**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**

**«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

Кафедра інформатики та інтелектуальної власності

**ЗВІТИ**

про виконання лабораторних робіт з дисципліни

«Методи та засоби обчислювальної математики»

Варіант 18

Група КН-321в

Виконавець Дмитро ХОМА

Викладач Дмитро ЄЛЬЧАНІНОВ

Харків 2023

**9 ЧИСЕЛЬНІ МЕТОДИ РОЗВ’ЯЗАННЯ НЕЛІНІЙНИХ РІВНЯНЬ**

**9.1 Завдання**

Для заданої функції знайти корінь рівняння на відрізку методами:

– ділення проміжку навпіл;

– хорд;

– дотичних (Ньютона).

Критерій зупинення ітераційного процесу для кожного метода: програма зупиняє роботу, якщо знайдено таке значення , що .

Функція та відрізок для заданого варіанта подані у табл. 9.1.

Таблиця 9.1 – Функція та відрізок для заданого варіанта

|  |  |  |
| --- | --- | --- |
| Варіант | Функція | Відрізок |
| 18 |  |  |

**9.2 Методи розв’язання нелінійних рівнянь**

Нехай функція неперервна на відрізку та приймає на його кінцях різні знаки: . Тоді на відрізку рівняння має хоча би один корінь.

Розглянемо три чисельні методи розв’язання нелінійних рівнянь – уточнення кореня – звуження границь відрізку , поки значення функції в певної точці відрізку не стане меншою, ніж наперед задана точність .

**9.2.1 Метод ділення проміжку навпіл**

Відрізок поділяється навпіл точкою :

Якщо , то точка – наближений корінь рівняння .

Інакше, звужуємо границі відрізку за формулою (9.1):

|  |  |
| --- | --- |
|  | (9.1) |

де – функція, яка визначає знак дійсного числа :

**9.2.2 Метод хорд**

Розглянемо хорду графіка функції на відрізку – пряму, що проходить через точки та . Ця хорда перетинає вісь у точці :

Якщо , то точка – наближений корінь рівняння .

Інакше, звужуємо границі відрізку за формулою (9.1).

**9.2.3 Метод дотичних (Ньютона)**

Розглянемо дотичну до графіка функції в точці . Ця дотична перетинає вісь у точці :

Якщо , то точка – наближений корінь рівняння .

Інакше, розглядаємо точку як точку та повторюємо процес.

**9.3 Коди програм**

Метод ділення проміжку навпіл реалізовано у коді програми мовою Python, як показано на рис. 9.1.

from math import cos, sin

from numpy import sign

import pandas as pd

a = -1

b = 0.5

h = 0.001

*def* F(*x*):

    y = cos(x)\*\*3\*sin(2.1\*x)\*(x\*\*2+x+1)

    return y

c = (a+b)/2

data\_A = [a]

data\_B = [b]

data\_C = [c]

data\_F\_A = [F(a)]

data\_F\_B = [F(b)]

data\_F\_C = [F(c)]

while abs(F(c)) > h:

    if sign(F(c)) == sign(F(a)):

        a = c

    else:

        b = c

    c = (a+b)/2

    data\_A.append(a)

    data\_B.append(b)

    data\_C.append(c)

    data\_F\_A.append(F(a))

    data\_F\_B.append(F(b))

    data\_F\_C.append(F(c))

data = pd.DataFrame({'a': data\_A, 'F(a)': data\_F\_A, 'c': data\_C, 'F(c)': data\_F\_C, 'b': data\_B, 'F(b)': data\_F\_B})

print(data)

print("В точці с",c,',f(c) = ', F(c))

Рисунок 9.1 – Код програми, що реалізує метод ділення проміжку навпіл

Метод хорд реалізовано у коді програми мовою Python, як показано на рис. 9.2.

from math import cos, sin

from numpy import sign

import pandas as pd

a = -1

b = 0.5

h = 0.01

*def* F(*x*):

    y = cos(x)\*\*3\*sin(2.1\*x)\*(x\*\*2+x+1)

    return y

c = (F(a)\*b-a\*F(b))/(F(a)-F(b))

data\_A = [a]

data\_B = [b]

data\_C = [c]

data\_F\_A = [F(a)]

data\_F\_B = [F(b)]

data\_F\_C = [F(c)]

while abs(F(c)) > h:

    if sign(F(c)) == sign(F(a)):

        a = c

    else:

        b = c

    c = (F(a)\*b-a\*F(b))/(F(a)-F(b))

    data\_A.append(a)

    data\_B.append(b)

    data\_C.append(c)

    data\_F\_A.append(F(a))

    data\_F\_B.append(F(b))

    data\_F\_C.append(F(c))

data = pd.DataFrame({'a': data\_A, 'F(a)': data\_F\_A, 'c': data\_C, 'F(c)': data\_F\_C, 'b': data\_B, 'F(b)': data\_F\_B})

print(data)

print("В точці с",c,',f(c) = ', F(c))

Рисунок 9.2 – Код програми, що реалізує метод хорд

Метод дотичних (Ньютона) реалізовано у коді програми мовою Python, як показано на рис. 9.3.

from math import cos, sin

import pandas as pd

c = -0.1

h = 0.001

*def* F(*x*):

    y = cos(x)\*\*3\*sin(2.1\*x)\*(x\*\*2+x+1)

    return y

*def* dF(*x*):

    return cos(x)\*\*2\*((-3)\*(x\*\*2+x+1)\*sin(x)\*sin(2.1\*x)+2.1\*(x\*\*2+x+1)\*cos(x)\*cos(2.1\*x)+(2\*x+1)\*sin(2.1\*x)\*cos(x))

data\_C = [c]

data\_F\_C = [F(c)]

while abs(F(c)) > h:

    c = c - F(c)/dF(c)

    data\_C.append(c)

    data\_F\_C.append(F(c))

data = pd.DataFrame({'c': data\_C, 'F(c)': data\_F\_C})

print(data)

print("В точці с",c,',f(c) = ', F(c))

Рисунок 9.3 – Код програми, що реалізує метод дотичних (Ньютона)

**9.4 Тестування програм**

Результат роботи програми, що реалізує метод ділення проміжку навпіл, показано на рис. 9.4.

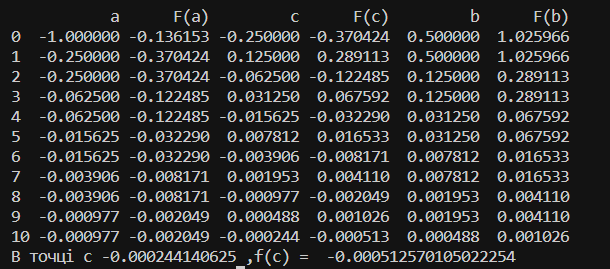


Рисунок 9.4 – Результат роботи програми, що реалізує метод ділення проміжку навпіл

Результати перевірки метода ділення проміжку навпіл засобами сервісу WolframAlpha показано на рис. 9.5.

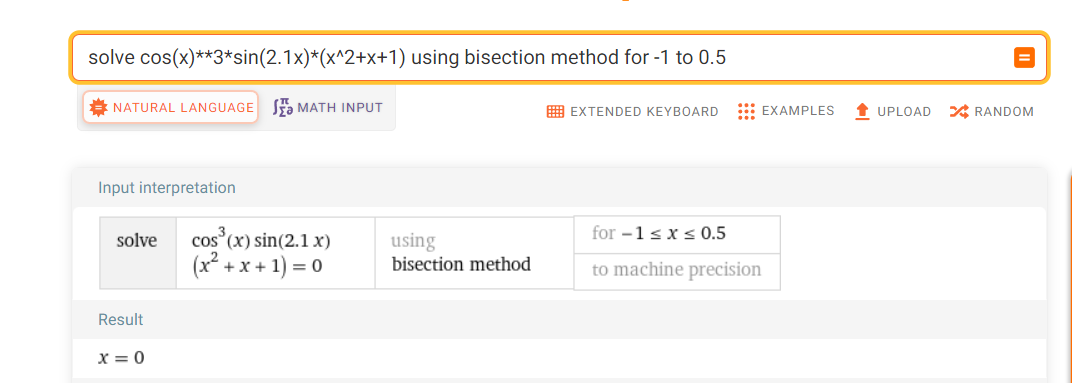


Рисунок 9.5 – Перевірка метода ділення проміжку навпіл

Результат роботи програми, що реалізує метод хорд, показано на рис. 9.6.

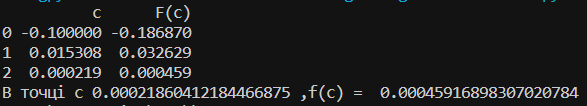


Рисунок 9.6 – Результат роботи програми, що реалізує метод хорд

Результати перевірки метода хорд засобами сервісу [Keisan Online Calculator](https://keisan.casio.com/exec/system/1222999093) показано на рис. 9.7.

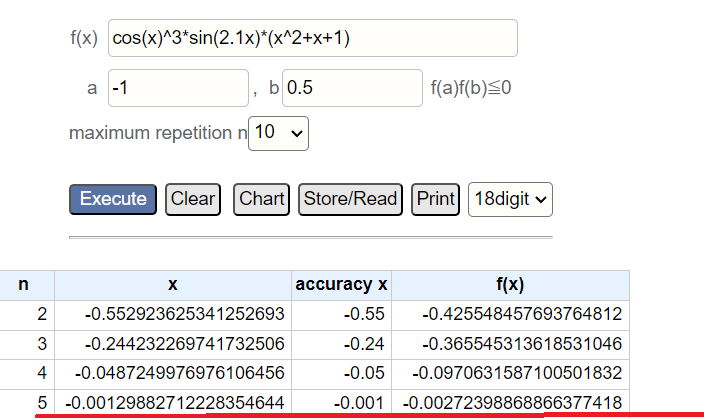


Рисунок 9.7 – Перевірка метода хорд

Результат роботи програми, що реалізує метод дотичних (Ньютона), показано на рис. 9.8.

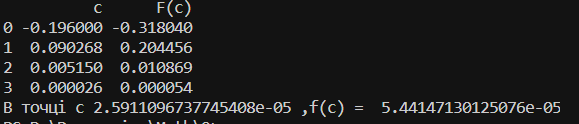
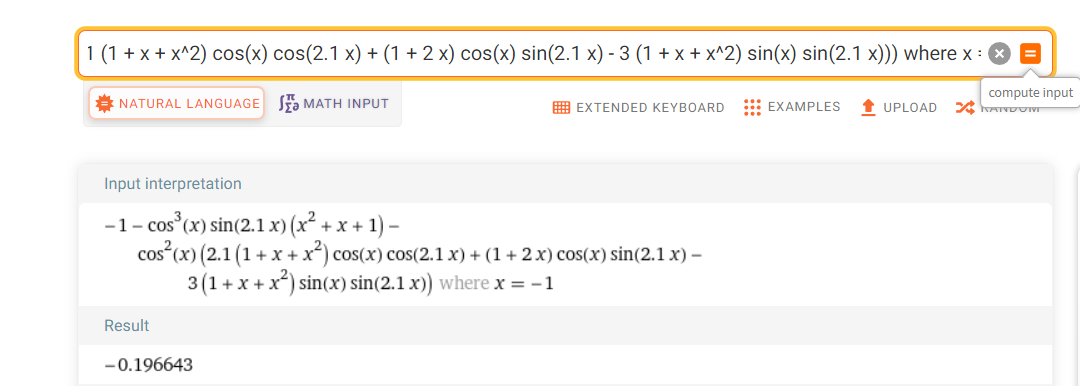


Рисунок 9.8 – Результат роботи програми, що реалізує метод дотичних (Ньютона)

Розраховано *с* на рис 9.8.1



Результати перевірки метода дотичних (Ньютона) засобами сервісу WolframAlpha показано на рис. 9.9.

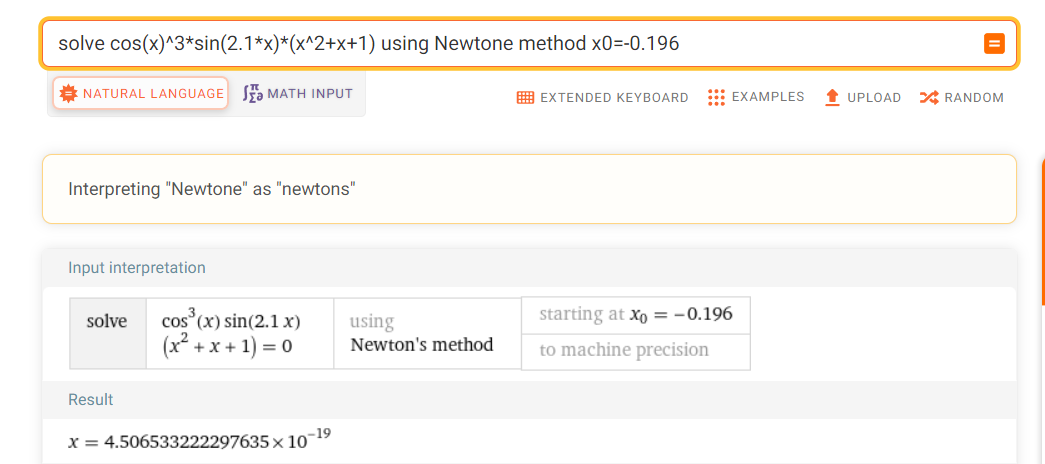


Рисунок 9.9 – Перевірка метода дотичних (Ньютона)

Результати обчислень програм за методами поділу проміжку навпіл та дотичних (Ньютона) співпадають з результатами обчислень засобами сервісу WolframAlpha.

Результати обчислення програми за методом хорд з початку співпадають з результатами обчислень засобами сервісу WolframAlpha, але потім відрізняються. Це пов’язано з тим, що метод secant, що використовується WolframAlpha, дещо відрізняється від методу хорд, що реалізований у програмі. Особливості метода secant описано за посиланням: <https://mathworld.wolfram.com/SecantMethod.html>.

Метод хорд, що використаний у програмі, відповідає методу, що має назву Method of False Position. Особливості цього методу описано за посиланням: <https://mathworld.wolfram.com/MethodofFalsePosition.html>. Але цей метод не реалізовано засобами сервісу WolframAlpha.