Московский авиационный институт

(Национальный исследовательский университет)

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра «Вычислительная математика и программирование»

**Курсовой проект**

**по курсу**

**«Фундаментальная информатика»**

**I семестр**

**Задание 4**

**«Процедуры и функции в качестве параметров»**

Студент: Горин Н.А.

Группа: М8О-113Б-22

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022 г.

Оглавление

[Задача 3](#_Toc122505329)

[Идея решения 3](#_Toc122505330)

[Протокол 3](#_Toc122505331)

[Вывод 5](#_Toc122505332)

# Задача

Составить программу на Си с процедурами решения трансцендентных алгебраических уравнений тремя методами:

* Итераций;
* Ньютона;
* Дихотомии

Вариант:



# Идея решения

Создать вспомогательные функции, которые будут возвращать значения функций из вариантов 6 и 7. Назовем их f6 и f7. Также вручную вычислим производные функций и назовем их соответственно dev6 и dev7. Выразим x. Назовем эти функции f6\_getx и f7\_getx. Тажке создадим отдельные функции для каждого из методов(дихотомии, Ньютона и итераций), в качестве параметров которых будут вышеперечисленные функции.

# Протокол

user@user-Katana:~$ gcc KP4.c -lm

user@user-Katana:~$ .cat KP4.c

/\* Курсовой проект по фундаментальной информатике

Задание 4, Вариант 6,7

М8О-113Б-22, Горин Никита

\*/

#include <stdio.h>

#include <locale.h>

#include <math.h>

const double EPS = 1e-7;

const double e = 2.718281828;

double f6(double x) {//Функция из варианта 6

return x + cos(pow(x, 0.52) + 2);

}

double f7(double x) {//Функция из варианта 7

return 3 \* pow(log(x), 2) + 6 \* log(x) - 5;

}

double dev6(double x) {//Производная f6

return 1 - (13 \* sin(pow(x, 0.52) + 2)) / (25 \* pow(x, 0.48));

}

double dev7(double x) {//Производная f7

return (6 \* log(x) + 6) / x;

}

double f6\_getx(double x) {//Выражение х из f6

return -cos(pow(x, 0.52) + 2);

}

double f7\_getx(double x) {//Выражение x из f7. Для метода простых итераций сходится

return pow(e, (5/(3\*log(x) + 6)));

}

//Метод дихотомии

double dichotomy(double function(double), double left, double right) {

double result;

while (fabs(left - right) > EPS) {

result = (right + left) / 2;

if (function(left) \* function(result) > 0)

left = result;

else

right = result;

}

return result;

}

//Метод итераций

double iteration(double f(double), double a, double b) {

double prevX = (a + b) / 2., x = f(prevX);

while (fabs(x - prevX) > EPS) {

prevX = x;

x = f(x);

}

return x;

}

//Метод Ньютона

double newton(double F(double), double F1(double), double a, double b, double eps) {

double x = (a + b / 2);

while (fabs(F(x) / F1(x)) > eps) {

x -= F(x) / F1(x);

}

return x;

}

int main() {

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

printf("| Уравнение | Отрезок | Метод | Результат |\n");

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

printf("| \t 1 | [0.5;1] | Дихотомии | %.10f |\n", dichotomy(f6, 0.5, 1));

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

printf("| \t 1 | [0.5;1] | Ньютона | %.10f |\n", newton(f6, dev6, 0.5, 1, EPS));

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

printf("| \t 1 | [0.5;1] | Итераций | %.10f |\n", iteration(f6\_getx, 0.5, 1));

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

printf("| \t 2 | [1;3] | Дихотомии | %.10f |\n", dichotomy(f7, 1, 3));

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

printf("| \t 2 | [1;3] | Ньютона | %.10f |\n", newton(f7, dev7, 1, 3, EPS));

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

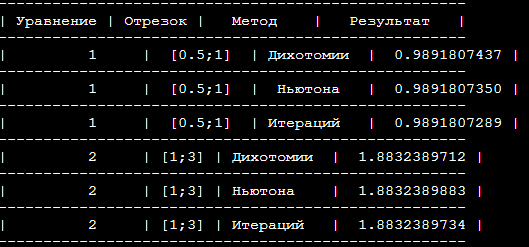
printf("| \t 2 | [1;3] | Итераций | %.10f |\n", iteration(f7\_getx, 1, 3));

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

return 0;

}

user@user-Katana~$ ./a.out



# Вывод

Таблица показывает, что приближенно все три корня совпали с исходными, следовательно, все три метода (дихотомии, Ньютона и итераций) достаточно точно работают для поиска корней на заданных отрезках.

Я научился писать программы на языке Си для поиска корней на заданных отрезках с помощью трех различных методов.