Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Методи оптимізації та планування експерименту**

Лабораторна робота №5:

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ КВАДРАТИЧНИХ ЧЛЕНІВ(ЦЕНТРАЛЬНИЙ ОРТОГОНАЛЬНИЙ КОМПОЗИЦІЙНИЙ ПЛАН)»

Виконала:

студентка групи ІВ-82

Чуясова Тетяна

Залікова книжка № 8228

Перевірив

Регіда П. Г.

Київ 2020р.

**Тема:** ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ КВАДРАТИЧНИХ ЧЛЕНІВ(ЦЕНТРАЛЬНИЙ ОРТОГОНАЛЬНИЙ КОМПОЗИЦІЙНИЙ ПЛАН).

**Мета:** провести трьохфакторний експеремент з урахуванням квадратичних членів, використовуючи центральний ортогональний композиційний план. Знайти рівняння регресії адекватне об'єкту.

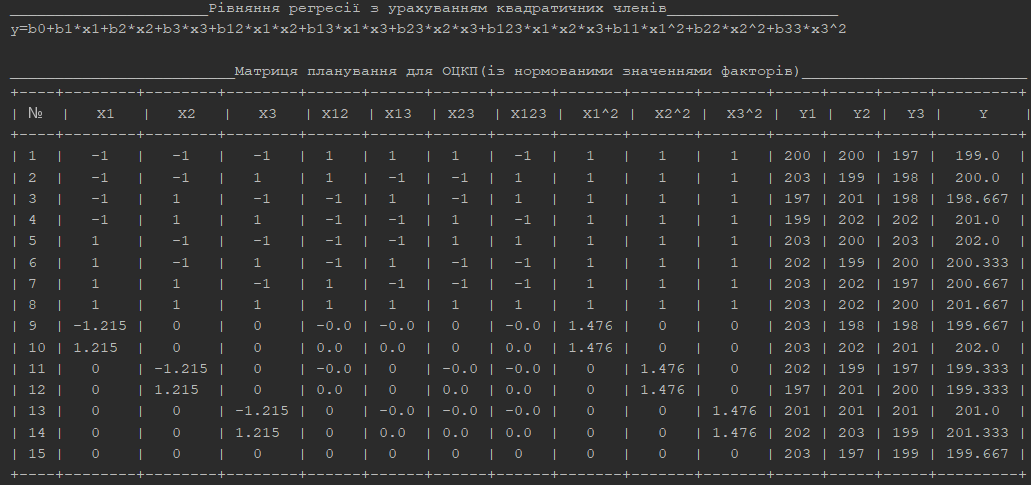
**Виконання:**

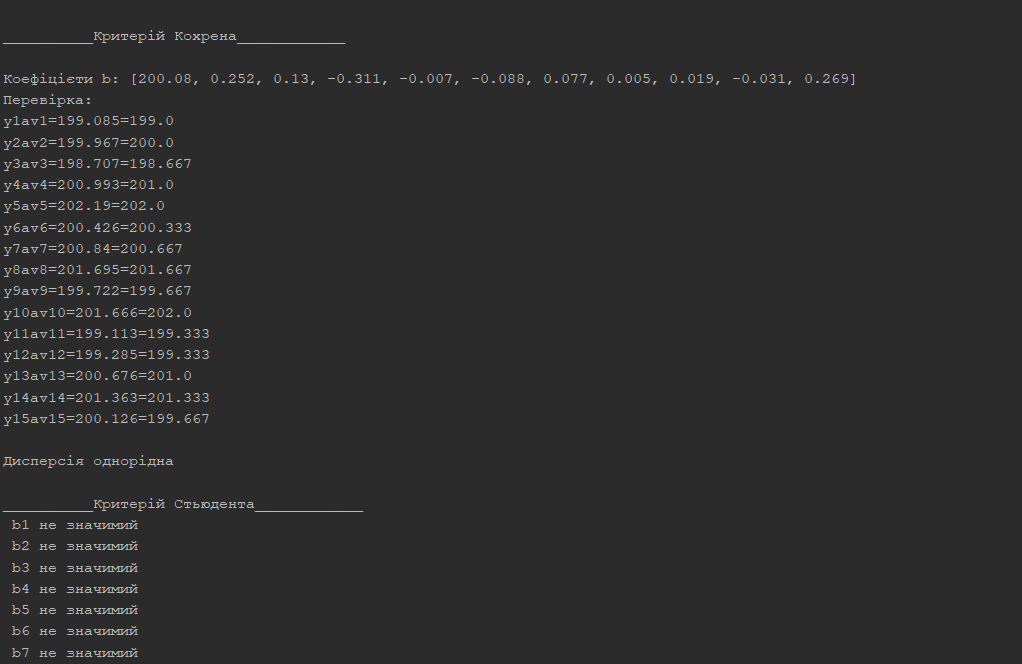
Варіант – 226

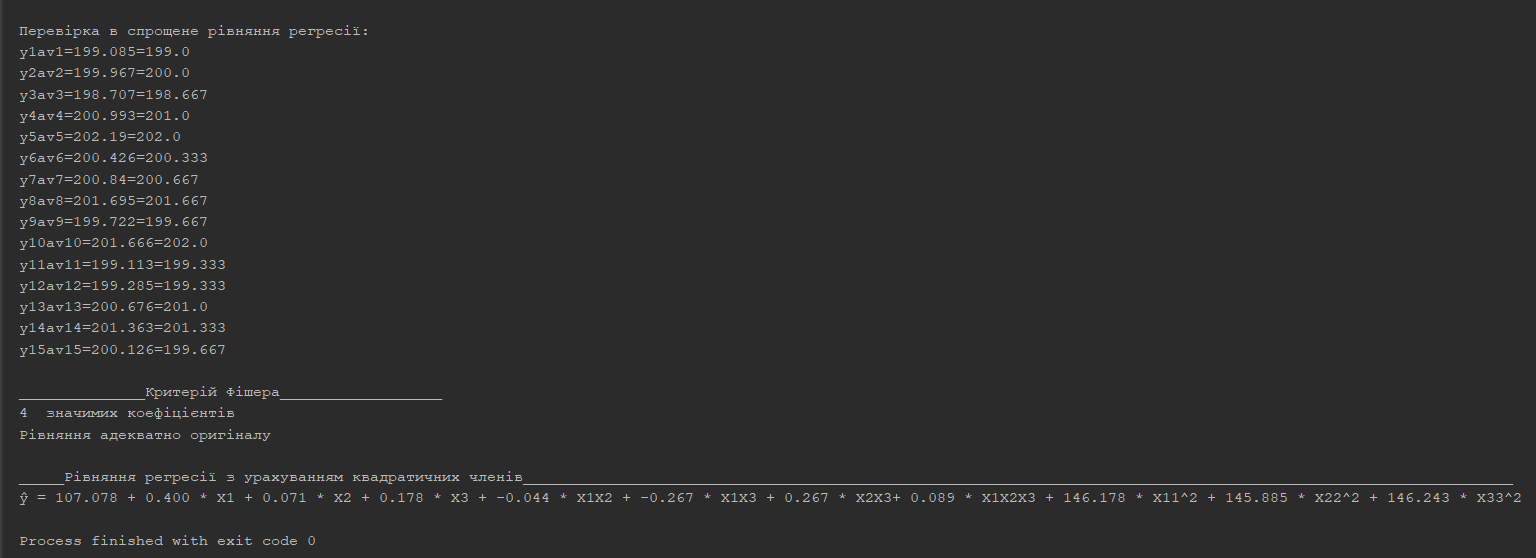


1. **Лістинг програми:**

import random  
from scipy.stats import f, t  
from prettytable import PrettyTable  
import numpy as np  
  
x1min = -5  
x1max = 4  
x2min = -2  
x2max = 7  
x3min = -1  
x3max = 2  
  
xAvmax = (x1max + x2max + x3max) / 3  
xAvmin = (x1min + x2min + x3min) / 3  
ymax = int(200 + xAvmax)  
ymin = int(200 + xAvmin)  
  
x01 = (x1max+x1min)/2  
x02 = (x2max+x2min)/2  
x03 = (x3max+x3min)/2  
deltax1 = x1max-x01  
deltax2 = x2max-x02  
deltax3 = x3max-x03  
  
m = 3  
  
X11 = [-1, -1, -1, -1, 1, 1, 1, 1, -1.215, 1.215, 0, 0, 0, 0, 0]  
X22 = [-1, -1, 1, 1, -1, -1, 1, 1, 0, 0, -1.215, 1.215, 0, 0, 0]  
X33 = [-1, 1, -1, 1, -1, 1, -1, 1, 0, 0, 0, 0, -1.215, 1.215, 0]  
  
  
  
def sumkf2(x1, x2):  
 xn = []  
 for i in range(len(x1)):  
 xn.append(round(x1[i] \* x2[i],3))  
 return xn  
  
  
def sumkf3(x1, x2, x3):  
 xn = []  
 for i in range(len(x1)):  
 xn.append(round(x1[i] \* x2[i] \* x3[i],3))  
 return xn  
  
  
def kv(x):  
 xn = []  
 for i in range(len(x)):  
 xn.append(round(x[i] \* x[i],3))  
 return xn  
  
  
X12 = sumkf2(X11, X22)  
X13 = sumkf2(X11, X33)  
X23 = sumkf2(X22, X33)  
X123 = sumkf3(X11, X22, X33)  
X1kv = kv(X11)  
X2kv = kv(X22)  
X3kv = kv(X33)  
  
for i in range(1, m + 1):  
 globals()['Y%s' % i] = [random.randrange(ymin, ymax, 1) for k in range(15)]  
  
  
y1av1, y2av2, y3av3, y4av4, y5av5, y6av6, y7av7, y8av8, y9av9, y10av10, y11av11, y12av12, y13av13, y14av14, y15av15 = 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0  
for i in range(1, m + 1):  
 for k in range(15):  
 globals()['y%sav%s' % (k + 1, k + 1)] += globals()['Y%s' % i][k]/m  
  
yav = []  
for i in range(15):  
 yav.append(round(globals()['y%sav%s' % (i+1, i+1)] ,3 ))  
  
print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Рівняння регресії з урахуванням квадратичних членів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
print("y=b0+b1\*x1+b2\*x2+b3\*x3+b12\*x1\*x2+b13\*x1\*x3+b23\*x2\*x3+b123\*x1\*x2\*x3+b11\*x1^2+b22\*x2^2+b33\*x3^2")  
table1 = PrettyTable()  
table1.add\_column("№", (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15))  
table1.add\_column("X1", X11)  
table1.add\_column("X2", X22)  
table1.add\_column("X3", X33)  
table1.add\_column("X12", X12)  
table1.add\_column("X13", X13)  
table1.add\_column("X23", X23)  
table1.add\_column("X123", X123)  
table1.add\_column("X1^2", X1kv)  
table1.add\_column("X2^2", X2kv)  
table1.add\_column("X3^2", X3kv)  
for i in range(1, m + 1):  
 table1.add\_column("Y" + str(i), globals()['Y%s' % i])  
table1.add\_column("Y", yav)  
print("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Матриця планування для ОЦКП(із нормованими значеннями факторів)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
print(table1)  
  
X1 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max, round(-1.215\*deltax1+x01,3), round(1.215\*deltax1+x01,3), x01, x01 ,x01 , x01, x01]  
X2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max, x02, x02, round(-1.215\*deltax2+x02,3), round(1.215\*deltax2+x02,3), x02, x02, x02]  
X3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max, x03, x03, x03, x03, round(-1.215\*deltax3+x03,3), round(1.215\*deltax3+x03,3), x03]  
X12 = sumkf2(X1, X2)  
X13 = sumkf2(X1, X3)  
X23 = sumkf2(X2, X3)  
X123 = sumkf3(X1, X2, X3)  
X1kv = kv(X1)  
X2kv = kv(X2)  
X3kv = kv(X3)  
for i in range(15):  
 globals()['d%s' % (i + 1)] = 0  
for k in range(1, m + 1):  
 for i in range(15):  
 globals()['d%s' % (i + 1)] += ((globals()['Y%s' % (k)][i]) - globals()['y%sav%s' % (i + 1, i + 1)] ) \*\* 2/m  
  
X0 =[1]\*15  
  
b = np.linalg.lstsq(list(zip(X0 , X1, X2, X3, X12, X13, X23, X123, X1kv, X2kv, X3kv)), yav, rcond=None)[0]  
b = [round(i , 3) for i in b]  
print("\n\_\_\_\_\_\_\_\_\_\_Критерій Кохрена\_\_\_\_\_\_\_\_\_\_\_\_")  
print("\nКоефіцієти b:" ,b)  
print("Перевірка:")  
for i in range(15):  
 print("y"+str(i+1)+"av"+str(i+1)+"="+str(round(b[0] + b[1]\*X1[i]+b[2]\*X2[i]+b[3]\*X3[i]+b[4]\*X1[i]\*X2[i]+b[5]\*X1[i]\*X3[i]+b[6]\*X2[i]\*X3[i]+b[7]\*X1[i]\*X2[i]\*X3[i]+b[8]\*X1kv[i]+b[9]\*X2kv[i]+b[10]\*X3kv[i],3))+"="+ str(round( globals()['y%sav%s' % (i + 1, i + 1)],3)))  
print()  
  
dcouple = []  
for i in range(15):  
 dcouple.append(round(globals()['d%s' % (i+1)] ,3 ))  
  
  
Gp = max(dcouple) / sum(dcouple)  
q = 0.05  
f1 = m - 1  
f2 = N = 15  
fisher = f.isf(\*[q / f2, f1, (f2 - 1) \* f1])  
Gt = round(fisher / (fisher + (f2 - 1)), 4)  
if Gp < Gt:  
 print("Дисперсія однорідна")  
 print("\n\_\_\_\_\_\_\_\_\_\_Критерій Стьюдента\_\_\_\_\_\_\_\_\_\_\_\_")  
 sb = sum(dcouple) / N  
 ssbs = sb / N \* m  
 sbs = ssbs \*\* 0.5  
  
 beta0 = (y1av1\*1+y2av2\*1+y3av3\*1+y4av4\*1+y5av5\*1+y6av6\*1+y7av7\*1+y8av8\*1+y9av9\*(-1.215)+y10av10\*1.215+y11av11\*0+y12av12\*0+y13av13\*0+y14av14\*0+y15av15\*0)/15  
 beta1 = (y1av1\*(-1)+y2av2\*(-1)+y3av3\*(-1)+y4av4\*(-1)+y5av5\*1+y6av6\*1+y7av7\*1+y8av8\*1+y9av9\*0+y10av10\*0+y11av11\*(-1.215)+y12av12\*1.215+y13av13\*0+y14av14\*0+y15av15\*0)/15  
 beta2 = (y1av1\*(-1)+y2av2\*(-1)+y3av3\*1+y4av4\*1+y5av5\*(-1)+y6av6\*(-1)+y7av7\*1+y8av8\*1+y9av9\*0+y10av10\*0+y11av11\*0+y12av12\*0+y13av13\*(-1.215)+y14av14\*1.215+y15av15\*0)/15  
 beta3 = (y1av1\*(-1)+y2av2\*1+y3av3\*(-1)+y4av4\*1+y5av5\*(-1)+y6av6\*1+y7av7\*(-1)+y8av8\*1)/15  
 beta4 = (y1av1\*1+y2av2\*1+y3av3\*(-1)+y4av4\*(-1)+y5av5\*(-1)+y6av6\*(-1)+y7av7\*1+y8av8\*1)/15  
 beta5 = (y1av1\*1+y2av2\*(-1)+y3av3\*1+y4av4\*(-1)+y5av5\*(-1)+y6av6\*1+y7av7\*(-1)+y8av8\*1)/15  
 beta6 = (y1av1\*1+y2av2\*(-1)+y3av3\*(-1)+y4av4\*1+y5av5\*1+y6av6\*(-1)+y7av7\*(-1)+y8av8\*1)/15  
 beta7 = (y1av1\*(-1)+y2av2\*1+y3av3\*1+y4av4\*(-1)+y5av5\*1+y6av6\*(-1)+y7av7\*(-1)+y8av8\*1)/15  
 beta8 = (y1av1\*1+y2av2\*1+y3av3\*1+y4av4\*1+y5av5\*1+y6av6\*1+y7av7\*1+y8av8\*1+y9av9\*1.46723+y10av10\*1.46723)/15  
 beta9 = (y1av1\*1+y2av2\*1+y3av3\*1+y4av4\*1+y5av5\*1+y6av6\*1+y7av7\*1+y8av8\*1+y11av11\*1.46723+y12av12\*1.46723)/15  
 beta10 = (y1av1\*1+y2av2\*1+y3av3\*1+y4av4\*1+y5av5\*1+y6av6\*1+y7av7\*1+y8av8\*1+y13av13\*1.46723+y14av14\*1.46723)/15  
  
 f3 = f1 \* f2  
 ttabl = round(abs(t.ppf(q / 2, f3)), 4)  
  
 d = 11  
 for i in range(11):  
 if ((abs(globals()['beta%s' % (i)]) / sbs) < ttabl):  
 print(" b%s не значимий" % (i))  
 globals()['b%s' % i ] = 0  
 d = d - 1  
 print("\nПеревірка в спрощене рівняння регресії:")  
 for i in range(15):  
 print("y"+str(i+1)+"av"+str(i+1)+"="+str(round(b[0] + b[1]\*X1[i]+b[2]\*X2[i]+b[3]\*X3[i]+b[4]\*X1[i]\*X2[i]+b[5]\*X1[i]\*X3[i]+b[6]\*X2[i]\*X3[i]+b[7]\*X1[i]\*X2[i]\*X3[i]+b[8]\*X1kv[i]+b[9]\*X2kv[i]+b[10]\*X3kv[i],3))+"="+ str(round( globals()['y%sav%s' % (i + 1, i + 1)],3)))  
  
 yy1 = b[0]+b[1]\*x1min+b[2]\*x2min+b[3]\*x3min+b[4]\*x1min\*x2min+b[5]\*x1min\*x3min+b[6]\*x2min\*x3min+b[7]\*x1min\*x2min\*x3min+b[8]\*x1min\*x1min+b[9]\*x2min\*x2min+b[10]\*x3min\*x3min  
 yy2 = b[0]+b[1]\*x1min+b[2]\*x2min+b[3]\*x3max+b[4]\*x1min\*x2min+b[5]\*x1min\*x3max+b[6]\*x2min\*x3max+b[7]\*x1min\*x2min\*x3max+b[8]\*x1min\*x1min+b[9]\*x2min\*x2min+b[10]\*x3max\*x3max  
 yy3 = b[0]+b[1]\*x1min+b[2]\*x2max+b[3]\*x3min+b[4]\*x1min\*x2max+b[5]\*x1min\*x3min+b[6]\*x2max\*x3min+b[7]\*x1min\*x2max\*x3min+b[8]\*x1min\*x1min+b[9]\*x2max\*x2max+b[10]\*x3min\*x3min  
 yy4 = b[0]+b[1]\*x1min+b[2]\*x2max+b[3]\*x3max+b[4]\*x1min\*x2max+b[5]\*x1min\*x3max+b[6]\*x2max\*x3max+b[7]\*x1min\*x2max\*x3max+b[8]\*x1min\*x1min+b[9]\*x2max\*x2max+b[10]\*x3max\*x3max  
 yy5 = b[0]+b[1]\*x1max+b[2]\*x2min+b[3]\*x3min+b[4]\*x1max\*x2min+b[5]\*x1max\*x3min+b[6]\*x2min\*x3min+b[7]\*x1max\*x2min\*x3min+b[8]\*x1max\*x1max+b[9]\*x2min\*x2min+b[10]\*x3min\*x3min  
 yy6 = b[0]+b[1]\*x1max+b[2]\*x2min+b[3]\*x3max+b[4]\*x1max\*x2min+b[5]\*x1max\*x3max+b[6]\*x2min\*x3max+b[7]\*x1max\*x2min\*x3max+b[8]\*x1max\*x1max+b[9]\*x2min\*x2min+b[10]\*x3min\*x3max  
 yy7 = b[0]+b[1]\*x1max+b[2]\*x2max+b[3]\*x3min+b[4]\*x1max\*x2max+b[5]\*x1max\*x3min+b[6]\*x2max\*x3min+b[7]\*x1max\*x2min\*x3max+b[8]\*x1max\*x1max+b[9]\*x2max\*x2max+b[10]\*x3min\*x3min  
 yy8 = b[0]+b[1]\*x1max+b[2]\*x2max+b[3]\*x3max+b[4]\*x1max\*x2max+b[5]\*x1max\*x3max+b[6]\*x2max\*x3max+b[7]\*x1max\*x2max\*x3max+b[8]\*x1max\*x1max+b[9]\*x2max\*x2max+b[10]\*x3min\*x3max  
  
 yy9 = b[0]+b[1]\*X1[8]+b[2]\*X2[8]+b[3]\*X3[8]+b[4]\*X12[8]+b[5]\*X13[8]+b[6]\*X23[8]+b[7]\*X123[8]+b[8]\*X1kv[8]+b[9]\*X2kv[8]+b[10]\*X3kv[8]  
 yy10 = b[0]+b[1]\*X1[9]+b[2]\*X2[9]+b[3]\*X3[9]+b[4]\*X12[9]+b[5]\*X13[9]+b[6]\*X23[9]+b[7]\*X123[9]+b[8]\*X1kv[9]+b[9]\*X2kv[9]+b[10]\*X3kv[9]  
 yy11 = b[0]+b[1]\*X1[10]+b[2]\*X2[10]+b[3]\*X3[10]+b[4]\*X12[10]+b[5]\*X13[10]+b[6]\*X23[10]+b[7]\*X123[10]+b[8]\*X1kv[10]+b[9]\*X2kv[10]+b[10]\*X3kv[10]  
 yy12 = b[0]+b[1]\*X1[11]+b[2]\*X2[11]+b[3]\*X3[11]+b[4]\*X12[11]+b[5]\*X13[11]+b[6]\*X23[11]+b[7]\*X123[11]+b[8]\*X1kv[11]+b[9]\*X2kv[11]+b[10]\*X3kv[11]  
 yy13 = b[0]+b[1]\*X1[12]+b[2]\*X2[12]+b[3]\*X3[12]+b[4]\*X12[12]+b[5]\*X13[12]+b[6]\*X23[12]+b[7]\*X123[12]+b[8]\*X1kv[12]+b[9]\*X2kv[12]+b[10]\*X3kv[12]  
 yy14 = b[0]+b[1]\*X1[13]+b[2]\*X2[13]+b[3]\*X3[13]+b[4]\*X12[13]+b[5]\*X13[13]+b[6]\*X23[13]+b[7]\*X123[13]+b[8]\*X1kv[13]+b[9]\*X2kv[13]+b[10]\*X3kv[13]  
 yy15 = b[0]+b[1]\*X1[14]+b[2]\*X2[14]+b[3]\*X3[14]+b[4]\*X12[14]+b[5]\*X13[14]+b[6]\*X23[14]+b[7]\*X123[14]+b[8]\*X1kv[14]+b[9]\*X2kv[14]+b[10]\*X3kv[14]  
 print("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_Критерій Фішера\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 print(d, " значимих коефіцієнтів")  
 f4 = N - d  
 sad = ((yy1-y1av1)\*\*2+(yy2-y2av2)\*\*2+(yy3-y3av3)\*\*2+(yy4-y4av4)\*\*2+(yy5-y5av5)\*\*2+(yy6-y6av6)\*\*2+(yy7-y7av7)\*\*2+(yy8-y8av8)\*\*2+ (yy9-y9av9)\*\*2+(yy10-y10av10)\*\*2+(yy11-y11av11)\*\*2+(yy12-y12av12)\*\*2+(yy13-y13av13)\*\*2+(yy14-y14av14)\*\*2+(yy15-y15av15)\*\*2)\*(m/(N-d))  
  
 Fp = sad / sb  
 Ft = round(abs(f.isf(q, f4, f3)), 4)  
  
 cont = 0  
 if Fp > Ft:  
 print("Рівняння неадекватно оригіналу")  
 cont = 1  
 m+=1  
 else:  
 print("Рівняння адекватно оригіналу")  
  
else:  
 print("Дисперсія неоднорідна(збільшемо кількість дослідів)")  
 m += 1  
print("\n\_\_\_\_\_Рівняння регресії з урахуванням квадратичних членів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
print("ŷ = {:.3f} + {:.3f} \* X1 + {:.3f} \* X2 + {:.3f} \* X3 + {:.3f} \* Х1X2 + {:.3f} \* Х1X3 + {:.3f} \* Х2X3"  
 "+ {:.3f} \* Х1Х2X3 + {:.3f} \* X11^2 + {:.3f} \* X22^2 + {:.3f} \* X33^2"  
 .format(beta0, beta1, beta2, beta3, beta4, beta5, beta6, beta7, beta8, beta9, beta10))

1. **Результат виконання роботи програми:**





**Висновок:**

У ході виконання даної роботи ми провели трьохфакторний експеримент при використанні рівняння з урахуванням квадратичних членів. Склали матрицю планування, знайшли коефіцієнти рівняння регресії, провели 3 статистичні перевірки. Була написана текстова програма, результати якої наведені вище.