Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №4

з дисципліни МОПЕ

на тему:

# «Проведення трьохфакторного експерименту

# при використанні рівняння регресії з урахуванням ефекту взаємодії.»

Виконав:

студент групи ІВ-82

Гевеленко Назар Романович

Залікова книжка № 8205

Варіант: 204

Перевірив:

Регіда П.Г.

Київ 2020

****

**Код програми:**

import math

import numpy as np

from scipy.stats import t,f

import random as r

from functools import partial

import prettytable as p

def make\_exp(m):

    return [[r.randint(math.floor(Y\_min), math.floor(Y\_max)) for \_ in range(m)] for i in range(N)]

def get\_dispersion(y\_aver, y):

    return sum([(i-y\_aver)\*\*2 for i in y])/len(y)

def get\_average(y):

    return sum(y)/len(y)

def count\_F(a, b):

    return max([a, b])/min([a, b])

def y\_regr\_norm(x1, x2, x3):

        return b0 + x1\*b1 + x2\*b2 + x3\*b3

def y\_regr\_abs(x1, x2, x3):

        return a0 + a1\*x1 + a2\*x2 + a3\*x3

def get\_fisher\_critical(prob,f3, f4):

    for i in [j\*0.001 for j in range(int(10/0.001))]:

        if abs(f.cdf(i,f4,f3)-prob) < 0.0001:

            return i

def get\_student\_critical(prob, f3):

    for i in [j\*0.0001 for j in range(int(5/0.0001))]:

        if abs(t.cdf(i,f3)-(0.5 + prob/0.1\*0.05)) < 0.000005:

            return i

def get\_cohren\_critical(prob, f1, f2):

    f\_crit = f.isf((1-prob)/f2, f1, (f2-1)\*f1)

    return f\_crit/(f\_crit+f2-1)

def is\_significant\_coef(t, tkr):

    return t > tkr

def craete\_table\_norm():

        global table1, X\_norm, Y\_exp

        table1 = p.PrettyTable()

        for i in range(len(X\_norm)):

            table1.add\_column(X\_types[i], X\_norm[i])

        for i in range(m):

            table1.add\_column("Y{0}".format(i+1), [j[i] for j in Y\_exp])

def create\_table\_abs():

        global table2, X\_abs, Y\_exp

        table2 = p.PrettyTable()

        for i in range(len(X\_abs)):

            table2.add\_column(X\_types[i], X\_abs[i])

        for i in range(m):

            table2.add\_column("Y{0}".format(i+1), [j[i] for j in Y\_exp])

def get\_aprox\_value\_abs(x1, x2, x3, x12, x13, x23, x123):

    return b\_abs[0]+b\_abs[1]\*x1+b\_abs[2]\*x2+b\_abs[3]\*x3+b\_abs[4]\*x12+b\_abs[5]\*x13+b\_abs[6]\*x23+b\_abs[7]\*x123

def get\_aprox\_value\_norm(x1, x2, x3, x12, x13, x23, x123):

    return b\_norm[0]+b\_norm[1]\*x1+b\_norm[2]\*x2+b\_norm[3]\*x3+b\_norm[4]\*x12+b\_norm[5]\*x13+b\_norm[6]\*x23+b\_norm[7]\*x123

m = 3

prob = 0.95

X1\_min = -20

X1\_max = 30

X2\_min = -25

X2\_max = 10

X3\_min = -25

X3\_max = -20

k = 3

X\_ranges = [[X1\_min, X1\_max], [X2\_min, X2\_max], [X3\_min, X3\_max]]

X\_types = ["X1", "X2", "X3", "X12", "X13", "X23", "X123"]

Xcp\_max = (X1\_max + X2\_max + X3\_max) / 3

Xcp\_min = (X1\_min + X2\_min + X3\_min) / 3

Y\_min = 200 + Xcp\_min

Y\_max = 200 + Xcp\_max

flag0 = True

ct = 3 #кількість спроб

while(flag0):

    flag0 = False

    X0\_norm = [1, 1, 1, 1]

    X1\_norm = [-1, -1, 1, 1]

    X2\_norm = [-1, 1, -1, 1]

    X3\_norm = [-1, 1, 1, -1]

    X\_norm = [X1\_norm, X2\_norm, X3\_norm]

    N = len(X1\_norm)

    X\_abs = [[max(X\_ranges[j]) if i == 1 else min(X\_ranges[j]) for i in X\_norm[j]] for j in range(k)]

    for i in range(len(X\_abs)):

        print("Абсолютні Х{0}: {1}".format(i+1, X\_abs[i]))

    Y\_exp = make\_exp(m)

    flag = True

    while(flag):

        table1 = p.PrettyTable()

        craete\_table\_norm()

        print("Нормалізована матриця:\n", table1)

        mx\_norm\_list = [get\_average(i) for i in X\_norm]

        y\_aver = [get\_average(i) for i in Y\_exp]

        my = get\_average(y\_aver)

        a1 = get\_average([X\_norm[0][i]\*y\_aver[i] for i in range(N)])

        a2 = get\_average([X\_norm[1][i]\*y\_aver[i] for i in range(N)])

        a3 = get\_average([X\_norm[2][i]\*y\_aver[i] for i in range(N)])

        a11 = get\_average([X\_norm[0][i]\*\*2 for i in range(N)])

        a22 = get\_average([X\_norm[1][i]\*\*2 for i in range(N)])

        a33 = get\_average([X\_norm[2][i]\*\*2 for i in range(N)])

        a12 = get\_average([X\_norm[0][i]\*X\_norm[1][i] for i in range(N)])

        a13 = get\_average([X\_norm[0][i]\*X\_norm[2][i] for i in range(N)])

        a23 = get\_average([X\_norm[1][i]\*X\_norm[2][i] for i in range(N)])

        a21 = a12

        a31 = a13

        a32 = a23

        znam = np.array([[1, mx\_norm\_list[0], mx\_norm\_list[1],  mx\_norm\_list[2]],

                        [mx\_norm\_list[0], a11, a12, a13],

                        [mx\_norm\_list[1], a12, a22, a32],

                        [mx\_norm\_list[2], a13, a23, a33]])

        b0\_matr = np.array([[my, mx\_norm\_list[0], mx\_norm\_list[1],  mx\_norm\_list[2]],

                        [a1, a11, a12, a13],

                        [a2, a12, a22, a32],

                        [a3, a13, a23, a33]])

        b1\_matr = np.array([[1, my, mx\_norm\_list[1],  mx\_norm\_list[2]],

                        [mx\_norm\_list[0], a1, a12, a13],

                        [mx\_norm\_list[1], a2, a22, a32],

                        [mx\_norm\_list[2], a3, a23, a33]])

        b2\_matr = np.array([[1, mx\_norm\_list[0], my,  mx\_norm\_list[2]],

                        [mx\_norm\_list[0], a11, a1, a13],

                        [mx\_norm\_list[1], a12, a2, a32],

                        [mx\_norm\_list[2], a13, a3, a33]])

        b3\_matr = np.array([[1, mx\_norm\_list[0], mx\_norm\_list[1],  my],

                        [mx\_norm\_list[0], a11, a12, a1],

                        [mx\_norm\_list[1], a12, a22, a2],

                        [mx\_norm\_list[2], a13, a23, a3]])

        zanm\_value = np.linalg.det(znam)

        b0 = np.linalg.det(b0\_matr)/zanm\_value

        b1 = np.linalg.det(b1\_matr)/zanm\_value

        b2 = np.linalg.det(b2\_matr)/zanm\_value

        b3 = np.linalg.det(b3\_matr)/zanm\_value

        print("Рівняння регресії для нормованих значень:\ny = {0} + {1}\*x1 + {2}\*x2 + {3}\*x3".format(b0, b1, b2, b3))

        print("Підставимо нормованi значення Х в рівння регресії")

        print("Р-ня регресії для Х11, Х21, Х31 =",y\_regr\_norm(X\_norm[0][0], X\_norm[1][0], X\_norm[2][0]))

        print("Середнє y1 =", y\_aver[0])

        print("Р-ня регресії для Х12, Х22, Х32 =", y\_regr\_norm(X\_norm[0][1], X\_norm[1][1], X\_norm[2][1]))

        print("Середнє y2 =", y\_aver[1])

        print("Р-ня регресії для Х13, Х23, Х33 =", y\_regr\_norm(X\_norm[0][2], X\_norm[1][2], X\_norm[2][2]))

        print("Середнє y3 =", y\_aver[2])

        delt\_x1 = (X1\_max - X1\_min)/2

        delt\_x2 = (X2\_max - X2\_min)/2

        delt\_x3 = (X3\_max - X3\_min)/2

        x10 = (X1\_max + X1\_min)/2

        x20 = (X2\_max + X2\_min)/2

        x30 = (X3\_max + X3\_min)/2

        a0 = b0 - b1\*(x10/delt\_x1) - b2\*(x20/delt\_x2) - b3\*(x30/delt\_x3)

        a1 = b1/delt\_x1

        a2 = b2/delt\_x2

        a3 = b3/delt\_x3

        print("Рівняння регресії для абсолютних значень:\ny = {0} + {1}\*x1 + {2}\*x2 + {3}\*x3".format(a0, a1, a2, a3))

        print("Підставимо абсолютні значення Х в рівння регресії")

        print("Р-ня регресії для Х11, Х21, Х31 =",y\_regr\_abs(X\_abs[0][0], X\_abs[1][0], X\_abs[2][0]))

        print("Середнє y1 =", y\_aver[0])

        print("Р-ня регресії для Х12, Х22, Х32 =", y\_regr\_abs(X\_abs[0][1], X\_abs[1][1], X\_abs[2][1]))

        print("Середнє y2 =", y\_aver[1])

        print("Р-ня регресії для Х13, Х23, Х33 =", y\_regr\_abs(X\_abs[0][2], X\_abs[1][2], X\_abs[2][2]))

        print("Середнє y3 =", y\_aver[2])

        print("Р-ня регресії для Х14, Х24, Х34 =", y\_regr\_abs(X\_abs[0][3], X\_abs[1][3], X\_abs[2][3]))

        print("Середнє y3 =", y\_aver[3])

        #Кохрен

        y\_disps = [get\_dispersion(y\_aver[i], Y\_exp[i]) for i in range(N)]

        f1 = m - 1

        f2 = N

        f3 = f2\*f1

        Gp = max(y\_disps)/sum(y\_disps)

        Gkr = get\_cohren\_critical(prob, f1, f2)

        print("--------------------------------------------------------")

        if(Gkr > Gp):

            print("Gkr = {0} > Gp = {1} ---> Дисперсії однорідні".format(Gkr, Gp))

            flag = False

        else:

            print("Gkr = {0} < Gp = {1} ---> Дисперсії неоднорідні, збільшимо m і проведемо розрахунки".format(Gkr, Gp))

            Y\_exp[0].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

            Y\_exp[1].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

            Y\_exp[2].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

            Y\_exp[3].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

            m += 1

    #Стьюдент

    S2B = sum(y\_disps)/N

    S2b = S2B/(N\*m)

    Sb = math.sqrt(S2b)

    beta0 = sum([y\_aver[i]\*X0\_norm[i] for i in range(N)])/N

    beta1 = sum([y\_aver[i]\*X1\_norm[i] for i in range(N)])/N

    beta2 = sum([y\_aver[i]\*X2\_norm[i] for i in range(N)])/N

    beta3 = sum([y\_aver[i]\*X3\_norm[i] for i in range(N)])/N

    t0 = abs(beta0)/Sb

    t1 = abs(beta1)/Sb

    t2 = abs(beta2)/Sb

    t3 = abs(beta3)/Sb

    tkr = get\_student\_critical(prob, f3)

    d = sum([1 if is\_significant\_coef(tkr, i) else 0 for i in [t0, t1, t2, t3]])

    a0 = a0 if is\_significant\_coef(t0, tkr) else 0

    a1 = a1 if is\_significant\_coef(t1, tkr) else 0

    a2 = a2 if is\_significant\_coef(t2, tkr) else 0

    a3 = a3 if is\_significant\_coef(t3, tkr) else 0

    y\_new = [y\_regr\_abs(X\_abs[0][i], X\_abs[1][i], X\_abs[2][i]) for i in range(N)]

    print("--------------------------------------------------------\nПісля перевірки значимості коефіцієнтів: ")

    print("Кількість значимих коефіцієнтів:", d)

    print("Рівняння регресії для абсолютних значень:\ny = {0} + {1}\*x1 + {2}\*x2 + {3}\*x3".format(a0, a1, a2, a3))

    print("Підставимо абсолютні значення Х в рівння регресії")

    print("Р-ня регресії для Х11, Х21, Х31 =", y\_new[0])

    print("Р-ня регресії для Х12, Х22, Х32 =", y\_new[1])

    print("Р-ня регресії для Х13, Х23, Х33 =", y\_new[2])

    print("Р-ня регресії для Х14, Х24, Х34 =", y\_new[3])

    #Фішер

    print("--------------------------------------------------------")

    f4 = N - d

    S2ad = (m/(N-d))\*sum([(y\_new[i] - y\_aver[i])\*\*2 for i in range(N)])

    Fp = S2ad/S2b

    Fkr = get\_fisher\_critical(prob, f3, f4)

    if(Fkr > Fp):

        print("Fkr = {0} > Fp = {1} ---> Р-ня адекватне оригіналу".format(Fkr, Fp))

    else:

        print("Рівняння неадекватне переходимо до ефекту взаємодії")

        print("-"\*45)

        X1\_norm = [-1, -1, -1, -1, 1, 1, 1, 1]

        X2\_norm = [-1, -1, 1, 1, -1, -1, 1, 1]

        X3\_norm = [-1, 1, -1, 1, -1, 1, -1, 1]

        N = len(X1\_norm)

        Xcp\_max = (X1\_max + X2\_max + X3\_max) / 3

        Xcp\_min = (X1\_min + X2\_min + X3\_min) / 3

        X\_norm = [X1\_norm, X2\_norm, X3\_norm]

        X\_abs = [[max(X\_ranges[j]) if i == 1 else min(X\_ranges[j]) for i in X\_norm[j]] for j in range(k)]

        Y\_exp = make\_exp(m)

        flag1 = True

        while(flag1):

            table1 = p.PrettyTable()

            X12\_norm = [X1\_norm[i]\*X2\_norm[i] for i in range(N)]

            X13\_norm = [X1\_norm[i]\*X3\_norm[i] for i in range(N)]

            X23\_norm = [X2\_norm[i]\*X3\_norm[i] for i in range(N)]

            X123\_norm = [X1\_norm[i]\*X2\_norm[i]\*X3\_norm[i] for i in range(N)]

            X\_norm.append(X12\_norm)

            X\_norm.append(X13\_norm)

            X\_norm.append(X23\_norm)

            X\_norm.append(X123\_norm)

            craete\_table\_norm()

            print("Нормалізована матриця:\n", table1)

            X\_abs = [[max(X\_ranges[j]) if i == 1 else min(X\_ranges[j]) for i in X\_norm[j]] for j in range(k)]

            X12\_abs, X13\_abs, X23\_abs, X123\_abs = [], [], [], []

            for i in range(N):

                if X1\_norm[i] == 1:

                    x1 = max(X\_ranges[0])

                else:

                    x1 = min(X\_ranges[0])

                if X2\_norm[i] == 1:

                    x2 = max(X\_ranges[1])

                else:

                    x2 = min(X\_ranges[1])

                X12\_abs.append(x1\*x2)

            X\_abs.append(X12\_abs)

            for i in range(N):

                if X1\_norm[i] == 1:

                    x1 = max(X\_ranges[0])

                else:

                    x1 = min(X\_ranges[0])

                if X3\_norm[i] == 1:

                    x3 = max(X\_ranges[2])

                else:

                    x3 = min(X\_ranges[2])

                X13\_abs.append(x1\*x3)

            X\_abs.append(X13\_abs)

            for i in range(N):

                if X3\_norm[i] == 1:

                    x3 = max(X\_ranges[2])

                else:

                    x3 = min(X\_ranges[2])

                if X2\_norm[i] == 1:

                    x2 = max(X\_ranges[1])

                else:

                    x2 = min(X\_ranges[1])

                X23\_abs.append(x3\*x2)

            X\_abs.append(X23\_abs)

            for i in range(N):

                if X1\_norm[i] == 1:

                    x1 = max(X\_ranges[0])

                else:

                    x1 = min(X\_ranges[0])

                if X2\_norm[i] == 1:

                    x2 = max(X\_ranges[1])

                else:

                    x2 = min(X\_ranges[1])

                if X3\_norm[i] == 1:

                    x3 = max(X\_ranges[2])

                else:

                    x3 = min(X\_ranges[2])

                X123\_abs.append(x1\*x2\*x3)

            X\_abs.append(X123\_abs)

            table2 = p.PrettyTable()

            create\_table\_abs()

            print("Абсолютна матриця:\n", table2)

            X1\_abs = X\_abs[0]

            X2\_abs = X\_abs[1]

            X3\_abs = X\_abs[2]

            mj0 = [N]

            for i in [sum(i) for i in X\_abs]:

                mj0.append(i)

            mj1 = [sum(X1\_abs),

                    sum([i\*\*2 for i in X1\_abs]),

                    sum(X12\_abs),

                    sum(X13\_abs),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum(X123\_abs),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i] for i in range(N)])]

            mj2 = [sum(X2\_abs),

                    sum(X12\_abs),

                    sum([i\*\*2 for i in X2\_abs]),

                    sum(X23\_abs),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2 for i in range(N)]),

                    sum(X123\_abs),

                    sum([X3\_abs[i]\*X2\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)])]

            mj3 = [sum(X3\_abs),

                    sum(X13\_abs),

                    sum(X23\_abs),

                    sum([i\*\*2 for i in X3\_abs]),

                    sum(X123\_abs),

                    sum([X1\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X3\_abs[i]\*\*2\*X2\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)])

                    ]

            mj4 = [sum(X12\_abs),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2 for i in range(N)]),

                    sum(X123\_abs),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)])

                    ]

            mj5 = [sum(X13\_abs),

                    sum([X1\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum(X123\_abs),

                    sum([X1\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)])

                    ]

            mj6 = [sum(X23\_abs),

                    sum(X123\_abs),

                    sum([X3\_abs[i]\*X2\_abs[i]\*\*2 for i in range(N)]),

                    sum([X3\_abs[i]\*\*2\*X2\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X2\_abs[i]\*\*2\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i]\*\*2 for i in range(N)])

                    ]

            mj7 = [sum(X123\_abs),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*\*2\*X3\_abs[i] for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*X2\_abs[i]\*\*2\*X3\_abs[i]\*\*2 for i in range(N)]),

                    sum([X1\_abs[i]\*\*2\*X2\_abs[i]\*\*2\*X3\_abs[i]\*\*2 for i in range(N)])

                    ]

            y\_aver = [get\_average(i) for i in Y\_exp]

            m\_list = [mj0, mj1, mj2, mj3, mj4, mj5, mj6, mj7]

            m0j = [i[0] for i in m\_list]

            m1j = [i[1] for i in m\_list]

            m2j = [i[2] for i in m\_list]

            m3j = [i[3] for i in m\_list]

            m4j = [i[4] for i in m\_list]

            m5j = [i[5] for i in m\_list]

            m6j = [i[6] for i in m\_list]

            m7j = [i[7] for i in m\_list]

            kf = [sum(y\_aver),

                    sum([y\_aver[i]\*X1\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X2\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X3\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X12\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X13\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X23\_abs[i] for i in range(N)]),

                    sum([y\_aver[i]\*X123\_abs[i] for i in range(N)])

                    ]

            b\_abs = list(np.linalg.solve([mj0, mj1, mj2, mj3, mj4, mj5, mj6, mj7], kf))

            print("-"\*45)

            print("Рівняння регресії для абсолютних значень: {0} + {1}\*x1 + {2}\*x2 + {3}\*x3 + {4}\*X12 + {5}\*X13 + {6}\*x23 + {7}\*x123".format(b\_abs[0],b\_abs[1],b\_abs[2],b\_abs[3],b\_abs[4],b\_abs[5],b\_abs[6],b\_abs[7]))

            print("Перевіримо")

            print("Y1 середнє =", y\_aver[0])

            a = [X\_abs[i][0] for i in range(7)]

            print("Y1 з рівняння =", get\_aprox\_value\_abs(\*a))

            print("Y4 середнє =", y\_aver[3])

            b = [X\_abs[i][3] for i in range(7)]

            print("Y4 з рівняння =", get\_aprox\_value\_abs(\*b))

            b\_norm = [get\_average(y\_aver),

                        get\_average([y\_aver[i]\*X1\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X2\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X3\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X12\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X13\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X23\_norm[i] for i in range(N)]),

                        get\_average([y\_aver[i]\*X123\_norm[i] for i in range(N)])

                        ]

            print("-"\*45)

            print("Рівняння регресії для нормалізовних значень: {0} + {1}\*x1 + {2}\*x2 + {3}\*x3 + {4}\*X12 + {5}\*X13 + {6}\*x23 + {7}\*x123".format(b\_norm[0],b\_norm[1],b\_norm[2],b\_norm[3],b\_norm[4],b\_norm[5],b\_norm[6],b\_norm[7]))

            print("Перевіримо")

            print("Y1 середнє =", y\_aver[0])

            a = [X\_norm[i][0] for i in range(7)]

            print("Y1 з рівняння =", get\_aprox\_value\_norm(\*a))

            print("Y4 середнє =", y\_aver[3])

            b = [X\_norm[i][3] for i in range(7)]

            print("Y4 з рівняння =", get\_aprox\_value\_norm(\*b))

            y\_disps = [get\_dispersion(y\_aver[i], Y\_exp[i]) for i in range(N)]

            f1 = m - 1

            f2 = N

            f3 = f2\*f1

            Gp = max(y\_disps)/sum(y\_disps)

            Gkr = get\_cohren\_critical(prob, f1, f2)

            print("--------------------------------------------------------")

            if(Gkr > Gp):

                print("Gkr = {0} > Gp = {1} ---> Дисперсії однорідні".format(Gkr, Gp))

                flag1 = False

            else:

                print("Gkr = {0} < Gp = {1} ---> Дисперсії неоднорідні, збільшимо m і проведемо розрахунки".format(Gkr, Gp))

                Y\_exp[0].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[1].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[2].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[3].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[4].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[5].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[6].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                Y\_exp[7].append(r.randint(math.floor(Y\_min), math.floor(Y\_max)))

                m += 1

        #Стьюдент

        S2B = sum(y\_disps)/N

        S2b = S2B/(N\*m)

        Sb = math.sqrt(S2b)

        beta0 = get\_average(y\_aver)

        beta1 = sum([y\_aver[i]\*X1\_norm[i] for i in range(N)])/N

        beta2 = sum([y\_aver[i]\*X2\_norm[i] for i in range(N)])/N

        beta3 = sum([y\_aver[i]\*X3\_norm[i] for i in range(N)])/N

        beta4 = sum([y\_aver[i]\*X12\_norm[i] for i in range(N)])/N

        beta5 = sum([y\_aver[i]\*X13\_norm[i] for i in range(N)])/N

        beta6 = sum([y\_aver[i]\*X23\_norm[i] for i in range(N)])/N

        beta7 = sum([y\_aver[i]\*X123\_norm[i] for i in range(N)])/N

        t0 = abs(beta0)/Sb

        t1 = abs(beta1)/Sb

        t2 = abs(beta2)/Sb

        t3 = abs(beta3)/Sb

        t4 = abs(beta4)/Sb

        t5 = abs(beta5)/Sb

        t6 = abs(beta6)/Sb

        t7 = abs(beta7)/Sb

        tkr = get\_student\_critical(prob, f3)

        d = sum([1 if is\_significant\_coef(i, tkr) else 0 for i in [t0, t1, t2, t3, t4, t5, t6, t7]])

        b\_abs[0] = b\_abs[0] if is\_significant\_coef(t0, tkr) else 0

        b\_abs[1] = b\_abs[1] if is\_significant\_coef(t1, tkr) else 0

        b\_abs[2] = b\_abs[2] if is\_significant\_coef(t2, tkr) else 0

        b\_abs[3] = b\_abs[3] if is\_significant\_coef(t3, tkr) else 0

        b\_abs[4] = b\_abs[4] if is\_significant\_coef(t4, tkr) else 0

        b\_abs[5] = b\_abs[5] if is\_significant\_coef(t5, tkr) else 0

        b\_abs[6] = b\_abs[6] if is\_significant\_coef(t6, tkr) else 0

        b\_abs[7] = b\_abs[7] if is\_significant\_coef(t7, tkr) else 0

        y\_new = [get\_aprox\_value\_abs(X1\_abs[i], X2\_abs[i], X3\_abs[i], X12\_abs[i], X13\_abs[i], X23\_abs[i], X123\_abs[i]) for i in range(N)]

        print("-"\*45)

        print("Після перевірки значимості коефіцієнтів: ")

        print("Кількість значимих коефіцієнтів:", d)

        print("Рівняння регресії для абсолютних значень:\ny = {0} + {1}\*x1 + {2}\*x2 + {3}\*x3 + {4}\*x12 + {5}\*x13 + {6}\*x23 + {7}\*x123".format(b\_abs[0], b\_abs[1], b\_abs[2], b\_abs[3], b\_abs[4], b\_abs[5], b\_abs[6], b\_abs[7]))

        print("Підставимо абсолютні значення Х в рівння регресії")

        print("Y1 середнє =", y\_aver[0])

        a = [X\_abs[i][0] for i in range(7)]

        print("Y1 з рівняння =", get\_aprox\_value\_abs(\*a))

        print("Y4 середнє =", y\_aver[3])

        b = [X\_abs[i][3] for i in range(7)]

        print("Y4 з рівняння =", get\_aprox\_value\_abs(\*b))

        #Фішер

        print("-"\*45)

        f4 = N - d

        S2ad = abs((m/(N-d))\*sum([(y\_new[i] - y\_aver[i])\*\*2 for i in range(N)]))

        Fp = S2ad/S2b

        Fkr = get\_fisher\_critical(prob, f3, f4)

        if(Fkr > Fp):

            print("Fkr = {0} > Fp = {1} ---> Р-ня адекватне оригіналу".format(Fkr, Fp))

            flag0 = False

        else:

            print("Fkr = {0} < Fp = {1} ---> Р-ня неадекватне оригіналу".format(Fkr, Fp))

            flag0 = ct!=0

            ct-=1