**Міністерство освіти і науки України**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Кафедра прикладної математики**

**ЕТАП №7**

«Розробка та перевірка програми

РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ»

з дисципліни: «Програмування» 1-й семестр

на тему: «Програма чисельного диференціювання (Обчислення першої похідної)»

Виконав: Чорний Дмитро Станіславович

Група КМ-02, факультет ФПМ

Керівник: Олефір О.С.

**Київ-2020**

1. Текст програми.

Повний текст програми виглядає наступним чином:

print("\n\n\n","\t\t\t\t\t ", "РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА", sep = "")

print("\t\t\t\t\t\t ", "З дисципліни:\n\t\t\t\t\t 'Програмування' 1-ий семестр.", sep = "")

print("\n\n\n","\t\t\t\t ", "Тема:'Програма чисельного диференціювання'", sep = "")

print("\n\n\n","\t\t\t", "Виконав","\t\t\t\t\t\t\t\t", "Чорний Д.С.", sep = "")

print("\n\n\n","\t\t\t\t\t\t\t", "2020","\n\n\n" , sep = "")

print("ІНФОРМАЦІЙНЕ ПОВІДОМЛЕННЯ ТА ВИМОГИ ДО ДАНИХ:",

"",

"Ця програма дозволяє обчислити приблизне значення\nпохідної по таблично заданим даним або аналітичній формулі.\nДля обчислення програма інтерполює табличні значення за\nдопомогою поліному Ньотона.",

"",

"ВИМОГИ ДО ДАНИХ:",

"",

"Програма обчислює поліном на основі рівновіддалених точок.\nТому, для табличного введення функції потрібно буде обрати крок,\nта місце розташування першого значення в стопвчику х-ів.\nУсі інші значення х програма обрахує з зазначеним вами кроком.\nПереконайтеся, що задана вами функція може бути диференційована\nу точці диференціювання. У разі вводу аналітичної формули вам буде\nзапропоновано ввести функцію за допомогою синтаксису мови Python.\nДалі, буде обрахована таблиця значень з кроком 0.1. Якщо ви бажаєте\nобрати інший крок, або функція не визначена в місцях з кроком 0.1\nвід точки диференціювання рекомендованим є ввести функцію\nвже в табличному вигляді. Точка має знаходитися в проміжку від\nнайпершого табличного х і до останнього.",

sep="\n")

import math

import itertools

def ask\_yes\_or\_no(firstmsg="", inputmsg=""):

print(firstmsg)

yes\_or\_not = True

while yes\_or\_not:

y\_o\_n = input(inputmsg)

if (y\_o\_n == "Так") or (y\_o\_n == "так"):

yes\_or\_not = False

return True

elif (y\_o\_n == "Ні") or (y\_o\_n == "ні"):

yes\_or\_not = False

return False

else:

print("Ви ввели некоректне значення. Введіть 'так' або 'ні'.")

def ask\_num(msg, type\_="float"):

num\_stay = True

while num\_stay:

try:

if type\_ == "int":

num = int(input(msg+"\n"))

num\_stay = False

return num

else:

num = float(input(msg+"\n"))

num\_stay = False

return num

except ValueError:

print("Ви ввели непідходяще значення. Введіть значення числом.")

def calculate(num, list\_, h):

is\_in\_table = False

for i in list\_:

if num == i[0]:

place = list\_.index(i)

is\_in\_table = True

elif i == list\_[0] and num < i[0]:

return ("Х не входить у проміжок заданих таблично даних." , "Х не входить у проміжок заданих таблично даних.")

elif i == list\_[-1] and num > i[0]:

return ("Х не входить у проміжок заданих таблично даних." , "Х не входить у проміжок заданих таблично даних.")

elif (list\_[list\_.index(i)-1][0] < num) and (num < i[0]):

if (num-list\_[list\_.index(i)-1][0] > i[0]-num) and (i != list\_[-1]):

place = list\_.index(i)

else:

place = list\_.index(i)-1

dif\_table = [[]] dif\_table = [[]]

for i in range(len(list\_)-1):

dif\_table[0].append(list\_[i+1][1]-list\_[i][1])

while len(dif\_table[-1])!=1:

dif\_table.append([])

for i in range(len(dif\_table[-2])-1):

dif\_table[-1].append(dif\_table[-2][i+1]-dif\_table[-2][i])

if is\_in\_table:

res = 0

for i in range(math.ceil((len(list\_)-place)/2)):

res += ((-1)\*\*i)\*((dif\_table[i][place])/(i+1))

res = res/h

error = (((-1)\*\*(len(list\_)-place-3))\*dif\_table[len(list\_)-place-2][place])/(h\*(len(list\_)-place-1))

else:

res = 0

q = (num-list\_[place][1])/h

for i in range(math.ceil((len(list\_)-place)/2)):

if i == 0:

res += dif\_table[i][place]

elif i == 1:

res += ((2\*q-1)/2)\*dif\_table[i][place]

else:

jopa = 1

for j in range(1, i+1):

jopa =jopa\*(q-j)

jui = 0

coefs = list(itertools.combinations(range(1, i+1), i-1))

for j in range(i):

jack = q

for c in coefs[j]:

jack=jack\*(q-c)

jui = jui+jack

res += ((jopa + jui)/(math.factorial(i+1)))\*dif\_table[i][place]

res = res/h

jopa = 1

for j in range(1, (len(list\_)-place-1)):

jopa =jopa\*(q-j)

jui = 0

coefs = list(itertools.combinations(range(1, (len(list\_)-place+1)), (len(list\_)-place-1)))

for j in range((len(list\_)-place-1)):

jack = q

for c in coefs[j-1]:

jack=jack\*(q-c)

jui = jui+jack

error = (((-1)\*\*(len(list\_)-place-3))\*dif\_table[len(list\_)-place-2][place])/(h\*(len(list\_)-place-1))

return (round(res, 6), round(error, 6))

stay = True

while stay:

# питання введення функції таблично, чи аналітично

if ask\_yes\_or\_no("Бажаєте ввести функцію таблично?", "Напишіть 'так' для табличного введення, або 'ні' для аналітичного.\n"):

point\_func\_stay = True

der\_stay = False

point\_stay = True

else:

point\_func\_stay = False

der\_func\_stay = True

der\_stay = True

while point\_func\_stay:

#Введення функції таблично

if point\_stay:

print("\nВведіть крок табличних значень, перший 'х' з таблиці та кожен 'у' відповідно. \nДля зупинки запису введіть пустий рядок.")

numbers = []

h = ask\_num("Введіть крок табличних значень.")

counter = 0

x = ask\_num("Введіть перший 'х'. (Усі інші будуть обраховані у відповідності з кроком.)")

while point\_stay:

inp = input("Введіть 'у'. Для зупинки запису введіть пустий рядок.\n")

if inp == "":

if len(numbers)==0:

print("Ви не ввели ні одного y. Спробуйте ще.")

else:

point\_stay = False

else:

try:

numbers.append(tuple([x+(h\*counter), float(inp)]))

counter +=1

except ValueError:

counter +=1

print("Значення в некоректному форматі. Введіть число.")

#Введення точки диференціювання

num = ask\_num("\nВведіть точку для диференціювання.")

#Обрахунок значення

result = calculate(num, numbers, h)

print("\nЗначення похідної у точці:", result[0])

print("Обрахована помилка:", result[1])

##Спробувати ще раз з цією функцією?

if not ask\_yes\_or\_no("\nБажаєте спробувати ще раз з цією функцією?", "Напишіть 'так', або 'ні'.\n"):

point\_func\_stay = False

else:

point\_stay = False

while der\_func\_stay:

#Ввід точки для диференціювання

num = ask\_num("\nВведіть точку для диференціювання.")

#Введення аналітично

while der\_stay:

func\_ = input("Введіть функцію від x за допомогою синтаксису мови Python.\nЗвертайтеся до модулю math через 'math.'\n")

try:

x = num

eval(func\_)

def func(x):

return eval(func\_)

der\_stay = False

except:

print("\nНеправильно введена функція або функція не визначена у точці диференціювання. Спробуйте ще.")

#Переведення фунціїї в табличну форму

iterations = 14

numbers = []

for i in range(math.floor(iterations/2)):

try:

numbers.append(tuple([num-math.floor(iterations/2)+(i/10), func(num-math.floor(iterations/2)+(i/10))]))

except:

pass

for i in range(math.ceil(iterations/2)):

try:

numbers.append(tuple([num+(i/10), func(num+(i/10))]))

except:

pass

#Обрахунок значення похідної від поліному

result = calculate(num, numbers, 0.1)

print("\nЗначення похідної у точці:", result[0])

print("Обрахована помилка:", result[1])

#Спробувати ще раз з цією функцією?

if not ask\_yes\_or\_no("\nБажаєте спробувати ще раз з цією функцією?", "Напишіть 'так', або 'ні'.\n"):

der\_func\_stay = False

else:

der\_stay = False

#Запустити програму ще раз?

if not ask\_yes\_or\_no("\nБажаєте спробувати ще раз з іншою функцією?", "Напишіть 'так', або 'ні'.\n"):

stay = False

1. Експериментальні розрахунки

Функціональність програми була перевірена на тестових даних, взятих з підручника «Основы вычислительной математики» за авторством Демидовича Б.П. та Марона И.А., які знаходяться на сторінці 566.

Тестові дані:

|  |  |  |
| --- | --- | --- |
|  | Час, с | Відстань, см |
| 0 | 0 | 0 |
| 1 | 0.01 | 1.519 |
| 2 | 0.02 | 6.031 |
| 3 | 0.03 | 13.397 |
| 4 | 0.04 | 23.396 |
| 5 | 0.05 | 35.721 |
| 6 | 0.06 | 50.0 |
| 7 | 0.07 | 65.798 |
| 8 | 0.08 | 82.635 |
| 9 | 0.09 | 100 |

Введення даних в програму:

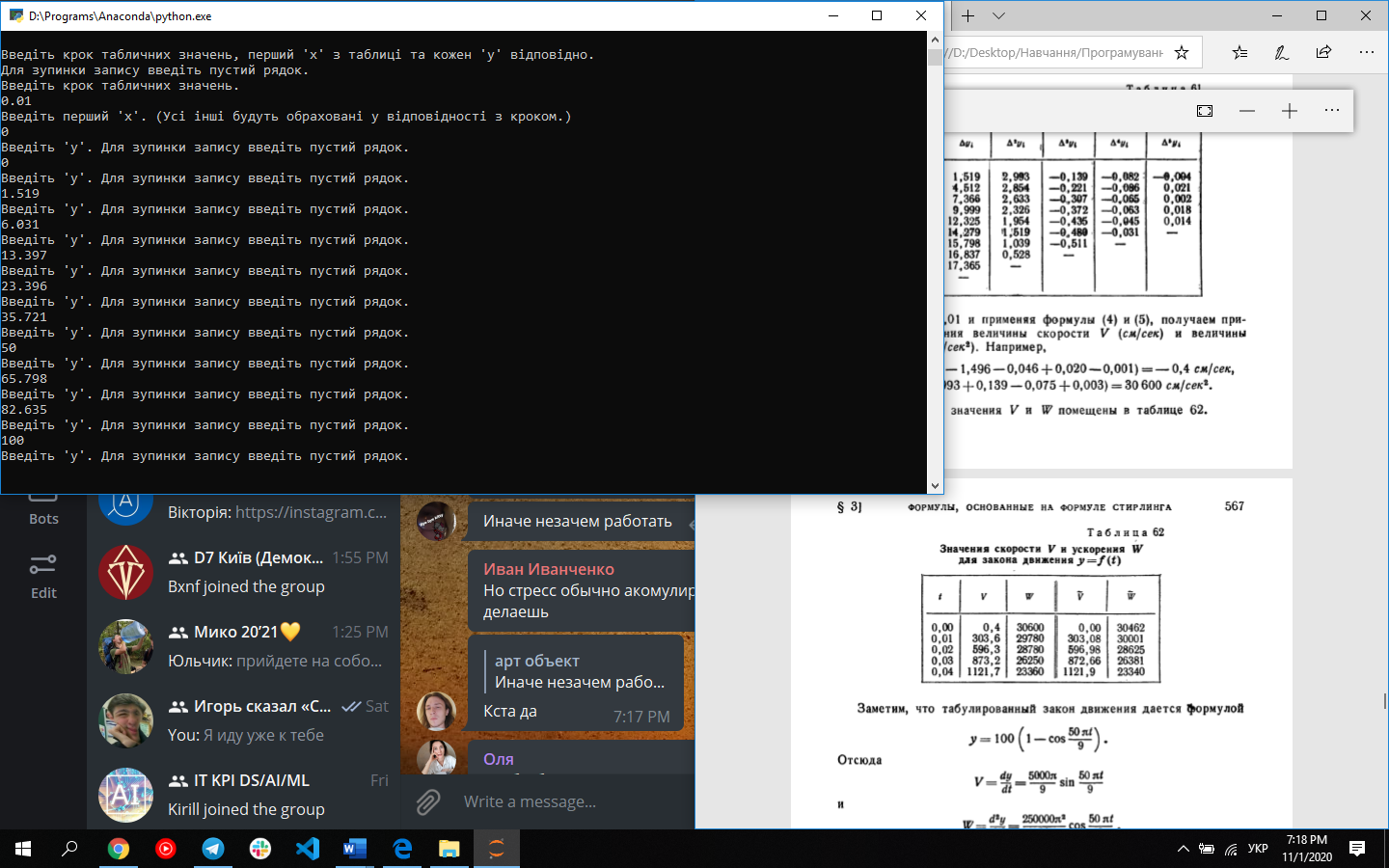


Рисунок 2.1

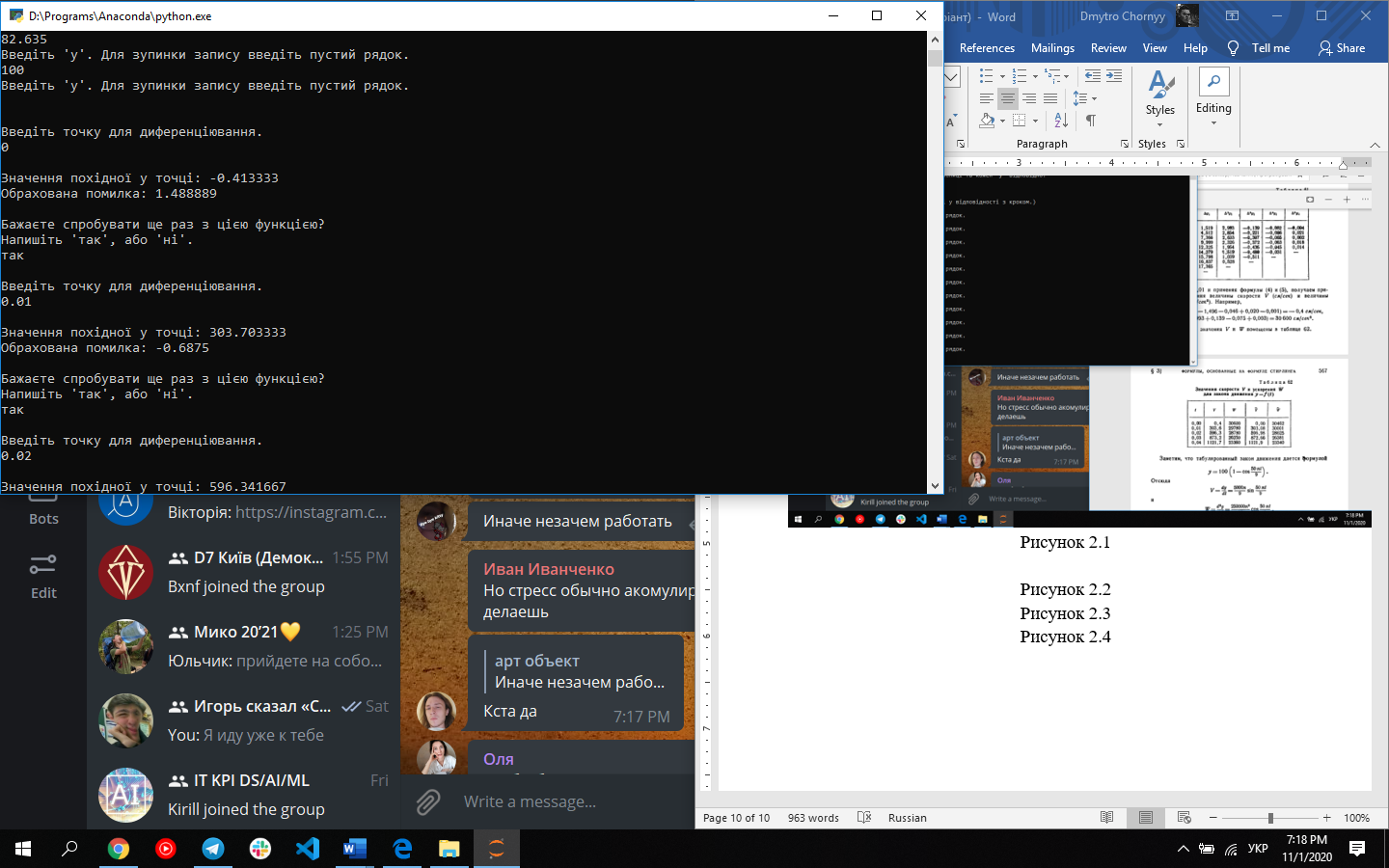


Рисунок 2.2

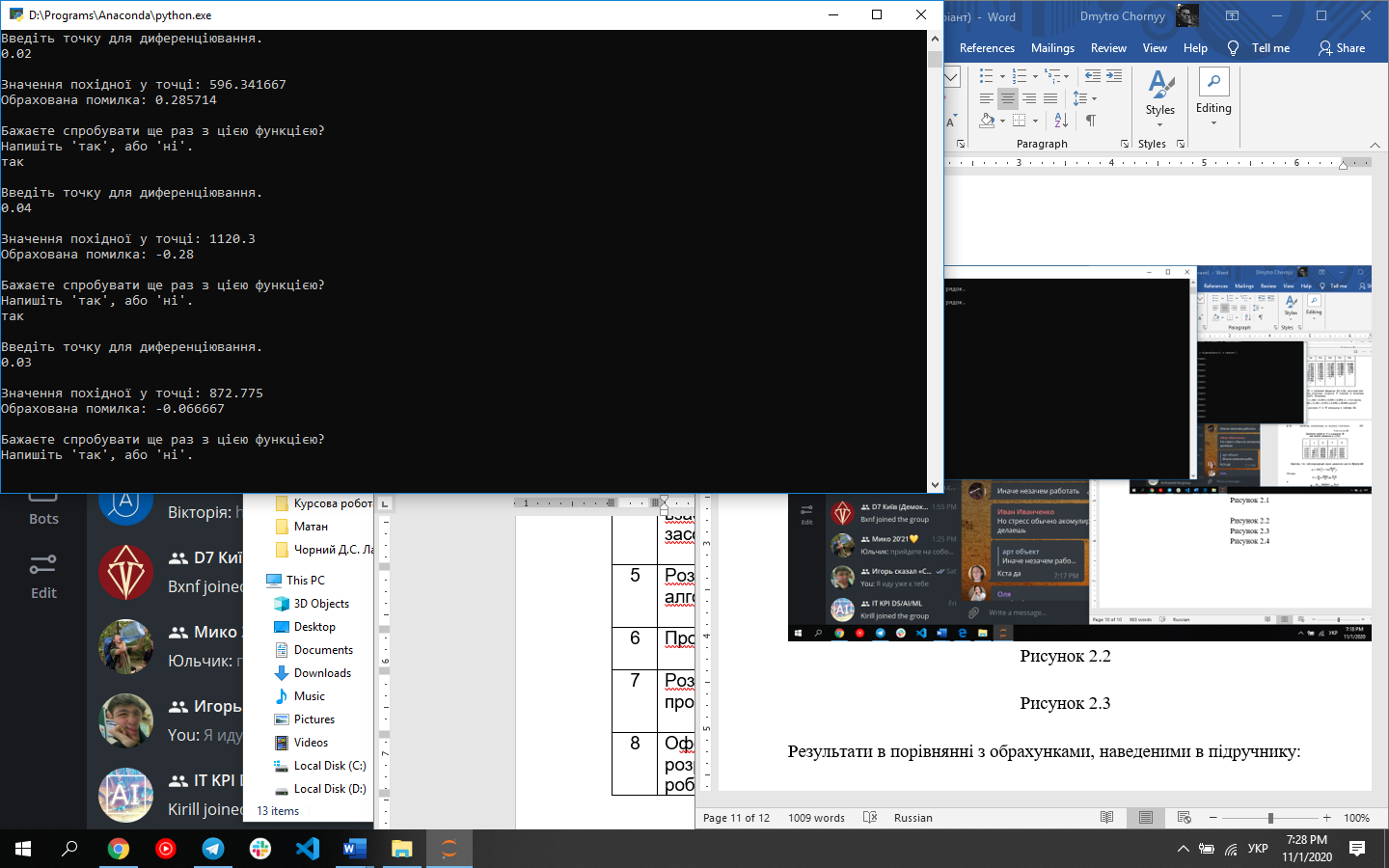


Рисунок 2.3

Результати в порівнянні з обрахунками, наведеними в підручнику:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Значення «у» | Результат роботи програми\* | Обрахована помилка\* | Обраховані значення за допомогою формули | Істинні значення похідних |
| 0 | -0,4133 | 1.4889 | 0,04 | 0 |
| 0,01 | 303,703 | -0.6875 | 303,6 | 303,08 |
| 0,02 | 596,342 | 0,285 | 596,3 | 596,98 |
| 0,03 | 872.775 | -0.0667 | 873,2 | 872,66 |
| 0,04 | 1120,3 | -0.28 | 1121,7 | 1121,9 |

\*невідповідність у 1-2% від обрахованих у підручнику найймовірніше пов’язана з особливостями представлення чисел у пам’яті комп’ютера.