МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



**ОТЧЁТ**

**по лабораторной работе № 1**

**«Методы вычисления корней*»***

по дисциплине:

**«Программирование»**

|  |  |
| --- | --- |
| Выполнил(а):  Дашкин С.М. Группа АТ-34 | Проверил:  Кухто А.В. |

Новосибирск

2024

**ЦЕЛЬ РАБОТЫ:**

6 вариант. Найти все корни уравнения на заданном интервале согласно варианту. 

Для этого:

1. Отделить корни:
   1. вычислить таблицу значений функции на интервале [X\_min, X\_max] с шагом h;
   2. программно определить отрезки, на концах которых функция меняет знак;
2. На каждом отрезке уточнить корень с точностью eps = 0.001. Метод уточнения для каждого отрезка выбрать самостоятельно при условии, что все методы различны. Для каждого метода вывести число итераций. Если корень всего один, то применить два метода.

**ХОД РАБОТЫ:**

Данный код содержит несколько функций и основную функцию main(), которая использует эти функции для решения уравнения и вывода результатов.

1. Функция double f(double x):

- Описание: Эта функция определяет уравнение, которое мы хотим решить. В данном случае, она возвращает значение функции f(x) = e^{-x} - (0.5 + \sqrt{|x|}) для заданного значения x.

- Входные параметры: x - значение переменной x, для которого нужно вычислить значение функции.

- Выходное значение: Значение функции f(x).

2. Функция double bisection(double a, double b):

- Описание: Эта функция реализует метод половинного деления для нахождения корня уравнения на заданном интервале [a, b].

- Входные параметры: a и b - границы интервала, на котором мы ищем корень уравнения.

- Выходное значение: Значение корня уравнения, найденное методом половинного деления.

3. Функция double hords(double a, double b):

- Описание: Эта функция реализует метод хорд для нахождения корня уравнения на заданном интервале [a, b].

- Входные параметры: a и b - границы интервала, на котором мы ищем корень уравнения.

- Выходное значение: Значение корня уравнения, найденное методом хорд.

4. Функция int main():

- Описание: Это основная функция программы. Она определяет интервал [a, b], шаг h и использует функции f(x), bisection(a, b) и hords(a, b) для нахождения корней уравнения и вывода результатов.

- Входные параметры: Нет.

- Выходное значение: Целое число, обычно 0, которое указывает на успешное выполнение программы.

Также в коде есть закомментированные функции kasat(double x0) и iterative(double x0), которые реализуют метод касательных и метод итераций соответственно. Они не используются в основной функции main(), так как у нас всего 1 корень, но могут быть раскомментированы и использованы для решения уравнения(метод итерации не работает).

**ЛИСТИНГ:**

#include <stdio.h>

#include <math.h>

#define EPS 0.001

double f(double x) {

return exp(-x) - (0.5 + sqrt(fabs(x)));

}

double bisection(double a, double b) { //метод половинного деления

int it = 0;

double x;

while ((b - a) >= EPS) {

x = (a + b) / 2.0; //делим отрезок на 2

if (f(x) == 0.0) // если есть корень - прерываем цикл

break;

//если нету, сужаем интервал поиска

else if (f(x) \* f(a) < 0)

b = x;

else

a = x;

it++;

}

printf("M1:cout iterations = %d, X = %lf\n", it, x);

return x;

}

double hords(double a, double b) { //метод хорд

int it = 0;

double x;

do {

x = (a \* f(b) - b \* f(a)) / (f(b) - f(a)); //уравнение прямой хорды, проходящей через крайние точки отрезка [a, f(a)] и [b, f(b)]

if (f(x) == 0.0) //опять же, если нашли корень - прерываем

break;

else if (f(x) \* f(a) < 0)// нет - уточняем до того момента, пока не дойдём до условия eps

b = x;

else

a = x;

it++;

} while (fabs(f(x)) < EPS);

printf("M2:cout iterations = %d, X = %lf\n", it, x);

return x;

}

/\*double kasat(double x0) { //метод касательных

int it = 0;

double x = x0;

do {

x0 = x; // елси abs(f(x))>=eps

x = x0 - f(x0) / (exp(-x0) + 0.5 / (2 \* sqrt(fabs(x0)))); //пересечение первой касательной с осью Ox в теле цикла

it++;

} while (fabs(x) < EPS);

printf("M3:cout iterations = %d, X = %lf\n", it, x);

return x;

}

double iterative(double x0) { // Метод итераций

int it = 0;

double x = x0;

printf("iter: x0 = %f\n", x0);

do {

x0 = x;

x = exp(-x0) + 0.5 / (2 \* sqrt(fabs(x0)));

it++;

} while (fabs(x-x0) > EPS); // ?????

printf("M4:cout iterations = %d, X = %lf\n", it, x);

return x;

}\*/

int main() {

double a = -1.0, b = 1.0, h = 0.05;

double x = a, fx\_prev = f(a);

printf("Tablica znachenii:\n");

for (double xx = a; xx <= b; xx += h) {

double result = f(xx);

printf("x = %.2f, y = %.6f\n", xx, result);

// Проверка на корень уравнения

if (fabs(result) < 0.09) {

printf("X-border = %.2f\n", xx);

//return 0;

}

}

while (x < b) {

x += h;

double fx = f(x);

if (fx \* fx\_prev < 0) {

printf("[%lf, %lf]:\n", x - h, x);

bisection(x - h, x);

hords(x - h, x);

/\*\*kasat((x - h + x) / 2.0);

iterative((x - h + x) / 2.0);\*/

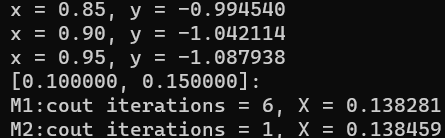
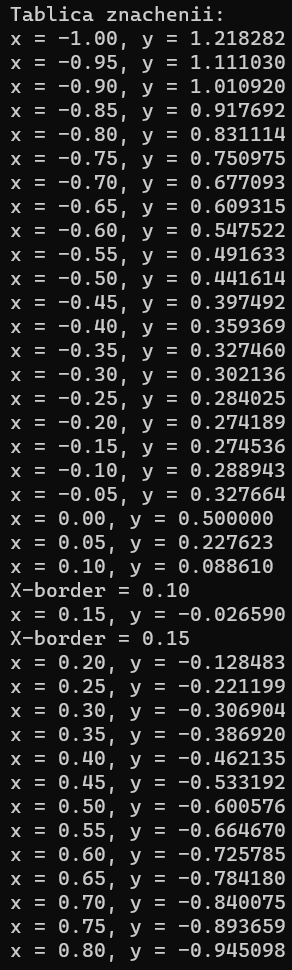
}

fx\_prev = fx;

}

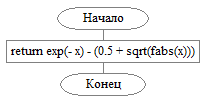
}

**ТЕСТИРОВАНИЕ:**

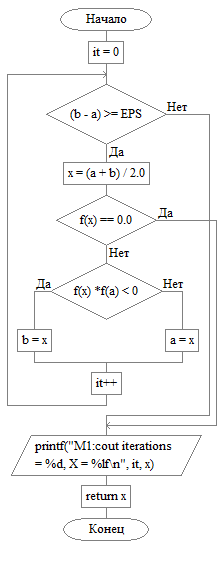
****

**БЛОК СХЕМЫ:**

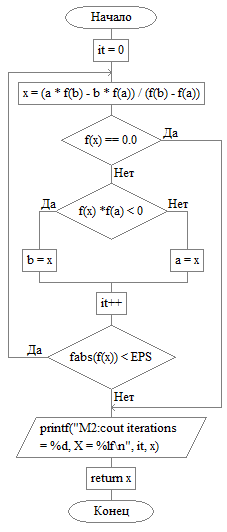
Блок схема функции f:



Блок схема функции bisection:



Блок схема функции hords:



Блок схема функции main:

