**Міністерство освіти і науки України**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”**

**Кафедра прикладної математики**

**ЕТАП №8**

«Пояснювальна записка

РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ»

З дисципліни: «Програмування» 1-й семестр

На тему: «Програма розв’язання рівнянь виду f(x)=0 методами послідовних наближень (метод дотичних)»

Виконала: Бордонос Катерина Юріївна.

Група КМ-02, факультет ФПМ

Залікова книжка № КМ-0201

Керівник: Олефір О.С.

**Київ-2020**

ЗМІСТ

[1 ПОСТАНОВКА ЗАДАЧІ 3](#_Toc58241357)

[2 ПРОЕКТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ 4](#_Toc58241358)

[2.1 Опис обраного методу розв’язання задачі 4](#_Toc58241359)

[2.2 Розв′язання контрольних прикладів 5](#_Toc58241360)

[2.3 Проектування схеми взаємодії програмних засобів 7](#_Toc58241361)

[2.4 Розробка і перевірка алгоритмів 8](#_Toc58241362)

[2.5 Проектування інтерфейсу 9](#_Toc58241363)

[3 РЕАЛІЗАЦІЯ ПРОГРАМИ 11](#_Toc58241364)

[3.1 Експериментальні розрахунки 11](#_Toc58241365)

[ВИСНОВКИ 12](#_Toc58241366)

[ДОДАТКИ 13](#_Toc58241367)

[ЛІТЕРАТУРА 15](#_Toc58241368)

# ПОСТАНОВКА ЗАДАЧІ

Для розв’язання рівнянь існує безліч методів. Серед найвідоміших можна назвати метод Ньютона, метод хорд, метод половинного ділення. Cлід зазначити, що будь-який з цих методів є наближеним і може лише уточнювати значення кореня. Проте це уточнення можна виконувати до будь-якої заданої нами точності. Дані методи використовуються в тих випадках, коли рівняння або неможливо, або занадто складно розв’язати прямими методами та вони значно полегшують процес знаходження коренів.

В даній розрахунковій роботі детально описано метод Ньютона, або його ще називають методом дотичних.

Метою даної розрахунково-графічної роботи є створення програми, що буде знаходити корені рівняння, використовуючи даний метод з точністю до вказаного значення. В цій пояснювальній записці наведений детальний опис методів, представлені контрольні приклади, за якими можна перевірити правильність роботи програми, детальний алгоритм роботи методів. Також наведений спроектований інтерфейс створеної програми.

# ПРОЕКТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ

## Опис обраного методу розв’язання задачі

Припустимо, що розглядається нелінійне рівняння виду *f(x)=0*, де *f(x)* — функція неперервна на відрізку [a;b] і має на даному відрізку похідні першого і другого порядків, відмінні від нуля.

Ідея **методу дотичних(методу Ньютона)** полягає в тому, що на кожній ітерації графік функції *f(x)* замінюється дотичною і точку перетину кожної з цих дотичних з віссю абсцис приймають за чергове наближення до шуканого кореня.

Перша дотична проходить через точку K0(x0;y0) — кінець відрізка, для якого виконується умова *f(x0)∙f ″(x0) >0*. В результаті вона перетне вісь Ox в деякій точці x1.

Далі, обчислюється значення функції  *f(x1)* і в знайденій точці знову виконується побудова дотичної. Продовжуючи даний процес далі, отримують послідовність значень x0,x1,x2…, яка збігається до точного **розв'язку рівняння** *f(x)=0*.

Виведемо розрахункові формули методу для **рішення нелінійного рівняння**. Рівняння дотичної до графіка функції *f(x)* в точці K0(x0;y0) :

*y=f ′(x0)∙(x-x0)+f(x0)* (1)

Вона перетнула вісь Ox в деякій точці (x1;0). Координати отриманої точки будуть задовольняти рівняння даної дотичної:

*f ′(x0)∙(x1-x0)+f(x0)=0* (2)

Звідси, знаходимо перше наближення до шуканого корення *x1=(x0-f(x0))/f ′(x0*). На наступному кроці, запишемо рівняння дотичної до графіка функції *f(x)* в точці K1(x1;y1=f(x1)):

*y=f ′(x1)∙(x-x1)+f(x1)* (3)

Зазначимо, що дана дотична також перетинає вісь Ox в деякій точці (x2;0). Підставляючи координати отриманої точки в рівняння (3), отримуємо *x2=(x1-f(x1))/f ′(x1)* . Звідси, приходимо до висновку, що у загальному випадку **формула методу дотичних** матиме вигляд:

xi+1=xi - (4)

## Розв′язання контрольних прикладів

**Приклад 1**: Використовуючи метод дотичних знайти, з точністю *ℇ=0.01*, розв'язок нелінійного рівняння *f(x)=x3+x-5=0* на проміжку *[-2;2].*

Отже, на першому кроці, визначимо першу та другу похідні заданої функції:

*f ′(x)=3x2+1; f ″(x)=6x*

Після цього, перевіримо виконання умови збіжності на кінцях заданого інтервалу:

*f (-2)∙ f ″(-2)=-15∙-12=180 >0 ; f (2)∙ f ″(2)=5∙12=60 >0*

Виходячи з того, що умова збіжності виконується для обох кінців проміжку

[-2;2], то в якості початкового наближення візьмемо, наприклад, значення лівого кінця *x0=-2*.

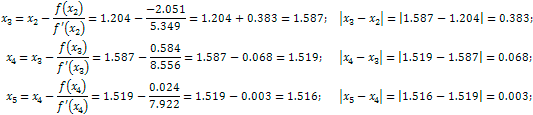
Далі, скориставшись формулою *xi+1=xi -*  , знаходимо перше наближення до шуканого кореня:

http://www.mathros.net.ua/wp-content/uploads/2012/08/metod_dotichnuh43.gif

Зважаючи на те, що для отриманого значення умова зупинки не виконується, переходимо до ітерації номер два, тобто, аналогічним чином підставляємо значення першого наближення у формулу *xi+1=xi -*  , і, таким чином, отримуємо наступне наближення:

http://www.mathros.net.ua/wp-content/uploads/2012/08/metod_dotichnuh41.gif

Як видно, умова зупинки для другого наближення також не виконується, тому продовжуючи ітераційний процес методу дотичних далі, на п'ятій ітерації отримаємо значення для якого умова зупинки виконується, і яке приймаємо в якості наближеного рішення нелінійного рівняння:



**Приклад 2**: Знайти корінь рівняння x3-x+1=0 методом дотичних з точністю *ℇ=0.001* на проміжку -2;-1.

Першим кроком знаходимо похідні даної функції. Маємо:

*f ′(x)=3x2-1; f ″(x)=6x*

Перевіряємо збіжність функції:

*f (-2)*∙ *f ″(-2)=-9∙(-12)=108 >0; f (-1)*∙ *f ″(-1)=-1∙(-6)=6 >0*

В якості початкового наближення обираємо *x0=-2*.

Результати розрахунків з використанням формули *xi+1=xi -* ,  зведемо в таблицю:

|  |  |  |
| --- | --- | --- |
| k | xi | |xi+1-xi| |
| 0 | –2 | – |
| 1 | –1,545455 | 0,454545 |
| 2 | –1,359615 | 0,185840 |
| 3 | –1,325801 | 0,033814 |
| 4 | –1,324719 | 0,001082 |
| 5 | –1,324718 | 0,000001 |

Оскільки *|x5-x4|=0.000001 < ℇ=0.001*, то процес завершується:

Відповідь*: x=-1.32472.*

## Проектування схеми взаємодії програмних засобів

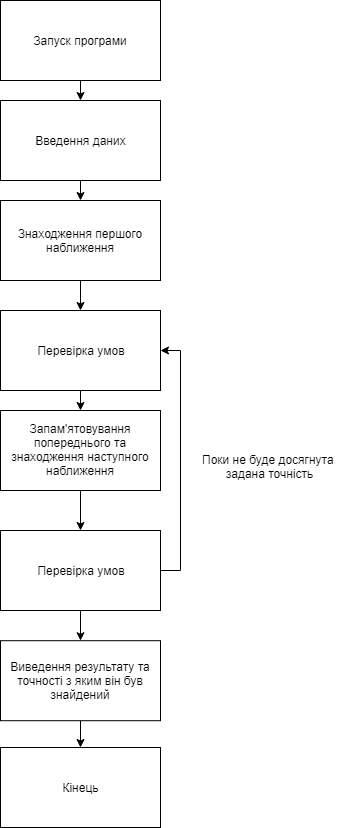


Рисунок 2.1. Схема взаємодії програмних засобів

## Розробка і перевірка алгоритмів

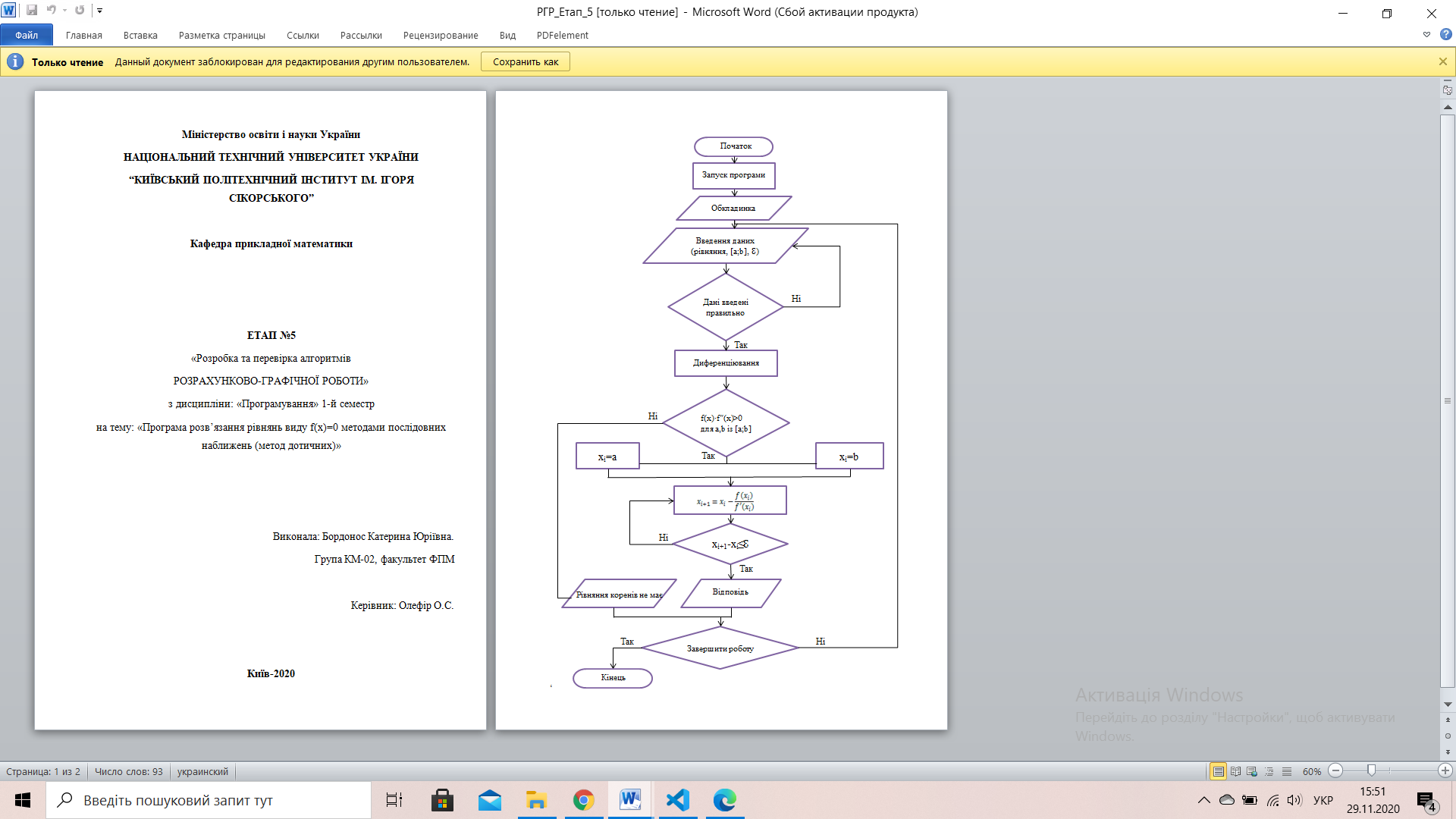


Рисунок 2.2. Блок-схеми алгоритмів програми

## Проектування інтерфейсу

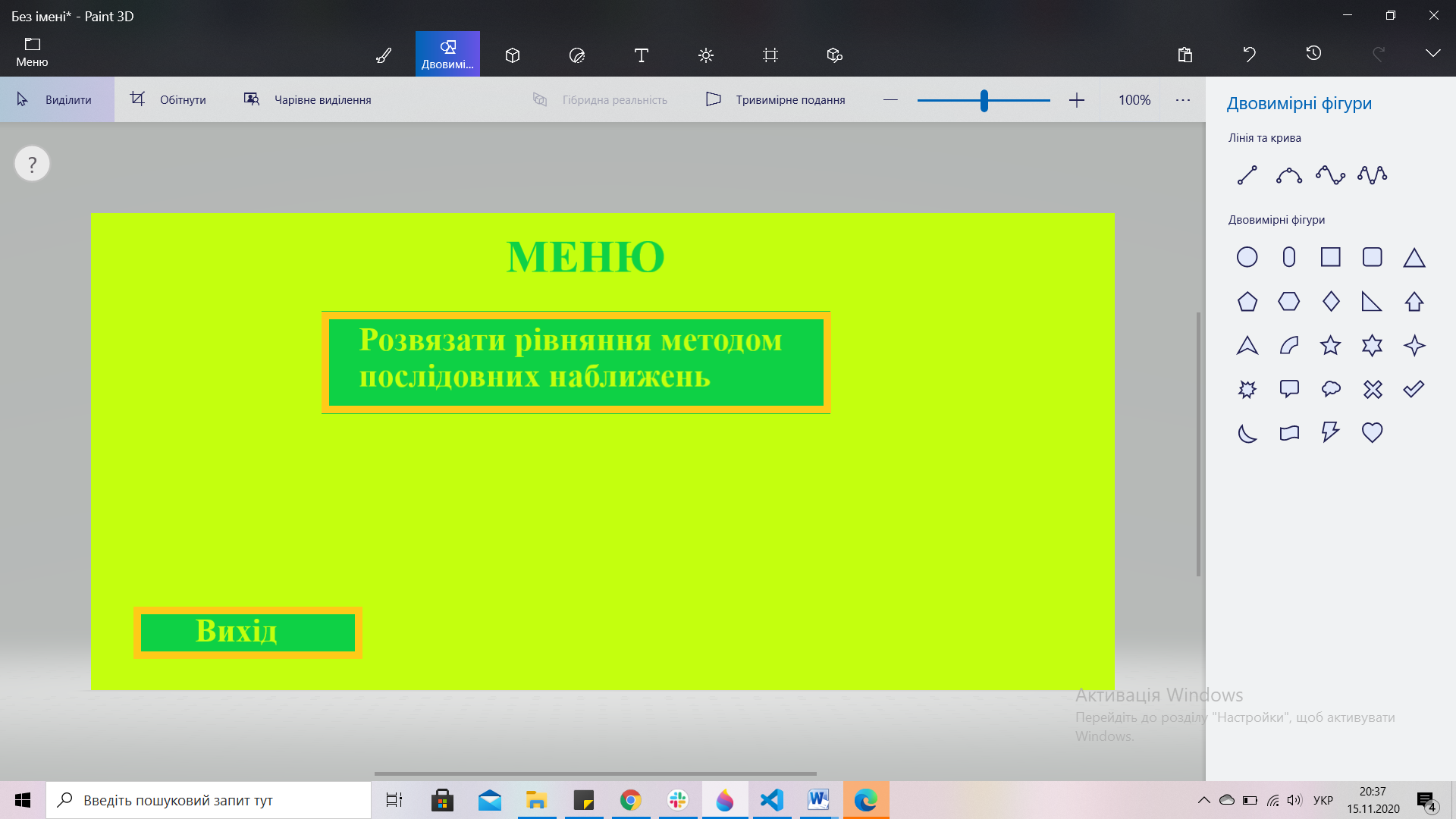


Рисунок 2.3. Головне меню програми

Містить такі активні кнопки як «Розв’язати рівняння методом послідовних наближень» та «Вихід». При виборі кнопки «Розв’язати рівняння методом послідовних наближень» програма переходить до наступного пункту, де буде використовуватись даний метод. При виборі кнопки «Вихід» програма закривається.

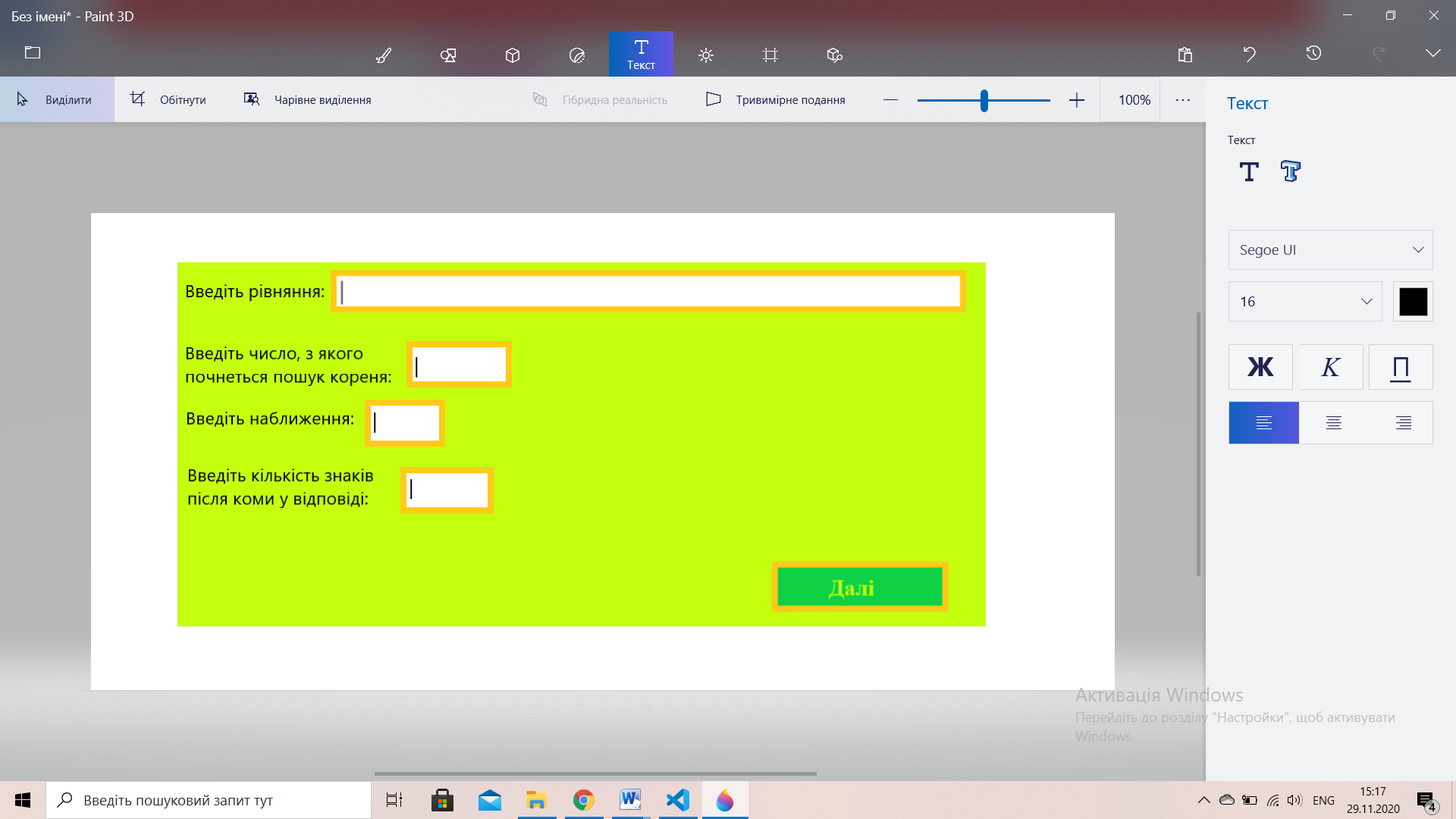


Рисунок 2.4. Інтерфейс вводу даних

Містить чотири поля для вводу самого рівняння, точності відповіді, наближення та числа, з якого програма почне шукати корені. Також містить кнопку «Далі», на яку користувач повинен натиснути після введення даних, і програма перейде до наступного пункту.

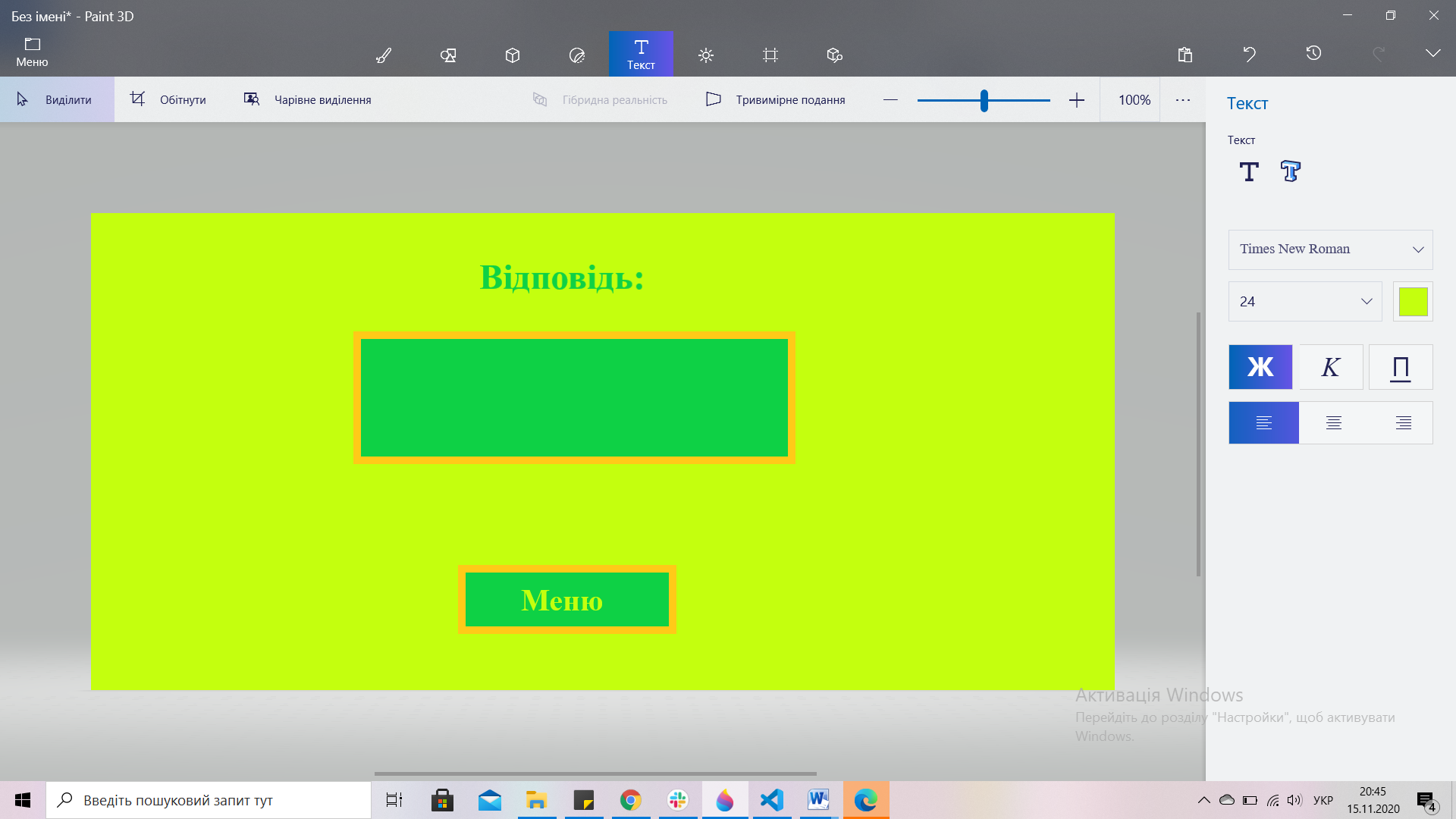
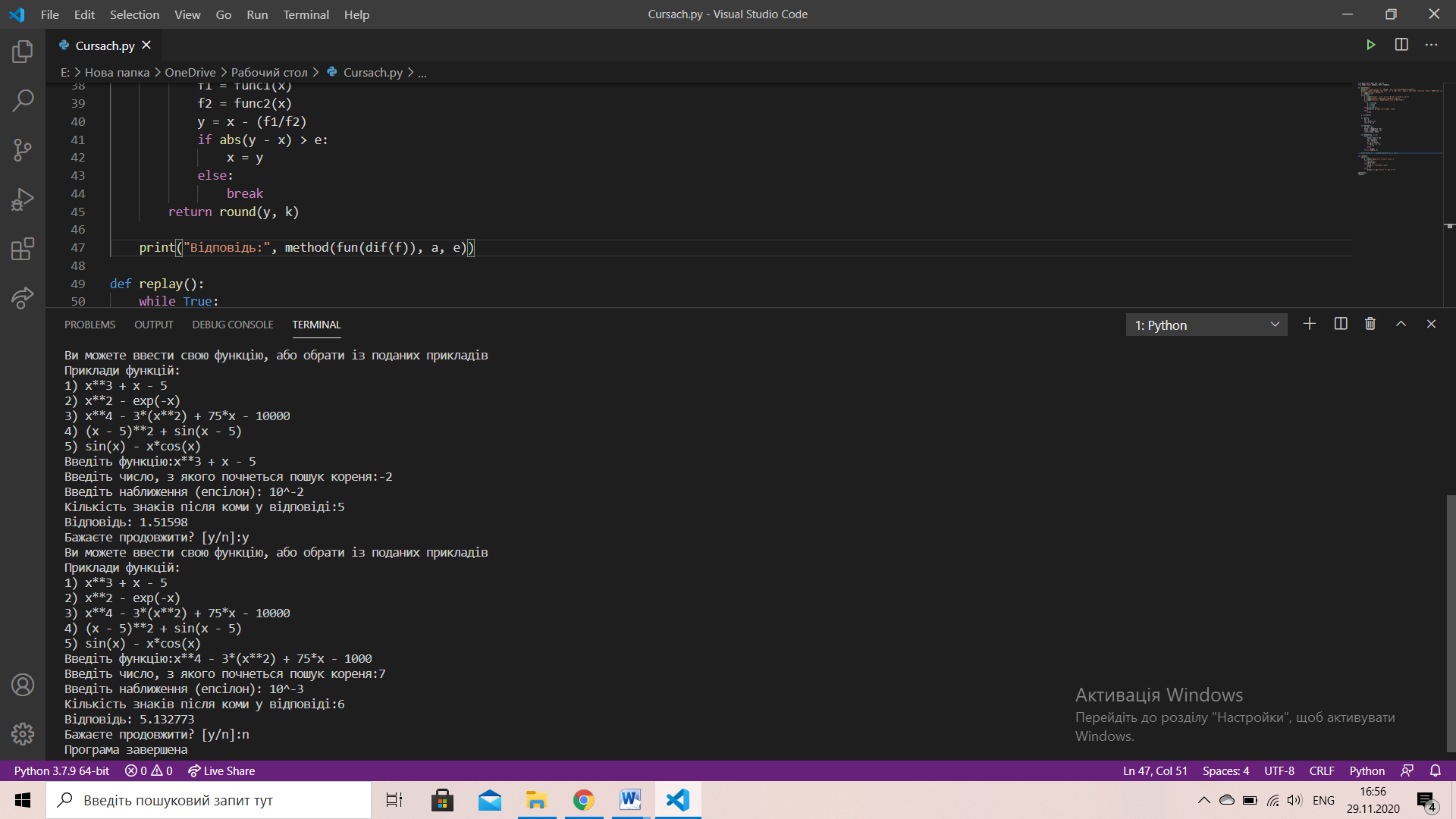


Рисунок 2.5. Меню відповіді

Демонструє відповідь, або вказує що дані введені не коректно, або вказує що відповідь не існує. Має єдину активну кнопку МЕНЮ, яка повертає в меню.

# РЕАЛІЗАЦІЯ ПРОГРАМИ

## Експериментальні розрахунки



Всі розрахунки перевірені та виконуються правильно

# ВИСНОВКИ

У цій розрахунково-графічній роботі був реалізований наближений метод розв’язання рівнянь виду F(x)=0, а саме метод дотичних(метод Ньютона). Під час створення розрахунково-графічної роботи було опановано теорію з обраної теми, розроблено алгоритми виконання обраних методів, вирішено контрольні приклади до методу, створено схему взаємодії програмних засобів, що розробляються, створено схему інтерфейсу користувача та компонування всіх елементів створення програми у цілісний результат.

# ДОДАТКИ

ТЕКСТ ПРОГРАМИ

from math import exp, sin, cos, pi

from sympy import symbols, diff, lambdify

print("Ви можете ввести свою функцію, або обрати із поданих прикладів")

def operation():

    print("Приклади функцій: \n1) x\*\*3 + x - 5 \n2) x\*\*2 - exp(-x) \n3) x\*\*4 - 3\*(x\*\*2) + 75\*x - 10000 \n4) (x - 5)\*\*2 + sin(x - 5) \n5) sin(x) - x\*cos(x)")

    f = input("\nВведіть функцію:")

    x = symbols('x')

    while type:

        a = input("Введіть число, з якого почнеться пошук кореня:")

        n = input("Введіть наближення (eпсілон): 10^-")

        k = input("Кількість знаків після коми у відповіді:")

        try:

            a = float(a)

            n = int(n)

            k = int(k)

        except ValueError:

            print("Ви повинні ввести цілi числa")

        else:

            break

    e = 1/(10\*\*n)

    def dif(f):

        f1 = f

        f2 = diff(f, x)

        return f1, f2

    def fun(dif):

        f1, f2 = dif

        func1 = lambdify(x, f1)

        func2 = lambdify(x, f2)

        return func1, func2

    def method(fun, x, e):

        while True:

            func1, func2 = fun

            f1 = func1(x)

            f2 = func2(x)

            y = x - (f1/f2)

            if abs(y - x) > e:

                x = y

            else:

                break

        return round(y, k)

    print("\nВідповідь:", method(fun(dif(f)), a, e))

def replay():

    while True:

        e = input("\nБажаєте продовжити? [y/n]:")

        if e == "y":

            operation()

        elif e == "n":

            print("Програма завершена")

            break

        else:

            print("Потрібно ввести 'y' або 'n'!")

operation()

replay()

# ЛІТЕРАТУРА

* 1. Демидович Б.П., Марон И.А. Основы вычислительной математики / Б.П.

Демидович, И.А. Марон - М. : Наука, 1970. - 660 с.

* 1. Волков Е. А. Глава 4. Методы решения нелинейных уравнений и систем. § 25. Метод Ньютона // Численные методы. – Учеб. Пособие для вузов. – 2-е изд., испр. – М. : Наука, 1987ю – С. 190.