**Міністерство освіти і науки України**

**Національний технічний університет України**

**“Київський політехнічний інституті ім. І. Сікорського”**

**Кафедра прикладної математики**

**ЕТАП № 7**

«РОЗРОБКА ТА ПЕРЕВІРКА ПРОГРАМИ»

з дисципліни: «Програмування» 1-й семестр

на тему: «Програма чисельного диференціювання

(Обчислення другої похідної)»

Виконав: Казновський А.Т.

Група КМ-02, факультет ФПМ

Керівник: Олефір О.С.

**Київ - 2020**

**Розробка та перевірка програми**

Код програми:

# Виведення початкової інформації

print("ПРОГРАМА ЧИСЕЛЬНОГО ДИФЕРЕНЦІЮВАННЯ (обчислення другої похідної \nРозробник: Казновський Антон\n")

print("Формат даних\n"+"-"\*13,

"\nПрограма обчислює значення другої похідної деякої функції, заданої таблично\n" +

"(з однаковою різницею між сусідніми аргументами), у заданій користувачем точці.\n" +

"Для цього програма спочатку запитує різницю аргументів, далі перший з них і значення функції для\n" +

"кожного наступного аргументу (користувач може ввести необмежену кількість залежних змінних).\n" +

"Абсциса досліджуваної точки має бути одним із аргументів досліджуваної функції, для якого\n" +

"користувач задав значення.\n")

print('Значення мають вводитися у форматі цілого, або десяткового дробового числа через символ ".".\n' +

'В інших випадках програма виведе помилку та запитає повторне введення.\n' +

'Після кінця введення всіх значень функцій відправте слово "готово".\n' +

'Кінцевий результат виводиться у вигляді числа, округленого до 4 знаку після коми.\n')

print("ПОРАДА\n" + "-"\*16 + "\nДля найбільш точного результату слід задати перший аргумент таким, в якому\n"+

"шукатиметься друга похідна та ввести як мінімум 5 значення функції.\n")

print("Виконання\n" + "-"\*16)

# Імпорт додаткових бібліотек

import numpy as np

from sympy import diff, symbols

# Задання допоміжних функцій

def table(lst):

'''Функція створення матриці значень дельта у'''

k = len(lst[0])

for i in lst:

if len(i) != k:

for d in range((k - len(i))):

i.append(float(0))

delta\_y\_table = np.array(lst).transpose()

return delta\_y\_table

def lisst(list\_of\_values, length):

'''Функція створення списку для таблиці значень y'''

delta\_y\_list = []

for d in range(length-1):

delta\_y\_list.append([])

for k in range(1, length):

delta\_y\_list[0].append(list\_of\_values[k] - list\_of\_values[k-1])

for i in range(1, length):

for k in range(len(delta\_y\_list[i-1]) - 1):

delta\_y\_list[i].append(delta\_y\_list[i-1][k+1] - delta\_y\_list[i-1][k])

delta\_y\_list.insert(0, list\_of\_values)

return delta\_y\_list

def polinome(delta\_y\_list, i):

'''Функція для створення поліному'''

a, q = symbols('a, q')

summa = 0

a=1

for j in range(len(list\_of\_arguments)):

summa += a\*(deltas\_ys[i, j])

a \*= q - j

a /= 1 + j

return summa

def pohidna(summa):

'''Функція, що рахує другу похідну у даній точці для отриманого поліному'''

q = symbols('q')

h = (delta)\*\*(-2)

second\_pohidna = h\*diff(summa, q, 2)

param = h\*(find\_x-list\_of\_arguments[0])

result = second\_pohidna.evalf(subs={q:param})

return result

def count\_measuring\_error(length, deltas\_ys, i):

'''Функція для розрахунку похибки обчислень'''

h = (delta)\*\*(-1)

return (-1)\*\*(length-i-2)\*h/(length-i)\*deltas\_ys[i, length-i-1]

#==========================

# Реалізація програми

while True:

#Введення різниці між х

while True:

try:

delta = float(input("Введіть різницю між сусідніми х: "))

break

except ValueError:

print("Будь-ласка, введіть ціле або дійсне число у правильному форматі.")

print()

#Введення першого значення х

while True:

try:

x = float(input("Введіть перший х: "))

break

except ValueError:

print("Будь-ласка, введіть ціле або дійсне число у правильному форматі.")

print()

#Введення та формування списку значень

list\_of\_arguments = []

list\_of\_values = []

while True:

n = input("Введіть у для х="+str(x)+ " : ")

if n == "готово":

if len(list\_of\_values) != 0:

break

else:

print("Введіть хочаб одне значення у")

else:

try:

list\_of\_values.append(float(n))

list\_of\_arguments.append(x)

x += delta

except ValueError:

print("Будь-ласка, введіть ціле або дійсне число у правильному форматі.")

print()

#Введення значення х, в якому шукатиметься друга похідна

while True:

try:

find\_x = float(input("Введіть х точки, для якої шукатиметься значення другої похідної: "))

if find\_x in list\_of\_arguments:

break

else:

print("Будь-ласка, введіть значення, для якого ви задали ординату.")

except ValueError:

print("Будь-ласка, введіть ціле або дійсне число у правильному форматі.")

print()

#Опрацювання даних

length = len(list\_of\_values)

i = list\_of\_arguments.index(find\_x)

deltas\_ys = table(lisst(list\_of\_values, length))

znachennya = pohidna(polinome(deltas\_ys, i))

znachennya = round(float(znachennya), 4)

pohibka = round(count\_measuring\_error(length, deltas\_ys, i), 4)

#Виведення результату

print("Значення другої похідної у даній точці становить:", znachennya)

print("Значення похибки становить:", pohibka)

#Запит на повторний запуск програми

cont = input("\nХочете перезапустити програму? Напишіть + або -\n")

while True:

if (cont != "-") and (cont != "+"):

cont = input("Неправильний формат введення. Напишіть + або -\n")

else: break

if cont == "-":

break

Перевірка роботи програми на контрольних прикладах:

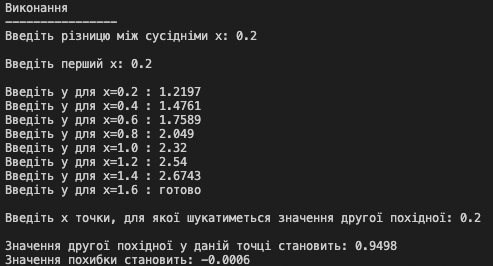
1. = esin x

Отримаємо значення для введення підставивши, наприклад х від 0,2 до 1,6 з кроком 0,2:  
= esin0,2 ≈ 1,2197  
= esin0,4 ≈ 1,4761  
= esin0,6 ≈ 1,7589  
= esin0,8 ≈ 2,049  
= esin1 ≈ 2,32

= esin1,2 ≈ 2,54  
= esin1,4 ≈ 2,6743  
= esin1,6 ≈ 2,717

Тепер знайдемо значення другої похідної у точці х = 2: ≈ 0,9293

Результат виконання програми:



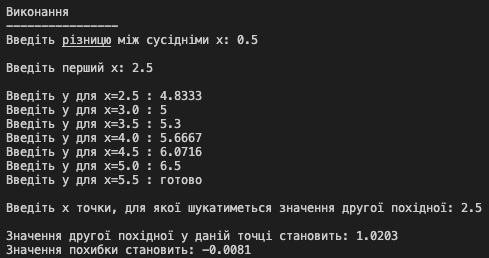
Додаток 1. Тестування програми за першим контрольним прикладом

Отримаємо значення для введення підставивши, наприклад х від 2,5 до 5 з кроком 0.5:

≈ 4,8333  
 = 5  
 = 5,3  
 ≈ 5,6667  
 ≈ 6,0716  
 = 6,5

Тепер знайдемо значення другої похідної в точці х = 2,5: ≈ 1,1851

Результат виконання програми:

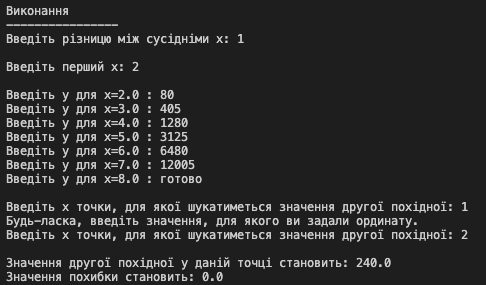


Додаток 2. Тестування програми за другим контрольним прикладом

Отримаємо значення для введення підставивши, наприклад х від 2 до 7:

= = 80  
= = 405  
= = 1280  
= = 3125  
= = 6480  
= = 12005

Тепер знайдемо значення другої похідної у точці х = 2: = 240

Результат виконання програми:

Додаток 3. Тестування програми за третім контрольним прикладом