Московский авиационный институт
(национальный исследовательский университет)

Институт № 8 «Информационные технологии и прикладная математика»

КУРСОВАЯ РАБОТА

по курсу «Фундаментальная информатика»

1 семестр

номер задания 4

Студент:	Мазаев В. В.
Группа:	М8О-104Б-22
Преподаватель:	Потенко М. А.
Подпись:	
Оценка:	

Москва, 2022

Тема: Процедуры и функции в качестве параметров

Задание: Составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итерации, Ньютона и половинного деления – дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины в случае необходимости. Применить каждую процедуру к решению двух уравнений, начиная с варианта с заданным номером. Если метод неприменим, дать математическое обоснование и графическую иллюстрацию с использованием gnuplot.

Вариант 15

Уравнение: $0{,}4 + arctg\sqrt{x} - x = 0$

Отрезок, содержащий корень: [1, 2]

Базовый метод: метод итераций

Приближённое значение корня: 1,2388

Вариант 16

Уравнение: $3sin\sqrt{x} + 0{,}35x - 3{,}8 = 0$

Отрезок, содержащий корень: [2, 3]

Базовый метод: метод итераций

Приближённое значение корня: 2,2985

Описание методов:

1. *Метод дихотомии (половинного деления)*

Численное нахождение приближенного значения корня функции строится на базе следствия из теоремы Больцано-Коши: «Если непрерывная функция на концах некоторого интервала имеет значение разных знаков, то внутри этого интервала у нее есть как минимум один корень».

Считаем, что отделение корней произведено и на интервале [a, b] расположен один корень, который необходимо уточнить с погрешностью ε.
Итак, имеем f(a)*f(b)<0. Метод дихотомии заключается в следующем. Определяем половину отрезка c=½(a+b) и вычисляем f(c). Проверяем следующие условия:

1) Если |f(c)| < ε, то c – корень. Здесь ε - заданная точность.
2) Если f(c)*f(a)<0, то корень лежит в интервале [a,c].
3) Если f(c)*f(b)<0, то корень лежит на отрезке[c,b].

Продолжая процесс половинного деления в выбранных подынтервалов, можно дойти до сколь угодно малого отрезка, содержащего корень ξ.
Так как за каждую итерацию интервал, где расположен корень уменьшается в два раза, то через n итераций интервал будет равен: $b_n - a_n = \frac{1}{2^n}(b - a)$, при этом $an \leq ξ \leq bn$,

$$|\xi - a_n| \leq \frac{1}{2^n}(b-a), |\xi - b_n| \leq \frac{1}{2^n}(b-a)$$

В качестве корня ξ возьмем $= ½(bn + an)$. Тогда погрешность определения корня будет равна $\frac{bn - an}{2}$. Если выполняется условие $\frac{b_n - a_n}{2} < \varepsilon$, то процесс поиска заканчивается и $\varepsilon = ½(bn + an)$.

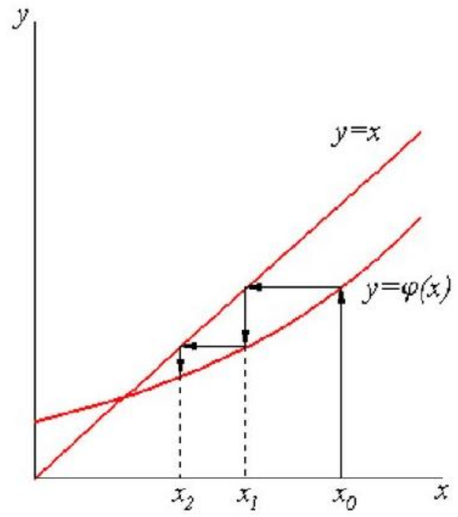
- Функция $f(x) = 0{,}4 + arctg\sqrt{x} - x$ непрерывна на отрезке $[1, 2]$ и на концах его принимает значения разных знаков: $f(1) \approx 0{,}185$; $f(2) \approx -0{,}645$
- Функция $f(x) = 3sin\sqrt{x} + 0{,}35x - 3{,}8$ непрерывна на отрезке $[2, 3]$ и на концах его принимает значения разных знаков: $f(2) \approx -0{,}137$; $f(3) \approx 0{,}211$

2. *Метод итераций*

   Суть метода в поиске по известному приближению искомой величины следующего, более точного приближения. Пусть дана функция $F(x)$. Заменим исходное уравнение $F(x) = 0$ на эквивалентное $f(x) = x$. Выберем начальное приближение корня x_0. Тогда получим некоторое число $x_1 = f(x_0)$. Теперь подставляя вместо x_0 число x_1 получим $x_2 = f(x_1)$. Повторяя этот процесс, будем иметь последовательность $x_n = f(x_{n-1}), n = 1, 2, \dots$. Если эта последовательность сходящаяся, то существует предел $\lim_{n\to\infty} x_n$, то данный предел является корнем уравнения и может быть вычислен по формуле $x_n = f(x_{n-1})$. Для преобразования $F(x) = 0$ в $f(x) = x$ используется уравнение вида $f(x) = x - \lambda F(x)$, где λ – некоторая постоянная, знак которой совпадает со знаком производной $F'(x)$ в некоторой окрестности корня.

Условие сходимости метода итераций: $|f'(x)| < 1$

Геометрическая интерпретация метода простой итерации



- Найдем $f(x)$ для функции F(x): $0{,}4 + arctg\sqrt{x} - x \Leftrightarrow 0{,}4 + arctg\sqrt{x}$

$$f'(x) = \frac{1}{2\sqrt{x}(x+1)}; \left|\frac{1}{2\sqrt{x}(x+1)}\right| < 1$$

  Условие сходимости выполняется на данном отрезке [1, 2].

- Найдем $f(x)$ для функции F(x): $3sin\sqrt{x} + 0{,}35x - 3{,}8 \Leftrightarrow 0{,}3x - 6sin\sqrt{x} + 7{,}6$

$$f'(x) = 0{,}3 - \frac{3\cos\sqrt{x}}{\sqrt{x}}; \left|0{,}3 - \frac{3\cos\sqrt{x}}{\sqrt{x}}\right| < 1$$

  Условие сходимости выполняется на данном отрезке [2, 3]

3. *Метод Ньютона*

Метод Ньютона – частный случай метода итераций ($f(x) = x - \frac{F(x)}{F'(x)}$). Суть метода состоит в разбиении отрезка $[a, b]$ на два отрезка с помощью касательной и выборе нового отрезка от точки пересечения касательной с осью абсцисс до неподвижной точки, на которой функция меняет знак и содержит решение.

Построение касательных продолжается до достижения необходимой точности ε.

Метод итераций сходится тогда и только тогда, когда $|f'(x)| < 1$, подставим в условие выражение для $f(x)$ и получим условие сходимости метода Ньютона:

$\left|\frac{F(x)F''(x)}{(F'(x))^2}\right| < 1$

Условие окончания: если $|x_{n+1} - x_n| < \varepsilon$, то значение x_{n+1}считается приближенным значением корня уравнения.

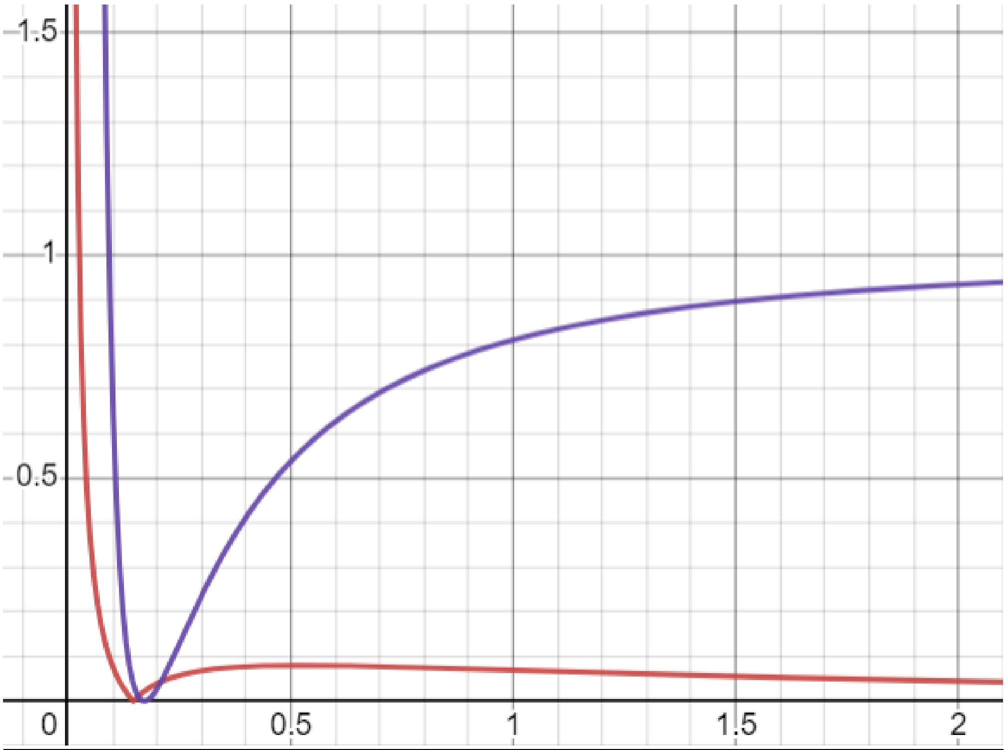
Проверим условие сходимости для заданных функций, вычислив их производные.

Вариант 15

$$f'(x) = \left(0{,}4 + arctg\sqrt{x} - x\right)' = \frac{1}{2\sqrt{x}(x+1)} - 1$$

$$f''(x) = \left(\frac{1}{2\sqrt{x}(x+1)} - 1\right)' = -\frac{\sqrt{x}\,(3x+1)}{4x^4 + 8x^3 + 4x^2}$$

$|f(x) \cdot f''(x)| < f'(x)^2$ на отрезке $[1, 2]$:



(Синим - $f'(x)^2$; красным - $f(x) \cdot f''(x)$)

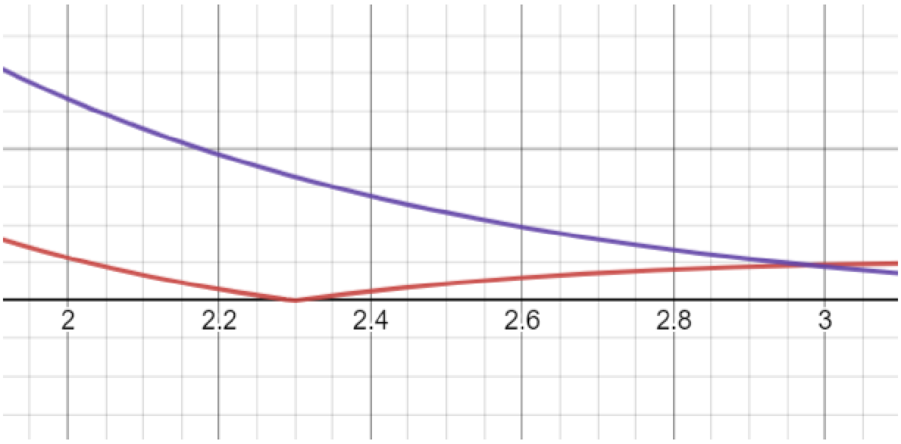
Условие сходимости выполняется.

Вариант 16

$$f'(x) = \left(3sin\sqrt{x} + 0{,}35x - 3{,}8\right)' = \frac{3\cos\sqrt{x}}{2\sqrt{x}} + 0{,}35$$

$$f''(x) = \left(\frac{3\cos\sqrt{x}}{2\sqrt{x}} + 0{,}35\right)' = -\frac{3\sin\sqrt{x}\,\sqrt{x} + 3\cos\sqrt{x}}{4x^{3/2}}$$

$|f(x) \cdot f''(x)| < f'(x)^2$ на отрезке $[2, 3]$:



(Синим - $f'(x)^2$; красным - $f(x) \cdot f''(x)$)

Условие сходимости выполняется.

Используемое оборудование:

ЭВМ *HP Pavilion Laptop 15*

Процессор *AMD Ryzen 5 5500U with Radeon Graphics*

ОП *8192 Мб*

НМД *524288 Мб*

Программное обеспечение:

Операционная система семейства *UNIX*

Наименование *Ubuntu*

Версия *18.04.6*

Интерпретатор команд *bash*

Редактор текстов *gedit*

Утилиты операционной системы *Терминал*

Местонахождение и имена файлов программ */Users/diss/kp3.c; /Users/diss/eps.c*

Программа будет написана на языке программирования Си, скомпилирована и запущена посредством терминала.

Функциональное назначение:

Программа находит корни уравнения посредством манипуляций над вещественными числами и математических операций над данными.

Описание функций и переменных, используемых в программе:

Название функции	*Тип функции*	*Описание переменных*	*Смысл функции*
function_1	double	double x – точка, в которой необходимо вычислить значение функции	Вычисляет значение функции в указанной точке (вариант 15)
function_1_iter	double	double x – точка, в которой необходимо вычислить значение доп. функции	Вычисляет значение дополнительной функции в указанной точке (вариант 15)
function_1_diff	double	double x – точка, в которой необходимо вычислить значение производной функции	Вычисляет производную функции в указанной точке (вариант 15)
function_2	double	double x – точка, в которой необходимо вычислить значение функции	Вычисляет значение функции в указанной точке (вариант 16)
function_2_iter	double	double x – точка, в которой необходимо вычислить значение доп. функции	Вычисляет значение дополнительной функции в указанной точке (вариант 15)
function_2_diff	double	double x – точка, в которой необходимо вычислить значение производной функции	Вычисляет производную функции в указанной точке (вариант 16)
dichotomy	double	double(*Func_main)(double)- указатель на функцию, корень которой нужно	Вычисляет корень уравнения методом

		вычислить double a, double b - значения концов отрезка DBL_EPSILON – библиотечное значение машинного эпсилон	дихотомии
iteration	double	double(*func)(double)- указатель на дополнительную функцию для функции, корень которой нужно вычислить double a, double b - значения концов отрезка double x_cur, x_last – вычисляемые приближения DBL_EPSILON – библиотечное значение машинного эпсилон	Вычисляет корень уравнения методом итераций
newton	double	double(*Func_main)(double)- указатель на функцию, корень которой нужно вычислить double (*func_diff)(double) – указатель на функцию, вычисляющую производную double x_cur, x_last - вычисляемые приближения double a, double b - значения концов отрезка DBL_EPSILON – библиотечное значение машинного эпсилон	Вычисляет корень уравнения методом Ньютона

Код программы:

```
#include <stdio.h>
#include <math.h>
#include <float.h>

double function_1(double x)
{
    return 0.4 + atan(sqrt(x)) - x;
}

double function_1_iter(double x)
{
    return 0.4 + atan(sqrt(x));
}

double function_1_diff(double x)
{
    return 1 / (2 * sqrt(x) * (x + 1)) - 1;
}

double function_2(double x)
{
    return 3 * sin(sqrt(x)) + 0.35 * x - 3.8;
}

double function_2_iter(double x)
{
    return 0.3 * x - 6 * sin(sqrt(x)) + 7.6;
}

double function_2_diff(double x)
{
    return (3 * cos(sqrt(x))) / (2 * sqrt(x)) + 0.35;
}


double dichotomy(double(*Func_main)(double), double a, double b)
{
    double x;
    while (fabs(a - b) > DBL_EPSILON) { //пока значения концов отрезка
отличаются с заданной точностью
        x = (a + b) / 2.0;
        if ((*Func_main)(a) * (*Func_main)(x) < 0.0) //если значение
функции на концах отрезка разных знаков - обновление значений концов
отрезка
            b = x;
        else
            a = x;
    }
    return x;
}

double iteration(double(*func)(double), double a, double b)
{
    double x_cur = (a + b) / 2.0, x_last; //первое и последующие
приближения
    while (fabs(x_cur - x_last) > DBL_EPSILON) { //пока значения
```

```
отличаются с заданной точностью, вычислять следующее приближение
        x_last = x_cur;
        x_cur = (*func)(x_last);
    }
    return x_cur;
}

double newton(double(*Func_main)(double), double (*func_diff)(double),
double a, double b)
{
    double x_cur = (a + b) / 2.0, x_last; //первое и последующие
приближения
    while (fabs(x_cur - x_last) > DBL_EPSILON) { //пока значения
отличаются с заданной точностью, вычислять следующее приближение (с
помощью касательной)
        x_last = x_cur;
        x_cur -= (*Func_main)(x_last) / (*func_diff)(x_cur);
    }
    return x_cur;
}

int main() {
    printf("Для перого уравнения на отрезке [1, 2]:\n");
    printf("Методом диохтомии: %.10lf\n", dichotomy(function_1, 1.0,
2.0)); //вывод корня уравнения, вычисленного методом диохтомии для
варианта 15
    printf("Методом итераций: %.10lf\n", iteration(function_1_iter,
1.0, 2.0)); //вывод корня уравнения, вычисленного методом итераций для
варианта 15
    printf("Методом Ньютона: %.10lf\n", newton(function_1,
function_1_diff, 1.0, 2.0)); //вывод корня уравнения, вычисленного
методом Ньютона для варианта 15
    printf("Для второго уравнения на отрезке [2, 3]:\n");
    printf("Методом диохтомии: %.10lf\n", dichotomy(function_2, 2.0,
3.0)); //вывод корня уравнения, вычисленного методом диохтомии для
варианта 16
    printf("Методом итераций: %.10lf\n", iteration(function_2_iter,
2.0, 3.0)); //вывод корня уравнения, вычисленного методом итераций для
варианта 16
    printf("Методом Ньютона: %.10lf\n", newton(function_2,
function_2_diff, 2.0, 3.0)); //вывод корня уравнения, вычисленного
методом Ньтона для варианта 16
    return 0;
}
```

Результат выполнения:

```
Для перого уравнения на отрезке [1, 2]:
Методом диохтомии: 1.2388399776
Методом итераций: 1.2388399776
Методом Ньютона: 1.2388399776
Для второго уравнения на отрезке [2, 3]:
Методом диохтомии: 2.2985361709
Методом итераций: 2.2985361709
Методом Ньютона: 2.2985361709
```

Вывод: В процессе выполнения данного задания я углубился в тему решения математических задач с помощью программных средств. Я научился использовать функции в качестве параметров других функций, то есть применять указатели.

Часто сложно аналитически, с помощью формул решит трансцендентные уравнения. А иногда на практике уравнение содержит коэффициенты, значения которых заданы приблизительно, так что говорить о точном решении уравнений в таких случаях вообще не имеет смысла. Таким образом, задача нахождения приблизительных значений корней уравнений с определённой точностью является актуальной.

В данном задании рассматривались 3 метода решения таких уравнений: метод дихотомии, метод итераций и метод Ньютона. С помощью этих методов были вычислены значения корней заданных уравнений. В итоге значения с точностью до 10 знаков не отличаются.

Численные методы являются основным инструментом решения современных прикладных задач. Численный анализ математических моделей является в настоящее время актуальным и наиболее эффективным аппаратом конструктивного исследования прикладных проблем.

Подпись студента