Лабораторна робота №9

*Наближення функцій поліномом Тейлора*

Виконала Гальчинська Софія, студентка ФІТ 2-8. Варіант 4.

[*https://github.com/GalchynskaSofiia/Numerical\_Methods\_Of\_Programming*](https://github.com/GalchynskaSofiia/Numerical_Methods_Of_Programming)

**Перепрошую за невчасно здану роботу, була в лікарні.**

**Завдання 1:**

Знайти перші три похідні функції та розрахуйте значення за многочленом Тейлора. Оцініть похибку. Побудуйте на одному малюнку графік функції та наближення.

Нехай *x = 0*, тоді маємо:

Обчислимо поліном Тейлора третього степеня за формулою (при цьому a = 0)

Маємо:

Оскільки sin(x) і cos(x) належать проміжку [-1;1], то четверта похідна не перевищуватиме (1 + 4 - 3) + (6 + 1) = 9. Тому M = 9 і за виразом маємо:

.

.

Код:

import sympy as sp

import numpy as np

import matplotlib.pyplot as plt

x = sp.symbols('x')

f = sp.sin(sp.cos(x))

f1 = sp.diff(f, x)

f2 = sp.diff(f1, x)

f3 = sp.diff(f2, x)

f4 = sp.diff(f3, x)

print("f'(x) =", f1)

print("f''(x) =", f2)

print("f'''(x) =", f3)

print("f''''(x) =", f4) #четверту похідну вивела, щоб перевірити свої аналітичні рлзрахунки

x0 = 0

f\_x0 = f.subs(x, x0).evalf()

f1\_x0 = f1.subs(x, x0).evalf()

f2\_x0 = f2.subs(x, x0).evalf()

f3\_x0 = f3.subs(x, x0).evalf()

T = f\_x0 + f1\_x0\*(x-x0) + (f2\_x0/2)\*(x-x0)\*\*2 + (f3\_x0/6)\*(x-x0)\*\*3

print("f(0) =", f\_x0)

print("T(x) =", T.evalf())

x\_vals = np.linspace(-2, 2, 1000)

f\_vals = np.array([f.subs(x, xi).evalf() for xi in x\_vals])

T\_vals = np.array([T.subs(x, xi).evalf() for xi in x\_vals])

fig, ax = plt.subplots()

ax.plot(x\_vals, f\_vals, label="f(x)")

ax.plot(x\_vals, T\_vals, label="T(x)")

ax.legend()

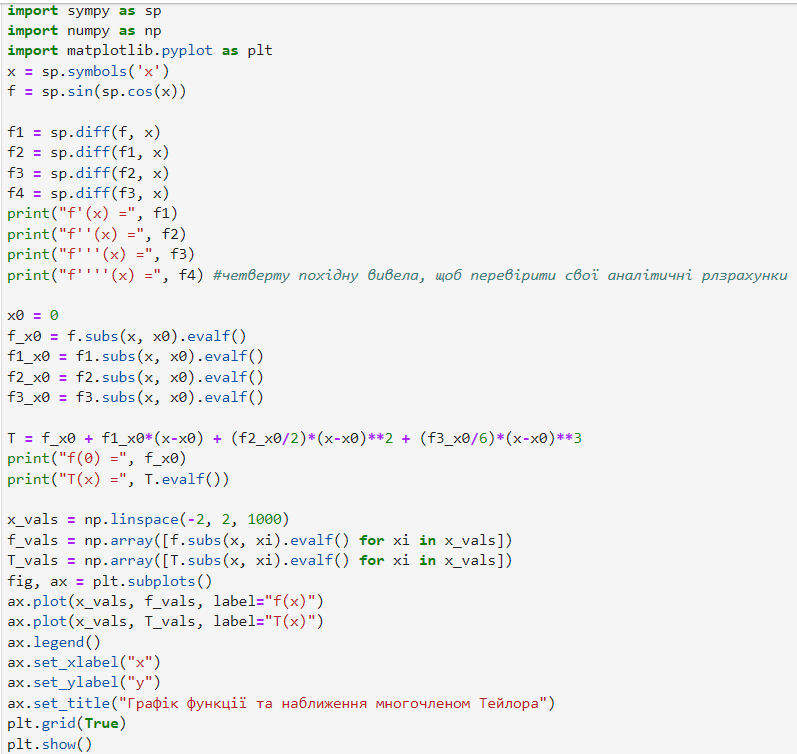
ax.set\_xlabel("x")

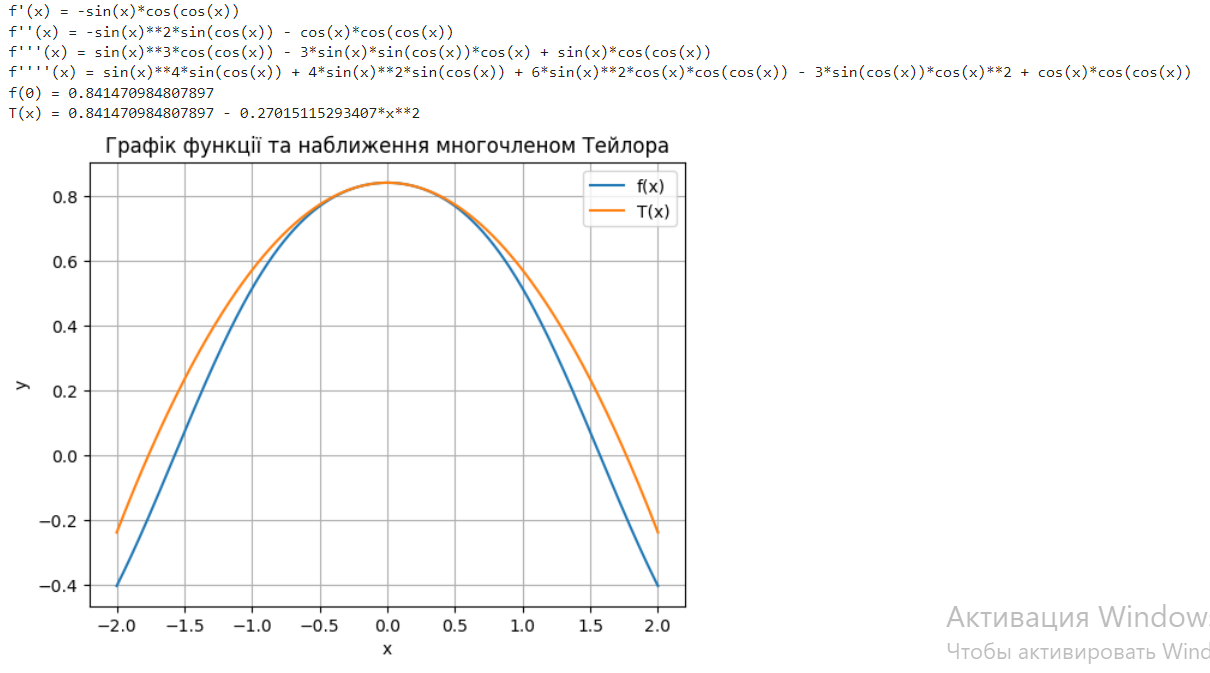
ax.set\_ylabel("y")

ax.set\_title("Графік функції та наближення многочленом Тейлора")

plt.grid(True)

plt.show()





**Завдання 2:**

Побудувати многочлен Тейлора (графік) за допомогою scipy.interpolate.approximate\_taylor\_polynomial(f, x, degree, scale, order=None)

Код:

import numpy as np

import matplotlib.pyplot as plt

from scipy.interpolate import approximate\_taylor\_polynomial

def f(x):

return np.sin(np.cos(x))

x = np.linspace(-2.0, 2.0, num=400)

plt.figure(figsize=(10, 6))

plt.plot(x, f(x), label="f(x) curve", color='blue')

degree = 3

taylor = approximate\_taylor\_polynomial(f, 0, degree, 1)

print('taylor=', taylor)

plt.plot(x, taylor(x), label=f"degree={degree}", color='red', linestyle='--' )

plt.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left',borderaxespad=0.0, shadow=True)

plt.xlabel("x")

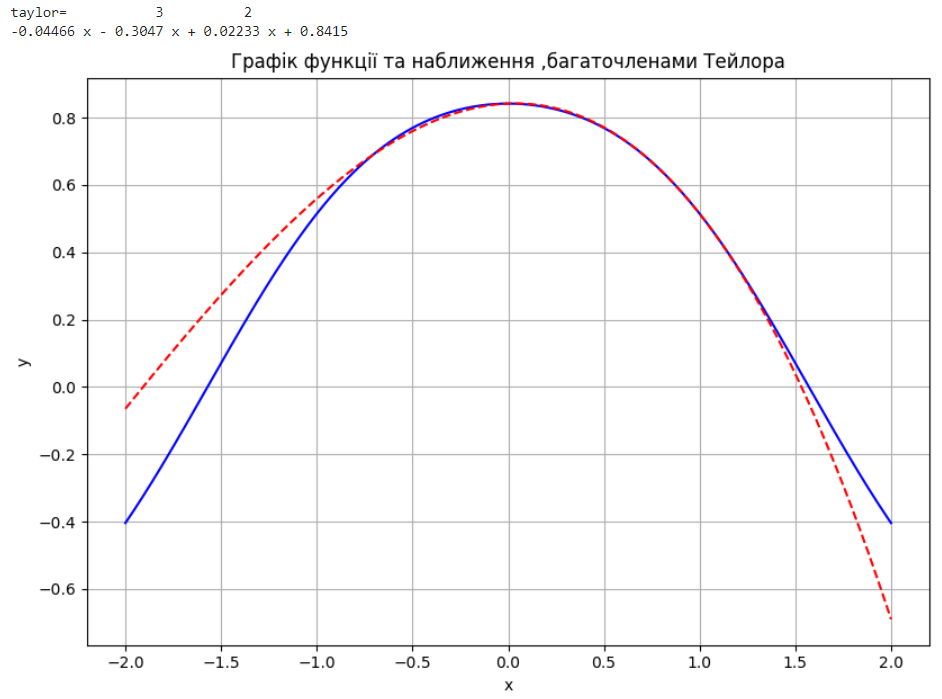
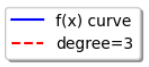
plt.ylabel("y")

plt.title("Графік функції та наближення ,багаточленами Тейлора")

plt.tight\_layout()

plt.grid()

plt.show()

**Висновки:** Отже, під час виконання практичної роботи провели аналітичні розрахунки та знайшли перші три похідні даної функції, розрахували значення за многочленом Тейлора. Потім оцінили похибку, яка становить 0,375. Розрахували похідні та многочлен Тейлора та побудували на одному малюнку графік функції та наближення за допомогою мови програмування Python. Також побудували многочлен Тейлора (графік) за допомогою scipy.interpolate.approximate\_taylor\_polynomial. **Важливим є те, що многочлен Тейлора, отриманий під час аналітичних розрахунків, повністю зійшовся з многочленом, отриманим за допомогою коду. Похідні також абсолютно ідентичні. Обидва графіка також вийшли однаковими.**