Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

**ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №3**

з дисципліни:« Інформатика1. Основи програмування та алгоритми»

|  |  |
| --- | --- |
|  | Виконав: Поляков Владислав  Група: РЕ-12  Викладач: Доцент Катін Ю.П |

Київ – 2021

Мета роботи: Створити програму для розв’язку не лінійного рівняння трьома методами(методом хорд, методом половинного ділення та методом Ньютона).

Обрана функція:

Похідна від функції:

Обраховане вручну значення кореня:

Основна блок-схема:

Код:

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <math.h>**

**#include <conio.h>**

**#include <time.h>**

**double F(double x)//функция1**

**{**

**return pow(2\*x-3,2)-4\*x+7;**

**}**

**double f(double x)//функция2**

**{**

**return (x-2)/2;**

**}**

**void info(double XI,double fault,int n,double Timer)**

**{**

**printf("XI = %lf\n",XI);**

**printf("Fault = %lf\n",fault);**

**printf("Count of iterations = %d\n",n);**

**printf("Calculation time = %lf\n\n",Timer);**

**}**

**void test(int quantity){**

**for( int i=0; i<quantity; i++ ){**

**printf("%d", "Hello World");**

**}**

**}**

**int main()**

**{**

**//My function**

**double x1, x2,XI,timer,fault;**

**unsigned int N, n = 0,var;**

**char i;**

**for(int q=0;q<=3;q++)**

**{**

**printf("1 (Chord method) | 2 (Half-Chord method) | 3 (Newton`s method(tangent method))\nChoose:");**

**scanf("%d",&var);**

**switch(var)**

**{**

**case 1://хорды**

**do**

**{**

**printf("Left limit: \n");**

**scanf("%lf",&x1);**

**printf("Right limit: \n");**

**scanf("%lf",&x2);**

**}**

**while(F(x1)\*F(x2)>0);**

**printf("Fault: \n");**

**scanf("%lf",&fault);**

**printf("Max count of iterations: \n");**

**scanf("%d",&N);**

**clock\_t T1 = clock();**

**do**

**{**

**XI=((F(x2)\*x1) - (F(x1)\*x2))/(F(x2) - F(x1));**

**if( F(XI) > 0 )**

**{**

**x2=XI;**

**}**

**else**

**{**

**x1=XI;**

**}**

**if(n==N)**

**{**

**printf("ESC to quit\nany key to continue\n");**

**i = getch();**

**if((int)i==27)**

**{**

**system("cls");**

**break;**

**}**

**else**

**{**

**system("cls");**

**}**

**}**

**n++;**

**}**

**while ( fabs (F(XI)) > fault );**

**clock\_t T2 = clock();**

**info(XI,fabs (F(XI)),n,(double)(T2 - T1) / CLOCKS\_PER\_SEC);**

**break;**

**case 2:**

**do**

**{**

**printf("Left limit: \n");**

**scanf("%lf",&x1);**

**printf("Right limit: \n");**

**scanf("%lf",&x2);**

**}**

**while(F(x1)\*F(x2)>0);**

**printf("Fault: \n");**

**scanf("%lf",&fault);**

**printf("Max num of iterations: \n");**

**scanf("%d",&N);**

**clock\_t T11 = clock();**

**do**

**{**

**XI=(x1+x2)/2;**

**if( F(XI) > 0 )**

**{**

**x2=XI;**

**}**

**else**

**{**

**x1=XI;**

**}**

**if(n==N)**

**{**

**printf("ESC to quit\nany key to continue\n");**

**i = getch();**

**if((int)i==27)**

**{**

**system("cls");**

**break;**

**}**

**else**

**{**

**system("cls");**

**}**

**}**

**n++;**

**}**

**while ( fabs (F(XI)) > fault );**

**clock\_t T22 = clock();**

**info(XI,fabs (F(XI)),n,(double)(T22 - T11) / CLOCKS\_PER\_SEC);**

**break;**

**case 3:**

**printf("X: \n");**

**scanf("%lf",&x1);**

**printf("Fault: \n");**

**scanf("%lf",&fault);**

**printf("Max count of iterations: \n");**

**scanf("%d",&N);**

**clock\_t T3 = clock();**

**do**

**{**

**XI = x1 - f(x1);**

**if( fabs(F(XI))< fabs(F(x1)) )**

**{**

**x1 = XI;**

**}**

**if(n==N)**

**{**

**printf("Max num of iterations has been reached\n");**

**printf("Press ESC to finish the calculation \n");**

**i = getch();**

**if((int)i==27)**

**{**

**system("cls");**

**break;**

**}**

**else**

**{**

**system("cls");**

**}**

**}**

**n++;**

**}**

**while ( fabs (F(XI)) > fault );**

**clock\_t T4 = clock();**

**info(XI,fabs (F(XI)),n,(double)(T4 - T3) / CLOCKS\_PER\_SEC);**

**break;**

**case 4:**

**printf("%d", "case 4");**

**break;**

**default:**

**printf("\nError, try again\n\n");**

**}**

**}**

**system("cls");**

**return 0;**

**}**