方程求根的迭代法

- 二分法求根
- 迭代法
- 牛顿法求根

假设我们这里需要求一个方程组$x^3 - x - 1 = 0$.

二分法

若$f(x)$在区间$[a, b]$内单调，且$f(a)f(b) < 0$,则$f(x) = 0$必定在该区间有唯一实数根。取中间值$x_0 = \frac{a+b}{2}$.

计算流程:

- 若$f(x_0) = 0$ 或$|f(x_0)| < \epsilon_f$，则直接返回x_0,x_0就是该方程的一个根。
- 若$f(a)f(x_0) < 0$,则在区间$[a, x_0]$内继续搜索;
- 若$f(x_0)f(b) < 0$,则在区间$[x_0, b]$内继续搜索;

设定误差ϵ_x,第k次迭代的区间为$[a_k, b_k]$,由于区间长度每次都会减半，故可得最大迭代次数:

$$k_{max} = \frac{\ln(b-a) - \ln 2\epsilon}{\ln 2}$$

> 在这里$f(x) = x^3 - x - 1$

```
public class BinaryIteration {
    private double a;
    private double b;
    private Function function;
    private double limitError;

    public BinaryIteration(double a,double b,Function function,double limi
        this.a = a;
        this.b = b;
        this.function = function;
        this.limitError = limitError;
    }

    public double calculate(){
        double mid;
        double start = a;
```

```
        double end = b;
        do {
            mid = (start + end) / 2;
            if(function.calculate(start) * function.calculate(mid) > 0){
                start = mid;
            }else{
                end = mid;
            }
        }while (Math.abs(start-end) > limitError);
        return start;
    }

    public int getMaxIteration(){
        return (int)Math.floor((Math.log(b-a) - Math.log(2*limitError)) / M
    }
}
```

迭代法

对于一个需要求解的方程$f(x) = 0$,将其变形为$x = g(x)$的形式。然后从一个初值x_0开始，反复使用迭代公式$x_{n+1} = g(x_n)$,直到满足精度ϵ要求，即$|x_{n+1} - x_n| < \epsilon$

> 这里我们将原方程组进行变化得到：$x = \sqrt[3]{x+1}$,因此这里的$g(x) = \sqrt[3]{x+1}$

```
public class Iteration {
    private double a;
    private double b;
    private Function function;
    private double limitError;

    public Iteration(double a, double b, Function function, double limitEr
        this.a = a;
        this.b = b;
        this.function = function;
        this.limitError = limitError;
    }

    public double calculate(){
        double start;
        double end = a;
        do {
            start = end;
            end = function.calculate(start);
        }while (Math.abs(start-end) > limitError);
        return end;
    }
}
```

牛顿迭代法

已知方程近似根x_0,则在x_0附近$f(x)$可用一阶泰勒多项式
$p(x) = f(x_0) + f'(x_0)(x - x_0)$近似代替。因此方程$f(x) = 0$可近似为$p(x) = 0$。

设$f'(x_0) \neq 0$,则有:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

重复这一过程，可得到迭代公式:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

> 我们可以设$h(x) = x - \frac{f(x)}{f'(x)}$，则$h(x) = x - \frac{x^3-x-1}{3x^2-1}$

```
public class NewTonIteration {
    private double a;
    private double b;
    private Function function;
    private double limitError;

    public NewTonIteration(double a, double b, Function function, double l
        this.a = a;
        this.b = b;
        this.function = function;
        this.limitError = limitError;
    }

    public double calculate(){
        double start,end;
        start = end = a;
        do {
            start = end;
            end = function.calculate(start);
        }while (Math.abs(start-end) > limitError);
        return end;

    }
}
```