Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра теоретичних основ радіотехніки

**ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ №3**

з дисципліни: «Інформатика 1»

|  |  |
| --- | --- |
|  | Виконав: Бурлака Тимофій  Група: РЕ-11  Викладачі: доцент Катін П.Ю.  Оцінка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Підпис: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ – 2021

**Мета роботи**: розробка програми розв’язання нелінійних рівнянь

**Ключові моменти**:

1)Для того щоб отримати час затрачений на знаходження кореня я використовую бібліотеку <time.h> та функцію clock() яка повертає значення часу у даний момент.

Пишу на початку обчислень clock\_t T1 = clock();

Функція clock() повертає час, пройдений з моменту старту програми, що визвала функцію clock().

2) За допомогою getch() записую у змінну і значення яке ввів користувач з клавіатури.

i = getch();

3) Для того щоб зчитати з клавіатури ESC я використовую таблицю Аскі.

if((int)i==27) – цим я перевію, чи дорівнює значення І == 27, бо саме 27 означає ESC

4)У кінці знаходження кореня я пишу clock\_t T2 = clock().Так я записую у змінну Т2 значення часу у даний момент, і щоб отримати значення за який час програма знайшла корінь я пишу:

(double)(T2 - T1) / CLOCKS\_PER\_SEC

CLOCKS\_PER\_SEC це змінна задана у бібліотеці <time.h> та дорівнює 1000.

Код:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <conio.h>

#include <time.h>

double F(double x)

{

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

}

void Fun(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",Timer);

}

int main()

{

//My function y=pow(x,3)-2\*pow(x,2)+2\*x+2

double x1, x2,xi,timer,fault;

unsigned int N, n = 0,var;

char i;

printf("1 - chord method\n2 - half chord method\n");

scanf("%d",&var);

switch(var)

{

case 1:

do

{

printf("Left limit: \n");

scanf("%lf",&x1);

printf("Right limit: \n");

scanf("%lf",&x2);

}

while(F(x1)>0 && F(x2)<0);

printf("Fault: \n");

scanf("%lf",&fault);

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

clock\_t T1 = clock();

scanf("%d",&N);

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("The maximum number of iterations is reached\n");

printf("Press ESC to finish the calculation \nor press any key to continue the calculation\n");

i = getch();

if((int)i==27)

{

system("cls");

break;

}

else

{

system("cls");

}

}

n++;

}

while ( fabs (F(xi)) > fault );

clock\_t T2 = clock();

Fun(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)>0 && F(x2)<0);

printf("Fault: \n");

scanf("%lf",&fault);

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

clock\_t T11 = clock();

scanf("%d",&N);

do

{

xi=(x1+x2)/2;

if( F(xi) > 0 )

{

x2=xi;

}

else

{

x1=xi;

}

if(n==N)

{

printf("The maximum number of iterations is reached\n");

printf("Press ESC to finish the calculation \nor press any key to continue the calculation\n");

i = getch();

if((int)i==27)

{

system("cls");

break;

}

else

{

system("cls");

}

}

n++;

}

while ( fabs (F(xi)) > fault );

clock\_t T22 = clock();

Fun(xi,fabs (F(xi)),n,(double)(T22 - T11) / CLOCKS\_PER\_SEC);

break;

}

return 0;

}