МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Московский авиационный институт

(национальный исследовательский университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра 806: «Вычислительная математика и программирование»

Курсовой проект

по курсу «Основы информатики»

I семестр

Задание 4. «Процедуры и функции в качестве параметров.»

|  |  |
| --- | --- |
| Группа: | М8о-107б-18 |
| Студент: | Тояков Артем |
| Преподаватель: | Ридли Александра Николаевна |
| Оценка: |  |
| Дата: |  |

# СОДЕРЖАНИЕ

[ЗАДАНИЕ 3](#_Toc536055977)

[МЕТОДЫ РЕШЕНИЯ 3](#_Toc536055978)

[ВЫЧИСЛЕННИЕ МАШИННОГО ЭПИСЛОН 4](#_Toc536055979)

[ЛОГИЧЕСКАЯ СТРУКТУРА 5](#_Toc536055980)

[ОПИСАНИЕ ПЕРЕМЕННЫХ 5](#_Toc536055981)

[ФУНКЦИИ 6](#_Toc536055982)

[ВХОДНЫЕ ДАННЫЕ 7](#_Toc536055983)

[ВЫХОДНЫЕ ДАННЫЕ 7](#_Toc536055984)

[КОД ПРОГРАММЫ 8](#_Toc536055985)

[РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ 10](#_Toc536055986)

[ВЫВОД 10](#_Toc536055987)

# ЗАДАНИЕ

Составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона и дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины в случае необходимости.

4.

5.

# МЕТОДЫ РЕШЕНИЯ

1. Метод дихотомии (половинного деления).

Очевидно, что если на отрезке [a,b] существует корень уравнения, то значения функции на концах отрезка имеют разные знаки: F(а) • F(b) < 0. Метод заключается в делении отрезка пополам и его сужении в два раза на каждом шаге итерационного процесса в зависимости от знака функции в середине отрезка.

Итерационный процесс строится следующим образом: за начальное приближение принимаются границы исходного отрезка a(0) = а, = b . Далее вычисления проводятся по формулам: а(k+1)= (а(k) + b(k))/ 2, b(k+1) = b(k) если F(a(k)) • F((a(k) + b(k))/2) > 0; или по формулам: a(k+1) = a(k), b(k+1) = (а(k) + b(k))/2, если F(b(k)) • F((a(k) +b(k))/2)>0.

Процесс повторяется до тех пор, пока не будет выполнено условие окончания |а(k) — b(k)| < ε .

Приближенное значение корня к моменту окончания итерационного процесса получается следующим образом:

х\* (а(конечное) +b(конечное))/2 .

2) Метод итераций.

Идея метода заключается в замене исходного уравнения F(x) = 0 уравнением вида х = f(х).

Достаточное условие сходимости метода: |(х)| < 1, x [a,b]. Это условие необходимо проверить перед началом решения задачи, так как функция f(х) может быть выбрана неоднозначно, причем в случае неверного выбора указанной функции метод расходится.

Начальное приближение корня: x(0) = (а + b)/2 (середина исходного отрезка).

Итерационный процесс: х(k+1) = f(x(k)).

Условие окончания: |x(k) – х(k-1)| < .

Приближенное значение корня: х\* х(конечное) .

3. Метод Ньютона.

Метод Ньютона является частным случаем метода итераций.

Условие сходимости метода:

|F(x) • F ''(x)| < (F'(x))2 на отрезке [а,b].

Итерационный процесс: x(k+1) = x(k) - F(x(k))/F'(x(k)).

# ВЫЧИСЛЕННИЕ МАШИННОГО ЭПИСЛОН

1. Подключить библиотеку limits.h и использовать константу DBL\_EPSILON

2. Делим 1.0 пополам пока не получится так, что мы не можем отличить одно от другого. Если так случилось, значит, разница на предыдущем шаге и есть машинное эпсилон.

double eps = 1.0;

while (1.0 + (eps / 2.0) > 1.0) {

eps /= 2.0;

}

Нужно найти число, у которого мантисса «сдвинута» на единицу левее другого числа. Например, числа 7/3 и 4/3:

7/3 = 10.010101010101…

4/3 = 1.0101010101010…

Запишем числа в формате IEEE-734:

7/3 = 1.0010101010101010101010101010101010101010101010101011 \* 2¹

4/3 = 1.0101010101010101010101010101010101010101010101010101 \* 2º

При вычитании все биты, кроме последнего, зануляются:

7/3 – 4/3 = 1.0000000000000000000000000000000000000000000000000001\*1

Получается, 7/3 – 4/3 = 1 + ε

ε = 7/3 – 4/3 – 1

double eps = 7.0 / 3.0 – 4.0 / 3.0 – 1.0;

# ЛОГИЧЕСКАЯ СТРУКТУРА

1) Описание вспомогательных функций:

Функции вида *F*(*x*)

Функции вида x=*f*(*x*)

Функции вида

2) Функции вычисления машинного эпсилон, дихотомии, метода итераций и метода Ньютона в общем виде.

3) Вычисление корней уравнений разными методами.

4) Печать результатов вычисления в виде таблицы.

# ОПИСАНИЕ ПЕРЕМЕННЫХ

|  |  |  |
| --- | --- | --- |
| Название переменной | Тип | Описание |
| left | double | Переменная, обозначающая левую границу вычислений |
| right | double | Переменная, обозначающая правую границу вычислений |
| eps | double | Машинное эпсилон |
| d14, d15 | double | Переменные, равные значению корня, вычисленного методом дихотомии |
| i14, i15 | double | Переменные, равные значению корня, вычисленного методом итераций |
| n14, n15 | double | Переменные, равные значению корня, вычисленного методом Ньютона |
| x | double | Вспомогательная переменная, используемая для возвращения значений вспомогательных функций |

# ФУНКЦИИ

1. double mainf4 (double x)

Возвращает значение функции вида:

=

1. double mainf5 (double x)

Возвращает значение функции вида:

=

1. double f4 (double x)

Возвращает значение функции вида:

=

1. double f5 (double x)

Возвращает значение функции вида:

=

1. double derivativef4 (double x)

Возвращает значение первой производной функции:

*F* '(*x*) =

1. double dirivativef5 (double x)

Возвращает значение первой производной функции:

*F* '(*x*) =

1. double eps ()

Возвращает значение машинного эпсилон.

# ВХОДНЫЕ ДАННЫЕ

Отсутствуют.

# ВЫХОДНЫЕ ДАННЫЕ

Значение машинного эпсилон с точностью до 20 знаков после запятой, таблица, в которой по столбцам будут выводиться решения уравнений разными методами.

# КОД ПРОГРАММЫ

#include <stdio.h>

#include <math.h>

double mainf4(double x)

{

return 3 \* x - 14 + exp(x) - exp(- x);

}

double mainf5(double x)

{

return sqrt(1 - x) - tan(x);

}

double f4(double x)

{

return log(14 + exp(- x) - 3 \* x);

}

double f5(double x)

{

return atan(sqrt(1 - x));

}

double derivativef4(double x)

{

return 3 + exp(x) + exp(- x);

}

double derivativef5(double x)

{

return -1/(2 \* sqrt(1 - x )) - 1/(cos(x) \* cos(x));

}

double epsilon()

{

double eps = 1.0;

while (1.0 + (eps / 2.0) > 1.0) {

eps /= 2;

}

return eps;

}

double dht(double left, double right, double (\*f)(double))

{

double eps = epsilon();

while (fabs(f(left) \* f(right)) >= eps) {

if (f(left) \* f((left + right) / 2.0) > 0) {

left = (left + right) / 2.0;

} else {

right = (left + right) / 2.0;

}

}

return (left + right) / 2.0;

}

double iter(double left, double right, double (\*f)(double))

{

double x, x0, eps = epsilon();

x0 = (left + right) / 2.0;

x = right;

while (fabs(x0 - x) >= eps) {

x0 = x;

x = f(x0);

}

return x;

}

double newton(double left, double right, double (\*f)(double), double (\*df)(double))

{

double x, eps = epsilon();

x = (left + right) / 2.0;

while (fabs(f(x) / df(x)) >= eps) {

x = x - (f(x) / df(x));

}

return x;

}

int main(void)

{

double eps, d4, d5, i4, i5, n4, n5;

d4 = dht(1.0, 3.0, \*mainf4);

d5 = dht(0.0, 1.0, \*mainf5);

i4 = iter(1.0, 3.0, \*f4);

i5 = iter(0.0, 1.0, \*f5);

n4 = newton(1.0, 3.0, \*mainf4, \*derivativef4);

n5 = newton(0.0, 1.0, \*mainf5, \*derivativef5);

eps = epsilon();

printf("Машинное эпсилон: %.20f \n", eps);

printf("| Выражение \t\t\t\t| Метод дихотомии \t| Метод итераций \t| Метод Ньютона \t|\n"); printf("-----------------------------------------------------------------------------------------------------------------\n");

printf("| 3 \* x - 14 + exp(x) - exp(-x) \t| %.15lf \t| %.15lf \t| %.15lf \t|\n", d4, i4, n4);

printf("| sqrt(1 - x) - tan(x) \t \t \t| %.15lf \t| %.15lf \t| %.15lf \t|\n", d5, i5, n5);

}

# РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

artoy@artem-trumpeter:~$ gcc -Wall -pedantic -std=c99 kp4.c -lm

artoy@artem-trumpeter:~$ ./a.out

Машинное эпсилон: 0.00000000000000022204

| Выражение | Метод дихотомии | Метод итераций | Метод Ньютона |

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

| 3 \* x - 14 + exp(x) - exp(-x) | 2.069218197837472 | 2.069218196373615 | 2.069218196373615

| sqrt(1 - x) - tan(x) | 0.576769810169935 | 0.576769807570752 | 0.576769807570752

# ВЫВОД

Написанная на языке СИ программа решает заданные алгебраические уравнения методом Ньютона, методом итераций и методом половинного деления – дихотомии. Для обеих функций работают все методы решения.