Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8

Кафедра №806

**Курсовой проект**

**по учебной дисциплине “Вычислительные системы”**

**I семестр**

**Задание 4.**

ФИО: Манташев А. У.

Группа: М8О-105Б-20

№ по списку: 17

Преподаватель: доц. каф. 806 Никулин Сергей Петрович

Оценка: Подпись преподавателя:

Дата сдачи: 19.12.2020

**Задание**

Составить программу на языке СИ с процедурами (функциями) решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона, половинного деления — дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины в случае необходимости. Применить каждую процедуру к решению уравнения, заданного двумя последовательными строками таблицы, начиная с варианта с заданным номером.





**Общий метод решения**

Каждое уравнение решаем 3 методами: итераций, дихотомии (половинного деления), Ньютона.

*Метод дихотомии* заключается в делении отрезка пополам и сужении его границ в 2 раза на каждом шаге итерационного процесса в зависимости от знака функции в середине отрезка. За начальное приближение принимаются границы исходного отрезка. Далее вычисления производятся по одной из двух формул:

)\*F()>0 => =, =

)\*F()>0 => =, =

Процесс повторяется до тех пор, пока выполняется условие >ε

*Метод итераций* заключается в замене исходного уравнения уравнением вида x=f(x). Условие сходимости: . Его необходимо проверить перед началом решения задачи, т. к. функция f(x) может быть выбрана неоднозначно, причем в случае неверного выбора указанной функции метод расходится.

Начальное приближение корня - середина отрезка.

Итерационный процесс: =f()

Условие окончания: >ε

*Метод Ньютона* является частным случаем метода итераций.

Условие сходимости: 

Итерационный процесс: =(.

**Описание программы**

В начале программы составляем функции для решения уравнения определённым методом и функции, послужащие им аргуметом. Для каждой функции нужно запрограммировать исходную формулу, самостоятельно вычисленную производную, и формулу для подстановки в итерационный метод. В теле основной программы делаем только вызов подпрограмм и вывод.

**Описание функций программы:**

|  |  |  |
| --- | --- | --- |
| Название функции | Аргументы и их тип | Описание функции |
| double epsil() | Отсутствуют | Функция вычисляет машинный эпсилон путём цикличного деления на два и проверки на различаемость полученного числа машиной, после чего возвращает  найденное значение |
| double func1() | double x | Функция вычисления значения x для функции №17 |
| double func2() | double x | Функция вычисления значения x для функции №18 |
| double func1\_der() | double x | Вычисление значения x для производной функции №17 |
| double func2\_der() | double x | Вычисление значения x для производной функции №18 |
| double func1\_iter() | double x | Вычисление значения x для метода итераций функции №17 |
| double func2\_iter() | double x | Вычисление значения x для метода итераций функции №18 |
| double dht\_meth() | double func,a,b,eps | Вычисляет методом дихотомии корень функции func на отрезке [a;bl с точностью до eps |
| double iter\_meth() | double func,a,b,eps | Вычисляет методом итераций корень функции func на отрезке [a;bl с точностью до eps |
| double newton\_meth() | double func,derivative,a,b,eps | Вычисляет методом Ньютона корень функции func на отрезке [a;bl с точностью до eps. Требует дополнительную функцию производной. |
| int main() | Отсутствуют | Главная функция программы |

*Входные данные* не предусмотрены.

*Выходные данные*: список корней уравнений, посчитанных тремя методами для каждой из двух функций.

**Сценарий выполнения.**

|  |  |  |  |
| --- | --- | --- | --- |
| Уравнения | Дихотомия | Метод Итераций | Метод Ньютона |
|  | 1.0001 | 1.0001 | 1.0001 |
|  | 0.7376 | 0.7376 | 0.7376 |

**Протокол.**

gladiator@asadec:~/Downloads$ cat kp4.c

#include <stdio.h>

#include <math.h>

typedef double(\*func\_pointer)(double);

double func1(double x){

return 0.25\*pow(x,3)+x-1.2502;

}

double func2(double x){

return x+pow(x,0.5)+pow(x,1.0/3.0)-2.5;

}

double func1\_der(double x){

return 0.75\*pow(x,2)+1;

}

double func2\_der(double x){

return 1+0.5\*pow(x,-0.5)+1.0/3.0\*pow(x,-2.0/3.0);

}

double func1\_iter(double x){

return 0.75\*x-0.25\*0.25\*pow(x,3)+0.31255;

}

double func2\_iter(double x){

return 2.0/3.0\*x-1.0/3.0\*pow(x,0.5)-1.0/3.0\*pow(x,1.0/3.0)+2.5/3.0;

}

double dht\_meth(func\_pointer func,double a,double b,double eps){

double x;

while(fabs(a-b)>=eps){

x=(a+b)/2.0;

if (func(a)\*func(x)>0.0)

a=x;

else

b=x;

}

return x;

}

double iter\_meth(func\_pointer func,double a,double b,double eps){

double x=(a+b)/2.0;

while(fabs(func(x)-x)>=eps)

x=func(x);

return x;

}

double newton\_meth(func\_pointer func,func\_pointer derivative,double a,double b,double eps){

double x=(a+b)/2.0;

while(fabs(func(x)/derivative(x))>=eps)

x-=func(x)/derivative(x);

return x;

}

int main(){

double eps = 0.00000001;

double a1=0.0,b1=2.0,a2=0.4,b2=1.0;

printf("--------------------------------------------------------------\n");

printf("| | Дихотомия | Метод итераций | Метод Ньютона |\n");

printf("--------------------------------------------------------------\n");

printf("| Уравнение №17 | %.6lf | %.6lf | %.6lf |\n",dht\_meth(func1,a1,b1,eps),iter\_meth(func1\_iter,a1,b1,eps),newton\_meth(func1,func1\_der,a1,b1,eps));

printf("--------------------------------------------------------------\n");

printf("| Уравнение №18 | %.6lf | %.6lf | %.6lf |\n",dht\_meth(func2,a2,b2,eps),iter\_meth(func2\_iter,a2,b2,eps),newton\_meth(func2,func2\_der,a2,b2,eps));

printf("--------------------------------------------------------------\n");

return 0;

}

gladiator@asadec:~/Downloads$ gcc kp4.c -lm -Wall --std=c99 -pedantic -o kp4.out && ./kp4.out

---------------------------------------------------------------------------------------------

| | Дихотомия | Метод итераций | Метод Ньютона |

---------------------------------------------------------------------------------------------

| Уравнение №17 | 1.000114 | 1.000114 | 1.000114 |

---------------------------------------------------------------------------------------------

| Уравнение №18 | 0.737619 | 0.737619 | 0.737619 |

---------------------------------------------------------------------------------------------

**Вывод**

В процессе выполнения этот задания, я получил навыки вычисления корней уравнений тремя разными способами. Я на практике применил знания, полученные на занятиях ранее. Также я отточил навыки использования функций.