НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

**Дисципліна:**

«Методи наукових досліджень»

**Звіт**

Лабораторної роботи №3

**на тему:** «ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З

ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

**Виконав:**

Студент групи: ІВ-92

Кубишка Юрій Сергійович

Варіант №13

**Перевірив:**

Регіда Павло Геннадійович

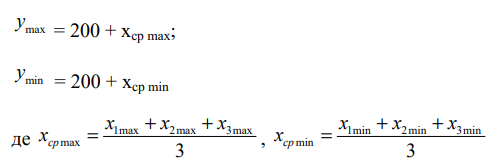
**Київ-2021**

**Мета:** провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти

коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

**Завдання на лабораторну роботу:**

1. Скласти матрицю планування для дробового трьохфакторного експерименту. Провести експеримент в усіх точках факторного простору, повторивши N експериментів, де N – кількість експериментів (рядків матриці планування) в усіх точках факторного простору – знайти значення функції відгуку У. Значення функції відгуку знайти у відповідності з варіантом діапазону, зазначеного далі (випадковим чином).

****

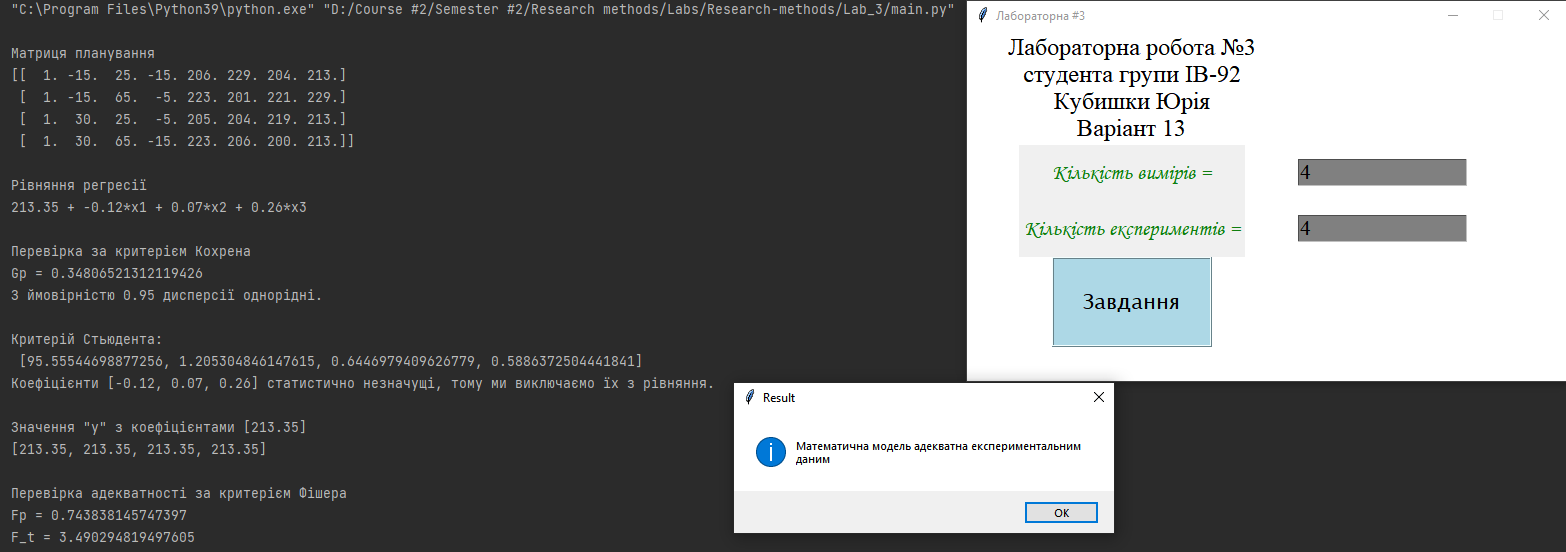
1. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.
2. Провести 3 статистичні перевірки.
3. Написати комп'ютерну програму, яка усе це виконує.

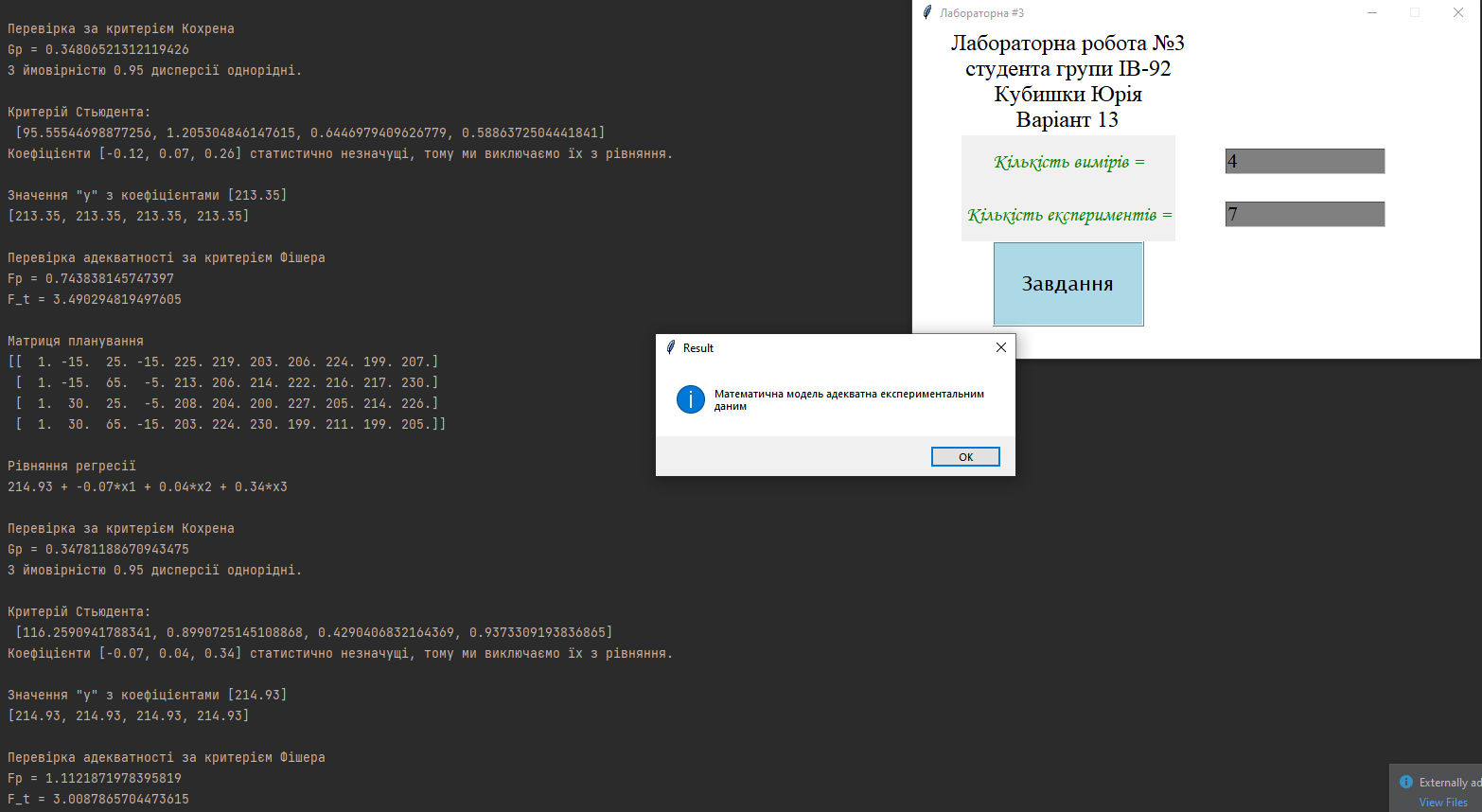
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | x1 | | x2 | | x3 | |
|  | min | max | min | max | min | max |
| **213** | -15 | 30 | 25 | 65 | -15 | -5 |

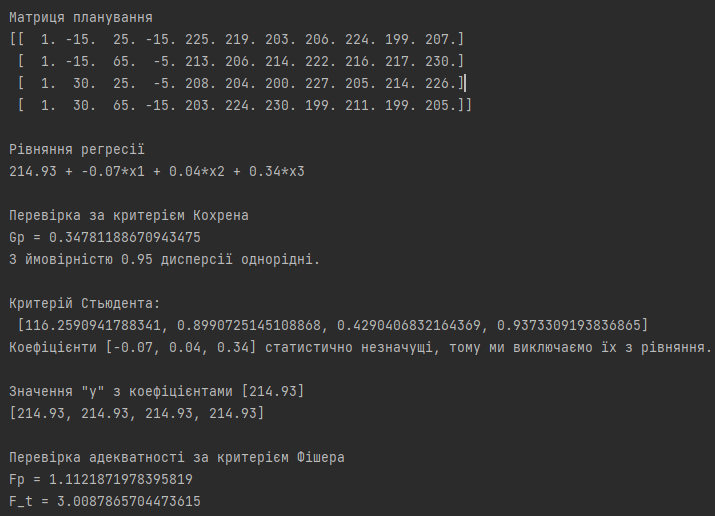
**Код програми:**

import numpy as np  
import random  
import tkinter as tk  
from numpy.linalg import solve  
from scipy.stats import f, t  
from functools import partial  
from tkinter.messagebox import showinfo, showerror  
  
  
def regression(x, b):  
 y = sum([x[i] \* b[i] for i in range(len(x))])  
 return y  
  
  
def find\_coefficient(x, y\_aver, n):  
 mx1 = sum(x[:, 1]) / n  
 mx2 = sum(x[:, 2]) / n  
 mx3 = sum(x[:, 3]) / n  
 my = sum(y\_aver) / n  
 a12 = sum([x[i][1] \* x[i][2] for i in range(len(x))]) / n  
 a13 = sum([x[i][1] \* x[i][3] for i in range(len(x))]) / n  
 a23 = sum([x[i][2] \* x[i][3] for i in range(len(x))]) / n  
 a11 = sum([i \*\* 2 for i in x[:, 1]]) / n  
 a22 = sum([i \*\* 2 for i in x[:, 2]]) / n  
 a33 = sum([i \*\* 2 for i in x[:, 3]]) / n  
 a1 = sum([y\_aver[i] \* x[i][1] for i in range(len(x))]) / n  
 a2 = sum([y\_aver[i] \* x[i][2] for i in range(len(x))]) / n  
 a3 = sum([y\_aver[i] \* x[i][3] for i in range(len(x))]) / n  
  
 X = [[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22, a23], [mx3, a13, a23, a33]]  
 Y = [my, a1, a2, a3]  
 B = [round(i, 2) for i in solve(X, Y)]  
 print('\nРівняння регресії')  
 print(f'{B[0]} + {B[1]}\*x1 + {B[2]}\*x2 + {B[3]}\*x3')  
  
 return B  
  
  
def s\_kv(y, y\_aver, n, m):  
 *"""квадратна дисперсія  
 """* res = []  
 for i in range(n):  
 s = sum([(y\_aver[i] - y[i][j]) \*\* 2 for j in range(m)]) / m  
 res.append(s)  
 return res  
  
  
def plan\_matrix(n, m, x\_range, y\_min, y\_max):  
 y = np.zeros(shape=(n, m))  
 for i in range(n):  
 for j in range(m):  
 y[i][j] = random.randint(y\_min, y\_max)  
 x\_norm = np.array([[1, -1, -1, -1],  
 [1, -1, 1, 1],  
 [1, 1, -1, 1],  
 [1, 1, 1, -1],  
 [1, -1, -1, 1],  
 [1, -1, 1, -1],  
 [1, 1, -1, -1],  
 [1, 1, 1, 1]])  
 x\_norm = x\_norm[:len(y)]  
  
 x = np.ones(shape=(len(x\_norm), len(x\_norm[0])))  
 for i in range(len(x\_norm)):  
 for j in range(1, len(x\_norm[i])):  
 if x\_norm[i][j] == -1:  
 x[i][j] = x\_range[j - 1][0]  
 else:  
 x[i][j] = x\_range[j - 1][1]  
  
 print('\nМатриця планування')  
 print(np.concatenate((x, y), axis=1))  
  
 return x, y, x\_norm  
  
  
def Cochrens\_criterion(y, y\_aver, n, m):  
 S\_kv = s\_kv(y, y\_aver, n, m)  
 Gp = max(S\_kv) / sum(S\_kv)  
 print('\nПеревірка за критерієм Кохрена')  
 return Gp  
  
  
def bs(x, y, y\_aver, n):  
 *""" оцінки коефіцієнтів  
 """* res = [sum(1 \* y for y in y\_aver) / n]  
 for i in range(3): # 4 - ксть факторів  
 b = sum(j[0] \* j[1] for j in zip(x[:, i], y\_aver)) / n  
 res.append(b)  
 return res  
  
  
def students\_criterion(x, y, y\_aver, n, m):  
 S\_kv = s\_kv(y, y\_aver, n, m)  
 s\_kv\_aver = sum(S\_kv) / n  
  
 s\_Bs = (s\_kv\_aver / n / m) \*\* 0.5 # статиcтична оцінка дисперсії  
 Bs = bs(x, y, y\_aver, n)  
 ts = [abs(B) / s\_Bs for B in Bs]  
  
 return ts  
  
  
def fishers\_criterion(y, y\_aver, y\_new, n, m, d):  
 S\_ad = m / (n - d) \* sum([(y\_new[i] - y\_aver[i]) \*\* 2 for i in range(len(y))])  
 S\_kv = s\_kv(y, y\_aver, n, m)  
 S\_kv\_aver = sum(S\_kv) / n  
  
 return S\_ad / S\_kv\_aver  
  
  
def cohren(f1, f2, q=0.05):  
 q1 = q / f1  
 fisher\_value = f.ppf(q=1 - q1, dfn=f2, dfd=(f1 - 1) \* f2)  
 return fisher\_value / (fisher\_value + f1 - 1)  
  
  
def main(n: str, m: str) -> None:  
 if n == "" or m == "":  
 showinfo("Warning", "Введіть бажані значення у відповідні поля")  
 try:  
 n = int(n)  
 m = int(m)  
 f1 = m - 1  
 f2 = n  
 f3 = f1 \* f2  
 q = 0.05  
  
 x\_range = [(-15, 30), (25, 65), (-15, -5)] # значення за варіантом  
 x\_aver\_max = (30 + 65 - 5) / 3  
 x\_aver\_min = (- 15 + 25 - 15) / 3  
  
 y\_max = 200 + int(x\_aver\_max)  
 y\_min = 200 + int(x\_aver\_min)  
  
 student = partial(t.ppf, q=1 - 0.025) # табличні значення  
 t\_student = student(df=f3)  
  
 G\_kr = cohren(f1, f2)  
  
 x, y, x\_norm = plan\_matrix(n, m, x\_range, y\_min, y\_max)  
 y\_aver = [round(sum(i) / len(i), 2) for i in y]  
  
 B = find\_coefficient(x, y\_aver, n)  
  
 Gp = Cochrens\_criterion(y, y\_aver, n, m)  
 print(f'Gp = {Gp}')  
 if Gp < G\_kr:  
 print(f'З ймовірністю {1 - q} дисперсії однорідні.')  
 else:  
 showinfo("Info", "Необхідно збільшити ксть дослідів")  
 m = int(m) + 1  
 main(int(n), m)  
  
 ts = students\_criterion(x\_norm[:, 1:], y, y\_aver, n, m)  
 print('\nКритерій Стьюдента:\n', ts)  
 res = [t for t in ts if t > t\_student]  
 final\_k = [B[ts.index(i)] for i in ts if i in res]  
 print('Коефіцієнти {} статистично незначущі, тому ми виключаємо їх з рівняння.'.format(  
 [i for i in B if i not in final\_k]))  
  
 y\_new = []  
 for j in range(n):  
 y\_new.append(regression([x[j][ts.index(i)] for i in ts if i in res], final\_k))  
  
 print(f'\nЗначення "y" з коефіцієнтами {final\_k}')  
 print(y\_new)  
  
 d = len(res)  
 f4 = n - d  
 F\_p = fishers\_criterion(y, y\_aver, y\_new, n, m, d)  
  
 fisher = partial(f.ppf, q=1 - 0.05)  
 f\_t = fisher(dfn=f4, dfd=f3) # табличне значення  
  
 print('\nПеревірка адекватності за критерієм Фішера')  
 print(f'Fp = {F\_p}')  
 print(f'F\_t = {f\_t}')  
 if F\_p < f\_t:  
 showinfo("Result", 'Математична модель адекватна експериментальним даним')  
 else:  
 showinfo("Result", 'Математична модель не адекватна експериментальним даним')  
 except:  
 showerror("Error", "Введіть числові значення у всі поля")  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 Width = 600  
 Height = 350  
 root = tk.Tk()  
 root.title("Лабораторна #3")  
 root.geometry("{Width}x{Height}".format(Width=Width, Height=Height))  
 root.resizable(width=False, height=False)  
  
 label\_a = tk.Label(root, width=20, height=2, text="Кількість вимірів =", font=("Monotype Corsiva", 16),  
 fg="#0F7F0F")  
 label\_b = tk.Label(root, width=20, height=2, text="Кількість експериментів =", font=("Monotype Corsiva", 16),  
 fg="#0F7F0F")  
  
 label\_a.grid(row=2, column=0)  
 label\_b.grid(row=3, column=0)  
  
 entry\_a = tk.Entry(root, bg="gray", width=15, font=("Times", 16)) # за однією й тією ж самою комбінації факторів  
 entry\_b = tk.Entry(root, bg="gray", width=15, font=("Times", 16)) # кількість експериментів (рядків матриці план.)  
  
 entry\_a.grid(row=2, column=1)  
 entry\_b.grid(row=3, column=1)  
  
 tk.Label(root, width=25, height=4, text='Лабораторна робота №3\nстудента групи ІВ-92\nКубишки Юрія\nВаріант 13',  
 font=('Times New Roman', 18), background="white").grid(row=0, column=0)  
 tk.Button(root, width=15, height=3, text="Завдання", font=('Roman', 16), background="lightblue",  
 activeforeground="#00FAFF",  
 activebackground="gray", relief='ridge', command=lambda: main(entry\_a.get(), entry\_b.get())).grid(row=4,  
 column=0)  
 root.config(background="White")  
 root.mainloop()

**Результати роботи програми:**







**Відповіді на контрольні запитання:**

1. **Що називається дробовим факторним експериментом?**

Дробовий факторний експеримент – це частина ПФЕ, який мінімізує число дослідів, за рахунок тієї інформації, яка не дуже істотна для побудови лінійної моделі.

1. **Для чого потрібно розрахункове значення Кохрена?**

Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.

1. **Для чого перевіряється критерій Стьюдента?**

За допомогою критерію Стьюдента перевіряється значущість коефіцієнтів

рівняння регресії.

1. **Чим визначається критерій Фішера і як його застосовувати?**

Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваному об’єкту.