# Московский Авиационный Институт (национальный исследовательский университет)

# Факультет Компьютерные науки и прикладная математика Кафедра вычислительной математики и программирования

# КУРСОВАЯ РАБОТА по курсам «Архитектура компьютера», «Фундаментальная информатика»

# I семестр Задание 4. Процедуры и функции в качестве параметров

# Студент: Галкин А.Д. Группа: М8О-108Б-22, № 3 по списку Руководитель: Сахарин Н.А. Оценка: Дата:

# Москва, 2022

**Содержание**

1. **ЗАДАЧА** …………………………………………………………………...…. 3

2. **ОБЩИЙ МЕТОД РЕШЕНИЯ** …………………………………………… 3

3. **ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ** …………………………………. 3

4. **ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ** …………………………………. 4

5. **ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ** …………………………… 4

6. **ОПИСАНИЕ ПЕРЕМЕННЫХ, КОНСТАНТ**……………………………. 4

7. **ПРОТОКОЛ** ………………………………………………………………….. 5

8. **ВХОДНЫЙ ДАННЫЕ** ……………………………………………………..... 7

9. **ВЫХОДНЫЕ ДАННЫЕ** ……………………………………………...…... 7

10. **ВЫВОДЫ** ………………………………………………………………….. 9

**1. Задача**

Составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона и половинного деления — дихотомии).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Уравнение | Отрезок, содержащий корень | Базовый метод | Приближенное значение корня |
| 3 |  | [1,1.5] | итераций | 1.1474 |
| 2 |  | [1,2] | дихотомии | 1.0804 |

**2. Общий метод решения**

Идея метода итераций заключается в замене исходного уравнения F(x) = 0 уравнением вида x = f(x). Достаточное условие сходимости метода . Это условие необходимо проверить перед началом решения задачи, так как функция может быть выбрана неоднозначно, причём в случае неверного выбора указанной функции метод расходится. Начальное приближение корня: . Итерационный процесс: . Условие окончания: . Приближённое значение корня: .

Метод Ньютона является частным случаем метода итераций. Условие сходимости метода: на отрезке [a, b]. Итерационный процесс: .

Идея метода дихотомии заключается в делении отрезка пополам и его сужении в два раза на каждом шаге итерационного процесса в зависимости от знака функции в середине отрезка. За начальное приближение принимаются границы исходного отрезка a(0) = a, b(0) = b. Далее вычисления проводятся по формулам: , b(k+1) = b(k), если F(a(k)) \* F((a(k) + b(k))/2)> 0; или по формулам , a(k+1) = a(k), , если F(b(k)) \* F((a(k) + b(k))/2)> 0. Процесс повторяется до тех пор, пока не будет выполнено условие окончания .

**3. Общие сведения о программе**

Аппаратное обеспечение: Компьютер

Операционная система: Linux Ubuntu, версия 22.04.1 LTS

Язык и система программирования: С, GNU

Число строк: 76

Местонахождение файлов: /home/Alexey

Имя файла: KP4.c

Компиляция программы: gcc –std=c18 KP4.c - lm

Вызов программы: ./a.out

**4. Функциональное назначение**

Программа предназначена для решения трансцендентных алгебраических уравнений различными численными методами.

**5. Описание логической структуры**

Программа получает на вход требуемый отрезок, далее находит корень уравнения F(x) = 0 различными методами и выводит значение на экран.

**6. Описание переменных, констант**

Назовём double dbl, а long double ldbl

Таблица 1. Описание функций программы

|  |  |  |
| --- | --- | --- |
| Функция | Входные аргументы | Описание |
| mach\_eps | - | Функция для подсчета машинного ε. Сравниваем 1+ε с 1. Последнее число, при стремлении ε к нулю, при котором 1+ε > 1 и будет машинным ε |
| func | dbl x | 1 функция, данная по условию |
| func\_2 | dbl x | 2 функция, данная по условию |
| f | dbl x | Аргумент 1 функции |
| f\_2 | dbl x | Аргумент 2 функции |
| DF | dbl x | Производная функции |
| DF\_2 | dbl x | Производная 2 функции |
| DDF | dbl x | Производная второго порядка |
| DDF\_2 | dbl x | Производная второго порядка |
| Iteration | dbl f(dbl), dbl Df(dbl), dbl a, dbl b | Метод Итерации |
| Newton | dbl func(dbl), dbl DF(dbl),dbl DDF(dbl), dbl a, dbl b | Метод Ньютона |
| Dichotomy | dbl func(dbl), dbl a, dbl b | Метод дихотомии |

Таблица 2. Описание переменных

|  |  |
| --- | --- |
| Переменная | Значение |
| Ldbl eps | Машинный эпсилон |
| dbl a, b | Границы отрезка |
| dbl x | Значение аргумента функции |

**7. Протокол**

Код программы:

#include <stdio.h>

#include <math.h>

typedef double dbl;

typedef long double ldbl;

ldbl mach\_eps();

dbl func (dbl x);

dbl func\_2 (dbl x);

dbl Df(dbl x);

dbl DF (dbl x);

dbl DDF (dbl x);

dbl Df\_2 (dbl x);

dbl DF\_2 (dbl x);

dbl DDF\_2 (dbl x);

dbl f (dbl x);

dbl f\_2 (dbl x);

dbl Iteration(dbl f(dbl), dbl Df(dbl), dbl a, dbl b);

dbl Newton(dbl func(dbl), dbl DF(dbl), dbl DDF(dbl), dbl a, dbl b);

dbl Dichotomy(dbl func(dbl), dbl a, dbl b);

int main(){

dbl a = 1, b = 1.5, a2 = 1, b2 = 2;

printf("Iteration method result for func\_2: %f\n", Iteration(f\_2, Df\_2, a2, b2));

printf("Newton method result for func\_2: %f\n", Newton(func\_2, DF\_2, DDF\_2, a2, b2));

printf("Dichotomy method result for func\_2: %f\n", Dichotomy(func\_2, a2, b2));

return 0;

}

ldbl mach\_eps(){

ldbl eps = 1;

while (1 + eps / 2.0 > 1) {

eps /= 2.0;

}

return eps;

}

dbl func (dbl x){

return (1 - x + sin(x) - log(1+x));

}

dbl func\_2 (dbl x){

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

}

dbl f (dbl x){

return (1 + sin(x) - log(1+x));

}

dbl f\_2 (dbl x){

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

}

dbl Df (dbl x){

return (cos(x) - (1/(1+x)));

}

dbl Df\_2 (dbl x){

return (sin(x) - x\*exp(-(x\*x)/2));

}

dbl DF (dbl x){

return (-1 + cos(x) - (1/(1+x)));

}

dbl DDF (dbl x){

return (-sin(x) + (1 / pow(x+1,2)));

}

dbl DF\_2 (dbl x){

return (-sin(x)+x\*exp(-(x\*x)/2) + 1);

}

dbl DDF\_2 (dbl x){

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

}

dbl Iteration(dbl f(dbl), dbl Df(dbl), dbl a, dbl b){

ldbl eps = mach\_eps ();

dbl x = a + (b - a) / 2;

if (fabs(Df(x)) < 1){

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

x = f(x);

}

}

return x;

}

dbl Newton(dbl func(dbl), dbl DF(dbl), dbl DDF(dbl), dbl a, dbl b){

ldbl eps = mach\_eps ();

dbl x = a + (b - a) / 2;

dbl x2 = x - (func(x) / DF(x));

if (fabs(func(a)\*DDF(a)) < pow(DF(a),2)) {

while (fabs(x - x2) >= eps){

x = x2;

x2 = x - (func(x) / DF(x));

}

return x2;

}

else{

return NAN;

}

}

dbl Dichotomy(dbl func(dbl), dbl a, dbl b) {

ldbl eps = mach\_eps ();

dbl x = (a + b) / 2;

if (func(a) \* func(b) < 0){

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

x = (a + b) / 2;

if (func(x) \* func(a) < 0) {

b = x;

}

else {

a = x;

}

}

return x;

}

else {

return NAN;

}

}**8. Входные данные**

Отсутствуют.

**9. Выходные данные**

**Протокол**

alexey@alexey-VirtualBox:~$ gcc -std=c18 3.c -lm

alexey@alexey-VirtualBox:~$ ./a.out

Iteration method result for func\_2: 1.089443

Newton method result for func\_2: 1.089443

Dichotomy method result for func\_2: 1.089443

**10. Вывод**

В ходе выполнения работы была составлена программа на языке Си с процедурами решения трансцендентных алгебраических уравнений, были изучены различные методы для вычисления приближенного значения корней.