МІНІСТЕРСТВО ОСВІТИ І НАУКИ В УКРАЇНІ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
ІПСА
Кафедра Системного проектування

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА

по дисципліні:
“Чисельні методи ”
на тему:

“ Дослідження збіжності ітеративних методів розв’язку нелінійних рівнянь ”

виконав:
студент ІІ курсу групи ДА-72
Кондратюк Т.Є.

Київ 2018

Зміст

Вступ

Ітераційний метод – чисельний метод вирішення математичних задач, наближений метод вирішення нелінійних алгебраїчних рівнянь. Суть полягає в знаходженні за наближеним значенням величину наступного наближення (котре вже є більш точним). Метод дозволяє отримати значення коренів рівняння з заданою точністю у вигляді границі послідовності деяких векторів (ітераційний процес). Характер збіжності і сам її факт залежить від вибору початкового наближеного кореня x0.

Для отримання розв’язку ріняння з необхідною точністю постановка задачі і умов вирішення повинна бути коректною, а метод, що використовується для отримання величин невідомої, повинен бути стійким і збіжним.
Якісним показником збіжності є сам факт отримання рішення при заданій точності, а кількісним показником буде число ітерацій К, за яке рішення потрапляє в окіл точного рішення, тобто швидкість вирішення, що вимірюється в циклах ітераційного алгоритму. Також слід врахувати дотримання достатньої умови збіжності ітераційного методу відносно похибки округлення.

Постановка задачі

Дана деяка функція f(x) і необхідно знайти всі або деякі значення x, для яких f(x) = 0;

Значення x^*, при котрому f(x^*) = 0, називається коренем (або рішенням) рівняння.

Відносно функції f(x) часто мається на увазі, що f(x) двічі неперервно диференційовна в околі кореня.

В процесі наближеного пошуку коренів рівняння зазвичай виділяють два етапи: локалізація (або відділення) кореня і його уточнення.

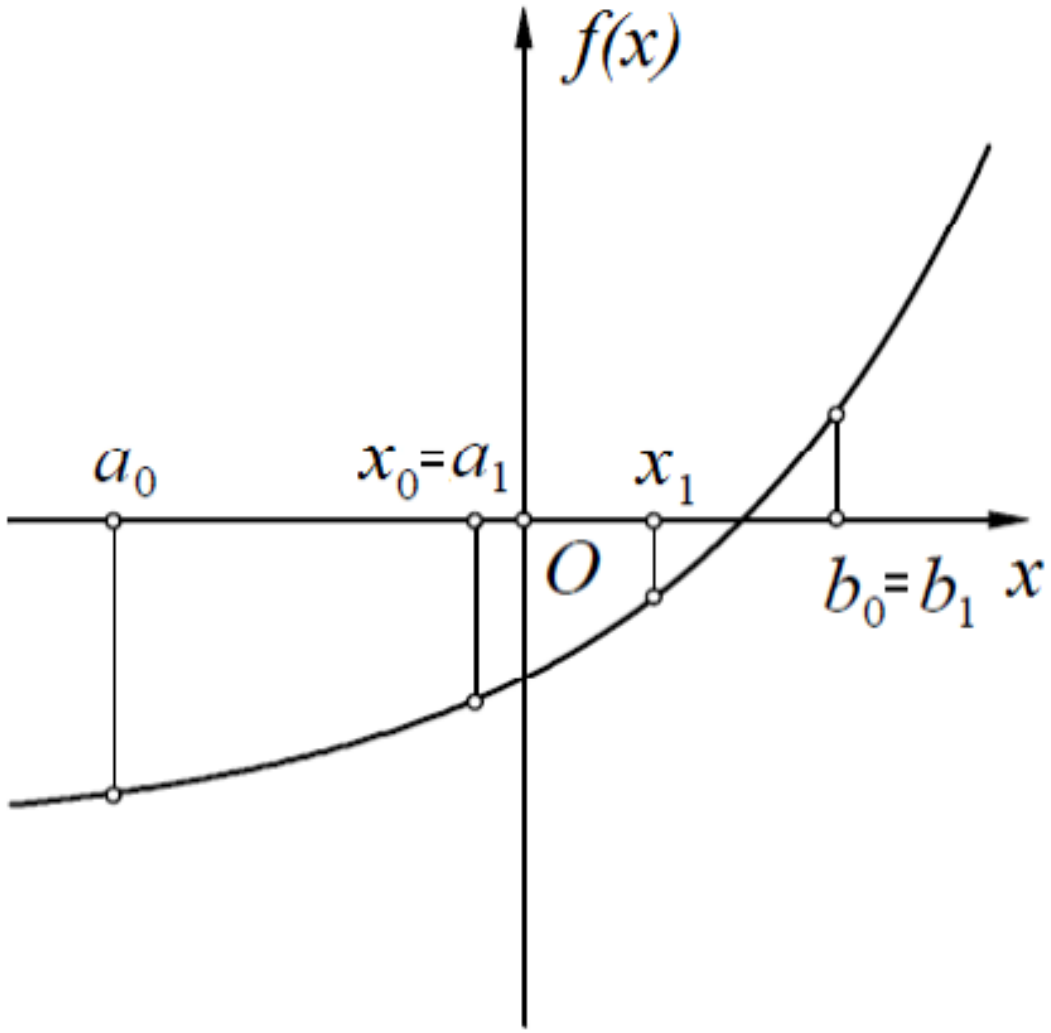
Локалізація полягає у визначенні відрізка [a,b], на якому знаходиться один і лише один корінь.

На етапі уточнення кореня вираховують наближене значення кореня з заданою точністю. Наближене значення уточнюють за допомогою ітераційних методів

Теоретичні відомості

Метод половинного ділення

Нехай із попереднього аналізу відомо, що корінь рівняння знаходиться на відрізку $[a_0, b_0]$, тобто $x^*[a_0, b_0]$, так, що f(x^*)=0.
Нехай функція f(x) неперервна на відрізку $[a_0, b_0]$ і приймає на кінцях відрізка значення різних знаків, тобто f(a_0)f(b_0) < 0.
Розділимо відрізок $[a_0, b_0]$ на дві половини. Отримаємо току $x_0=(a_0+b_0)/2$
Обчислимо значення функції в цій точці : f(x0) Якщо f(x0) = 0, то x_0 – шуканий корінь і задача вирішена.
В іншому випадку знаходимо знаки f(x) на кінцях відрізків $[a_0, x_0]$ і $[x_0, b_0]$.
Той із них, на кінцях якого f(x) має значення різних знаків приймають за новий відрізок $[a_1, b_1]$, і вираховують наступне наближення x1=$(a_1+b_1)/2$.



Похибка методу

Після кожної ітерації відрізок, на якому розташований корінь, зменшується вдвічі, а після n ітерацій в 2^n разів:
b_n - a_n = (b_0 - a_0) / 2^n
Оскільки корінь належить відрізку [a_n, b_n], а x_n – середина цього відрізка, то величина | x^*- x_n | завжди буде менше половини довжини цього відрізка:
|x^*- x_n | < (b_n - a_n) / 2, отже | x^* - x_n | < (b_0 - a_0) / 2^n.

Критерій завершення

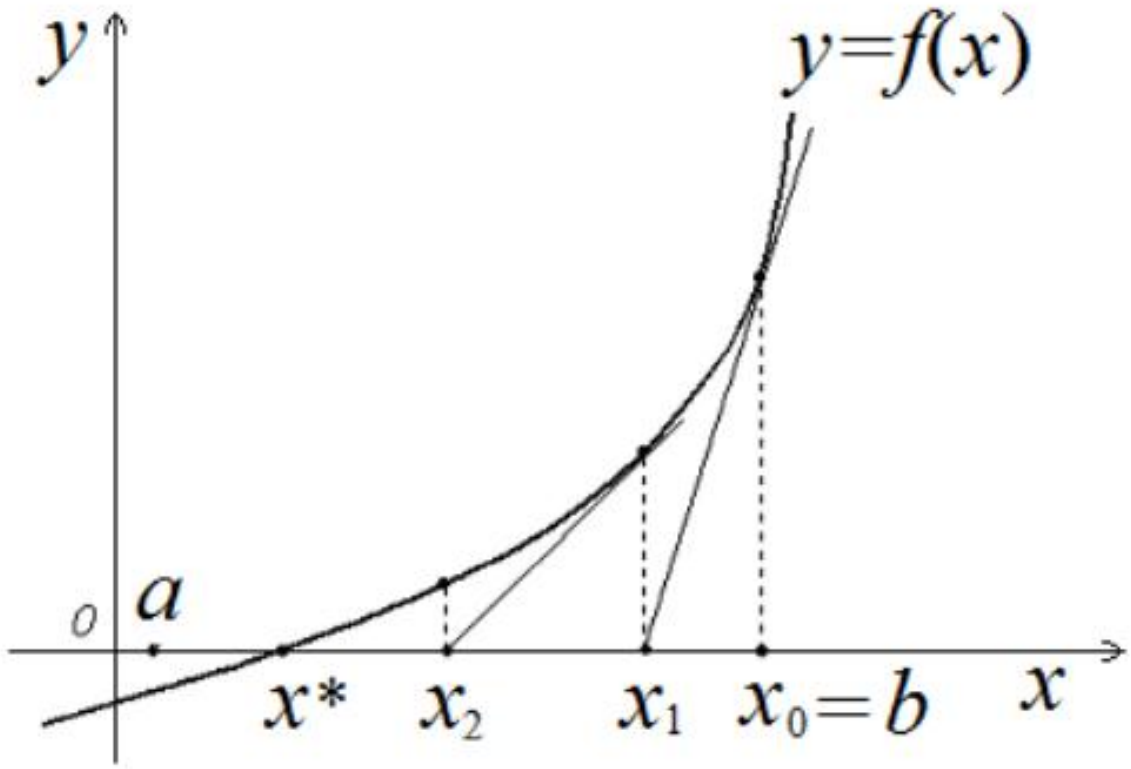
При заданій точності наближення ε обчислення
закінчуються, коли буде виконано нерівність
b_n - a_n < 2ε або нерівність n > log2 ((b_0 - a_0) / ε) - 1.
Таким чином, кількість ітерацій можна визначити заздалегідь.
За наближене значення кореня береться величина x_n.

Збіжність методу

На відміну від більшості інших методів уточнення, метод
половинного ділення сходиться завжди, тобто володіє безумовною збіжністю.
З кожним кроком похибка наближеного значення зменшується в два рази, тобто
| x^*- x_n | < | x^* - x_n-1 | / 2.
Тому даний метод є методом з лінійною збіжністю

Метод Ньютона (метод дотичних)

Нехай корінь x^* ∈ [a, b], так, що f (a) f (b) <0. Припускаємо, що
функція f (x) неперервна й двічі неперервно диференційовна на відрізку [a, b]. А її похідні f '(x) і f' '(x) зберігають свій знак на [a, b]. Приймемо за x_0 той кінець відрізка, в якому f (x) має той же знак що і f " (x). Рівняння дотичної до f (x) в точці (x_0, f (x_0)) буде мати вигляд: y = f (x_0) + f '(x_0) (x -x_0). Перший перетин отримаємо, взявши абсцису точки перетину цієї дотичної з віссю OX: x_1 = x_0 - f (x_0) / f '(x_0). Аналогічно вчинимо з точкою (x1, f (x1)), потім з точкою (x2, f (x_2)), і так далі. В результаті одержимо послідовність наближень:
x_n + 1 = x_n - f (x_n) / f '(x_n).



Похибка методу

Для метода Ньютона справедлива наступна оцінка похибки :

$$|x_n - x^*| \leq \frac{M_2}{2m_1}|x_n - x_{n-1}|^2$$

Де $M_2 = \max\limits_{a \leq x \leq b}|f''(x)|, m_1 = \min\limits_{a \leq x \leq b}|f'(x)|$

Критерій завершення

При заданій точності ε > 0 обчислення потрібно вести до тих пір, поки не буде виконано нерівність:

$\frac{M_2}{2m_1}|x_n - x_{n-1}|^2 < \varepsilon$ або $|x_n - x_{n-1}| < \sqrt{\frac{2m_1\varepsilon}{M_2}}$

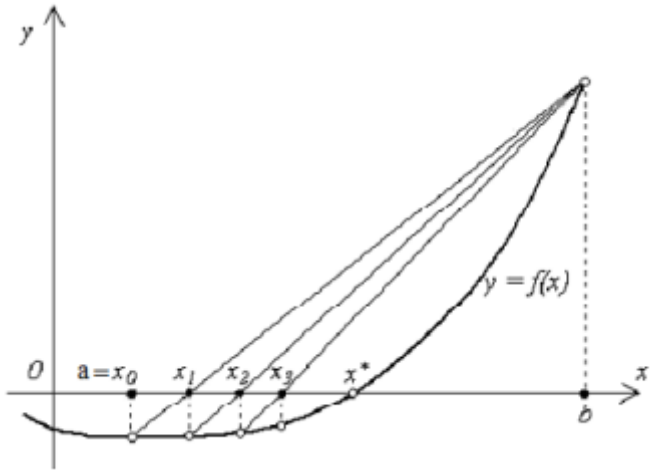
Можна використовувати спрощену умову: $|x_n - x_{n-1}| < \varepsilon$

Збіжність методу

Збіжність методу Ньютона залежить від вибору початкового наближення. Якщо в якості x0 вибрати той з кінців відрізка, для якого f (x) f '' (x) ≥ 0 то ітерації сходяться.

Метод хорд

Метод хорд є ще однією модифікацією методу Ньютона. Нехай відомо, що корінь x * рівняння f (x) = 0 знаходиться на відрізку [a, b] і виконується умова f (b) f "(b) ≥ 0, тоді x0 = a. Будемо проводити з точки (b, f (b)) прямі через розташовані на графіку функції точки з координатами (x_n, f (x_n).
Абсциса точки перетину такої прямої з віссю OX є черговим наближенням x_{n+1}



Прямі на цьому рисунку заміняють дотичні в методі Ньютона. Ця заміна базується на наближеній рівності

$$f'(x_n) \approx \frac{f(b) - f(x_n)}{b - x_n}$$

Замінимо в розрахунковій формулі Ньютона похідну $f'(x_n)$ правою частиною наближеної нерівності. В результаті отримаємо розрахункову формулу методу хорд:

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

Похибка методу

Похибка методу оцінюється відношенням:

$$\left|x_n - x^*\right| \le \frac{M_1 - m_1}{m_1} \left|x_n - x_{n-1}\right|,$$

де $$M_1 = \max_{a \le x \le b} \left|f'(x)\right|, m_1 = \min_{a \le x \le b} \left|f'(x)\right|$$

Збіжність і критерій завершення

Метод хорд має лінійну збіжність
Критерій завершення ітерацій методу хорд:

$$|x_n - x_{n-1}| < \varepsilon$$

Метод простих ітерацій

Для застосування цього методу вихідне нелінійне рівняння f (x) = 0 замінюють еквівалентним: x = φ (x)
Нехай на відрізку [a, b] розташований єдиний корінь. Приймемо за x0 будь-яке значення з інтервалу [a, b]. Обчислимо значення функції φ (x) при x = x_0 і знайдемо уточнене значення x1 = φ (x_0). Продовжуючи цей процес необмежено, отримаємо послідовність наближень до кореня: x_{n+1}= φ (x_n)

Похибка методу

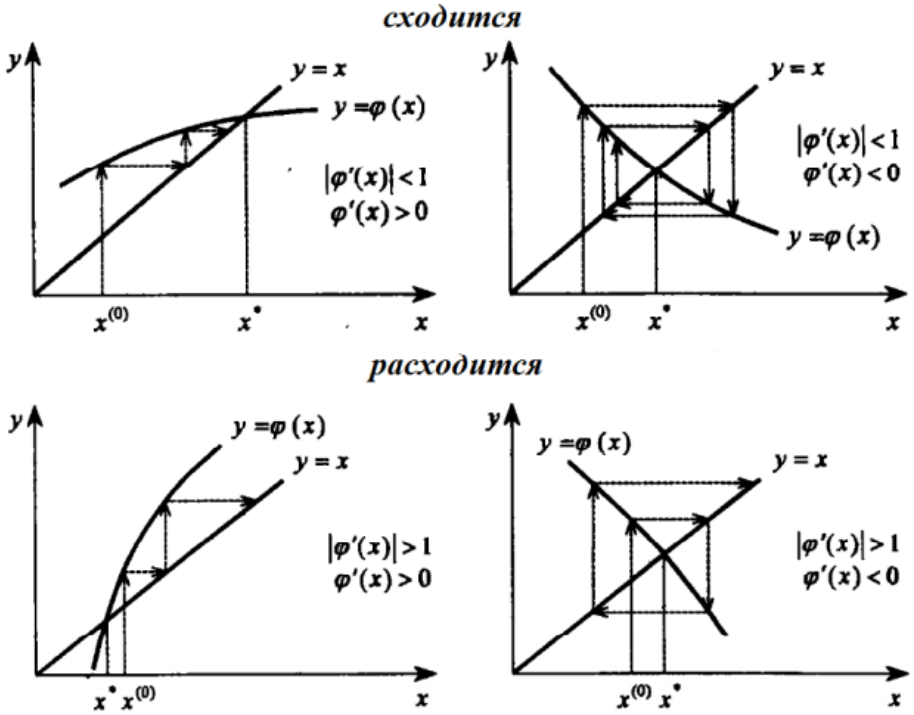
$$\left|x_n - x^*\right| \le \frac{q}{1-q}\left|x_n - x_{n-1}\right| \qquad q = \max_{a \le x \le b}\left|\varphi'(x)\right|$$

Збіжність методу

Якщо функція φ (x) визначена і неперервна на інтервалі [a, b] і |φ '(x)| < 1, x ∈ [a,b] то процес ітерацій сходиться з будь-якою точністю при будь-якому початковому значенні x0 з інтервалу [a, b].

Геометрична ілюстрація методу

Коренем вихідного нелінійного рівняння є абсциса точки перетину лінії y = φ (x) з прямою y = x. З графіків можна побачити, що в методі простих ітерацій можливі як збіжні, так і розбіжні ітераційні процеси. Швидкість збіжності залежить від абсолютної величини φ '(x). Тому вибір способу зведення вихідного рівняння до виду x = φ (x) є важливим



Критерій завершення

При заданій точності ε > 0 обчислення потрібно вести до тих пір, поки не буде виконано нерівність: $|x_n - x_{n-1}| < \frac{1-q}{q}\varepsilon$

Якщо q ≤ 0.5 можна використовувати спрощене умова: $|x_n - x_{n-1}| < \varepsilon$

Якщо функція f(x) неперервна разім зі своєю першою похідною на відрізку [a,b] і 0 < m < f'(x) < M на [a,b], то зведення рівняння f(x) = 0 до вигляду x = φ(x) виконують наступним чином:

λf(x) = 0

x = x + λf(x)

x = φ(x), де φ(x) = x + λf(x)

φ'(x) = 1 + λf'(x)

Якщо в якості константи λ взяти $\lambda = -\frac{1}{M}$, то

$$\varphi'(x) = 1 + \lambda f'(x) = 1 - \frac{1}{M} f'(x) < 1 - \frac{m}{M} < 1,$$

$$\varphi'(x) = 1 + \lambda f'(x) = 1 - \frac{1}{M} f'(x) > 1 - \frac{M}{M} = 0$$

Тобто $0 < \varphi'(x) < k = 1 - \frac{m}{M} < 1$

Реалізація на java

Метод половинного ділення

```
import static java.lang.Math.abs;
public class Test {

   static double f(double x) {
      return Math.cos(x) - Math.sqrt(x);
   }

   public static void main(String[] args) {

      double a, b, x = 0, c, eps = 0.00001f;
      int iter = 0;
      System.out.println("Знайти корінь рівняння cos(x)-sqrt(x)=0 на інтервалі [0;10] при eps=0.00001");
      System.out.println();
      long startTime = System.currentTimeMillis();
      a = 0;
      b = 10;

      while (abs(a - b) > eps) {
         iter++;
         c = (a + b) / 2;
         if (f(a) * f(c) <= 0) b = c;
         else {
            a = c;
            x = (a + b) / 2;
         }
      }
      long timeSpent = System.currentTimeMillis() - startTime;
      System.out.println("x = " + x + " f(x) = " + f(x));
      System.out.println("Кількість ітерацій : "+ iter);
      System.out.println("Час роботи : " + timeSpent + " мс");
   }
}
```

Вивід :

```
Знайти корінь рівняння cos(x)-sqrt(x)=0 на інтервалі [0;10] при eps=0.00001


x = 0.6417465209960938 f(x) = -3.931121899514167E-5

Кількість ітерацій : 20

Час роботи : 0 мс
```

Метод Ньютона

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

 class Test {

    public static double f(double x) {
        return Math.cos(x) - Math.sqrt(x);
    }
    public static double df(double x) {
        return (-1)/(2 * Math.sqrt(x)) - Math.sin(x);
    }
    public static double g(double x) {
        return x - f(x) / df(x);
    }

    public static void main (String[] args) throws Exception {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        double x;
        double eps;
        System.out.println("Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001");
        System.out.print("Введіть приблизне значення кореня : ");
        x = Double.parseDouble(reader.readLine());
        eps = 0.00001;
        double iter;
        long startTime = System.currentTimeMillis();
        for(iter = 1; eps < Math.abs(f(x)); iter = iter + 1) {
            if(df(x) == 0)
            {
                System.out.println("Ділення на 0");
                break;
            }
            x = g(x);
        }
        long timeSpent = System.currentTimeMillis() - startTime;
        System.out.printf("Кількість ітерацій : %.0f%n", iter);
        System.out.println("x    = " + x);
        System.out.println("g(x) = " + g(x));
        System.out.println("df(x)= " + df(x));
        System.out.println("f(x) = " + f(x));
        System.out.println("Час роботи : " + timeSpent + " мс");
    }
}
```

Вивід :

```
Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001

Введіть приблизне значення кореня : 1

Кількість ітерацій : 4

x    = 0.6417143710025017

g(x) = 0.6417143708728826

df(x)= -1.2227342355853725

f(x) = -1.584896658357593E-10

Час роботи : 0 мс
```

```
Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001
Введіть приблизне значення кореня : 0
Нескінченний цикл
Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001
Введіть приблизне значення кореня : 50
Нескінченний цикл
```

Метод хорд

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Test {

   public static double f(double x) {
      return Math.cos(x) - Math.sqrt(x);
   }

   public static void main(String[] args)throws Exception {
      BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
      double  x,eps,a,b,c;
      int n;
      System.out.println("Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001");
      System.out.println("Введіть значення a");
      a = Double.parseDouble(reader.readLine());
      System.out.println("Введіть значення b");
      b = Double.parseDouble(reader.readLine());
      eps = 0.00001;

      n = 0;
      do {
         c = (f(b) * a - f(a) * b) / (f(b) - f(a));
         if (f(a) * f(c) > 0)
            a = c;
         else b = c;
         n++;
      }
      while(Math.abs((f(b) * a - f(a) * b) / (f(b) - f(a)) - c) < eps);

      x = c;
      System.out.printf("Корінь x =%10.7f%n", x);
      System.out.println("Кількість ітерацій = " + n);
   }
}
```

Вивід :

```
Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001
Введіть значення a
0
Введіть значення b
1
Корінь x = 0,6850734
Кількість ітерацій = 1
```

Метод простих ітерацій

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class Test {

   public static double f(double x) {
      return Math.cos(x)*Math.cos(x);
   }

   public static void main(String[] args)throws Exception {
      BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
      double  x,eps,a,b,x0;
      int n;
      System.out.println("Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001");
      System.out.println("Введіть значення a");
      a = Double.parseDouble(reader.readLine());
      System.out.println("Введіть значення b");
      b = Double.parseDouble(reader.readLine());
      System.out.println("Введіть значення x0");
      x0 = Double.parseDouble(reader.readLine());
      eps = 0.00001;
      int iter;
      long startTime = System.currentTimeMillis();
      for(iter=0;;iter++)
      {
         x=f(x0);
         if((Math.abs(x-x0))<eps)
            break;
         x0=x;
      }
      long timeSpent = System.currentTimeMillis() - startTime;
      System.out.printf("Корінь x =%10.7f%n", x);
      System.out.println("Кількість ітерацій = " + iter);
      System.out.println("Час роботи : " + timeSpent + " мс");0
   }
}
```

Вивід:

```
Знайти корінь рівняння cos(x)-sqrt(x)=0 при eps=0.00001
Введіть значення a
0
Введіть значення b
1
Введіть значення x0
0.5
Корінь x = 0,6417097
Кількість ітерацій = 241
Час роботи : 0 мс
```

Висновок

Для знаходження кореня кожен із запропонованих методів потребує початкового задання інтервалу [a,b], на якому розташований корінь. На цьому інтервалі функція повинна бути неперервною.

Методи половинного ділення хорд збіжні завжди, якщо правильно вказати інтервал [a,b], в якому лежить корінь. У нашому прикладі знадобилось 20 ітерацій, щоб знайти x методом половинного ділення і всього 1 ітерація для методу хорд.

Метод Ньютона збіжний лише тоді, коли f(x) і f''(x) мають однакові знаки. В іншому випадку слід обрати інший кінець відрізка. Також функція на відрізку [a,b] повинна бути двічі неперервно диференційовною. Для нашого прикладу знадобилося 4 ітерації

Метод простих ітерацій збіжний, якщо функція φ (x) визначена і неперервна на інтервалі [a, b] і $|\varphi'(x)| < 1$. Нам знадобилася 241 ітерація для знаходження x.

Не дивлячись на те, що методи суттєво різняться кількістю ітерацій, час виконання жодного з них не перевищив однієї мілісекунди. Це пояснюється тим, що ітерації різних методів мають різну швидкість виконання.

Список джерел

http://www.cyberforum.ru
https://life-prog.ru/1_7967_metodi-resheniya-nelineynih-uravneniy.html
http://statistica.ru/branches-maths/chislennye-metody-resheniya-uravneniy/#s5