Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №4
З дисципліни «Методи наукових досліджень»
За темою:

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАВ:
Студент ІІ курсу ФІОТ
Групи ІВ-91
Щоткін М.А.
Варіант - 29

ПЕРЕВІРИВ:
асистент
Регіда П.Г.

Київ 2021 р.

Мета: Провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об'єкту.

Завдання:

Завдання на лабораторну роботу

1. Скласти матрицю планування для повного трьохфакторного експерименту.
2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

$$y_{i\max} = 200 + x_{cp\max}$$
$$y_{i\min} = 200 + x_{cp\min}$$

де $x_{cp\max} = \frac{x_{1\max} + x_{2\max} + x_{3\max}}{3}$, $x_{cp\min} = \frac{x_{1\min} + x_{2\min} + x_{3\min}}{3}$

3. Знайти коефіцієнти рівняння регресії і записати його.
4. Провести 3 статистичні перевірки – за критеріями Кохрена, Стьюдента, Фішера.
5. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
6. Написати комп'ютерну програму, яка усе це моделює.

129	-15	30	30	80	30	35

Програмний код

Main.pt

```
import random
import math
import numpy as np
from scipy.stats import f,t

Gt = 0.5157
Ft = 2.7
m = 3
N = 8
d = 0

x1min = -15
x1max = 30
x2min = 30
x2max = 80
x3min = 30
x3max = 35

ymin = 200 + (x1min + x2min + x3min)/3
ymax = 200 + (x1max + x2max + x3max)/3
y1 = [random.randint(int(ymin), int(ymax) + 1) for i in range(N)]
y2 = [random.randint(int(ymin), int(ymax) + 1) for i in range(N)]
y3 = [random.randint(int(ymin), int(ymax) + 1) for i in range(N)]
```

```
yAverage = [0]*N
for i in range(0,N):
    yAverage[i] = (y1[i] + y2[i] + y3[i])/3

x1iR = [random.randint(x1min, x1max + 1) for i in range(N)]
x2iR = [random.randint(x2min, x2max + 1) for i in range(N)]
x3iR = [random.randint(x3min, x3max + 1) for i in range(N)]

rx = [0]*N
ry = [0]*N
for i in range(0,N):
    rx[i] = [x1iR[i], x2iR[i], x3iR[i]]
    ry[i] = [y1[i], y2[i], y3[i]]
matrix0fY = np.array([ry[0], ry[1], ry[2], ry[3], ry[4], ry[5], ry[6],
ry[7]])
matrix0fX = np.array([rx[0], rx[1], rx[2], rx[3], rx[4], rx[5], rx[6],
rx[7]])
print('X:\n', matrix0fX)
print('\nY:\n', matrix0fY)
print('\nСередні значення Y:\n', yAverage)

r0 = [0]*N
r1 = [0]*N
r2 = [0]*N
r3 = [0]*N
r4 = [0]*N
r5 = [0]*N
r6 = [0]*N
r7 = [0]*N
r0[0] = N

for i in range(0,N):
    r0[1] += x1iR[i]
    r2[0] += x2iR[i]
    r3[0] += x3iR[i]
    r4[0] += x1iR[i] * x2iR[i]
    r5[0] += x1iR[i] * x3iR[i]
    r6[0] += x2iR[i] * x3iR[i]
    r7[0] += x1iR[i] * x2iR[i] * x3iR[i]
    r1[1] += x1iR[i] ** 2
    r4[1] += x1iR[i] * x1iR[i] * x2iR[i]
    r5[1] += x1iR[i] * x1iR[i] * x3iR[i]
    r7[1] += x1iR[i] * x1iR[i] * x2iR[i] * x3iR[i]
    r2[2] += x2iR[i] ** 2
    r4[2] += x1iR[i] * x2iR[i] * x2iR[i]
    r6[2] += x2iR[i] * x2iR[i] * x3iR[i]
    r7[2] += x1iR[i] * x2iR[i] * x2iR[i] * x3iR[i]
    r3[3] += x3iR[i] ** 2
    r5[3] += x1iR[i] * x3iR[i] * x3iR[i]
    r6[3] += x2iR[i] * x3iR[i] * x3iR[i]
    r7[3] += x1iR[i] * x2iR[i] * x3iR[i] ** 2
    r4[4] += x1iR[i] * x1iR[i] * x2iR[i] * x2iR[i]
    r5[4] += x1iR[i] * x1iR[i] * x2iR[i] * x3iR[i]
    r7[4] += x1iR[i] * x1iR[i] * x2iR[i] * x2iR[i] * x3iR[i]
    r5[5] += x1iR[i] * x1iR[i] * x3iR[i] * x3iR[i]
    r7[5] += x1iR[i] * x1iR[i] * x2iR[i] * x3iR[i] * x3iR[i]
    r6[6] += x2iR[i] * x2iR[i] * x3iR[i] * x3iR[i]
    r7[6] += x1iR[i] * x2iR[i] * x2iR[i] * x3iR[i] * x3iR[i]
    r7[7] += x1iR[i] * x1iR[i] * x2iR[i] * x3iR[i] * x3iR[i]
r0[1] = r1[0]
r2[0] = r0[2]
r3[0] = r0[3]
r4[0] = r0[4] = r2[1] = r1[2]
r5[0] = r0[5] = r3[1] = r1[3]
```

```
r6[0] = r0[6] = r3[2] = r2[3]
r7[0] = r0[7] = r6[1] = r1[6] = r5[2] = r2[5] = r4[3] = r3[4]
r4[1] = r1[4]
r5[1] = r1[5]
r7[1] = r1[7]
r4[2] = r2[4]
r6[2] = r2[6]
r7[2] = r2[7] = r6[4] = r4[6]
r5[3] = r3[5]
r6[3] = r3[6]
r7[3] = r3[7] = r6[5] = r5[6]
r5[4] = r4[5]
r7[4] = r4[7]
r7[5] = r5[7]
r7[6] = r6[7]

mass = np.array([r0, r1, r2, r3, r4, r5, r6, r7])

k = [0]*N
for i in range(0,N):
    k[0] += yAverage[i]
    k[1] += yAverage[i] * x1iR[i]
    k[2] += yAverage[i] * x2iR[i]
    k[3] += yAverage[i] * x3iR[i]
    k[4] += yAverage[i] * x1iR[i] * x2iR[i]
    k[5] += yAverage[i] * x1iR[i] * x3iR[i]
    k[6] += yAverage[i] * x2iR[i] * x3iR[i]
    k[7] += yAverage[i] * x1iR[i] * x2iR[i] * x3iR[i]

vyznachnyk = [0]*8
commonVyznachnyk = r0[0]*r1[1]*r2[2]*r3[3]*r4[4]*r5[5]*r6[6]*r7[7] +
r0[1]*r1[2]*r2[3]*r3[4]*r4[5]*r5[6]*r6[7]*r7[0] +
r0[2]*r1[3]*r2[4]*r3[5]*r4[6]*r5[7]*r6[0]*r7[1] +
r0[3]*r1[4]*r2[5]*r3[6]*r4[7]*r5[0]*r6[1]*r7[2] +
r0[4]*r1[5]*r2[6]*r3[7]*r4[0]*r5[1]*r6[2]*r7[3] +
r0[5]*r1[6]*r2[7]*r3[0]*r4[1]*r5[2]*r6[3]*r7[4] +
r0[6]*r1[7]*r2[0]*r3[1]*r4[2]*r5[3]*r6[4]*r7[5] +
r0[7]*r1[0]*r2[1]*r3[2]*r4[3]*r5[4]*r6[5]*r7[6] -
(r7[0]*r6[1]*r5[2]*r4[3]*r3[4]*r2[5]*r1[6]*r0[7] +
r7[1]*r6[2]*r5[3]*r4[4]*r3[5]*r2[6]*r1[7]*r0[0] +
r7[2]*r6[3]*r5[4]*r4[5]*r3[6]*r2[7]*r1[0]*r0[1] +
r7[3]*r6[4]*r5[5]*r4[6]*r3[7]*r2[0]*r1[1]*r0[2] +
r7[4]*r6[5]*r5[6]*r4[7]*r3[0]*r2[1]*r1[2]*r0[3] +
r7[5]*r6[6]*r5[7]*r4[0]*r3[1]*r2[2]*r1[3]*r0[4] +
r7[6]*r6[7]*r5[0]*r4[1]*r3[2]*r2[3]*r1[4]*r0[5] +
r7[7]*r6[0]*r5[1]*r4[2]*r3[3]*r2[4]*r1[5]*r0[6])
vyznachnyk[0] = k[0]*r1[1]*r2[2]*r3[3]*r4[4]*r5[5]*r6[6]*r7[7] +
r0[1]*r1[2]*r2[3]*r3[4]*r4[5]*r5[6]*r6[7]*k[7] +
r0[2]*r1[3]*r2[4]*r3[5]*r4[6]*r5[7]*k[6]*r7[1] +
r0[3]*r1[4]*r2[5]*r3[6]*r4[7]*k[5]*r6[1]*r7[2] +
r0[4]*r1[5]*r2[6]*r3[7]*k[4]*r5[1]*r6[2]*r7[3] +
r0[5]*r1[6]*r2[7]*k[3]*r4[1]*r5[2]*r6[3]*r7[4] +
r0[6]*r1[7]*k[2]*r3[1]*r4[2]*r5[3]*r6[4]*r7[5] +
r0[7]*k[1]*r2[1]*r3[2]*r4[3]*r5[4]*r6[5]*r7[6] -
(k[7]*r6[1]*r5[2]*r4[3]*r3[4]*r2[5]*r1[6]*r0[7] +
r7[1]*r6[2]*r5[3]*r4[4]*r3[5]*r2[6]*r1[7]*k[0] +
r7[2]*r6[3]*r5[4]*r4[5]*r3[6]*r2[7]*k[1]*r0[1] +
r7[3]*r6[4]*r5[5]*r4[6]*r3[7]*k[2]*r1[1]*r0[2] +
r7[4]*r6[5]*r5[6]*r4[7]*k[3]*r2[1]*r1[2]*r0[3] +
r7[5]*r6[6]*r5[7]*k[4]*r3[1]*r2[2]*r1[3]*r0[4] +
r7[6]*r6[7]*k[5]*r4[1]*r3[2]*r2[3]*r1[4]*r0[5] +
r7[7]*k[6]*r5[1]*r4[2]*r3[3]*r2[4]*r1[5]*r0[6])
vyznachnyk[1] = r0[0]*k[1]*r2[2]*r3[3]*r4[4]*r5[5]*r6[6]*r7[7] +
k[0]*r1[2]*r2[3]*r3[4]*r4[5]*r5[6]*r6[7]*r7[0] +
```

```
r0[2]*r1[3]*r2[4]*r3[5]*r4[6]*r5[7]*r6[0]*k[7] +
r0[3]*r1[4]*r2[5]*r3[6]*r4[7]*r5[0]*k[6]*r7[2] +
r0[4]*r1[5]*r2[6]*r3[7]*r4[0]*k[5]*r6[2]*r7[3] +
r0[5]*r1[6]*r2[7]*r3[0]*k[4]*r5[2]*r6[3]*r7[4] +
r0[6]*r1[7]*r2[0]*k[3]*r4[2]*r5[3]*r6[4]*r7[5] +
r0[7]*r1[0]*k[2]*r3[2]*r4[3]*r5[4]*r6[5]*r7[6] -
(r7[0]*k[6]*r5[2]*r4[3]*r3[4]*r2[5]*r1[6]*r0[7] +
k[7]*r6[2]*r5[3]*r4[4]*r3[5]*r2[6]*r1[7]*r0[0] +
r7[2]*r6[3]*r5[4]*r4[5]*r3[6]*r2[7]*r1[0]*k[0] +
r7[3]*r6[4]*r5[5]*r4[6]*r3[7]*r2[0]*k[1]*r0[2] +
r7[4]*r6[5]*r5[6]*r4[7]*r3[0]*k[2]*r1[2]*r0[3] +
r7[5]*r6[6]*r5[7]*r4[0]*k[3]*r2[2]*r1[3]*r0[4] +
r7[6]*r6[7]*r5[0]*k[4]*r3[2]*r2[3]*r1[4]*r0[5] +
r7[7]*r6[0]*k[5]*r4[2]*r3[3]*r2[4]*r1[5]*r0[6])
vyznachnyk[2] = r0[0]*r1[1]*k[2]*r3[3]*r4[4]*r5[5]*r6[6]*r7[7] +
r0[1]*k[1]*r2[3]*r3[4]*r4[5]*r5[6]*r6[7]*r7[0] +
k[0]*r1[3]*r2[4]*r3[5]*r4[6]*r5[7]*r6[0]*r7[1] +
r0[3]*r1[4]*r2[5]*r3[6]*r4[7]*r5[0]*r6[1]*k[7] +
r0[4]*r1[5]*r2[6]*r3[7]*r4[0]*r5[1]*k[6]*r7[3] +
r0[5]*r1[6]*r2[7]*r3[0]*r4[1]*k[5]*r6[3]*r7[4] +
r0[6]*r1[7]*r2[0]*r3[1]*k[4]*r5[3]*r6[4]*r7[5] +
r0[7]*r1[0]*r2[1]*k[3]*r4[3]*r5[4]*r6[5]*r7[6] -
(r7[0]*r6[1]*k[5]*r4[3]*r3[4]*r2[5]*r1[6]*r0[7] +
r7[1]*k[6]*r5[3]*r4[4]*r3[5]*r2[6]*r1[7]*r0[0] +
k[7]*r6[3]*r5[4]*r4[5]*r3[6]*r2[7]*r1[0]*r0[1] +
r7[3]*r6[4]*r5[5]*r4[6]*r3[7]*r2[0]*r1[1]*k[0] +
r7[4]*r6[5]*r5[6]*r4[7]*r3[0]*r2[1]*k[1]*r0[3] +
r7[5]*r6[6]*r5[7]*r4[0]*r3[1]*k[2]*r1[3]*r0[4] +
r7[6]*r6[7]*r5[0]*r4[1]*k[3]*r2[3]*r1[4]*r0[5] +
r7[7]*r6[0]*r5[1]*k[4]*r3[3]*r2[4]*r1[5]*r0[6])
vyznachnyk[3] = r0[0]*r1[1]*r2[2]*k[3]*r4[4]*r5[5]*r6[6]*r7[7] +
r0[1]*r1[2]*k[2]*r3[4]*r4[5]*r5[6]*r6[7]*r7[0] +
r0[2]*k[1]*r2[4]*r3[5]*r4[6]*r5[7]*r6[0]*r7[1] +
k[0]*r1[4]*r2[5]*r3[6]*r4[7]*r5[0]*r6[1]*r7[2] +
r0[4]*r1[5]*r2[6]*r3[7]*r4[0]*r5[1]*r6[2]*k[7] +
r0[5]*r1[6]*r2[7]*r3[0]*r4[1]*r5[2]*k[6]*r7[4] +
r0[6]*r1[7]*r2[0]*r3[1]*r4[2]*k[5]*r6[4]*r7[5] +
r0[7]*r1[0]*r2[1]*r3[2]*k[4]*r5[4]*r6[5]*r7[6] -
(r7[0]*r6[1]*r5[2]*k[4]*r3[4]*r2[5]*r1[6]*r0[7] +
r7[1]*r6[2]*k[5]*r4[4]*r3[5]*r2[6]*r1[7]*r0[0] +
r7[2]*k[6]*r5[4]*r4[5]*r3[6]*r2[7]*r1[0]*r0[1] +
k[7]*r6[4]*r5[5]*r4[6]*r3[7]*r2[0]*r1[1]*r0[2] +
r7[4]*r6[5]*r5[6]*r4[7]*r3[0]*r2[1]*r1[2]*k[0] +
r7[5]*r6[6]*r5[7]*r4[0]*r3[1]*r2[2]*k[1]*r0[4] +
r7[6]*r6[7]*r5[0]*r4[1]*r3[2]*k[2]*r1[4]*r0[5] +
r7[7]*r6[0]*r5[1]*r4[2]*k[3]*r2[4]*r1[5]*r0[6])
vyznachnyk[4] = r0[0]*r1[1]*r2[2]*r3[3]*k[4]*r5[5]*r6[6]*r7[7] +
r0[1]*r1[2]*r2[3]*k[3]*r4[5]*r5[6]*r6[7]*r7[0] +
r0[2]*r1[3]*k[2]*r3[5]*r4[6]*r5[7]*r6[0]*r7[1] +
r0[3]*k[1]*r2[5]*r3[6]*r4[7]*r5[0]*r6[1]*r7[2] +
k[0]*r1[5]*r2[6]*r3[7]*r4[0]*r5[1]*r6[2]*r7[3] +
r0[5]*r1[6]*r2[7]*r3[0]*r4[1]*r5[2]*r6[3]*k[7] +
r0[6]*r1[7]*r2[0]*r3[1]*r4[2]*r5[3]*k[6]*r7[5] +
r0[7]*r1[0]*r2[1]*r3[2]*r4[3]*k[5]*r6[5]*r7[6] -
(r7[0]*r6[1]*r5[2]*r4[3]*k[3]*r2[5]*r1[6]*r0[7] +
r7[1]*r6[2]*r5[3]*k[4]*r3[5]*r2[6]*r1[7]*r0[0] +
r7[2]*r6[3]*k[5]*r4[5]*r3[6]*r2[7]*r1[0]*r0[1] +
r7[3]*k[6]*r5[5]*r4[6]*r3[7]*r2[0]*r1[1]*r0[2] +
k[7]*r6[5]*r5[6]*r4[7]*r3[0]*r2[1]*r1[2]*r0[3] +
r7[5]*r6[6]*r5[7]*r4[0]*r3[1]*r2[2]*r1[3]*k[0] +
r7[6]*r6[7]*r5[0]*r4[1]*r3[2]*r2[3]*k[1]*r0[5] +
r7[7]*r6[0]*r5[1]*r4[2]*r3[3]*k[2]*r1[5]*r0[6])
vyznachnyk[5] = r0[0]*r1[1]*r2[2]*r3[3]*r4[4]*k[5]*r6[6]*r7[7] +
r0[1]*r1[2]*r2[3]*r3[4]*k[4]*r5[6]*r6[7]*r7[0] +
```

```
r0[2]*r1[3]*r2[4]*k[3]*r4[6]*r5[7]*r6[0]*r7[1] +
r0[3]*r1[4]*k[2]*r3[6]*r4[7]*r5[0]*r6[1]*r7[2] +
r0[4]*k[1]*r2[6]*r3[7]*r4[0]*r5[1]*r6[2]*r7[3] +
k[0]*r1[6]*r2[7]*r3[0]*r4[1]*r5[2]*r6[3]*r7[4] +
r0[6]*r1[7]*r2[0]*r3[1]*r4[2]*r5[3]*r6[4]*k[7] +
r0[7]*r1[0]*r2[1]*r3[2]*r4[3]*r5[4]*k[6]*r7[6] -
(r7[0]*r6[1]*r5[2]*r4[3]*r3[4]*k[2]*r1[6]*r0[7] +
r7[1]*r6[2]*r5[3]*r4[4]*k[3]*r2[6]*r1[7]*r0[0] +
r7[2]*r6[3]*r5[4]*k[4]*r3[6]*r2[7]*r1[0]*r0[1] +
r7[3]*r6[4]*k[5]*r4[6]*r3[7]*r2[0]*r1[1]*r0[2] +
r7[4]*k[6]*r5[6]*r4[7]*r3[0]*r2[1]*r1[2]*r0[3] +
k[7]*r6[6]*r5[7]*r4[0]*r3[1]*r2[2]*r1[3]*r0[4] +
r7[6]*r6[7]*r5[0]*r4[1]*r3[2]*r2[3]*r1[4]*k[0] +
r7[7]*r6[0]*r5[1]*r4[2]*r3[3]*r2[4]*k[1]*r0[6])
vyznachnyk[6] = r0[0]*r1[1]*r2[2]*r3[3]*r4[4]*r5[5]*k[6]*r7[7] +
r0[1]*r1[2]*r2[3]*r3[4]*r4[5]*k[5]*r6[7]*r7[0] +
r0[2]*r1[3]*r2[4]*r3[5]*k[4]*r5[7]*r6[0]*r7[1] +
r0[3]*r1[4]*r2[5]*k[3]*r4[7]*r5[0]*r6[1]*r7[2] +
r0[4]*r1[5]*k[2]*r3[7]*r4[0]*r5[1]*r6[2]*r7[3] +
r0[5]*k[1]*r2[7]*r3[0]*r4[1]*r5[2]*r6[3]*r7[4] +
k[0]*r1[7]*r2[0]*r3[1]*r4[2]*r5[3]*r6[4]*r7[5] +
r0[7]*r1[0]*r2[1]*r3[2]*r4[3]*r5[4]*r6[5]*k[7] -
(r7[0]*r6[1]*r5[2]*r4[3]*r3[4]*r2[5]*k[1]*r0[7] +
r7[1]*r6[2]*r5[3]*r4[4]*r3[5]*k[2]*r1[7]*r0[0] +
r7[2]*r6[3]*r5[4]*r4[5]*k[3]*r2[7]*r1[0]*r0[1] +
r7[3]*r6[4]*r5[5]*k[4]*r3[7]*r2[0]*r1[1]*r0[2] +
r7[4]*r6[5]*k[5]*r4[7]*r3[0]*r2[1]*r1[2]*r0[3] +
r7[5]*k[6]*r5[7]*r4[0]*r3[1]*r2[2]*r1[3]*r0[4] +
k[7]*r6[7]*r5[0]*r4[1]*r3[2]*r2[3]*r1[4]*r0[5] +
r7[7]*r6[0]*r5[1]*r4[2]*r3[3]*r2[4]*r1[5]*k[0])
vyznachnyk[7] = r0[0]*r1[1]*r2[2]*r3[3]*r4[4]*r5[5]*r6[6]*k[7] +
r0[1]*r1[2]*r2[3]*r3[4]*r4[5]*r5[6]*k[6]*r7[0] +
r0[2]*r1[3]*r2[4]*r3[5]*r4[6]*k[5]*r6[0]*r7[1] +
r0[3]*r1[4]*r2[5]*r3[6]*k[4]*r5[0]*r6[1]*r7[2] +
r0[4]*r1[5]*r2[6]*k[3]*r4[0]*r5[1]*r6[2]*r7[3] +
r0[5]*r1[6]*k[2]*r3[0]*r4[1]*r5[2]*r6[3]*r7[4] +
r0[6]*k[1]*r2[0]*r3[1]*r4[2]*r5[3]*r6[4]*r7[5] +
k[0]*r1[0]*r2[1]*r3[2]*r4[3]*r5[4]*r6[5]*r7[6] -
(r7[0]*r6[1]*r5[2]*r4[3]*r3[4]*r2[5]*r1[6]*k[0] +
r7[1]*r6[2]*r5[3]*r4[4]*r3[5]*r2[6]*k[1]*r0[0] +
r7[2]*r6[3]*r5[4]*r4[5]*r3[6]*k[2]*r1[0]*r0[1] +
r7[3]*r6[4]*r5[5]*r4[6]*k[3]*r2[0]*r1[1]*r0[2] +
r7[4]*r6[5]*r5[6]*k[4]*r3[0]*r2[1]*r1[2]*r0[3] +
r7[5]*r6[6]*k[5]*r4[0]*r3[1]*r2[2]*r1[3]*r0[4] +
r7[6]*k[6]*r5[0]*r4[1]*r3[2]*r2[3]*r1[4]*r0[5] +
k[7]*r6[0]*r5[1]*r4[2]*r3[3]*r2[4]*r1[5]*r0[6])

result = [0]*N
for i in range(0,N):
    result[i] = vyznachnyk[i]/commonVyznachnyk
print('\nКоефіцієнти лінійного рівняння регресії:\n', result)

devariation = [0]*N
Sdevariation = 0
for i in range(0,N):
    devariation[i] = ((yAverage[i] - y1[i])**2 + (yAverage[i] - y2[i])**2 +
(yAverage[i] - y3[i])**2)/3
    Sdevariation += devariation[i]
Gp = max(devariation)/Sdevariation

print('\nПеревірка однорідності дисперсії за критерієм Кохрена:')
print('Gp =',Gp, '\nGt =', Gt)
if Gp < f.ppf(0.95, Gt, N):
    print('Gp <= Gt  Дисперсія однорідна')
```

```
else:
    print('Gp > Gt  Дисперся не однорідна, при m =', m, 'Потрібно збільшити
m')

devariationVidtvoriuvanosty = Sdevariation/N
s2Betta = devariationVidtvoriuvanosty/(N*m)
sBetta = math.sqrt(s2Betta)

x1i = [-1, -1, -1, -1, 1, 1, 1, 1]
x2i = [-1, -1, 1, 1, -1, -1, 1, 1]
x3i = [-1, 1, -1, 1, -1, 1, -1, 1]

b = [0]*N
for i in range(0,N):
    b[0] += yAverage[i]
    b[1] += yAverage[i]*x1i[i]
    b[2] += yAverage[i]*x2i[i]
    b[3] += yAverage[i]*x3i[i]
    b[4] += yAverage[i]*x1i[i]*x2i[i]
    b[5] += yAverage[i]*x1i[i]*x3i[i]
    b[6] += yAverage[i]*x2i[i]*x3i[i]
    b[7] += yAverage[i]*x1i[i]*x2i[i]*x3i[i]

print('\nОцінка значимості коефіцієнтів регресії згідно критерію Стьюдента')
t = [0]*N
for i in range(0,N):
    t[i] = abs(b[i])/(sBetta)

Tt = (m - 1)*N
temp = [0]*N
coef_1 = []
coef_2 = []
for i in range(0,N):
    if t[i] < f.ppf(0.95, Tt, N):
        print(' b[',i,'] - не значний коефіцієнт')
        temp[i] = 0
    else:
        print(' b[',i,'] - значний коефіцієнт')
        temp[i] = b[i]
        d +=1

y_2 = [0]*N
for i in range(0,N):
    y_2[i] = temp[0] + temp[1]*x1i[i] + temp[2]*x2i[i] + temp[3]*x3i[i] +
temp[4]*x1i[i]*x2i[i] + temp[5]*x1i[i]*x3i[i] + temp[6]*x2i[i]*x3i[i] +
temp[7]*x1i[i]*x2i[i]*x3i[i]

sum = 0
for i in range(0,N):
    sum +=(y_2[i] - yAverage[i])**2
sAdecvatnosti = (m/(N - d))*(sum/10**5)
print('\nКритерій Фішера:')

Fp = (sAdecvatnosti)/(devariationVidtvoriuvanosty)
print('d=',devariationVidtvoriuvanosty, 's=',sAdecvatnosti)
print('Fp =', Fp)
print('Ft =', Ft)

if Fp < f.ppf(0.95, Fp, Tt):
    print('Fp <= Ft => Рівняння регресії адекватне щодо оригіналу при рівні
значимості 0,05')
else:
    print('Fp > Ft => Рівняння регресії НЕадекватне щодо оригіналу при рівні
значимості 0,05')
```

Результати роботи програми

```
X:
 [[  2  34  31]
 [ -2  38  35]
 [ -3  73  31]
 [-10  67  36]
 [ 10  64  36]
 [ 12  59  30]
 [ 18  68  34]
 [ 29  48  36]]

Y:
 [[222 220 248]
 [222 228 239]
 [228 231 234]
 [233 234 231]
 [243 216 234]
 [218 228 219]
 [234 241 248]
 [228 227 237]]

Середні значення Y:
 [230.0, 229.66666666666666, 231.0, 232.66666666666666, 231.0, 221.66666666666666, 241.0, 230.66666666666666]

Коефіцієнти лінійного рівняння регресії:
 [230.95833333333334, 8.504805591961555, 3.874803451819386, 6.838558281083856, 0.14679991855455626, 0.23745910339932919, 0.1148094813092131, 0.2352286834839176]

Перевірка однорідності дисперсії за критерієм Кохрена:
Gp = 0.38832891246684353
Gt = 0.5157
Gp <= Gt  Дисперсія однорідна
Середні значення Y:
 [230.0, 229.66666666666666, 231.0, 232.66666666666666, 231.0, 221.66666666666666, 241.0, 230.66666666666666]

Коефіцієнти лінійного рівняння регресії:
 [230.95833333333334, 8.504805591961555, 3.874803451819386, 6.838558281083856, 0.14679991855455626, 0.23745910339932919, 0.1148094813092131, 0.2352286834839176]

Перевірка однорідності дисперсії за критерієм Кохрена:
Gp = 0.38832891246684353
Gt = 0.5157
Gp <= Gt  Дисперсія однорідна

Оцінка значимості коефіцієнтів регресії згідно критерію Стьюдента
 b[ 0 ] - значний коефіцієнт
 b[ 1 ] - не значний коефіцієнт
 b[ 2 ] - значний коефіцієнт
 b[ 3 ] - значний коефіцієнт
 b[ 4 ] - значний коефіцієнт
 b[ 5 ] - значний коефіцієнт
 b[ 6 ] - не значний коефіцієнт
 b[ 7 ] - не значний коефіцієнт

Критерій Фішера:
d= 52.36111111111111 s= 209.1934611111111
Fp = 3.9952066843501326
Ft = 2.7
Fp > Ft => Рівняння регресії НЕадекватне щодо оригіналу при рівні значимості 0,05

Process finished with exit code 0
```