НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп’ютерних систем

**Лабораторна робота № 1**

з дисципліни:

«Алгоритми та методи обчислень»

Виконав: Стецюренко І. С,

Студент групи КВ-03

Перевірив(ла):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ-2022

***ЛАБОРАТОРНА РОБОТА №1.***

***НЕЛІНІЙНІ РІВНЯННЯ З ОДНИМ НЕВІДОМИМ***

**Постановка задачі**

1. Розробити програму на мові програмування *С*# у середовищі розробки Visual Studio 2013 (або вище), яка буде працювати у віконному режимі та реалізовувати задані методи розв’язання нелінійних рівнянь (*табл*. 1.1, *табл*. 1.4). Розроблена програма повинна виводити на екран всі проміжні результати.

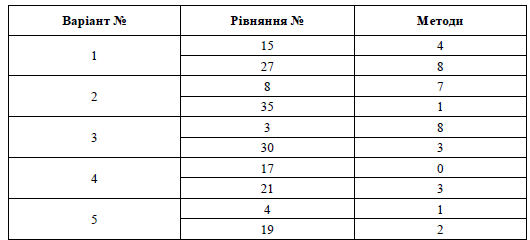
2. За допомогою розробленої програми з п.1 розв'язати задані за варіантом рівняння (*табл*. 1.1, *табл*. 1.2) на заданому проміжку з точністю

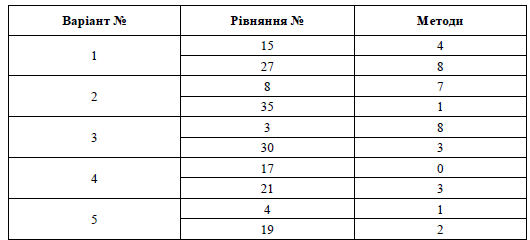
ε ≤ 10^(-7).

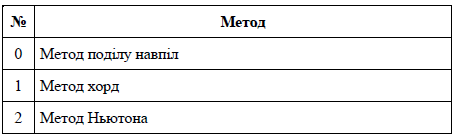
3. При виконанні завдання з п.2 необхідно побудувати графіки залежності наближеного значення кореня рівняння від кількості ітерацій починаючи з початкового наближення. Якщо рівняння має більше двох коренів, то побудувати графіки для двох будь- яких коренів.

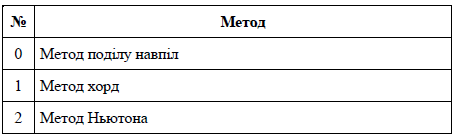
4. Задані за варіантом, рівняння розв’язати у MatLab 7.0 (або вище) або у MathCAD 15.0 (або вище), або у Mathematica 7.0 (або вище). Задане за варіантом, алгебраїчного рівняння необхідно розв’язати, як мінімум двома функціями наявними у відповідному математичному пакеті. Наприклад, в математичному пакеті MatLab 7.0 наявна функція solve для розв’язання будь- якого нелінійного рівняння та функція roots для розв’язання алгебраїчного рівняння.

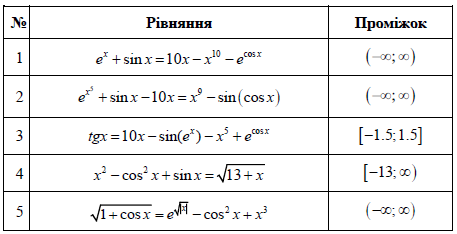
**Завдання для 5 варіанту:**

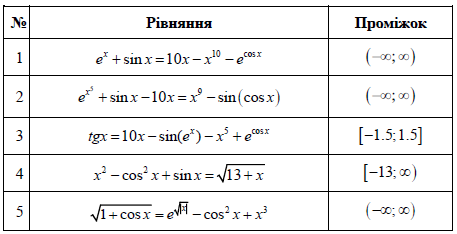
****

****











**Математичне підґрунтя**

***Метод хорд***

Можливо досягти кращих результатів збіжності, якщо відрізок поділити точкою на частини не навпіл, а пропорційно величинам ординат ƒ*(b)* та ƒ*(a)* графіка даної функції ƒ*(x)*:

(1.1)

***Метод Ньютона (метод дотичних)***

Ітераційний процес метода Ньютона визначається наступною формулою

(1.2)

де *k =* 0, 1, 2… та вважається, що на елементах послідовності *(xk)* перша похідна даної функції не дорівнює нулю.

**Теорема 1.4.** *Нехай на відрізку* [*a;b*] *функція* ƒ*(x) має першу (не дорівнює нулю) та другу похідні сталого знаку та нехай*

*Тоді якщо точка* обрана на таким чином, що

(1.3)

*то послідовність (xk)*  *, що починається з x0 та визначається методом Ньютона (1.2), монотонно збігається до кореня ξ ∈* (*a;b*) *рівняння* *виду ƒ(x) = 0.*

**Основні етапи процесу локалізації коренів**

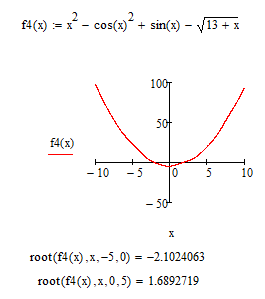
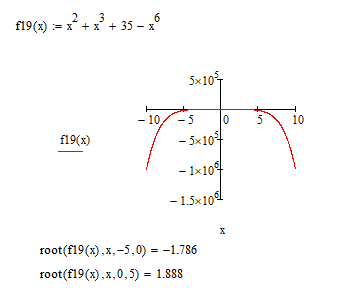
*** ***

Рис.1 Графік та корені функції №4 Рис.2 Графік та корені функції №19

**Значення коренів**

Рівняння №4

|  |  |  |
| --- | --- | --- |
| Метод | Програма | MathCAD15 |
| Метод хорд | -2.1024063 | -2.1024063 |
| Метод хорд | 1,6892719 | 1,6892719 |

Рівняння №19

|  |  |  |
| --- | --- | --- |
| Метод | Програма | MathCAD15 |
| Метод Ньютона | -1,786324 | -1,786324 |
| Метод Ньютона | 1,8880263 | 1,8880263 |

**Графіки** залежності наближеного значення кореня рівняння від кількості ітерацій починаючи з початкового наближення

Рис. 4 Метод хорд, рівняння №4, відрізок [; ]

Рис. 5 Метод простих ітерацій, рівняння №13, відрізок [3; 3.5]

**Висновок**

Виконуючи дану лабораторну роботу я опанував різні методи

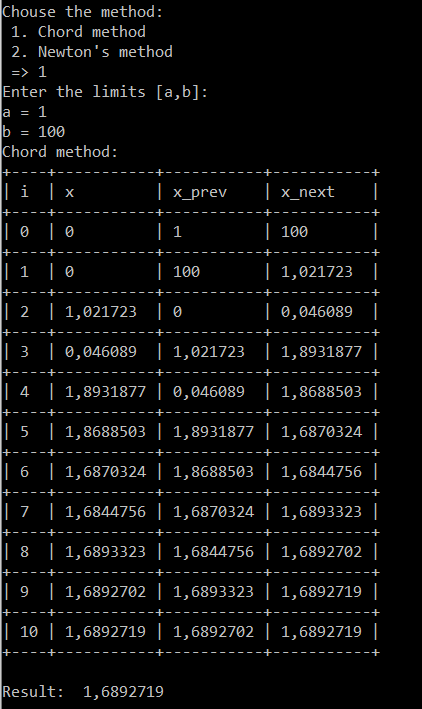
наближеного розв’язання нелінійних рівнянь. Я розробив програму на мові

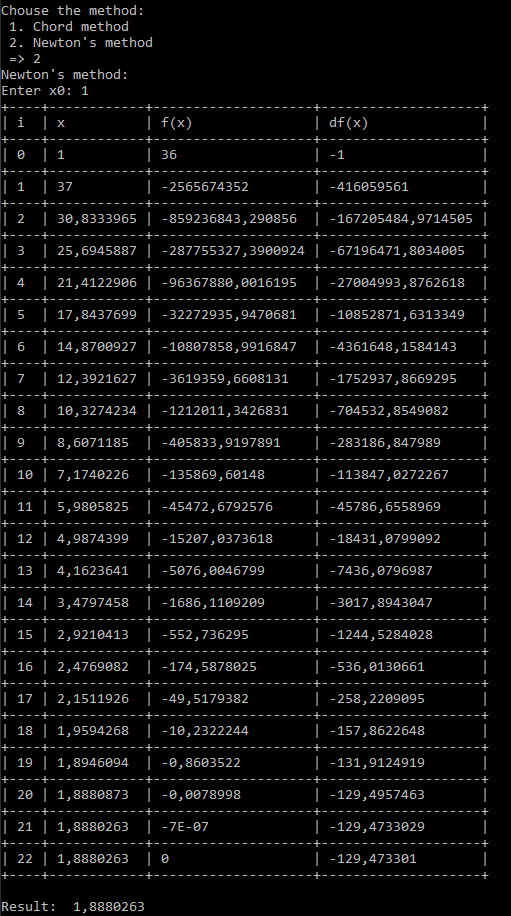
С#, яка реалізує задані за моїм варіантом методи. Результати

програми було перевірено у математичному пакеті MathCAD15.

Порівнюючи кількість ітерацій двох методів можна прийти до

висновку, що метод хорд набагато простіший в реалізації. Метод хорд, на відміну методу Ньютона, має плюс у цьому, що розрахунку не потрібно обчислення похідних, але при цьому метод хорд повільніший.





Код:

**using** System;

**using** System.Collections.Generic;

**using** System.Linq;

**using** System.Text;

**using** System.Threading.Tasks;

**using** ConsoleTables;

**namespace** lab1

{

**class** Program

{

**static** **void** Main(**string**[] args)

{

Console.Write("Chouse the method: **\n** 1. Chord method **\n** 2. Newton's method **\n** => ");

**int** num = Console.Read();

**if** (num == '1')

{

Console.ReadLine();

Console.WriteLine("Enter the limits [a,b]: ");

Console.Write("a = ");

**double** x0 = Convert.ToDouble(Console.ReadLine());

Console.Write("b = ");

**double** x1 = Convert.ToDouble(Console.ReadLine());

**double** e = 0.0000001;

Console.WriteLine("Chord method:");

**double** x = method\_chord(x0, x1, e); *// 1 -3 - done*

*// 2 -1 -done*

Console.WriteLine("Result: " + Math.Round(x,7));

}

**else** **if** (num == '2')

{

**double** e = 0.0000001;

Console.ReadLine();

Console.WriteLine("Newton's method: ");

Console.Write("Enter x0: ");

**double** x0 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Result: " + Math.Round(Newtons\_method(x0, e), 7)); *// 1, -2 -done*

}

}

**public** **static** **double** method\_chord(**double** x\_prev, **double** x\_curr, **double** e)

{

**double** x\_next = 0;

**double** tmp;

**int** i = 1;

**var** table = new ConsoleTable("i", "x", "x\_prev", "x\_next");

table.AddRow(0, 0, x\_prev, x\_curr);

**do**

{

tmp = x\_next;

x\_next = x\_curr - f(x\_curr) \* (x\_prev - x\_curr) / (f(x\_prev) - f(x\_curr));

x\_prev = x\_curr;

x\_curr = tmp;

table.AddRow(i, Math.Round(x\_curr, 7), Math.Round(x\_prev,7), Math.Round(x\_next,7));

i++;

} **while** (Math.Abs(x\_next - x\_curr) > e);

table.Write(Format.Alternative);

**return** x\_next;

}

**public** **static** **double** f(**double** x)

{

**return** Math.Pow(x, 2) - Math.Pow(Math.Cos(x), 2) + Math.Sin(x) - Math.Sqrt(13+x);

}

*// Метод Ньютона*

**public** **static** **double** fx(**double** x)

{

**return** Math.Pow(x, 2) + Math.Pow(x, 3) + 35 - Math.Pow(x, 6);

}

**public** **static** **double** dfx(**double** x)

{

**return** 2 \* x + 3 \* Math.Pow(x, 2) - 6 \* Math.Pow(x, 5);

}

**public** **static** **double** Newtons\_method(**double** x0, **double** e)

{

**var** table = new ConsoleTable("i", "x", "f(x)", "df(x)");

table.AddRow(0, x0, fx(x0), dfx(x0));

**int** i = 1;

**double** xi = x0;

**while** (Math.Abs(fx(xi)) >= e)

{

xi = xi - fx(xi) / dfx(xi);

table.AddRow(i, Math.Round(xi,7), Math.Round(fx(xi),7), Math.Round(dfx(xi),7));

i++;

}

table.Write(Format.Alternative);

**return** xi;

}

}

}