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)»

Виконав:

студент групи ІВ-82

Онищук В.В.

Залікова книжка № 8218

Перевірив Регіда П. Г.

Київ 2020р.

**Лабораторна робота №5**

**Тема:** Проведення трьохфакторного експерименту при використанні рівняння регресії з урахуванням квадратичних членів.

(центральний ортогональний композиційний план)

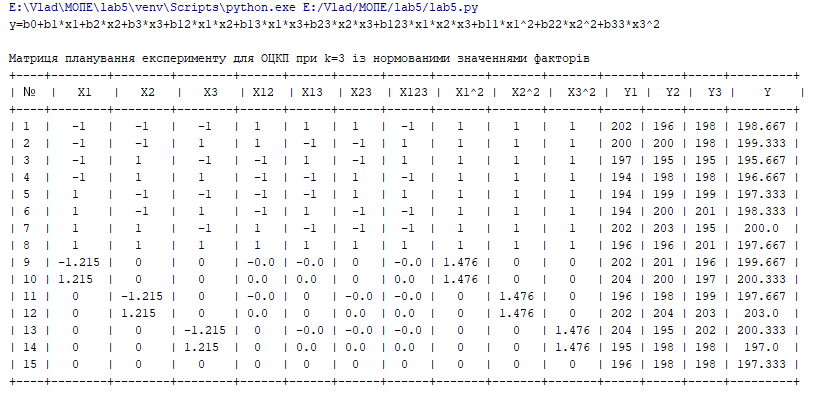
**Мета:** Провести трьохфакторний експеримент з урахуванням квадратичних членів ,використовуючи центральний ортогональний композиційний план. Знайти рівняння регресії, яке буде адекватним для опису об'єкту.

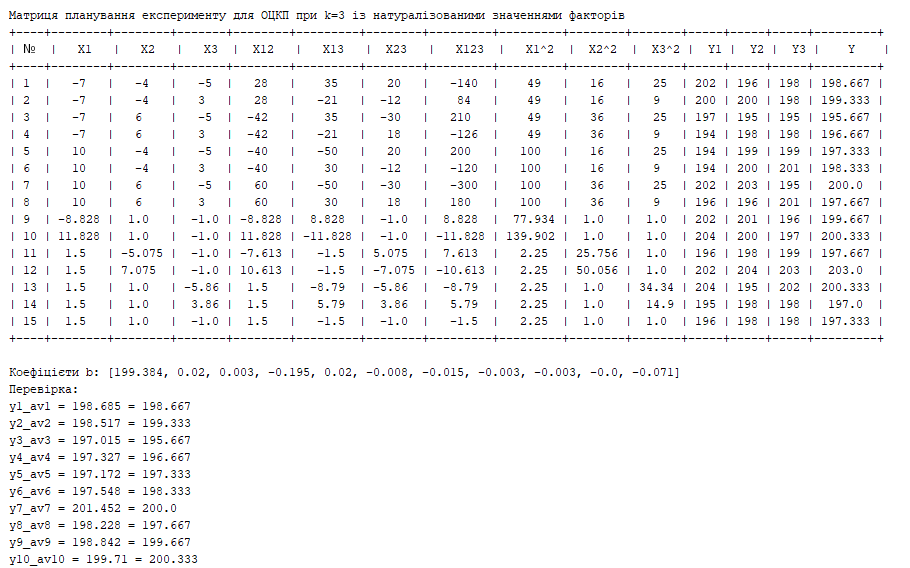


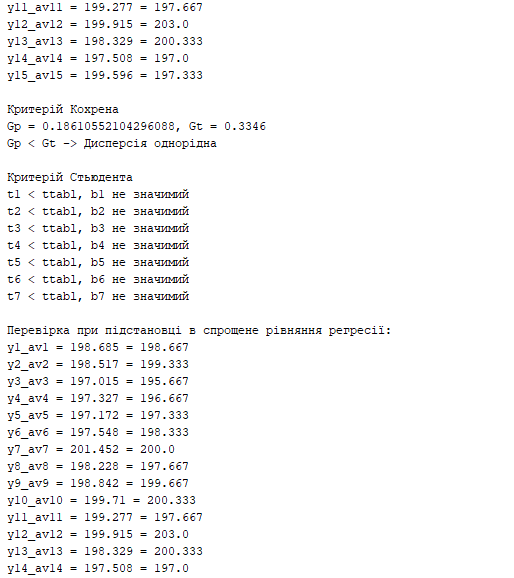
1. Лістинг програми:

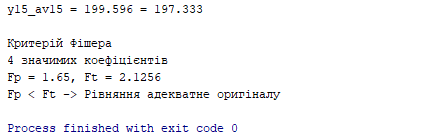
**import** random  
**from** scipy.stats **import** f, t  
**from** prettytable **import** PrettyTable  
**import** numpy **as** np  
  
*# Варіант 116*x1\_min = -7  
x1\_max = 10  
x2\_min = -4  
x2\_max = 6  
x3\_min = -5  
x3\_max = 3  
  
x\_av\_max = (x1\_max + x2\_max + x3\_max) / 3  
x\_av\_min = (x1\_min + x2\_min + x3\_min) / 3  
y\_max = int(200 + x\_av\_max)  
y\_min = int(200 + x\_av\_min)  
  
x01 = (x1\_max + x1\_min) / 2  
x02 = (x2\_max + x2\_min) / 2  
x03 = (x3\_max + x3\_min) / 2  
deltax1 = x1\_max - x01  
deltax2 = x2\_max - x02  
deltax3 = x3\_max - 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)  
X1\_kv = kv(X11)  
X2\_kv = kv(X22)  
X3\_kv = kv(X33)  
  
**for** i **in** range(1, m + 1):  
 globals()[**'Y%s'** % i] = [random.randrange(y\_min, y\_max, 1) **for** k **in** range(15)]  
  
*# Середнє значення функції відгуку за рядками*y1\_av1, y2\_av2, y3\_av3, y4\_av4, y5\_av5, y6\_av6, y7\_av7, y8\_av8, y9\_av9, y10\_av10, y11\_av11, y12\_av12, y13\_av13, \  
y14\_av14, y15\_av15 = 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%s\_av%s'** % (k + 1, k + 1)] += globals()[**'Y%s'** % i][k] / m  
  
y\_av = []  
**for** i **in** range(15):  
 y\_av.append(round(globals()[**'y%s\_av%s'** % (i + 1, i + 1)], 3))  
  
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\n"**)  
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"**, X1\_kv)  
table1.add\_column(**"X2^2"**, X2\_kv)  
table1.add\_column(**"X3^2"**, X3\_kv)  
**for** i **in** range(1, m + 1):  
 table1.add\_column(**"Y"** + str(i), globals()[**'Y%s'** % i])  
table1.add\_column(**"Y"**, y\_av)  
print(**"Матриця планування експерименту для ОЦКП при k=3 із нормованими значеннями факторів"**)  
print(table1, **"\n"**)  
  
X1 = [x1\_min, x1\_min, x1\_min, x1\_min, x1\_max, x1\_max, x1\_max, x1\_max, round(-1.215 \* deltax1 + x01, 3),  
 round(1.215 \* deltax1 + x01, 3), x01, x01, x01, x01, x01]  
X2 = [x2\_min, x2\_min, x2\_max, x2\_max, x2\_min, x2\_min, x2\_max, x2\_max, x02, x02, round(-1.215 \* deltax2 + x02, 3),  
 round(1.215 \* deltax2 + x02, 3), x02, x02, x02]  
X3 = [x3\_min, x3\_max, x3\_min, x3\_max, x3\_min, x3\_max, x3\_min, x3\_max, 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)  
X1\_kv = kv(X1)  
X2\_kv = kv(X2)  
X3\_kv = kv(X3)  
  
table2 = PrettyTable()  
table2.add\_column(**"№"**, (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15))  
table2.add\_column(**"X1"**, X1)  
table2.add\_column(**"X2"**, X2)  
table2.add\_column(**"X3"**, X3)  
table2.add\_column(**"X12"**, X12)  
table2.add\_column(**"X13"**, X13)  
table2.add\_column(**"X23"**, X23)  
table2.add\_column(**"X123"**, X123)  
table2.add\_column(**"X1^2"**, X1\_kv)  
table2.add\_column(**"X2^2"**, X2\_kv)  
table2.add\_column(**"X3^2"**, X3\_kv)  
**for** i **in** range(1, m + 1):  
 table2.add\_column(**"Y"** + str(i), globals()[**'Y%s'** % i])  
table2.add\_column(**"Y"**, y\_av)  
print(**"Матриця планування експерименту для ОЦКП при k=3 із натуралізованими значеннями факторів"**)  
print(table2)  
  
*# Дисперсія за рядками***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%s\_av%s'** % (i + 1, i + 1)]) \*\* 2 / m  
  
X0 = [1] \* 15  
  
b = np.linalg.lstsq(list(zip(X0, X1, X2, X3, X12, X13, X23, X123, X1\_kv, X2\_kv, X3\_kv)), y\_av, rcond=**None**)[0]  
b = [round(i, 3) **for** i **in** b]  
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] \* X1\_kv[i] + b[9] \* X2\_kv[i] + b[10] \* X3\_kv[i], 3)) + **" = "** + str(  
 round(globals()[**'y%s\_av%s'** % (i + 1, i + 1)], 3)))  
print()  
  
dcouple = []  
**for** i **in** range(15):  
 dcouple.append(round(globals()[**'d%s'** % (i + 1)], 3))  
  
print(**"Критерій Кохрена"**)  
**while True**:  
 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)  
 print(**"Gp = "** + str(Gp) + **", Gt = "** + str(Gt))  
 **if** Gp > Gt:  
 print(**"Дисперсія неоднорідна , збільшуємо m"**)  
 m += 1  
 **else**:  
 print(**"Gp < Gt -> Дисперсія однорідна\n"**)  
 print(**"Критерій Стьюдента"**)  
 sb = sum(dcouple) / N  
 ssbs = sb / N \* m  
 sbs = ssbs \*\* 0.5  
  
 beta0 = (y1\_av1 \* 1 + y2\_av2 \* 1 + y3\_av3 \* 1 + y4\_av4 \* 1 + y5\_av5 \* 1 + y6\_av6 \* 1 + y7\_av7 \* 1 + y8\_av8 \* 1 +  
 y9\_av9 \* (-1.215) + y10\_av10 \* 1.215 + y11\_av11 \* 0 + y12\_av12 \* 0 + y13\_av13 \* 0 + y14\_av14 \* 0 +  
 y15\_av15 \* 0) / 15  
 beta1 = (y1\_av1 \* (-1) + y2\_av2 \* (-1) + y3\_av3 \* (-1) + y4\_av4 \* (  
 -1) + y5\_av5 \* 1 + y6\_av6 \* 1 + y7\_av7 \* 1 + y8\_av8 \* 1 + y9\_av9 \* 0 + y10\_av10 \* 0 + y11\_av11 \* (  
 -1.215) + y12\_av12 \* 1.215 + y13\_av13 \* 0 + y14\_av14 \* 0 + y15\_av15 \* 0) / 15  
 beta2 = (y1\_av1 \* (-1) + y2\_av2 \* (-1) + y3\_av3 \* 1 + y4\_av4 \* 1 + y5\_av5 \* (-1) + y6\_av6 \* (  
 -1) + y7\_av7 \* 1 + y8\_av8 \* 1 + y9\_av9 \* 0 + y10\_av10 \* 0 + y11\_av11 \* 0 + y12\_av12 \* 0 + y13\_av13 \* (  
 -1.215) + y14\_av14 \* 1.215 + y15\_av15 \* 0) / 15  
 beta3 = (y1\_av1 \* (-1) + y2\_av2 \* 1 + y3\_av3 \* (-1) + y4\_av4 \* 1 + y5\_av5 \* (-1) + y6\_av6 \* 1 + y7\_av7 \* (  
 -1) + y8\_av8 \* 1) / 15  
 beta4 = (y1\_av1 \* 1 + y2\_av2 \* 1 + y3\_av3 \* (-1) + y4\_av4 \* (-1) + y5\_av5 \* (-1) + y6\_av6 \* (  
 -1) + y7\_av7 \* 1 + y8\_av8 \* 1) / 15  
 beta5 = (y1\_av1 \* 1 + y2\_av2 \* (-1) + y3\_av3 \* 1 + y4\_av4 \* (-1) + y5\_av5 \* (-1) + y6\_av6 \* 1 + y7\_av7 \* (  
 -1) + y8\_av8 \* 1) / 15  
 beta6 = (y1\_av1 \* 1 + y2\_av2 \* (-1) + y3\_av3 \* (-1) + y4\_av4 \* 1 + y5\_av5 \* 1 + y6\_av6 \* (-1) + y7\_av7 \* (  
 -1) + y8\_av8 \* 1) / 15  
 beta7 = (y1\_av1 \* (-1) + y2\_av2 \* 1 + y3\_av3 \* 1 + y4\_av4 \* (-1) + y5\_av5 \* 1 + y6\_av6 \* (-1) + y7\_av7 \* (  
 -1) + y8\_av8 \* 1) / 15  
 beta8 = (y1\_av1 \* 1 + y2\_av2 \* 1 + y3\_av3 \* 1 + y4\_av4 \* 1 + y5\_av5 \* 1 + y6\_av6 \* 1 + y7\_av7 \* 1 + y8\_av8 \* 1  
 + y9\_av9 \* 1.46723 + y10\_av10 \* 1.46723) / 15  
 beta9 = (y1\_av1 \* 1 + y2\_av2 \* 1 + y3\_av3 \* 1 + y4\_av4 \* 1 + y5\_av5 \* 1 + y6\_av6 \* 1 + y7\_av7 \* 1 + y8\_av8 \* 1  
 + y11\_av11 \* 1.46723 + y12\_av12 \* 1.46723) / 15  
 beta10 = (y1\_av1 \* 1 + y2\_av2 \* 1 + y3\_av3 \* 1 + y4\_av4 \* 1 + y5\_av5 \* 1 + y6\_av6 \* 1 + y7\_av7 \* 1 + y8\_av8 \* 1  
 + y13\_av13 \* 1.46723 + y14\_av14 \* 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(**"t%s < ttabl, b%s не значимий"** % (i, i))  
 globals()[**'b%s'** % i] = 0  
 d = d - 1  
 print(**"\nПеревірка при підстановці 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] \* X1\_kv[i] + b[9] \* X2\_kv[i] + b[10] \* X3\_kv[i],  
 3)) + **" = "** + str(round(globals()[**'y%s\_av%s'** % (i + 1, i + 1)], 3)))  
  
 yy1 = b[0] + b[1] \* x1\_min + b[2] \* x2\_min + b[3] \* x3\_min + b[4] \* x1\_min \* x2\_min + b[5] \* x1\_min \* x3\_min + b[  
 6] \* x2\_min \* x3\_min + b[7] \* x1\_min \* x2\_min \* x3\_min + b[8] \* x1\_min \* x1\_min + b[9] \* x2\_min \* x2\_min + b[  
 10] \* x3\_min \* x3\_min  
 yy2 = b[0] + b[1] \* x1\_min + b[2] \* x2\_min + b[3] \* x3\_max + b[4] \* x1\_min \* x2\_min + b[5] \* x1\_min \* x3\_max + b[  
 6] \* x2\_min \* x3\_max + b[7] \* x1\_min \* x2\_min \* x3\_max + b[8] \* x1\_min \* x1\_min + b[9] \* x2\_min \* x2\_min + b[  
 10] \* x3\_max \* x3\_max  
 yy3 = b[0] + b[1] \* x1\_min + b[2] \* x2\_max + b[3] \* x3\_min + b[4] \* x1\_min \* x2\_max + b[5] \* x1\_min \* x3\_min + b[  
 6] \* x2\_max \* x3\_min + b[7] \* x1\_min \* x2\_max \* x3\_min + b[8] \* x1\_min \* x1\_min + b[9] \* x2\_max \* x2\_max + b[  
 10] \* x3\_min \* x3\_min  
 yy4 = b[0] + b[1] \* x1\_min + b[2] \* x2\_max + b[3] \* x3\_max + b[4] \* x1\_min \* x2\_max + b[5] \* x1\_min \* x3\_max + b[  
 6] \* x2\_max \* x3\_max + b[7] \* x1\_min \* x2\_max \* x3\_max + b[8] \* x1\_min \* x1\_min + b[9] \* x2\_max \* x2\_max + b[  
 10] \* x3\_max \* x3\_max  
 yy5 = b[0] + b[1] \* x1\_max + b[2] \* x2\_min + b[3] \* x3\_min + b[4] \* x1\_max \* x2\_min + b[5] \* x1\_max \* x3\_min + b[  
 6] \* x2\_min \* x3\_min + b[7] \* x1\_max \* x2\_min \* x3\_min + b[8] \* x1\_max \* x1\_max + b[9] \* x2\_min \* x2\_min + b[  
 10] \* x3\_min \* x3\_min  
 yy6 = b[0] + b[1] \* x1\_max + b[2] \* x2\_min + b[3] \* x3\_max + b[4] \* x1\_max \* x2\_min + b[5] \* x1\_max \* x3\_max + b[  
 6] \* x2\_min \* x3\_max + b[7] \* x1\_max \* x2\_min \* x3\_max + b[8] \* x1\_max \* x1\_max + b[9] \* x2\_min \* x2\_min + b[  
 10] \* x3\_min \* x3\_max  
 yy7 = b[0] + b[1] \* x1\_max + b[2] \* x2\_max + b[3] \* x3\_min + b[4] \* x1\_max \* x2\_max + b[5] \* x1\_max \* x3\_min + b[  
 6] \* x2\_max \* x3\_min + b[7] \* x1\_max \* x2\_min \* x3\_max + b[8] \* x1\_max \* x1\_max + b[9] \* x2\_max \* x2\_max + b[  
 10] \* x3\_min \* x3\_min  
 yy8 = b[0] + b[1] \* x1\_max + b[2] \* x2\_max + b[3] \* x3\_max + b[4] \* x1\_max \* x2\_max + b[5] \* x1\_max \* x3\_max + b[  
 6] \* x2\_max \* x3\_max + b[7] \* x1\_max \* x2\_max \* x3\_max + b[8] \* x1\_max \* x1\_max + b[9] \* x2\_max \* x2\_max + b[  
 10] \* x3\_min \* x3\_max  
  
 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] \* X1\_kv[8] + b[9] \* X2\_kv[8] + b[10] \* X3\_kv[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] \* X1\_kv[9] + b[9] \* X2\_kv[9] + b[10] \* X3\_kv[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] \* X1\_kv[10] + b[9] \* X2\_kv[10] + b[10] \* X3\_kv[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] \* X1\_kv[11] + b[9] \* X2\_kv[11] + b[10] \* X3\_kv[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] \* X1\_kv[12] + b[9] \* X2\_kv[12] + b[10] \* X3\_kv[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] \* X1\_kv[13] + b[9] \* X2\_kv[13] + b[10] \* X3\_kv[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] \* X1\_kv[14] + b[9] \* X2\_kv[14] + b[10] \* X3\_kv[14]  
 print(**"\nКритерій Фішера"**)  
 print(d, **"значимих коефіцієнтів"**)  
 f4 = N - d  
 sad = ((yy1 - y1\_av1) \*\* 2 + (yy2 - y2\_av2) \*\* 2 + (yy3 - y3\_av3) \*\* 2 + (yy4 - y4\_av4) \*\* 2 + (  
 yy5 - y5\_av5) \*\* 2 + (  
 yy6 - y6\_av6) \*\* 2 + (yy7 - y7\_av7) \*\* 2 + (yy8 - y8\_av8) \*\* 2 + (yy9 - y9\_av9) \*\* 2 + (  
 yy10 - y10\_av10) \*\* 2 + (yy11 - y11\_av11) \*\* 2 + (yy12 - y12\_av12) \*\* 2 + (yy13 - y13\_av13) \*\* 2 + (  
 yy14 - y14\_av14) \*\* 2 + (yy15 - y15\_av15) \*\* 2) \* (m / (N - d))  
  
 Fp = sad / sb  
  
 Ft = round(abs(f.isf(q, f4, f3)), 4)  
  
 cont = 0  
 print(**"Fp = "** + str(round(Fp, 2)) + **", Ft = "** + str(Ft))  
 **if** Fp > Ft:  
 print(**"Fp > Ft -> Рівняння неадекватне оригіналу"**)  
 cont = 1  
 **else**:  
 print(**"Fp < Ft -> Рівняння адекватне оригіналу"**)  
 **break**

1. Результат виконання роботи програми:









**Висновок:** Отже, у ході виконання лабораторної роботи №5 провели трьохфакторний експеримент з урахуванням квадратичних членів ,використовуючи центральний ортогональний композиційний план. Знайшли рівняння регресії, яке буде адекватним для опису об'єкту, результати наведені вгорі.