Алгоритм нахождения корня n-ной степени

Арифметическим корнем n-ной степени n√A положительного действительного числа A называется положительное действительное решение уравнения

x^{n}=A

(для целого n существует n комплексных решений данного уравнения, если A > 0, но только одно является положительным действительным).

Существует быстросходящийся алгоритм нахождения корня n-ной степени:

Сделать начальное предположение x\_{0};

Задать {\displaystyle x\_{k+1}={\frac {1}{n}}\left({(n-1)x\_{k}+{\frac {A}{x\_{k}^{n-1}}}}\right);}

Повторять шаг 2, пока не будет достигнута необходимая точность.

Частным случаем является итерационная формула Герона для нахождения квадратного корня, которая получается подстановкой n = 2 в шаг 2:

x\_{{k+1}}={\frac {1}{2}}\left(x\_{k}+{\frac {A}{x\_{k}}}\right).

Существует несколько выводов данного алгоритма. Одно из них рассматривает алгоритм как частный случай метода Ньютона (также известного как метод касательных) для нахождения нулей функции f(x) с заданием начального предположения. Хотя метод Ньютона является итерационным, он сходится очень быстро. Метод имеет квадратичную скорость сходимости — это означает, что число верных разрядов в ответе удваивается с каждой итерацией (то есть увеличение точности нахождения ответа с 1 до 64 разрядов требует всего лишь 6 итераций). По этой причине данный алгоритм используют в компьютерах как очень быстрый метод нахождения квадратных корней.

Для больших значений n данный алгоритм становится менее эффективным, так как требуется вычисление x\_{k}^{{n-1}} на каждом шаге, которое, тем не менее, может быть выполнено с помощью алгоритма быстрого возведения в степень.

import java.util.Scanner;

public class Number14 {

public void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.println("a=");

double a = input.nextDouble();

System.out.println("n=");

double n = input.nextDouble();

System.out.println("~x0=");

double x0 = input.nextDouble();

System.out.println("e=");

double e = input.nextDouble();

while((x0-x0%1)>e){

x0=1/n \* ((n-1)\*x0 + a/(Math.pow(x0, n-1)));

}

System.out.println(x0);

}

}