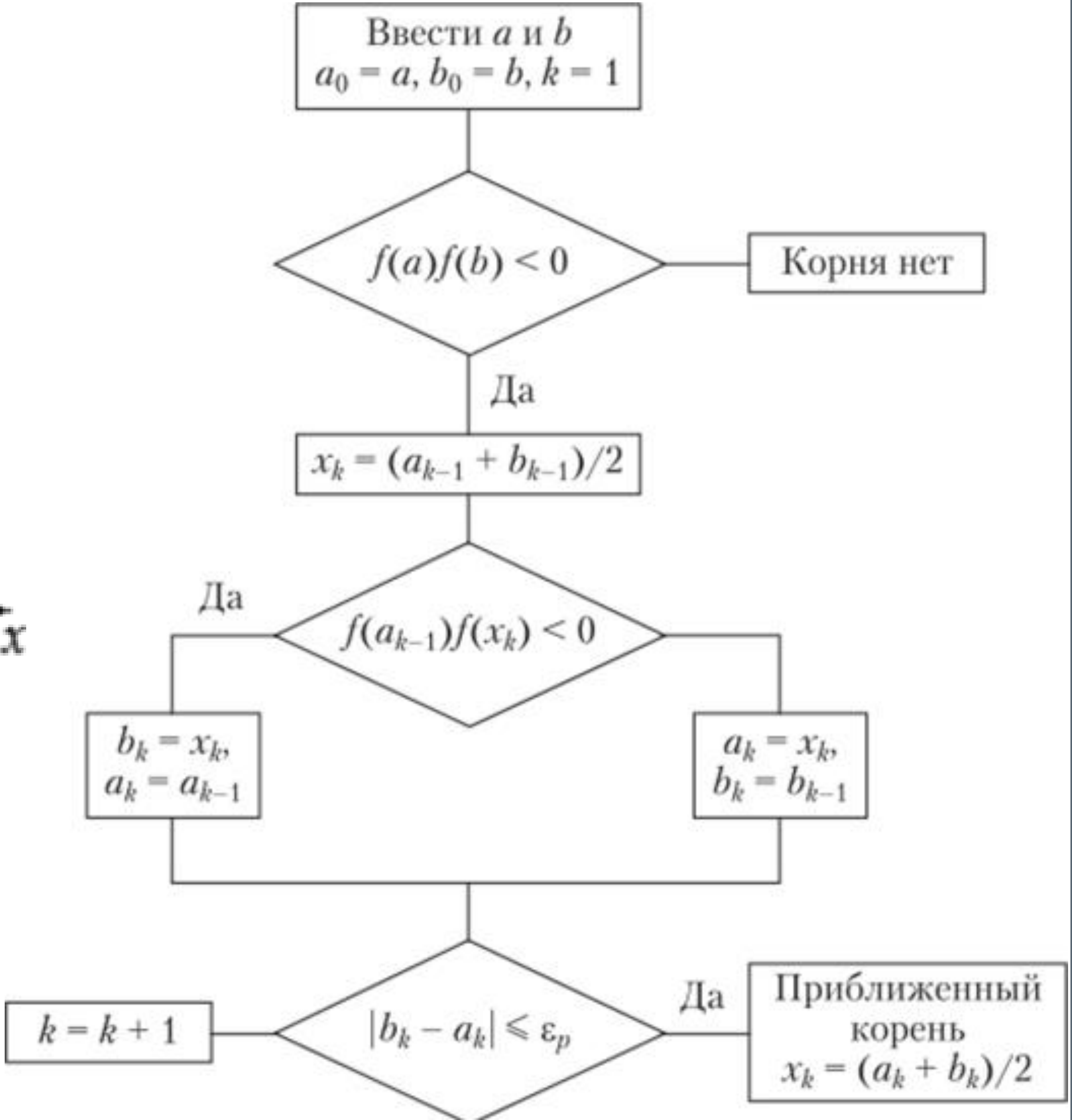
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \ldots .$$





Алгоритм метода деления пополам





Алгоритм Евклида – НОД (A,B)

- Если A = 0, тогда НОД(A,B) = B, поскольку НОД(0,B) = B, и алгоритм останавливается.
- Если B = 0, тогда НОД(A,B) = A, поскольку GCD(A,0) = A, и алгоритм останавливается.
- Делим A на B с остатком (A = B·Q + R)
- Находим НОД(B,R) при помощи алгоритма Евклида, поскольку НОД(A,B) = НОД(B,R)

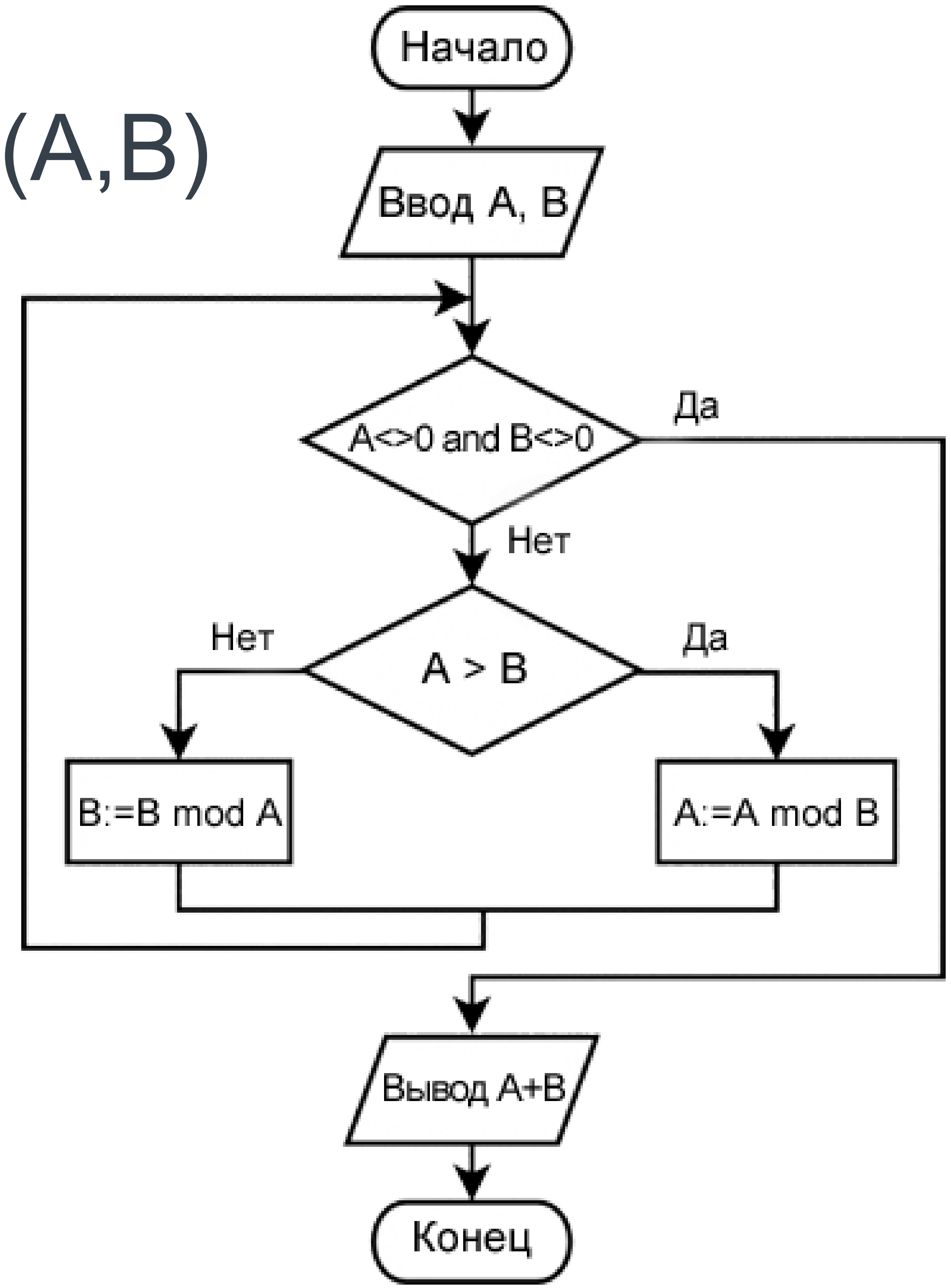


Схема Горнера

$$P_n(x)=\sum_{i=0}^{n} a_i x^{n-i} = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-3} + \ldots + a_{n-1}x + a_n$$

	a_0	a_1	a_2	a_3	…	a_{n-1}	a_n
a							

Схема Горнера

$5x^4 + 5x^3 + x^2 - 11$ на $(x-1)$

$$5\cdot x^4 + 5\cdot x^3 + 1\cdot x^2 + 0\cdot x - 11$$

	5	5	1	0	−11
1					

	5	5	1	0	−11
1	5				

	5	5	1	0	−11
1	5	10			

	5	5	1	0	−11
1	5	10	11		

	5	5	1	0	−11
1	5	10	11	11	

	5	5	1	0	−11
1	5	10	11	11	0

	5	5	1	0	−11
1	5	10	11	11	0

$$5x^4 + 5x^3 + x^2 - 11 = (x-1)(5x^3 + 10x^2 + 11x + 11) + 0$$

π Алгоритм Карацубы $O(n^2) \longrightarrow O(n^{\log_2 3})$

$$A(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n$$
$$= a_0 + a_1 \cdot 10 + a_2 \cdot 10^2 + \cdots + a_n \cdot 10^n$$

$$a(x) = a_1(x) + x^k a_2(x) \qquad p_1(x) = a_1(x) \cdot b_1(x)$$
$$b(x) = b_1(x) + x^k b_2(x) \qquad p_2(x) = a_2(x) \cdot b_2(x)$$

$$t(x) = (a_1(x) + a_2(x)) \cdot (b_1(x) + b_2(x))$$

Алгоритм Карацубы

$$c(x) = a(x) \cdot b(x) = p_1(x) + x^k \cdot (t(x) - p_1(x) - p_2(x)) + x^{2k} \cdot p_2(x)$$

$$\Theta(n^{\log_2 3}) \approx \Theta(n^{1.58})$$

Похожий метод можно применить к матричному умножению

Алгоритм Штрассена

$$O(n^{\log_2 7}) \approx O(n^{2.81})$$