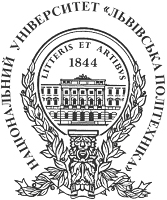
Міністерство освіти і науки, молоді та спорту України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення



**ЗВІТ**

**Про виконання лабораторної роботи № 2**

«Розв’язування нелінійних рівнянь методом дотичних та методом послідовних наближень»

**з дисципліни «Чисельні методи»**

**Лектор:**

доцент кафедри ПЗ

Мельник Н.Б.

**Виконав:**

студ. групи ПЗ-16

Шеремета А.І.

**Прийняла:**

асистент кафедри ПЗ

Бутрак І. О.

«\_\_\_» \_\_\_\_\_\_\_\_2022 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2022

**Тема роботи:** Розв’язування нелінійних рівнянь методом дотичних та методом послідовних наближень.

**Мета роботи:** Ознайомлення на практиці з методом дотичних та методом послідовних наближень для розв’язування нелінійних рівнянь

**Індивідуальне завдання**

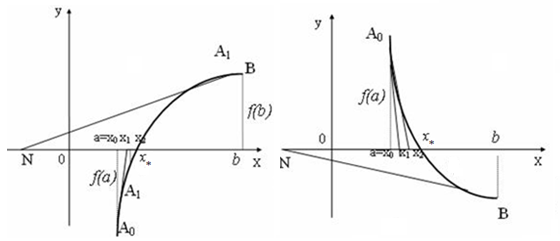
Скласти програму розв’язування нелінійного рівняння методом дотичних та методом простої ітерації.



**Теоретичні відомості**

1. Назвати методи уточнення дійсних коренів нелінійних рівнянь. Метод Ньютона (метод дотичних), метод простої ітерації (метод послідовних наближень). (Метод хорд та метод дихотомії).
2. Навести алгоритм методу Ньютона і геометрично його зобразити. Записати основні ітераційні формули методу Ньютона.

Геометричний змістметоду Ньютона полягає в тому, що дугу кривої на відрізку замінюють дотичною до цієї кривої, а наближене значення кореня визначають як абсцису точки перетину дотичної з віссю *,* проведеної через один із кінців відрізка.



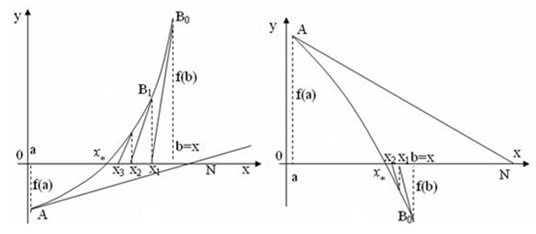


Рис.1. Геометричний зміст методу Ньютона

Основні ітераційні формули методу Ньютона:

* Обчислення абсциси наближення до кореня: ;

1. Показати особливості методу простої ітерації та його обмеження. Графічно зобразити метод простої ітерації та записати основні ітераційні формули методу. Сформулювати умову завершення ітераційного процесу обчислень за методом простої ітерації.

Рівняння треба записати у канонічній формі . Довільним способом визначити наближене значення кореня рівняння і підставити його в праву частину співвідношення. У результаті отримаємо формулу: . Необхідно, щоб виконувалась умова: .

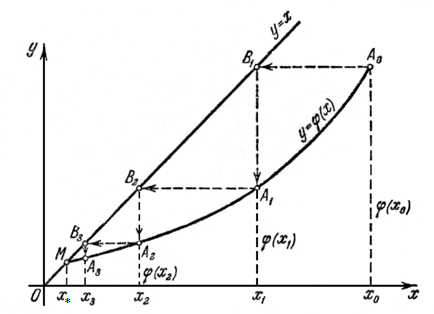


Рис.2. Графічна інтерпретація методу ітерацій

Основні ітераційні формули методу ітерацій:

* Ітераційна формула: .

Умова завершення ітераційного процесу обчислень за методом простої ітерації:

.

**Хід роботи**

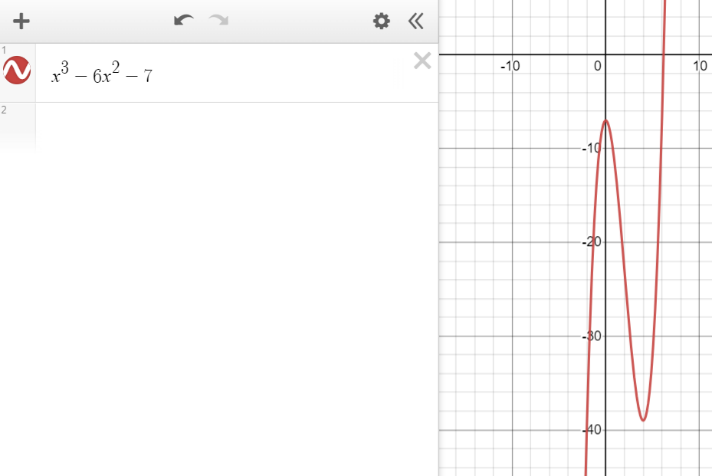


Рис.3. Графік функції

Канонічне рівняння функції:

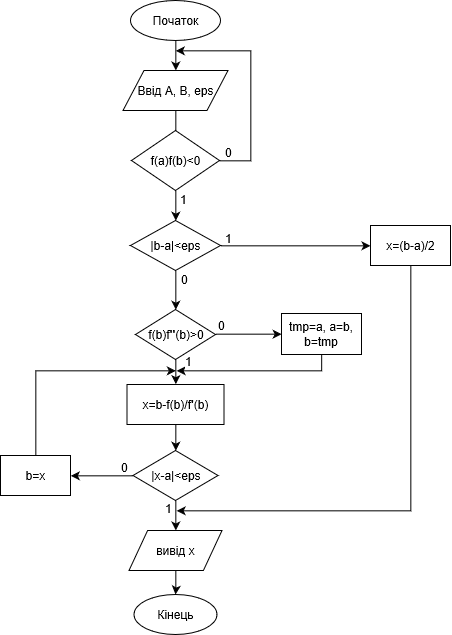


Рис.4. Схема алгоритму розв'язання нелінійного рівняння методом Ньютона

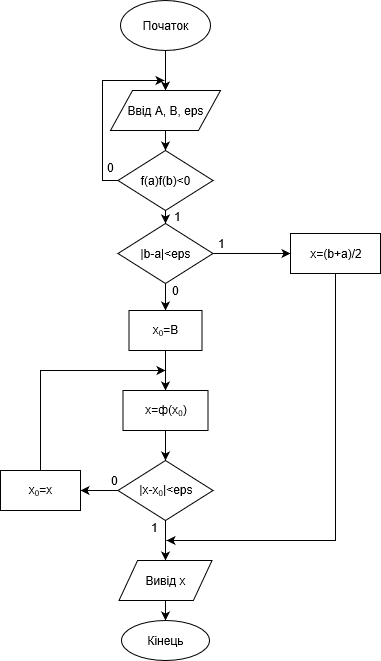


Рис.5. Схема алгоритму розв'язання нелінійного рівняння методом простої ітерації

**Код програми**

#include <iostream>

#include <math.h>

#include <stdlib.h>

#include <iomanip>

#define f(x) (x \* x \* x) - (6 \* x \* x) - 7

#define g(x) (3 \* x \* x)-(12 \* x)

using namespace std;

void NewtonMethod(float a, float e);

void IterationMethod(float a, float b, float e);

double canonical(double x);

double polinom(double x);

double pohidna(double x);

double canonical\_pohidna(double x);

int main(){

    float a, b, e;

    cout << setprecision(7) << fixed;

    cout << "Enter a:";

    cin >> a;

    cout << "Enter b:";

    cin >> b;

    cout << "Enter error:";

    cin >> e;

    cout << endl<< "-------------------" <<endl;

    cout << "Newton method:" << endl;

    cout<< "-------------------" <<endl;

    NewtonMethod(a, e);

    cout << endl<< "-------------------" <<endl;

    cout << "Iterations method:" << endl;

    cout<< "-------------------" <<endl;

    IterationMethod(a, b, e);

    return 0;

}

void NewtonMethod(float a, float e){

    float x1, f0, f1, g0, temp, temp2;

    int step = 1;

    do

     {

          g0 = g(a);

          f0 = f(a);

          if(g0 == 0.0)

          {

               cout<<"Mathematical Error.";

               exit(0);

          }

          x1 = a - f0/g0;

    if(x1 == temp || x1 == temp2){

       cout << "Root found. " << endl;

       break;

      }

          temp = x1;

          a = x1;

          step = step+1;

          if(step > 71)

          {

               cout<<"Not Convergent.";

               exit(0);

          }

          f1 = f(x1);

          if(step%2 == 0){

              temp2 = x1;

          }

     }while(step<71);

     cout<< endl<<"Root is: "<< x1 << endl;

     cout << "Iterations: " << step-1 << endl;

}

void IterationMethod(float a, float b, float e)

{

    short int Sign = 1;

    bool canonicalcheck = true;

    double x0, x1, temp = 0, k, maxd;

    maxd = fabs(pohidna(a));

        while (temp <= b)

        {

            if (fabs(canonical\_pohidna(temp)) > 1)

            canonicalcheck = 0;

            if (fabs(pohidna(temp) > maxd))

            {

                maxd = fabs(pohidna(temp));

                if (pohidna(temp) < 0)

                    Sign = -1;

                else

                    Sign = 1;

            }

            temp = temp + 0.00001;

        }

        int i = 0;

        x1 = a;

        if(!canonicalcheck)

        {

            k = (int)maxd/2;

            if(Sign > 0)

            k = k;

            else

            k = -k;

            do{

                x0 = x1;

                x1 = x0 -(polinom(x0)/k);

                ++i;

            }while (fabs(x0-x1) > e);

        }   else

        {

            do{

                x0 = x1;

                x1 = canonical(x0);

                ++i;

            }while (fabs(x0 - x1) > e);

        }

        cout << endl << "Root is " << x1 << endl;

        cout << "Iterations: " << i << endl;

        }

double canonical(double x){

    return 7/((x\*x) - (6\*x));

}

double polinom(double x){

    return (x \* x \* x) - (6 \* x \* x) - 7;

}

double pohidna(double x){

    return (3\*x\*x) - (12 \* x);

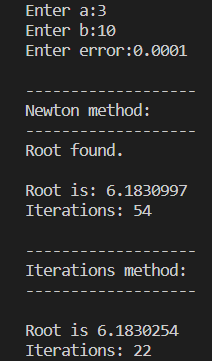
}

double canonical\_pohidna(double x){

    return (3\*x\*x)/12;

}

**Виконання лаборатоної роботи**



**Висновки**

На цій лабораторній роботі я ознайомився на практиці з методом дотичних та методом послідовних наближень для розв’язування нелінійних рівнянь.