МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Брянский государственный технический университет

Утверждаю
Ректор университета
_____А.В. Лагерев
«_____»_____2013 г.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧ

Методические указания
к выполнению лабораторной работы
для студентов очной формы обучения
специальностей 090303 – «Информационная безопасность автоматизированных систем»,
090900 – «Информационная безопасность»

Брянск 2013

УДК 004.43

Языки программирования. Итерационные методы решения задач: методические указания к выполнению лабораторной работы для студентов очной формы обучения специальностей 090303 – «Информационная безопасность автоматизированных систем», 090900 – «Информационная безопасность». – Брянск: БГТУ, 2013. – 19 с.

Разработали:
Ю.А. Леонов, к.т.н., доц.,
Е.А. Леонов, к.т.н., доц.

Рекомендовано кафедрой «Компьютерные технологии и системы» БГТУ (протокол № 2 от 19.09.2013)

1. ЦЕЛЬ РАБОТЫ

Целью работы является ознакомление с итерационными методами решения задач на примере вычисления суммы сходящегося ряда и нахождение корней нелинейных уравнений.

Продолжительность работы – 6 ч.

2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

2.1. Сумма сходящегося ряда

Если существует конечный предел частичных сумм $S = \lim_{n\to\infty} S_n$, то числовой ряд называется *сходящимся*. Число S_n называется n-ой частичной суммой ряда $S_n = \sum_{i=1}^{\infty} u_i$.

Сумму сходящегося ряда можно представить в виде суммы членов ряда:

$$S = u_1 + u_2 + ... + u_n, \quad (1)$$

где u_1, u_2, u_n – члены сходящегося ряда.

2.2. Итерационные методы

Для решения систем алгебраических уравнений, нахождения экстремумов функций и т.д. существуют прямые и итерационные методы.

Прямые методы позволяют получить точное решение, выполнив конечное число операций. Примером прямого метода может служить правило Крамера для решения системы совместных линейных алгебраических уравнений. Обычно для сложных систем уравнений прямые методы неэффективны, так как при их применении требуется выполнение огромного объема вычислений. Поэтому чаще пользуются итерационными методами.

Сущность итерационных методов заключается в многократном повторении одного и того же простого алгоритма, который дает результат, постепенно приближающийся к точному решению. Итерации начинаются с задания начального приближенного решения. Затем начальные значения переменных последовательно изменяются, пока не достигается заданная точность решения. Скорость сходимости

итерационного метода сильно зависит от близости выбранного начального приближения к истинному значению корня.

2.3. Решение нелинейных уравнений

Любое уравнение можно представить в виде $f(x) = 0$, если перенести все компоненты уравнения в одну сторону, тогда поиск корней уравнения сводится к поиску точек пересечения функции $f(x)$ с осью абсцисс.

В случае, когда невозможно или крайне затруднителен поиск прямых методов решения для нахождения корней нелинейных уравнений, применяют итерационные методы решения.

Пусть задана непрерывная функция $f(x)$. Требуется найти корни уравнения $f(x) = 0$ итерационными методами.

Чтобы воспользоваться итерационными методами для нахождения корней нужно решить следующие подзадачи.

1. Анализ количества, характера и расположения корней (обычно путем построения графика функции или исходя из физического смысла исследуемой модели).
2. Локализация корней (разбиение на интервалы) и выбор начального приближения к каждому корню. В простейшем случае можно протабулировать функцию с заданным шагом.
3. Вычисление корней уравнения с требуемой точностью.

Ниже представлены итерационные методы решения нелинейных уравнений:

- метод половинного деления;
- метод хорд;
- метод Ньютона (метод касательных);
- метод секущих;
- метод простых итераций.

2.3.1. Метод половинного деления

Дана математическая функция $f(x)$, которая имеет один корень на числовом интервале $[x_0, x_1]$. Требуется найти значение корня p с точностью e (рис. 1).

Суть алгоритма нахождения корня уравнения заданным методом можно представить совокупностью повторяющихся операций, называемых *итерациями*.

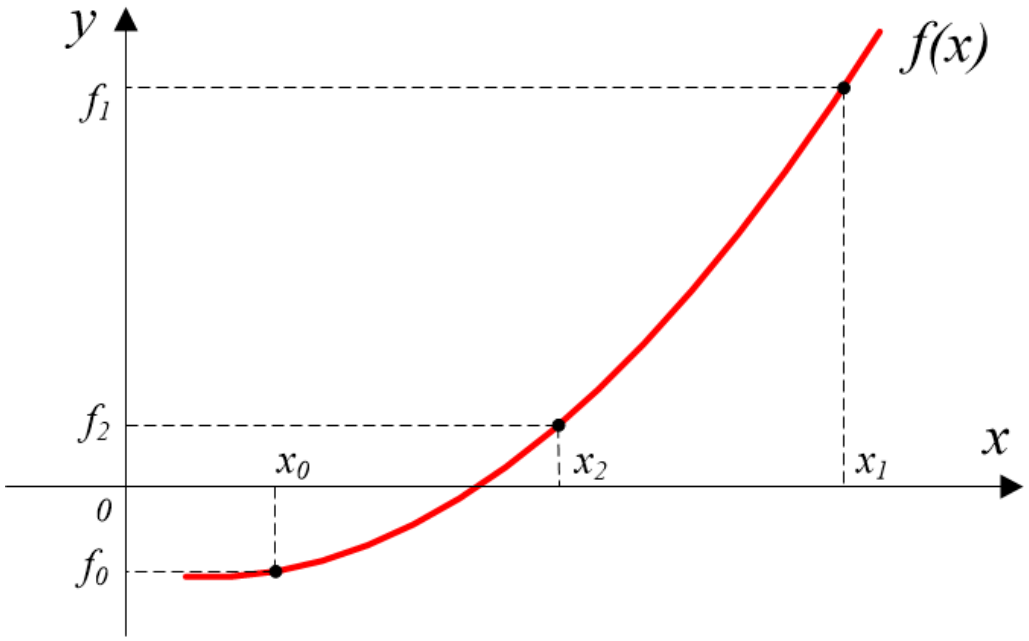

Рис. 1. Графическая интерпретация метода половинного деления

Алгоритм одной итерации можно представить в следующем виде.

1. Делим отрезок $[x_0, x_1]$, на котором имеется один корень уравнения, на две равные части, полученную точку назовем «x_2» (рис. 1).

2. Далее необходимо определить, на каком из полученных отрезков $[x_0, x_2]$ или $[x_2, x_1]$ находится корень уравнения. На рис. 1 хорошо видно, что знаки функции на концах отрезков $[x_0, x_2]$ имеют различные знаки, а для отрезка $[x_2, x_1]$ одинаковые, следовательно, корень уравнения находится на том отрезке, где значения функций для аргументов концов отрезка будут иметь различные знаки:

$$f(x_0) \cdot f(x_2) < 0. \qquad (2)$$

3. После того как мы нашли новый отрезок, задачу можно свести к предыдущей, переименовав точку x_2 в точку x_1. Так как задача свелась к предыдущей, то можно применить описанный алгоритм для того же самого отрезка $[x_0, x_1]$.

4. Итерации повторяются до тех пор, пока разница между границами числового интервала $[x_0, x_1]$ не будет меньше либо равным точности вычисления $|x_1 - x_0| \le e$ или значение функции в середине отрезка не окажется меньше либо равным точности вычисления $|f(x_2)| \le e$.

Скорость сходимости метода половинного деления невелика, однако он прост и надежен. Метод неприменим к корням четной кратности (там, где функция не изменяет свой знак).

2.3.2. Метод хорд

Дана математическая функция $f(x)$, которая имеет один корень на числовом интервале $[x_0, x_1]$. Требуется найти значение корня p с точностью e (рис. 2).

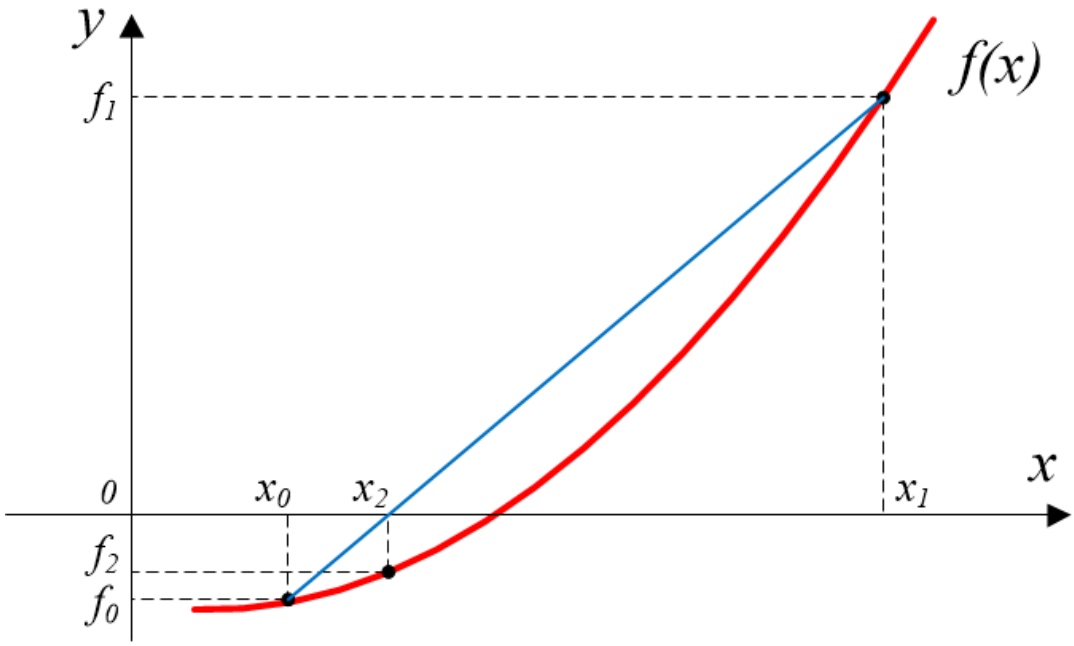

Рис. 2. Графическая интерпретация метода хорд

Приведем алгоритм нахождения корня уравнения методом хорд.

1. Между точками $(x_0, f(x_0))$ и $(x_1, f(x_1))$ проводится прямая линия (хорда), которая пересекает ось абсцисс в точке x_2 (рис. 2). Формула для нахождения точки x_2:

$$x_2 = x_0 - \frac{x_1 - x_0}{f(x_1) - f(x_0)} f(x_0). \qquad (3)$$

2. Также как и в методе половинного деления, необходимо выбрать числовой отрезок после деления отрезка $[x_0, x_1]$ точкой x_2, т.е. выбрать отрезок, на котором находится корень p из отрезков $[x_0, x_2]$ и $[x_1, x_2]$. Если выполняется условие $f(x_0) \cdot f(x_2) < 0$, то корень на отрезке $[x_0, x_2]$, иначе на отрезке $[x_2, x_1]$.

3. Переименуем точку x_2 в точку x_0, т.к. корень находится на отрезке $[x_2, x_1]$. Таким образом, задача свелась к начальной, а именно корень находится на отрезке $[x_0, x_1]$.

4. Описанный процесс повторяется до тех пор, пока разница между границами числового интервала $[x_0, x_1]$ не будет меньше либо равным точности вычисления $|x_1 - x_0| \le e$ или значение функции в се-

редине отрезка не окажется меньше либо равным точности вычисления $|f(x_2)| \le e$.

Метод хорд в большинстве случаев работает быстрее, чем метод половинного деления.

2.3.3. Метод Ньютона (метод касательных)

Дана математическая функция $f(x)$. Требуется найти значение корня p с точностью e (рис. 3).

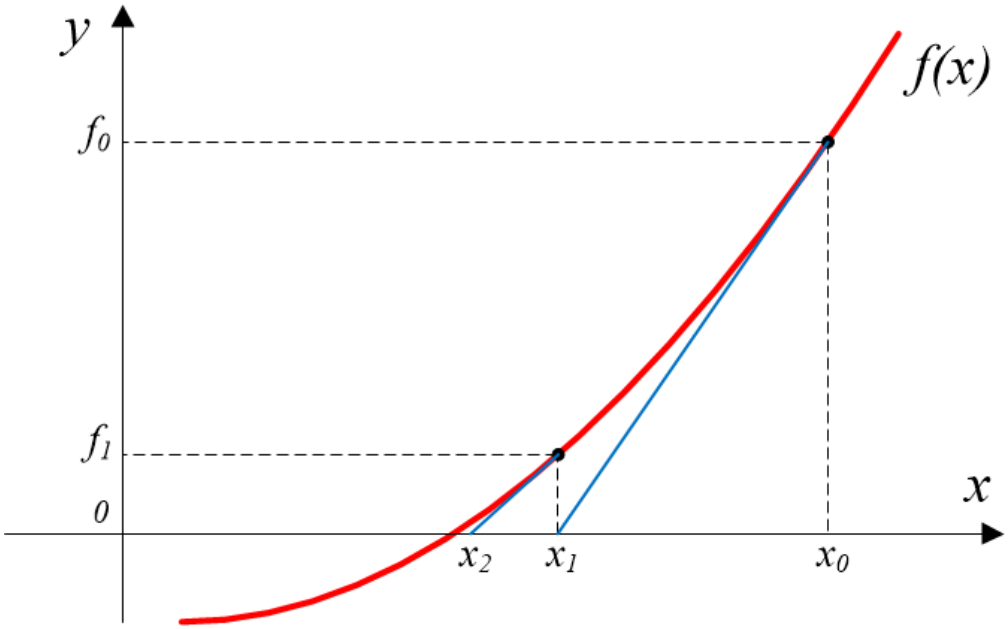

Рис. 3. Графическая интерпретация метода Ньютона

Пусть x_0 – начальное приближение к корню, а $f(x)$ имеет непрерывную производную. Следующее приближение к корню найдем в точке x_1, где касательная к функции $f(x)$, проведенная из точки (x_0, f_0), пересекает ось абсцисс. Затем точно так же обрабатываем точку (x_1, f_1), организуя итерационный процесс. Выход из итерационного процесса производится по условиям:

$$|x_n - x_{n+1}| \le e \qquad (4)$$

$$\text{или } |f(x_{n+1})| \le e. \qquad (5)$$

Общая формула вычисления последующего приближения к корню имеет вид:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \qquad (6)$$

Итерации будут сходиться к корню с той стороны, с которой выполняется следующее неравенство:

$$\left| f(x) \cdot f''(x) \right| \geq 0. \qquad (7)$$

Метод обладает самой высокой скоростью сходимости: погрешность очередного приближения примерно равна квадрату погрешности предыдущего приближения.

В качестве недостатка метода Ньютона можно указать необходимость знать явный вид первой и второй производных, так как их численный расчет приведет к уменьшению скорости сходимости метода. Для упрощения расчетов, используют *модифицированный метод Ньютона*, в котором значение производной функции вычисляется только в точке начального приближения к корню x_0:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_0)}, \qquad (8)$$

при этом число итераций увеличивается, но расчеты на каждой итерации упрощаются.

2.3.4. Метод секущих

Дана математическая функция $f(x)$. Требуется найти значение корня p с точностью e (рис. 4).

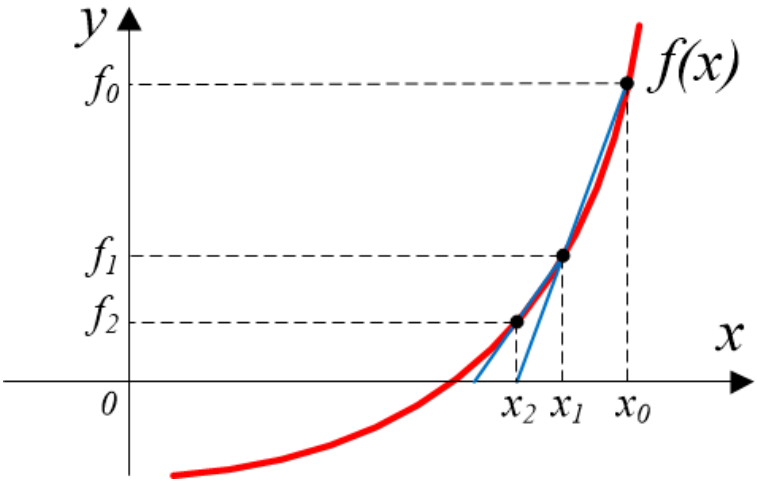

Рис. 4. Графическая интерпретация метода секущих

В отличие от метода Ньютона, можно заменить производную первой разделенной разностью, найденной по двум последним итерациям, т.е. заменить касательную секущей. При этом получим формулу для нахождения следующего приближения методом секущих:

$$x_{n+1} = x_n - \frac{f(x_n) \cdot (x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}. \qquad (9)$$

Использование этого метода избавляет от необходимости расчета производной функции в процессе вычислений.

Для выполнения первой итерации требуется задание двух начальных точек x_0 и x_1. Выбор начальной точки x_0 осуществляется по тому же принципу, что и в методе касательных, например по условию (7). Вторая начальная точка x_1 выбирается в непосредственной близости от x_0, желательно между точкой x_0 и искомым корнем.

Окончание итерационного процесса по методу секущих можно контролировать по формулам (4, 5).

2.3.5. Метод простых итераций

Суть метода простых итераций в принципе совпадает с методом, изложенным для решения систем линейных алгебраических уравнений. Для нелинейного уравнения метод основан на переходе от уравнения $f(x)=0$ к эквивалентному уравнению $x=\varphi(x)$. Этот переход можно осуществить разными способами, в зависимости от вида $f(x)$, например можно положить:

$$\varphi(x)=x+b\cdot f(x), \tag{10}$$

где $b=const,$ при этом корни исходного уравнения не изменяются. Если известно начальное приближение к корню x_0, то новое приближение $x_1=\varphi(x_0),$ т.е. в общем виде формула для нахождения следующего приближения будет:

$$x_{n+1}=\varphi(x_n), \tag{11}$$

Для окончания итерационного процесса поиска корня можно использовать формулы (4, 5). Если вблизи корня $|\varphi'(x)|<1,$ то итерации сходятся.

Исследуем выбор константы b в функции (10) с точки зрения обеспечения максимальной скорости сходимости. В соответствии с критерием сходимости наибольшая скорость сходимости обеспечивается при $|\varphi'(x)|=0.$ Если подставить в формулу (10) $b=1/f'(x),$ то формула для нахождения следующего приближения (11) переходит в

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

т.е. в формулу нахождения следующего приближения методом Ньютона (6). Таким образом, метод Ньютона является частным случаем метода простых итераций, обеспечивающим самую высокую

скорость сходимости из всех возможных вариантов выбора функции $\varphi(x)$.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Для выполнения лабораторной работы необходимо первоначально ознакомиться с теоретической частью. В случае если материала представленного в теоретической части недостаточно для выполнения задания необходимо воспользоваться лекциями и книгами по данной тематике.

Затем написать две программы на языке C# выполняющие два задания:

1) вычисление суммы сходящегося ряда;
2) нахождение корней уравнения заданным итерационным методом.

Для нахождения корня уравнения во втором задании необходимо первоначально проанализировать функцию с целью правильного задания исходных данных.

3.1. Примеры выполнения программ

Задача 1. Требуется вычислить сумму S сходящегося ряда с заданной точностью e.

Дано: числовой ряд $\sum_{i=1}^{\infty} 1/i$, точность вычисления суммы сходящегося ряда $e = 0.0001$.

Решение (программный код на языке C#).

```
using System;
namespace SumSeries
{
    class Program
    {
        static void Main()
        {
            const float e = 0.0001f;
            float i = 1, term, sum = 0;
            do {
                term = 1 / i;
                sum += term;
                i++;
            } while (term > e);
```

```
                Console.WriteLine("Сумма заданного числового
ряда = {0}", sum);
                Console.WriteLine("Количество членов ряда =
{0}", i);
                Console.Read();
            }
        }
    }
```

Задача 2. Требуется найти любой корень уравнения методом половинного деления.

Дано: уравнение $y = x^2 - 5$, точность вычисления корня $e = 0.0001$.

Решение (программный код на языке C#).

```
using System;
namespace Bisection
{
    class Program
    {
        static void Main()
        {
            const float e = 0.0001f;
            float x0, x1, x2 = 0, fx0, fx2;
            Console.WriteLine("Введите начало и конец чис-
лового промежутка");
            x0 = Convert.ToSingle(Console.ReadLine());
            x1 = Convert.ToSingle(Console.ReadLine());
            while (Math.Abs(x1 - x0) > e) {
                x2 = (x0 + x1) / 2;
                fx0 = x0 * x0 - 5;
                fx2 = x2 * x2 - 5;
                if (fx0 * fx2 < 0) x1 = x2; else x0 = x2;
            }
            Console.WriteLine("Значение корня уравнения на
заданном числовом промежутке = {0}", x2);
            Console.Read();
        }
    }
}
```

Задача 3. Требуется найти любой корень уравнения методом хорд.

Дано: уравнение $y = x^2 - 5$, точность вычисления корня $e = 0.0001$.

Решение (программный код на языке C#).

```
using System;
namespace Chord
{
    class Program
    {
        static void Main()
        {
            const float e = 0.0001f;
            float x0, x1, x2 = 0, fx0, fx1, fx2;
            Console.WriteLine("Введите начало и конец чис-
лового промежутка");
            x0 = Convert.ToSingle(Console.ReadLine());
            x1 = Convert.ToSingle(Console.ReadLine());
            while (Math.Abs(x1 - x0) > e) {
                fx0 = x0 * x0 - 5;
                fx1 = x1 * x1 - 5;
                x2 = x0 - (x1 - x0) / (fx1 - fx0) * fx0;
                fx2 = x2 * x2 - 5;
                if (fx0 * fx2 < 0) x1 = x2; else x0 = x2;
            }
            Console.WriteLine("Значение корня уравнения на
заданном числовом промежутке = {0}", x2);
            Console.Read();
        }
    }
}
```

Задача 4. Требуется найти любой корень уравнения методом Ньютона.

Дано: уравнение $y = x^2 - 5$, точность вычисления корня $e = 0.0001$.

Решение (программный код на языке C#).

```
using System;
namespace Newton
{
```

```
        class Program
        {
            static void Main()
            {
                const float e = 0.0001f;
                float x, xLast, fx, _fx;
                Console.WriteLine("Введите начальное приближе-
ние к корню");
                x = Convert.ToSingle(Console.ReadLine());
                do {
                    fx = x * x - 5;
                    _fx = 2 * x;
                    xLast = x;
                    x = xLast - fx / _fx;
                } while (Math.Abs(fx) > e);
                Console.WriteLine("Значение корня уравнения =
{0}", x);
                Console.Read();
            }
        }
    }
```

Задача 5. Требуется найти любой корень уравнения методом секущих.

Дано: уравнение $y = x^2 - 5$, точность вычисления корня $e = 0.0001$.

Решение (программный код на языке C#).

```
    using System;
    namespace Secant
    {
        class Program
        {
            static void Main()
            {
                const float e = 0.0001f;
                float x0, x1, x2, fx0, fx1, fx2;
                Console.WriteLine("Введите начало и конец чис-
лового отрезка");
                x0 = Convert.ToSingle(Console.ReadLine());
                x1 = Convert.ToSingle(Console.ReadLine());
                do {
```

```
                fx0 = x0 * x0 - 5;
                fx1 = x1 * x1 - 5;
                x2 = x1 - fx1 * (x1 - x0) / (fx1 - fx0);
                fx2 = x2 * x2 - 5;
                x0 = x1; x1 = x2;
            } while (Math.Abs(fx2) > e);
            Console.WriteLine("Значение корня уравнения на
заданном числовом отрезке = {0}", x2);
            Console.Read();
        }
    }
}
```

Задача 6. Требуется найти любой корень уравнения методом простых итераций.

Дано: уравнение $y = x^2 - 5$, точность вычисления корня $e = 0.0001$.

Решение (программный код на языке C#).

```
using System;
namespace SimpleIteration
{
    class Program
    {
        static void Main()
        {
            const float e = 0.0001f;
            const float b = 0.01f;
            float x, fx;
            Console.WriteLine("Введите начальное приближе-
ние к корню");
            x = Convert.ToSingle(Console.ReadLine());
            do {
                fx = x * x - 5;
                x = x + b * fx;
            } while (Math.Abs(fx) > e);
            Console.WriteLine("Значение корня уравнения =
{0}", x);
            Console.Read();
        }
    }
}
```

4. СПИСОК ЗАДАНИЙ

4.1. Список заданий на вычисление суммы сходящегося ряда

В списке заданий (табл. 1) ряд может быть представлен как последовательностью членов ряда, так и в общем виде: $\sum_{n=1}^{\infty} K_n$, $n = 1,2,3...$

Таблица 1

№	Задание	№	Задание
1	$\frac{x}{n^p}$	**16**	$\frac{1}{n\cdot(n+1)}$
2	$\frac{1}{n!}$	**17**	$\frac{0}{(0+1)^n}-\frac{3}{(1+2)^n}+\frac{9}{(2+3)^n}-$
3	$\frac{1}{n\cdot(n+1)}$	**18**	$\frac{(1-\sin x)}{n!}$
4	$\frac{1}{n^p}$	**19**	$\frac{1}{n\cdot(4n-5)}$
5	$\frac{x^n}{n!}$	**20**	$-\frac{0^0}{1!}+\frac{1^1}{2!}-\frac{2^0}{3!}+\frac{3^1}{4!}$
6	$\frac{5-x}{n!}$	**21**	$\frac{9+x^n}{n!}$
7	$-\frac{x^1}{2!}+\frac{x^2}{4!}-\frac{x^3}{6!}+\frac{x^4}{8!}$	**22**	$-\frac{x^1}{1!}+\frac{x^2}{2!}-\frac{x^3}{4!}+\frac{x^4}{8!}$
8	$\frac{1}{2^n}-\frac{1}{3^n}+\frac{1}{4^n}-$	**23**	$\frac{(x-10)\cdot x}{(1+n!)}$
9	$\frac{1}{1!}-\frac{1}{2!}+\frac{1}{3!}-$	**24**	$\frac{1}{5^n}-\frac{1}{8^n}+\frac{1}{11^n}-\frac{1}{14^n}+$
10	$\frac{1}{(0+1)^n}-\frac{1}{(1+3)^n}+\frac{1}{(2+7)^n}-$	**25**	$\frac{\cos x+1}{5n!}$
11	$\frac{(5x-1)}{(1-n!)}$	**26**	$\frac{2}{(1+1)^n}-\frac{4}{(2-1)^n}+\frac{8}{(3+1)^n}-$
12	$-\frac{1^1}{1!}+\frac{2^2}{2!}-\frac{1^3}{3!}+\frac{2^4}{4!}-$	**27**	$\frac{(1-4x)}{5n!}$
13	$\frac{(\sin x-1)}{(3n!)}$	**28**	$\frac{4}{3n\cdot(n+8)!}$

Окончание табл. 1

14	$\frac{2}{1!}-\frac{2}{2!}+\frac{2}{4!}-\frac{2}{7!}+$	**29**	$\frac{(1-\cos x)}{(n^n)}$
15	$\frac{(tgx+1)}{(2n!)}$	**30**	$\frac{1}{2^n}-\frac{2}{2^n}+\frac{1}{4^n}-\frac{2}{4^n}+$

Примечание: x, p – это константы, объявленные в программе; n – натуральные числа, принимающие значения в интервале [1; ∞].

4.2. Список заданий на нахождение корней уравнений

В списке заданий (табл. 2) используются следующие сокращения:
A – метод половинного деления;
B – метод хорд;
C – метод Ньютона (метод касательных);
D – метод секущих;
E – метод простых итераций.

Таблица 2

№	Уравнение	Метод	№	Уравнение	Метод
1	$\ln x-\frac{1}{x^2}$	*A*	**16**	$e^{-x}-\sqrt{x-1}$	*A*
2	$2\cdot\ln x-\frac{x}{2}+1$	*B*	**17**	$2\sin(3x)-1.5x$	*B*
3	$\frac{1-x}{x}-3\cos(4x)$	*C*	**18**	$0.1e^{-x}-\frac{x}{2}$	*C*
4	$ctg(x)-x^2$	*D*	**19**	$\ln(1.2x)-1.5x+2$	*D*
5	$tg\frac{x}{4}-x-2$	*E*	**20**	$tg(2.5x)-5x$	*E*
6	$\sqrt{x}-3\sin x$	*A*	**21**	$\ln x-2\cos x$	*A*
7	$\sqrt{x}-3\cos\frac{x}{2}$	*B*	**22**	$\sqrt{2-x^2}-e^x$	*B*
8	$2\cdot\ln x-\frac{1}{x}$	*C*	**23**	$e^{-(x+1)}+x^2+2x-1$	*C*
9	$x-3\cos^2 x$	*D*	**24**	$e^{-x}-2+x^2$	*D*
10	$tg(7.5x)-2(x+1)$	*E*	**25**	$x\cdot e^x-x-1$	*E*
11	$\ln x-\frac{7}{2x+6}$	*A*	**26**	$(\cos x)^2-\frac{1}{2}\cos x+\frac{1}{4}$	*A*

Окончание табл. 2

12	$e^{-x}-(x-1)^2$	*B*	**27**	$\sin(x+2)-x^2+2x$	*B*
13	$e^{x}+x^2-2$	*C*	**28**	$x-e^{-x^2}$	*C*
14	$e^{x}-2(x-1)^2$	*D*	**29**	$x\ln x-x^2+3x$	*D*
15	$e^{x}+2x^2-3$	*E*	**30**	$e^{-x}-5x^2+10x$	*E*

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие методы решения задач называют прямыми, а какие итерационными?
2. Чем следует руководствоваться при выборе метода решения задачи?
3. Какие итерационные методы решения уравнений вы знаете?
4. Как выбираются начальные приближения для итерационных методов решения задач?
5. В чем принцип метода половинного деления?
6. Чем отличается метод Ньютона от метод секущих?
7. Без каких конструкций языка C# не обходится решение задач итерационными методами?
8. Назовите формулы для нахождения следующего приближения методами: хорд, Ньютона, секущих, половинного деления.
9. Дайте графическое представление итерационных методов решения нелинейных уравнений.

6. СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Основная

1. Колдаев В.Д. Численные методы и программирование. Учебное пособие. – Изд.: ИД «Форум», 2009. – 336с.

2. Кутищев Г.П. Решение алгебраических уравнений произвольной степени. Теория, методы, алгоритмы. – Изд.: ЛКИ, 2010. – 232с.

3. Павловская Т. А. C#. Программирование на языке высокого уровня. – Изд.: Питер, 2009. – 432с.

4. Эндрю Троелсен. Язык программирования C# 2010 и платформа .NET 4. – Изд.: Вильямс, 2011. – 1392с.

5. Кристиан Нейгел, Билл Ивьен, Джей Глинн, Карли Уотсон, Морган Скиннер. C# 4.0 и платформа .NET 4 для профессионалов. – Изд.: Питер, 2011. – 1440с.

Дополнительная

6. Тыртышников Е.Е. Методы численного анализа. Учебное пособие. – Изд.: МГУ, 2006. – 281с.

7. Киреев В.И., Пантелеев А.В. Численные методы в примерах и задачах. Учебное пособие. – Изд.: Высшая школа, 2008. – 481с.

8. Джесс Либерти. Программирование на C#. – Изд.: КноРус, 2003. – 688с.

9. Харви Дейтел. C# в подлиннике. Наиболее полное руководство. – Изд.: БХВ-Петербург, 2006. – 1056с.

Языки программирования. Итерационные методы решения задач: методические указания к выполнению лабораторной работы для студентов очной формы обучения специальностей 090303 – «Информационная безопасность автоматизированных систем», 090900 – «Информационная безопасность». – Брянск: БГТУ, 2013. – 19 с.

ЮРИЙ АЛЕКСЕЕВИЧ ЛЕОНОВ
ЕВГЕНИЙ АЛЕКСЕЕВИЧ ЛЕОНОВ

Научный редактор: Ю.М. Казаков
Редактор издательства: Л.И. Афонина
Компьютерный набор: Ю.А. Леонов

Темплан 2013г., п.

Подписано в печать Формат 60x84 1/16. Бумага офсетная.
Офсетная печать.
Усл. печ. л. 1,1 Уч. – изд. л. 1,1 Тираж 50 экз. Заказ Бесплатно

Издательство брянского государственного технического университета,
241035, Брянск, бульвар 50-летия Октября, 7, БГТУ. 58-82-49
Лаборатория оперативной полиграфии БГТУ, ул. Харьковская, 9